От издателей русского перевода

На мировом рынке компьютерной литературы существует множество книг, предназначенных для обучения основным алгоритмам и используемых при программировании. Их довольно много, и они в значительной степени конкурируют между собой. Однако среди них есть особая книга. Это трехтомник "Искусство программирования" Д. Э. Кнута, который стоит вне всякой конкуренции, входит в золотой фонд мировой литературы по информатике и является настольной книгой практически для всех, кто связан с программированием.

Мы как издатели видим ценность книги в том, что она предназначена не столько для обучения технике программирования, сколько для обучения, если это возможно, "искусству" программирования, предлагает массу рецептов усовершенствования программ и, что самое главное, учит самостоятельно находить эти рецепты.

Ни для кого не секрет, что наши программисты являются одними из наиболее высококвалифицированных специалистов в мире. Они достойно представляют за рубежом отечественную школу программирования и информатики, которая внесла значительный вклад в формирование фундаментальных основ компьютерных наук. Для сохранения такого уровня и продвижения вперед необходимо своевременное издание на русском языке книг, отражающих основные мировые достижения в этой области. Трехтомник "Искусство программирования" Д. Э. Кнута — одна из таких книг.

Мы гордимся тем, что библиотеки программистов, преподавателей, студентов, старшеклассников и многих других пополнятся этой классической книгой и что тем самым мы внесем свой вклад в формирование более глубокого понимания основ компьютерных наук. Мы глубоко убеждены, что книга "Искусство программирования" Д. Э. Кнута способна приблизить человека к совершенству. Надеемся, наше издание на русском языке этой замечательной книги еще раз подтвердит, что истинные ценности с годами не устаревают.

— Виктор Штонда, Геннадий Петриковец, Алексей Орлович, издатели

О книге "Искусство программирования"

У каждой книги своя судьба. Одни появляются незаметно и так же незаметно исчезают в потоке времени, покрываясь пылью на полках библиотек. Другие в определенный период пользуются спросом у узкого круга специалистов, пока им на смену не приходят новые справочники. Третьи, поднимаясь над временем, оказывают мощное влияние на технологическое развитие общества. Книг, относящихся к последней категории, не так уж и много. Их выход в свет — всегда праздник. Проходят годы, изменяются технологии, но новые поколения с постоянным интересом перечитывают их страницы. Именно к таким книгам относится предлагаемый читателю многотомный труд известного американского ученого Дональда Эрвина Кнута "Искусство программирования".

Прошло почти 30 лет со времени первого издания в 1972 году в США этой книги. Она была переведена на большинство языков мира, в том числе и на русский. К настоящему времени на территории стран СНГ трехтомник Д. Э. Кнута стал библиографической редкостью. В 1998 году в США вышло третье издание "Искусства программирования". В нем сохранена последовательность изложения материала прежних версий, но значительно расширен список ссылок, в который включены свежие и наиболее важные результаты, добавлены новые упражнения и комментарии, устранены неточности. Учитывая популярность во всем мире "Искусства программирования", давно следовало ожидать появления нового переводного издания на русском языке, которое вы и держите в руках.

В чем же успех "Искусства программирования" Д. Э. Кнута?

Во-первых, эта книга — великолепное учебное пособие по составлению и анализу компьютерных алгоритмов. Ее разделы могут быть включены во многие университетские курсы по технологиям программирования, теории алгоритмов, дискретной математике. Книгу могут изучать и школьники старших классов, знакомые с основами программирования. В качестве основного языка записи алгоритмов автор выбрал язык машинных команд гипотетического универсального компьютера `MIX`. Это позволяет строить оптимальные программы с учетом особенностей вычислительных машин. Перенести `MIX`-программы на реальные ЭВМ или переписать их на языках высокого уровня не составляет особого труда. Логика работы программ почти всегда поясняется простыми блок-схемами.

Во-вторых, тщательно подобранный материал, вошедший в книгу, включает в себя основные фундаментальные классы алгоритмов, которые в том или ином виде наиболее часто встречаются в практике программирования.

В-третьих, немаловажным фактором успеха книги Д. Э. Кнута является энциклопедичность изложения. Профессор Кнут отличается уникальной способностью отслеживать проблему от исторических предпосылок ее зарождения до современного состояния. Многочисленные ссылки на работы старых мастеров (вплоть до времен античности), заключенные в современный контекст, создают у читателя особое чувство причастности к историческому развитию научных идей и методов.

В-четвертых, следует отметить мастерство изложения. Книга рассчитана на широкий круг читателей — от начинающих студентов до программистов-профессионалов. Каждому будет интересно изучать компьютерные алгоритмы на своем уровне. Материал

самодостаточен. Для понимания сути методов не требуется знания особых разделов математики или специальных технологий программирования. Прослеживается определенная "музыкальная" композиция сюжетного построения (дома у Д. Э. Кнута есть небольшой орган, на котором он играет).

Список составляющих успеха "Искусства программирования" можно легко продолжить.

Автор этих строк прослушал курс "Искусство программирования" в изложении профессора Кнута в 1976–1977 годах во время стажировки в Станфордском университете. Тогда формировалась алгоритмическая основа технологий программирования, у истоков которой стоял Д. Э. Кнут. Было много обсуждений, семинаров, творческих замыслов.

Значительные книги всегда связаны с судьбой автора. Дональд Эрвин Кнут начал работу над "Искусством программирования" в 1962 году. Продолжает ее и сейчас. У него много планов. Впереди новые тома "Искусства программирования", которых с нетерпением ждут читатели.

— Профессор Анатолий Анисимов

От редактора перевода

Со времени первого издания книги "Искусство программирования" Д. Э. Кнута прошло около 25 лет. Тем не менее книга не только не устарела, но по-прежнему остается основным руководством по искусству программирования, книгой, по которой учатся понимать суть и особенности этого искусства.

За эти годы на английском языке вышло уже третье издание 1-го и 2-го томов, а также второе издание 3-го тома. Автор внес в них значительные изменения и существенные дополнения. Достаточно сказать, что число упражнений практически удвоилось, а многие упражнения, включенные в предыдущие издания (особенно ответы к ним), модифицированы. Существенно дополнены и переделаны многие главы и разделы, исправлены неточности и опечатки, добавлены многочисленные новые ссылки на литературу, использованы теоретические результаты последних лет.

Значительно преобразилась глава 3, особенно разделы 3.5 и 3.6, а также разделы 4.5.2, 4.7, 5.1.4, 5.3, 5.4.9, 6.2.2, 6.4, 6.5 и др.

Естественно, возникла необходимость в новом издании книги.

Перевод выполнен по третьему изданию 1-го и 2-го томов и второму изданию 3-го тома. Кроме того, учтены дополнения и исправления, любезно предоставленные автором.

При переводе мы старались сохранить стиль автора, обозначения и манеру изложения материала. В большинстве случаев использовались термины, принятые в научной литературе на русском языке. При необходимости приводились английские эквиваленты. По многим причинам, в частности из-за сложности некоторых разделов, читать книгу "Искусство программирования" далеко непросто. Одной из причин, которые затрудняют понимание книги, является манера изложения автора; привыкнув к ней, можно существенно облегчить чтение.

Из-за обилия материала (часто мало связанного между собой) невозможно построить книгу так, чтобы различные понятия и определения вводились сразу же при первом упоминании о них. Поэтому в главе 1 могут обсуждаться без ссылок понятия, строгие определения которых приводятся в 3-м томе. Именно поэтому так велика роль предметного указателя, без которого понимание книги было бы существенно затруднено. Надеемся, что читатель не будет удивлен, найдя ссылки на главы 7, 8 и последующие не вошедшие в предлагаемые три тома главы. Мы вместе с автором надеемся, что очень скоро они будут опубликованы и, безусловно, сразу же появятся в русском переводе в качестве продолжения этого издания.

Следует также обратить внимание на далеко не всегда стандартные обозначения, которыми пользуется автор. Так же, как и определения, эти обозначения могут появиться в 1-м томе, а вводиться во 2-м. Поэтому без указателя обозначений пользоваться книгой было бы чрезвычайно трудно. Хочу также обратить внимание на запись `[A]`, где `A` — некоторое высказывание. Эта запись встречается в формулах, а иногда и в тексте, и обозначает величину, равную индикатору `A`.

— Профессор Ю. В. Козаченко

ПРЕДИСЛОВИЕ

Дорогая Офелия!
Мне плохо от этих чисел:
Я не способен сосчитать мои стоны!
— *Гамлет* (акт II, сцена 2, строка 120)

Алгоритмы, описываемые в этой книге, имеют непосредственное отношение к числам. Я считаю, что их справедливо называют *получисленными*, так как они находятся на границе между численными и символьными методами. Каждый алгоритм должен не только находить числовое решение проблемы, но и хорошо сочетаться с внутренними операциями цифрового компьютера. В большинстве случаев человек не может оценить всю красоту подобных алгоритмов, если только он не владеет машинным языком компьютера. Эффективность соответствующей компьютерной программы — это жизненно важный фактор, который нельзя отделить от самого алгоритма. Проблема заключается в том, чтобы найти оптимальные способы работы компьютеров с числами, а это включает вопросы тактики и численного анализа. Поэтому предмет данной книги, без сомнения, относится к компьютерной науке так же, как к разделам математики, занимающимся численным анализом.

Некоторые математики, работающие в "высоких сферах" численного анализа, будут считать, что предлагаемые в этом томе темы относятся к сфере влияния системных программистов. А специалисты, работающие в "высоких сферах" системного программирования, наоборот, решат, что изучать рассматриваемые темы — дело численных аналитиков. Но я все-таки надеюсь, что найдутся читатели, которые захотят внимательно изучить эти фундаментальные методы. Хотя данные методы можно отнести, скорее всего, к нижнему уровню, на них основаны все грандиозные компьютерные приложения, предназначенные для решения числовых задач. Отсюда следует, насколько важно хорошо в них разобраться. В настоящем томе будет рассмотрена область, которая находится на стыке численного анализа и программирования; именно это и делает предмет книги весьма интересным.

В данном томе по сравнению с другими содержится значительно больший объем математического материала; это обусловлено спецификой изучаемых тем. Причем в большинстве случаев нужные математические темы раскрываются прямо на страницах книги практически с нуля (или на основании результатов, доказанных в томе 1). Но в некоторых разделах предполагается, что читатель знаком с используемым материалом.

В этом томе содержатся главы 3 и 4. Глава 3 посвящена случайным числам: здесь изучаются не только различные методы генерирования случайных чисел, но

и статистические критерии случайности, а также преобразование равномерно распределенных случайных чисел в другие типы случайных величин. Последняя тема позволяет проиллюстрировать практическое применение случайных чисел. В эту главу также включен раздел, повествующий о природе самой случайности. Глава 4 — это захватывающая история о том, какие открытия были сделаны в арифметике в результате многовекового прогресса. В ней обсуждаются различные системы представления чисел и способы преобразования одной системы в другую; рассматривается арифметика чисел с плавающей точкой, целых чисел высокой точности, рациональных дробей, полиномов и степенных рядов, а также вопросы разложения на множители и нахождения наибольших общих делителей.

Каждая из глав 3 и 4 может использоваться в качестве основы для семестрового университетского курса, причем изложение материала можно построить так, чтобы его можно было излагать на разных уровнях: как для первокурсников, так и для выпускников. В настоящее время курсы "Случайные числа" и "Арифметика" не входят в программы многих университетов. Но я надеюсь, читатель увидит, что в этих главах в едином ключе освещается материал, имеющий реальную образовательную ценность. Мой собственный опыт подтверждает, что он служит прекрасным способом ознакомления студентов с элементарной теорией вероятности и теорией чисел. Почти все темы, которые обычно рассматривают в таких вводных курсах, естественно возникают в связи с вопросами практического применения теории. Кроме того, обсуждение на лекциях вопросов практического использования результатов может стать той движущей силой, которая вызовет у студентов интерес к учебе и поможет понять важность и значение теории. Более того, в каждой главе содержатся упоминания о более сложных темах, которые у многих студентов вызовут интерес к дальнейшему изучению математики.

Этот том, в основном, представляет собой полную и самостоятельную книгу; исключение составляют только вопросы, касающиеся компьютера MIX, который описывался в томе 1. В приложении Б приведены использованные в данной книге математические обозначения, которые иногда отличаются от принятых в традиционной математической литературе.

Предисловие к третьему изданию

Когда в 1980 году второе издание этой книги было закончено, в ней впервые были использованы системы компьютерного набора TEX и METAFONT. А теперь я рад отметить завершение разработки этих систем возвратом к книге, которая вдохновила меня на их создание. Наконец-то мне удалось внести все тома в персональный компьютер и таким образом получить ее электронную версию, что позволит в дальнейшем вносить любые изменения в технологию печати и отображения на экране. Такой способ работы предоставил мне возможность сделать буквально тысячи улучшений, и я добился того, о чем так долго мечтал.

В этом новом издании я смог проверить каждое слово в тексте, стараясь сохранить юношеский задор оригинальных предложений и в то же время внести бóльшую зрелость суждений. Были добавлены десятки новых упражнений, а на десятки старых даны новые или улучшенные ответы. Изменения коснулись всего текста,

но особенно это относится к разделам 3.5 (теоретические основы случайности), 3.6 (универсальные генераторы случайных чисел), 4.5.2 (двоичный алгоритм нахождения наибольшего общего делителя) и 4.7 (композиция и итерация степенных рядов).

Таким образом, работа над книгой *Искусство программирования* продолжается. Исследования получисленных алгоритмов продвигаются с феноменальной скоростью. Именно поэтому некоторые части данной книги начинаются пиктограммой "В процессе построения" (это своеобразное извинение за то, что приведены не самые новые данные). Мои файлы переполнены важными материалами, которые я планирую включить в окончательное, знаменательное четвертое издание тома 2 (оно выйдет, вероятно, через 16 лет). Но сначала я должен закончить тома 4 и 5. Я хочу, чтобы они были опубликованы сразу же, как только будут готовы к печати.

Я чрезвычайно благодарен сотням людей, которые помогали мне собирать материал в течение последних 35 лет. Большая часть тяжелой работы по подготовке этого нового издания была выполнена Сильвио Леви (Silvio Levy), который профессионально отредактировал электронную версию текста, а также Джеффри Олдхэмом (Jefferey Oldham), который конвертировал почти все оригинальные иллюстрации в формат METAPOST. Я исправил все ошибки, которые бдительные читатели обнаружили во втором издании (а также ошибки, которых, увы, не заметил никто), и постарался избежать появления новых ошибок. Тем не менее я допускаю, что некоторые огрехи все же остались, и хотел бы их исправить как можно скорее. Поэтому за каждую опечатку*, а также ошибку, относящуюся к сути излагаемого материала или к приведенным историческим сведениям, я охотно заплачу $2,56 тому, кто первым ее найдет. На Web-странице, адрес которой приведен на обложке книги, содержится текущий список всех ошибок, о которых мне сообщили.

Стенфорд, Калифорния D. E. K.
Июль 1997

Когда работа над книгой продолжается в течение восьми лет, то появляется очень много людей — коллег, наборщиков, студентов, преподавателей и друзей, которых нужно поблагодарить. Но я не собираюсь освобождать их от ответственности за ошибки, которые остались в тексте. Они должны были их исправить! Иногда они даже несут ответственность за идеи, которые в конце концов оказываются ошибочными. Но в любом случае я благодарен всем своим сотрудникам.

— ЭДВАРД Ф. КЭМПБЕЛЛ (мл.) (EDWARD F. CAMPBELL, JR.) (1975)

Defendit numerus [В числах ты найдешь покой] — *это истина дураков;*
Deperdit numerus [В числах ты найдешь погибель] — *истина мудрых.*

— Ч. К. КОЛТОН (C. C. COLTON) (1820)

* Имеется в виду оригинал настоящего издания. — *Прим. ред.*

ПРИМЕЧАНИЯ К УПРАЖНЕНИЯМ

Упражнения, приведенные в этой серии книг, предназначены как для самостоятельной проработки, так и для семинарских занятий. Очень трудно и, наверное, просто невозможно выучить предмет, только читая теорию и не применяя ее для решения конкретных задач, которые заставляют задуматься о прочитанном. Более того, мы лучше всего заучиваем то, до чего дошли самостоятельно, своим умом. Поэтому упражнения занимают важное место в данном издании. Я приложил немало усилий, чтобы сделать их как можно более информативными, а также отобрать задачи, которые были бы не только поучительны, но и позволяли читателю получить удовольствие от их решения.

Во многих книгах простые упражнения даются вместе с исключительно сложными. Это не всегда удобно, так как читателю хочется знать заранее, сколько времени ему придется затратить на решение задач (иначе в лучшем случае он их только просмотрит). В качестве классического примера подобной ситуации можно привести книгу Ричарда Беллмана (Richard Bellman) *Динамическое программирование* (М.: Изд-во иностр. лит., 1960). Это очень важная, новаторская работа, но у нее есть один недостаток: в конце некоторых глав в разделе "Упражнения и научные проблемы" среди серьезных, еще нерешенных проблем приводятся простейшие вопросы. Говорят, что кто-то однажды спросил д-ра Беллмана, как отличить упражнения от научных проблем, и он ответил: "Если вы можете решить задачу, значит, это упражнение; в противном случае это научная проблема".

Совершенно очевидно, что в книге, подобной этой, должны быть приведены и сложные научные проблемы, и простейшие упражнения. Поэтому, чтобы читатель не ломал голову, пытаясь отличить одно от другого, были введены рейтинги, которые определяют степень сложности каждого упражнения. Эти рейтинги имеют следующее значение.

Рейтинг *Объяснение*

00 Чрезвычайно простое упражнение, на которое можно ответить сразу же, если прочитанный материал понят. Упражнения подобного типа почти всегда можно решить "в уме".

10 Простая задача, которая заставляет задуматься над прочитанным, но не представляет особых трудностей. На ее решение вы затратите не больше минуты; в процессе решения могут понадобиться карандаш и бумага.

20 Средняя задача, которая позволяет проверить, понял ли читатель основные положения изложенного материала. Чтобы получить исчерпывающий ответ, может понадобиться примерно 15–20 минут.

30 Задача умеренной сложности. Для ее решения может понадобиться более двух часов (а если одновременно вы смотрите телевизор, то еще больше).

40 Достаточно сложная или трудоемкая задача, которую вполне можно включить в план семинарских занятий. Предполагается, что студент должен справиться с ней, затратив не слишком много времени, и решение будет нетривиальным.

50 Научная проблема, которая (насколько известно автору в момент написания книги) пока еще не получила удовлетворительного решения, хотя найти его пытались очень многие. Если вы нашли решение подобной проблемы, то опубликуйте его; более того, автор данной книги будет очень признателен, если ему сообщат решение как можно скорее (при условии, что оно правильно).

Интерполируя по этой "логарифмической" шкале, можно понять, что означает любой промежуточный рейтинг. Например, рейтинг *17* говорит о том, что упражнение немного проще, чем задача средней сложности. Если задача с рейтингом *50* будет впоследствии решена каким-либо читателем, то в следующих изданиях данной книги и в списке ошибок, опубликованных в Internet, она может иметь рейтинг *45* (адрес Web-страницы приводится на обложке книги).

Остаток от деления рейтинга на 5 показывает, какой объем рутинной работы потребуется для решения данной задачи. Таким образом, для выполнения упражнения с рейтингом *24* может потребоваться больше времени, чем для упражнения с рейтингом *25*, но для последнего необходим более творческий подход.

Автор очень старался правильно присвоить рейтинги упражнениям, но тому, кто составляет задачи, трудно предвидеть, насколько сложными они окажутся для кого-то другого. К тому же одному человеку некая задача может показаться простой, а другому — сложной. Таким образом, определение рейтингов — дело достаточно субъективное и относительное. Я надеюсь, что рейтинги помогут вам получить правильное представление о степени трудности задач, но их следует воспринимать в качестве ориентира, а не в качестве абсолюта.

Эта книга написана для читателей с различным уровнем математической подготовки и научного кругозора, поэтому некоторые упражнения рассчитаны исключительно на тех, кто серьезно интересуется математикой или занимается ею профессионально. Если рейтингу предшествует буква M, значит, математические понятия и обоснования используются в упражнении в большей степени, чем это необходимо тому, кто интересуется в основном программированием алгоритмов. Если же упражнение отмечено буквами *HM*, то для его решения необходимо знание высшей математики в большем объеме, чем дается в настоящей книге. Но пометка *HM* совсем необязательно означает, что упражнение трудное.

Перед некоторыми упражнениями стоит стрелка "▶", которая означает, что они особенно поучительны и их очень рекомендуется выполнить. Само собой разумеется, никто не ожидает, что читатель (или студент) будет решать *все* задачи, поэтому наиболее важные из них и были выделены. Но это ни в коем случае не означает, что другие упражнения выполнять не стоит! Каждый читатель должен хотя бы попытаться решить все задачи, рейтинг которых меньше или равен *10*. Стрелки помогут выбрать задачи с более высокими рейтингами, которые следует решать в первую очередь.

К большинству упражнений приведены ответы, помещенные в отдельном разделе в конце книги. Пожалуйста, пользуйтесь ими разумно: ответ смотрите только

после того, как приложите все усилия, чтобы решить задачу самостоятельно, либо если у вас совершенно нет времени на ее решение. Ответ будет поучителен и полезен для вас только в том случае, если вы ознакомитесь с ним *после* того, как найдете свое решение или изрядно потрудитесь над задачей. Ответы к задачам излагаются очень кратко и схематично, так как предполагается, что читатель честно пытался решить задачу собственными силами. Иногда в приведенном решении дается меньше информации, чем спрашивалось, но чаще бывает наоборот. Вполне возможно, что полученный вами ответ окажется лучше того, который помещен в книге, или вы найдете ошибку в ответе. В таком случае автор был бы очень признателен, если бы вы как можно скорее подробно сообщили ему об этом; тогда в последующих изданиях книги будет опубликовано более удачное решение, а также имя его автора.

Решая задачи, вы, как правило, можете пользоваться ответами к предыдущим упражнениям, за исключением случаев, когда это будет оговорено особо. Рейтинги упражнениям присваивались в расчете именно на это, и вполне возможно, что рейтинг упражнения $n + 1$ ниже рейтинга упражнения n, даже если результат упражнения n является его частным случаем.

Условные обозначения			
		00	Простейшее (ответ дать немедленно)
		10	Простое (на одну минуту)
▸	Рекомендуется	*20*	Средней трудности (на четверть часа)
M	С математическим уклоном	*30*	Повышенной трудности
HM	Требует знания высшей математики	*40*	Высокой трудности
		50	Научная проблема

УПРАЖНЕНИЯ

▸ **1.** [*00*] Что означает рейтинг *M20*?

2. [*10*] Какое значение для читателя имеют упражнения, которые приводятся в учебниках?

3. [*34*] Леонард Эйлер (Leonhard Euler) в 1772 году предположил, что уравнение $w^4+x^4+y^4 = z^4$ не имеет решения в целых положительных числах, но Ноам Элкис (Noam Elkies) доказал в 1987 году, что существует бесконечное множество решений [см. *Math. Comp.* **51** (1988), 825–835]. Найдите все цёлочисленные решения, такие, что $0 \le w \le x \le y < z < 10^6$.

4. [*M50*] Докажите, что если n — целое число, $n > 4$, то уравнение $w^n + x^n + y^n = z^n$ неразрешимо в целых положительных числах w, x, y, z.

Упражнения — лучший инструмент познания.

— РОБЕРТ РЕКОРД (ROBERT RECORDE),
The Whetstone of Witte (1557)

СЛУЧАЙНЫЕ ЧИСЛА

Каждый, кто использует арифметические методы генерирования случайных чисел, безусловно, грешит.

— ДЖОН ФОН НЕЙМАН (JOHN VON NEUMANN) (1951)

О вероятности коль кто забудет, обманщиком вовек не будет.

— ДЖОН ГЕЙ (JOHN GAY) (1727)

Достаточно лишь нескольких лучей света, чтобы помочь людям в совершенствовании их "стохастических" способностей.

— ДЖОН ОУЭН (JOHN OWEN) (1662)

3.1. ВВЕДЕНИЕ

Числа, которые выбираются случайным образом, находят множество полезных применений.

a) *Моделирование.* При использовании компьютера для моделирования естественных явлений случайные числа нужны для того, чтобы сделать эти модели похожими на реальные явления. Моделирование применяется во многих областях, начиная от исследований в ядерной физике (где частицы испытывают случайные столкновения) и заканчивая исследованием операций (где люди прибывают, например, в аэропорт через случайные промежутки времени).

b) *Выборочный метод.* Часто бывает невозможно исследовать все варианты, но случайная выборка обеспечивает понимание того, что можно назвать "типичным" поведением.

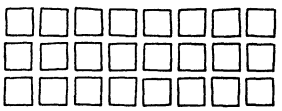
c) *Численный анализ.* Для решения сложных задач численного анализа была разработана остроумная техника, использующая случайные числа. Об этом написано несколько книг.

d) *Компьютерное программирование.* Случайные величины являются хорошим источником данных для тестирования эффективности компьютерных алгоритмов.

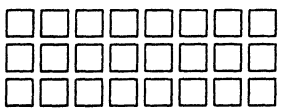
Более важно то, что они играют решающую роль при использовании *рандомизированных алгоритмов*, которые часто намного превосходят своих детерминированных двойников. В этой серии книг нас, в первую очередь, интересует именно такое применение случайных чисел. Этим объясняется то, что случайные числа рассматриваются уже здесь, в главе 3, прежде чем появится большинство других компьютерных алгоритмов.

e) *Принятие решений.* Говорят, что многие администраторы принимают решения, бросая монету, игральную кость либо каким-нибудь другим подобным способом. Сплетничают, что некоторые профессора в колледжах ставят оценки, используя тот же метод. Иногда важно принять полностью "беспристрастное" решение. Случайность является также важной частью оптимальных стратегий в теории матричных игр.

f) *Эстетика.* Небольшая добавка случайности оживляет музыку и компьютерную графику. Например, рисунок



в определенном смысле выглядит привлекательнее, чем



[См. D. E. Knuth, *Bull. Amer. Math. Soc.* **1** (1979), 369.]

g) *Развлечения.* Многие считают, что они замечательно проводят время, бросая игральные кости, тасуя колоду карт, вращая колесо рулетки и т. п. Такие традиционные способы использования случайных чисел получили название *метод Монте-Карло*. Это общее название всех алгоритмов, использующих случайные числа.

Те же, кто интересуются этой темой, постоянно вовлекаются в философские дискуссии о значении слова "случайный". Возникает вопрос "А что является не случайным числом?". Например, будет ли число 2 случайным? Охотнее говорят о *последовательности независимых случайных чисел* с заданным *распределением*, и это означает, если говорить не строго, что каждое число было получено случайно, не имея ничего общего с другими числами в последовательности, и что каждое число имеет заданную вероятность появления в любой заданной области значений.

Равномерным распределением на конечном множестве чисел (в дальнейшем — просто *равномерным распределением*) называется такое распределение, при котором любое из возможных чисел имеет одинаковую вероятность появления. Если не задано определенное распределение на конечном множестве чисел, то принято считать его равномерным.

Каждая из десяти цифр от 0 до 9 будет появляться примерно один раз из 10 в равномерной последовательности случайных цифр. Каждой паре двух последовательных цифр следует появиться один раз из ста и т. д. Однако если взять конкретную случайную последовательность длиной в миллион цифр, то она не всегда будет содержать 100 000 нулей, 100 000 единиц и т. д. Действительно, возможность появления такой последовательности незначительна; на самом деле, если рассматривать достаточно большую совокупность таких *последовательностей*, то в среднем будет появляться 100 000 нулей, 100 000 единиц и т. д.

Любая конкретная последовательность, содержащая миллион цифр, так же вероятна, как и любая другая. Если мы выберем миллион цифр наудачу и если ока-

жется, что первые 999 999 из них — нули, то вероятность того, что последняя цифра в этой последовательности — также нуль, все еще останется точно равной одной десятой в истинно случайной ситуации. Это утверждение большинству кажется парадоксальным, однако оно не противоречит реальности.

Существует несколько приличных возможностей дать абстрактное определение случайности, и мы вернемся к этой интересной теме в разделе 3.5; пока что достаточно интуитивного понимания данной концепции.

В течение многих лет те, кому случайные числа были необходимы для научной работы, вынуждены были таскать шары из урны, предварительно хорошо перемешав их, либо бросать игральные кости, либо раскладывать карты. Таблица, содержащая более 40 000 взятых наудачу из отчетов о переписи случайных цифр, была опубликована в 1927 году Л. Х. К. Типпеттом (L. H. C. Tippett).

С тех пор были построены механические генераторы случайных чисел. Первая такая машина была использована в 1939 году М. Ж. Кендаллом (M. G. Kendall) и Б. Бабингтон-Смитом (B. Babington-Smith) для построения таблицы, содержащей 100 000 случайных цифр. Компьютер Ferranti Mark I, впервые запущенный в 1951 году, имел встроенную программу, использующую резисторный генератор шума, которая поставляла 20 случайных битов на сумматор. Этот метод был предложен А. М. Тьюрингом (A. M. Turing). В 1955 году RAND Corporation опубликовала широко используемые таблицы, в которых содержался миллион случайных цифр, полученных с помощью других специальных устройств. Известный генератор случайных чисел ERNIE применялся на протяжении многих лет для определения выигрышных номеров британской лотереи. [См. статьи Kendall and Babington-Smith *J. Royal Stat. Soc.* **A101** (1938), 147–166; **B6** (1939), 51–61, а также дискуссию S. H. Lavington's с Mark I в *CACM* **21** (1978), 4–12; обозрение в *Math. Comp.* **10** (1956), 39–43; дискуссию об ERNIE W. E. Thomson, *J. Royal Stat. Soc.* **A122** (1959), 301–333.]

Короче говоря, после изобретения компьютеров начались исследования эффективного способа получения случайных чисел, встроенных программно в компьютеры. Можно было применять таблицы, но пользы от этого метода было мало из-за ограниченной памяти компьютера и требуемого времени ввода, поэтому таблицы могли быть лишь слишком короткими; кроме того, было не особенно приятно составлять таблицы и пользоваться ими. Генератор ERNIE мог быть встроен в компьютер, как в Ferranti Mark I, но это оказалось неудобно, поскольку невозможно точно воспроизвести вычисления даже сразу по окончании работы программы; более того, такие генераторы часто давали сбои, что было крайне трудно обнаружить. Технологический прогресс позволил вернуться к использованию таблиц в 90-е годы, так как миллиарды тестированных случайных байтов можно было разместить на компакт-дисках. Дж. Марсалья (George Marsaglia) помог оживить табличный метод в 1995 году, подготовив демонстрационный диск с 650 Мбайт случайных чисел, при генерировании которых запись шума диодной цепи сочеталась с определенным образом скомпонованной музыкой в стиле "рэп". (Он назвал это "белым и черным шумом".)

Несовершенство первых механических методов вначале пробудило интерес к получению случайных чисел с помощью обычных арифметических операций, заложенных в компьютер. Джон фон Нейман (John von Neumann) первым предложил

такой подход около 1946 года; его идея заключалась в том, чтобы возвести в квадрат предыдущее случайное число и выделить средние цифры. Например, мы хотим получить 10-значное число и предыдущее число равнялось 5772156649. Возводим его в квадрат и получаем

$$33317792380594909201;$$

значит, следующим числом будет 7923805949.

Совершенно очевидны претензии, предъявляемые к этому методу: как может быть случайной последовательность, генерируемая таким образом, если каждое число полностью определяется предыдущим? (См. эпиграф фон Неймана к этой главе.) Ответ состоит в том, что эта последовательность *не* случайна, но *кажется* такой. В типичных приложениях фактически существующая связь между двумя числами, следующими одно за другим, на самом деле не имеет значения; поэтому нельзя утверждать, что неслучайный характер последовательности нежелателен. Интуитивно ясно, что метод средин квадратов должен быть достаточно хорошим перемешиванием и заменой цифр предыдущего числа.

В "заумной" технической литературе последовательности, генерируемые детерминистическим путем, таким как этот, называются *псевдослучайными* или *квазислучайными*, однако в данной книге мы, в основном, просто будем называть их случайными последовательностями, понимая, что они только *кажутся* случайными. "Кажущаяся случайность" — это, возможно, все, что так или иначе может быть сказано о любой случайной последовательности. Случайные числа, генерируемые детерминистическими методами на компьютере, почти всегда работали достаточно хорошо при условии, что метод был выбран удачно. Конечно, детерминистическая последовательность не всегда применима; ею, безусловно, не следует заменять ERNIE в лотерее.

Метод средин квадратов фон Неймана, как было показано, фактически является сравнительно бедным источником случайных чисел. Опасность состоит в том, что последовательность стремится войти в привычную колею, т. е. короткий цикл повторяющихся элементов. Например, каждое появление нуля как числа последовательности приведет к тому, что все последующие числа также будут нулями.

Некоторые ученые экспериментировали с методом средин квадратов в начале 50-х годов. Работая с четырехзначными числами вместо десятизначных, Дж. Э. Форсайт (G. E. Forsythe) испытал 16 различных начальных значений и обнаружил, что 12 из них приводят к циклическим последовательностям, заканчивающимся циклом 6100, 2100, 4100, 8100, 6100, ..., в то время как две из них приводят к последовательностям, вырождающимся в 0. Более интенсивные исследования, главным образом в двоичной системе счисления, провел Н. К. Метрополис (N. C. Metropolis). Он показал, что если использовать 20-разрядное число, то последовательность случайных чисел, полученная методом средин квадратов, вырождается в 13 различных циклов, причем длина самого большого периода равна 142.

Достаточно легко запустить метод средин квадратов с новым исходным значением, если обнаружить число "нуль", однако избавиться от длинных циклов довольно трудно. В упр. 6 и 7 обсуждается несколько интересных вариантов определения циклов периодических последовательностей, использующих достаточно малый объем памяти.

Теоретические недостатки метода средин квадратов приведены в упр. 9 и 10. С другой стороны, используя 38-разрядные числа, Метрополис получил невырожденную последовательность, содержащую около 750 000 чисел (прежде чем произошло вырождение), и полученные $750\,000 \times 38$ бит удовлетворительно прошли статистический тест на случайность. [*Symp. on Monte Carlo Methods* (Wiley, 1956), 29–36.] Эти опыты показали, что метод средин квадратов *может* давать удовлетворительные результаты, но ему опасно доверять, пока не выполнены тщательные вычисления.

Когда автор работал над первым изданием этой книги, многие генераторы случайных чисел (в литературе на русском языке параллельно употребляется термин "датчик случайных чисел". — *Прим. ред.*), которые тогда использовались, были недостаточно хороши. Программисты традиционно не интересовались информацией о таких подпрограммах; старые методы, сравнительно неудовлетворительные, слепо переходили от одного программиста к другому, поскольку пользователи не понимали ограничений, при которых можно применять эти методы. Мы увидим здесь, что наиболее важные сведения о генераторах случайных чисел нетрудно изучить, несмотря на то что необходимо быть осторожным, чтобы избежать обычных ловушек.

Так нелегко придумать понятный всем и каждому датчик случайных чисел! В этом автор убедился в 1959 году, когда он попытался создать фантастически хороший генератор случайных чисел, используя следующий необычный подход.

Алгоритм К (*Супергенератор случайных чисел*). Пусть задано 10-значное десятичное число X и этот алгоритм использует замену X другим числом так, чтобы получить случайную последовательность. Несмотря на то что от алгоритма можно ожидать на выходе всецело случайную последовательность, соображения, приведенные ниже, показывают, что это, к сожалению, не всегда так. (Читатель не обязан изучать данный алгоритм во всех деталях, но рекомендуется обратить внимание на его сложность; отметим, в частности, шаги К1 и К2.)

К1. [Выбрать число итераций.] Присвоить $Y \leftarrow \lfloor X/10^9 \rfloor$ наибольшую значащую цифру X. (Мы выполним шаги К2–К13 точно $Y + 1$ раз, т. е. применим рандомизированные преобразования *случайное* число раз.)

К2. [Выбрать случайный шаг.] Присвоить $Z \leftarrow \lfloor X/10^8 \rfloor \bmod 10$ следующую наибольшую значащую цифру X. Переходим к шагу К$(3 + Z)$, т. е. к *случайно* выбранному шагу в программе.)

К3. [Обеспечить $\geq 5 \times 10^9$.] Если $X < 5000000000$, присвоить $X \leftarrow X + 5000000000$.

К4. [Средина квадрата.] Заменить X числом $\lfloor X^2/10^5 \rfloor \bmod 10^{10}$, т. е. серединой квадрата X.

К5. [Умножить.] Заменить X числом $(1001001001\,X) \bmod 10^{10}$.

К6. [Псевдодополнение.] Если $X < 100000000$, то присвоить $X \leftarrow X + 9814055677$; иначе присвоить $X \leftarrow 10^{10} - X$.

К7. [Переставить половины.] Поменять местами пять младших по порядку знаков X со старшими по порядку пятью знаками, т. е. присвоить $X \leftarrow 10^5(X \bmod 10^5) + \lfloor X/10^5 \rfloor$; это то же самое, что взять десять средних цифр числа $(10^{10} + 1)X$.

К8. [Умножить.] Выполнить шаг К5.

Таблица 1

КОЛОССАЛЬНОЕ СОВПАДЕНИЕ: АЛГОРИТМ К ПРЕОБРАЗОВАЛ ЧИСЛО 6065038420 САМО В СЕБЯ

Шаг	X (после)	
К1	6065038420	
К3	6065038420	
К4	6910360760	
К5	8031120760	
К6	1968879240	
К7	7924019688	
К8	9631707688	
К9	8520606577	
К10	8520506578	
К11	8520506578	
К12	0323372207	$Y = 6$
К6	9676627793	
К7	2779396766	
К8	4942162766	
К9	3831051655	
К10	3830951656	
К11	3830951656	
К12	1905867781	$Y = 5$
К12	3319967479	$Y = 4$
К6	6680032521	
К7	3252166800	
К8	2218966800	
К9	1107855700	
К10	1107755701	
К11	1107755701	
К12	1226919902	$Y = 3$
К5	0048821902	
К6	9862877579	
К7	7757998628	
К8	2384626628	
К9	1273515517	
К10	1273415518	
К11	1273415518	
К12	5870802097	$Y = 2$
К11	5870802097	
К12	3172562687	$Y = 1$
К4	1540029446	
К5	7015475446	
К6	2984524554	
К7	2455429845	
К8	2730274845	
К9	1620163734	
К10	1620063735	
К11	1620063735	
К12	6065038420	$Y = 0$

К9. [Уменьшить цифры.] Уменьшить каждую не равную нулю цифру десятичного представления числа X на единицу.

К10. [Модифицировать на 99999.] Если $X < 10^5$, присвоить $X \leftarrow X^2 + 99999$; иначе присвоить $X \leftarrow X - 99999$.

К11. [Нормировать.] (На этом шаге X не может быть равным нулю.) Если $X < 10^9$, присвоить $X \leftarrow 10X$ и повторить этот шаг.

К12. [Модификация метода средин квадратов.] Заменить X на $\lfloor X(X-1)/10^5 \rfloor \bmod 10^{10}$, т. е. средними 10 цифрами числа $X(X-1)$.

К13. [Повторить?] Если $Y > 0$, уменьшить Y на 1 и возвратиться к шагу К2. Если $Y = 0$, алгоритм завершен. Значение числа X, полученное на предыдущем шаге, и будет желаемым "случайным" значением. ▮

(Программа, реализующая этот алгоритм, обещала быть настолько сложной, что тот, кто прочел бы ее распечатку, без толковых комментариев не смог бы в ней разобраться.)

После рассмотрения всех преобразований алгоритма К не кажется ли правдоподобным, что можно было бы обеспечить бесконечное снабжение невероятно случайными числами? Нет! На самом деле, когда этот алгоритм впервые был реализован на компьютере, он почти немедленно сошелся к 10-значному числу 6065038420,

которое по невероятному совпадению преобразовалось само в себя по алгоритму (табл. 1). С другим стартовым числом последовательность начала повторяться после 7401 значения с периодом длиной 3178.

Мораль этой истории в том, что *случайные числа не следует генерировать методом, выбранным наудачу*. Не мешало бы использовать немного теории.

В следующих разделах будут рассмотрены генераторы случайных чисел более высокого уровня, чем метод средин квадратов и алгоритм К. Соответствующие последовательности гарантированно обладают желаемыми случайными свойствами и не вырождаются. Мы исследуем некоторые причины такого, похожего на случайное, поведения, а также покажем, как можно обращаться со случайными числами. Например, одно из наших исследований будет посвящено программе, которая тасует смоделированную на компьютере колоду карт.

В разделе 3.6 приводятся итоги к этой главе и содержатся некоторые библиографические источники.

УПРАЖНЕНИЯ

▶ **1.** [*20*] Предположим, вы хотите получить случайную десятичную цифру. Какой из следующих методов вы выберете?

a) Откроете телефонный справочник, ткнув пальцем куда-нибудь, выберете первый попавшийся номер телефона и возьмете младшую цифру (*цифру младшего разряда*) этого номера.

b) Поступите, как в (a), но выберете младшую цифру номера *страницы*.

c) Покатите игральную кость в форме икосаэдра, имеющую двадцать граней, которые помечены цифрами $0, 0, 1, 1, \ldots, 9, 9$. Когда кость остановится, выберете верхнюю цифру. (Для катания игральной кости рекомендуется использовать стол с хорошо натянутым сукном.)

d) На одну минуту выставите счетчик Гейгера у источника радиоактивного излучения (предварительно обезопасив себя) и воспользуетесь младшей цифрой показаний счетчика. Предполагается, что на счетчике Гейгера представлены числа в десятичной системе счисления и вначале на нем был установлен нуль.

e) Бросите быстрый взгляд на свои часы и, если секундная стрелка находится между $6n$ и $6(n+1)$ секундами, выберете цифру n.

f) Попросите приятеля задумать любую цифру и воспользуетесь ею.

g) Попросите врага задумать любую цифру и воспользуетесь ею.

h) Предположите, что 10 лошадей участвуют в забеге и вам о них абсолютно ничего не известно. Присвоите каждой лошади в произвольном порядке номер от 0 до 9, а после забега выберете в качестве случайной цифры номер победителя.

2. [*M22*] Какова вероятность того, что в случайной последовательности из миллиона десятичных цифр каждая возможная цифра встречается ровно 100 000 раз?

3. [*10*] Какое число следует за 1010101010 в методе средин квадратов?

4. [*20*] (a) Почему на шаге К11 алгоритма К значение X не может быть равно нулю? Какая ошибка возникла бы в алгоритме, если бы X мог принимать значение "нуль"? (b) Используя табл. 1, установите, что происходит, когда алгоритм К применяется повторно со стартовым значением $X = 3830951656$.

5. [*15*] Объясните, почему в любом случае, даже если совпадение, приведенное в табл. 1, не произошло, алгоритм К не сможет выдать бесконечную последовательность случайных чисел в том смысле, что любая последовательность, генерируемая алгоритмом К, станет в конце концов периодичной.

▸ **6.** [*M21*] Предположим, что необходимо получить последовательность целых случайных чисел X_0, X_1, X_2, ... на интервале $0 \le X_n < m$. Пусть $f(x)$ — любая функция, такая, что неравенство $0 \le x < m$ влечет $0 \le f(x) < m$. Рассмотрим последовательность $X_{n+1} = f(X_n)$. (Примеры таких последовательностей — метод средин квадратов и алгоритм К.)

a) Покажите, что такая последовательность периодична в том смысле, что существуют числа λ и μ, для которых значения

$$X_0,\ X_1,\ \ldots,\ X_\mu,\ \ldots, X_{\mu+\lambda-1}$$

различны, однако $X_{n+\lambda} = X_n$, когда $n \ge \mu$. Определите возможные максимальное и минимальное значения μ и λ.

b) (Р. Флойд (R. W. Floyd).) Покажите, что существует такое $n > 0$, что $X_n = X_{2n}$ и наименьшее такое значение n лежит в интервале $\mu \le n \le \mu + \lambda$. Более того, значение X_n является единственным в том смысле, что если $X_n = X_{2n}$ и $X_r = X_{2r}$, то $X_r = X_n$.

c) Используя идеи (b), составьте алгоритм вычисления μ и λ для любой заданной функции f и любого заданного X_0, используя только $O(\mu+\lambda)$ шагов и только ограниченный объем памяти.

▸ **7.** [*M21*] (Р. П. Брент (R. P. Brent), 1977.) Пусть $\ell(n)$ — наибольшее целое число, такое, что $\ell(n) \le n$, $\ell(n) = 2^k$, где k — целое число. Так, например, $\ell(15) = 8$ и $\ell(\ell(n)) = \ell(n)$.

a) Покажите, что в терминах обозначений упр. 6 существует такое $n > 0$, что $X_n = X_{\ell(n)-1}$. Найдите формулу для наименьшего такого n в терминах чисел μ и λ, определяющих период.

b) Примените этот результат для составления алгоритма, который может быть использован совместно с любым генератором случайных чисел типа $X_{n+1} = f(X_n)$, чтобы предотвратить циклическую неопределенность. Вашему алгоритму следует вычислять период длиной λ и использовать только небольшой объем памяти — вы просто не должны заполнять всю память вычисленными значениями последовательности!

8. [*23*] Выполните полную проверку метода средин квадратов для случая, когда десятичные числа состоят из двух цифр.

a) Можете начать процесс с любого из 100 допустимых чисел $00, 01, \ldots, 99$. Сколько из этих значений в конечном счете приведут к повторению цикла (зацикливанию) $00, 00, \ldots$? [*Пример.* Начиная с числа 43, получим последовательность 43, 84, 05, 02, 00, 00, 00, ...]

b) Сколько существует финальных циклов? Каков размер самого длинного цикла?

c) Какое начальное значение (или значения) даст наибольшее число различных элементов, прежде чем последовательность повторится?

9. [*M14*] Докажите, что метод средин квадратов, использующий $2n$-значные числа в b-ичной системе счисления, имеет следующие недостатки: если последовательность включает любое число, в котором n старших значащих цифр — нули, то последующие числа становятся все меньше и меньше, пока не превратятся в нули.

10. [*M16*] Пусть выполняются предположения предыдущего упражнения. Что можно сказать о последовательности чисел, следующих за X, если *младшие* значащие n цифр числа X равны нулю? Что если $n+1$ младших значащих цифр равны нулю?

▸ **11.** [*M26*] Рассмотрим последовательности генераторов случайных чисел, имеющих вид, описанный в упр. 6. Если выбрать $f(x)$ и X_0 наудачу (другими словами, если предположить, что каждая из m^m возможных функций $f(x)$ равновероятна и каждое из m возможных значений X_0 равновероятно), то какова вероятность того, что последовательность в конечном счете выродится в цикл длиной $\lambda = 1$? [*Замечание.* Предположения этой задачи дают повод задуматься о "случайности" генераторов случайных чисел такого типа. Можно

ожидать, что метод, подобный алгоритму K, отчасти ведет себя так же, как рассмотренный здесь генератор; ответ на эту задачу дает колоссальное число совпадений в табл. 1.]

▸ **12.** [*M31*] Какова средняя длина финального цикла, если выполняются предположения предыдущего упражнения? Какова средняя длина последовательности до вхождения в цикл? (В обозначениях упр. 6 необходимо определить средние значения λ и $\mu + \lambda$.)

13. [*M42*] Если $f(x)$ выбрана наудачу, как в упр. 11, какова средняя длина самого *длинного* цикла, полученного путем варьирования начального значения X_0? [*Замечание.* Мы уже рассмотрели аналогичную проблему для случая, когда $f(x)$ — это случайные перестановки; см. упр. 1.3.3–23.]

14. [*M38*] Если $f(x)$ выбрано наудачу, как в упр. 11, каково среднее число различных финальных циклов, полученных в результате варьирования начальных значений? [См. упр. 8, (b).]

15. [*M15*] Если $f(x)$ выбрано наудачу, как и в упр. 11, чему равна вероятность, что ни один из финальных циклов не имеет длину, равную 1, невзирая на выбор X_0?

16. [*15*] Последовательность, генерируемая, как в упр. 6, должна повторяться после того, как было сгенерировано не более m значений. Предположим, что мы обобщим метод таким образом, что X_{n+1} будет зависеть от X_{n-1} так же, как от X_n; формально пусть $f(x, y)$ — такая функция, для которой $0 \le x, y < m$ влечет неравенства $0 \le f(x, y) < m$. Последовательность строится так: сначала произвольно выбирают X_0 и X_1, а затем полагают, что

$$X_{n+1} = f(X_n, X_{n-1}), \qquad \text{где } n > 0.$$

Чему предположительно равен максимальный период в этом случае?

17. [*10*] Обобщите ситуацию из предыдущего упражнения так, чтобы X_{n+1} зависело от предыдущих k значений последовательности.

18. [*M20*] Придумайте метод, аналогичный методу из упр. 7, для определения цикла генератора случайных чисел, описанного в упр. 17, в общем виде.

19. [*M48*] Выполните упр. 11, используя упр. 15, в более общем случае, когда X_{n+1} зависят от k предыдущих значений последовательности; каждая из m^{m^k} функций $f(x_1, \ldots, x_k)$ считается равновероятной. [*Замечание.* Число функций, которые дают *максимальный* период, анализируется в упр. 2.3.4.2–23.]

20. [*30*] Найдите все неотрицательные числа $X < 10^{10}$, которые при использовании алгоритма K в конечном счете приводят к самовоспроизводящимся числам из табл. 1.

21. [*42*] Докажите или опровергните следующее утверждение: отображение $X \mapsto f(X)$, определенное алгоритмом K, имеет ровно пять циклов длиной 3178, 1606, 1024, 943 и 1.

22. [*21*] (Г. Роллетшек (H. Rolletschek).) Хороша ли идея генерирования случайных чисел с помощью последовательности $f(0)$, $f(1)$, $f(2)$, ..., где f — случайная функция, вместо того, чтобы использовать x_0, $f(x_0)$, $f(f(x_0))$ и т. д.?

▸ **23.** [*M26*] (Д. Фоата (D. Foata) и А. Фучс (A. Fuchs), 1970.) Покажите, что каждая из m^m функций $f(x)$, рассмотренных в упр. 6, может быть представлена как последовательность $(x_0, x_1, \ldots, x_{m-1})$, имеющая такие свойства.

i) $(x_0, x_1, \ldots, x_{m-1})$ — это перестановки последовательности $(f(0), f(1), \ldots, f(m-1))$.

ii) $(f(0), \ldots, f(m-1))$ может быть единственным образом восстановлена из последовательности $(x_0, x_1, \ldots, x_{m-1})$.

iii) Элементы, которые появляются в циклах из f, имеют вид $\{x_0, x_1, \ldots, x_{k-1}\}$, где k — самый большой индекс, такой, что эти k элементов различны.

iv) $x_j \notin \{x_0, x_1, \ldots, x_{j-1}\}$ влечет $x_{j-1} = f(x_j)$, если x_j не является наименьшим элементом в цикле из f.

v) $(f(0), f(1), \ldots, f(m-1))$ — это перестановка последовательности $(0, 1, \ldots, m-1)$ тогда и только тогда, когда $(x_0, x_1, \ldots, x_{m-1})$ представляет собой *обратную* перестановку к той перестановке, которая в разделе 1.3.3 названа необычным соответствием.

vi) $x_0 = x_1$ тогда и только тогда, когда $(x_1, \ldots, x_{m-1})$ представляет собой ориентированное дерево, построенное в упр. 2.3.4.4–18, с $f(x)$ порождающим x.

3.2. ГЕНЕРИРОВАНИЕ РАВНОМЕРНО РАСПРЕДЕЛЕННЫХ СЛУЧАЙНЫХ ЧИСЕЛ

В ЭТОМ РАЗДЕЛЕ будут рассмотрены методы генерирования последовательности случайных дробей, т. е. случайных *действительных чисел U_n, равномерно распределенных между нулем и единицей (на интервале* $[0, 1)$*)*. Так как компьютер может представлять действительные числа только с определенной точностью, мы будем генерировать целое число X_n между нулем и некоторым числом m: дробь

$$U_n = X_n/m$$

будет, следовательно, лежать между нулем и единицей. Обычно m выбирают равным размеру слова в компьютере. (В этой книге размером слова (*word size*) автор называет число b^e, где b — основание системы счисления, используемой в компьютере, а e — число разрядов машины. — *Прим. ред.*) Поэтому X_n можно по традиции рассматривать как целое число, занимающее все компьютерное слово, с точкой, которая отделяет целую часть числа от дробной, стоящей в правом конце слова, а U_n — если хотите, как содержание того же слова с разделяющей точкой, стоящей в левом конце слова.

3.2.1. Линейный конгруэнтный метод

В настоящее время наиболее популярными генераторами случайных чисел являются генераторы, в которых используется следующая схема, предложенная Д. Г. Лехмером (D. H. Lehmer) в 1949 году [см. Proc. 2nd Symp. on Large-Scale Digital Calculating Machinary (Cambridge, Mass.: Harvard University Press, 1951, 141–146)]. Выберем четыре "волшебных числа":

$$\begin{array}{rll} m, & \text{модуль;} & 0 < m; \\ a, & \text{множитель;} & 0 \le a < m; \\ c, & \text{приращение;} & 0 \le c < m; \\ X_0, & \text{начальное значение;} & 0 \le X_0 < m. \end{array} \tag{1}$$

Затем получим желаемую последовательность случайных чисел $\langle X_n \rangle$, полагая

$$X_{n+1} = (aX_n + c) \bmod m, \qquad n \ge 0. \tag{2}$$

Эта последовательность называется *линейной конгруэнтной последовательностью*. Получение остатков по модулю m отчасти напоминает предопределенность, когда шарик попадает в ячейку крутящегося колеса рулетки. Например, для $m = 10$ и $X_0 = a = c = 7$ получим последовательность

$$7,\ 6,\ 9,\ 0,\ 7,\ 6,\ 9,\ 0,\ \ldots\ . \tag{3}$$

Как показывает этот пример, такая последовательность не может быть "случайной" при некоторых наборах чисел m, a, c и X_0. Принципы выбора подходящих волшебных чисел будут подробно исследованы в следующих разделах этой главы.

В примере (3) иллюстрируется тот факт, что конгруэнтная последовательность всегда образует петли, т. е. обязательно существует цикл, повторяющийся бесконечное число раз. Это свойство является общим для всех последовательностей вида $X_{n+1} = f(X_n)$, где f преобразует конечное множество само в себя (см. упр. 3.1–6).

Повторяющиеся циклы называются *периодами*; длина периода последовательности (3) равна 4. Безусловно, последовательности, которые мы будем использовать, имеют относительно длинный период.

Заслуживает внимания случай, когда $c = 0$, так как генерируемые числа будут иметь меньший период, чем при $c \neq 0$. Мы убедимся в дальнейшем, что ограничение $c = 0$ уменьшает длину периода последовательности, хотя при этом все еще возможно сделать период достаточно длинным. В оригинальном методе, предложенном Д. Г. Лехмером, c выбиралось равным нулю, хотя он и допускал случай, когда $c \neq 0$, как один из возможных. Тот факт, что условие $c \neq 0$ может приводить к появлению более длинных периодов, был установлен В. Е. Томсоном (W. E. Thomson) [Comp. J. 1 p. 83, 86] и независимо от него А. Ротенбергом (A. Rotenberg) [JACM 7 (1960), 75–77]. Многие авторы называют линейную конгруэнтную последовательность при $c = 0$ *мультипликативным конгруэнтным методом*, а при $c \neq 0$ — *смешанным конгруэнтным методом*. Буквы m, a, c и X_0 будут использованы в этой главе в том смысле, в каком они вводились раньше. То же самое относится и к константе

$$b = a - 1, \tag{4}$$

которая вводится для упрощения многих наших формул.

Можно сразу отбросить случай, когда $a = 1$, при котором последовательность X_n представима в виде $X_n = (X_0 + nc) \bmod m$ и ведет себя явно не как случайная последовательность. Случай, когда $a = 0$, даже хуже предыдущего. Следовательно, для практических целей предполагаем, что

$$a \geq 2, \qquad b \geq 1. \tag{5}$$

Сейчас можно обобщить формулу (2)

$$X_{n+k} = \big(a^k X_n + (a^k - 1)c/b\big) \bmod m, \qquad k \geq 0, \quad n \geq 0, \tag{6}$$

где $(n + k)$-й член выражается непосредственно через n-й. (Случай, когда $n = 0$, в этом уравнении также достоин внимания.) Из (4) следует, что подпоследовательность, содержащая каждый k-й член последовательности $\langle X_n \rangle$, является также линейной конгруэнтной последовательностью, множитель которой равен $a^k \bmod m$ и приращение равно $\big((a^k - 1)c/b\big) \bmod m$. Важным следствием из (6) является то, что общая последовательность, определенная с помощью a, c и X_0, может быть очень просто выражена в терминах специального случая, когда $c = 1$ и $X_0 = 0$. Пусть

$$Y_0 = 0, \qquad Y_{n+1} = (aY_n + 1) \bmod m. \tag{7}$$

В соответствии с (6) получим $Y_k \equiv (a^k - 1)/b$ (по модулю m). Значит, последовательность, определенная в (2), будет имеет вид

$$X_n = (AY_n + X_0) \bmod m, \qquad \text{где } A = (X_0 b + c) \bmod m. \tag{8}$$

УПРАЖНЕНИЯ

1. [*10*] В примере (3) показана ситуация, когда $X_4 = X_0$, так что последовательность начинается сначала. Приведите пример линейной конгруэнтной последовательности при $m = 10$, для которой число X_0 никогда снова не появится.

▸ **2.** [*M20*] Покажите, что если a и m взаимно простые, то X_0 всегда появится в периоде.

3. [*M10*] Объясните, почему последовательность имеет определенные недостатки и, вероятно, не очень случайна, если a и m — не взаимно простые числа. Поэтому следует выбирать a и m так, чтобы они были взаимно простыми.

4. [*11*] Докажите формулу (6).

5. [*M20*] Соотношение (6) справедливо при $k \geq 0$. Если это возможно, получите формулы для X_{n+k} в терминах X_n для *отрицательных* значений k.

3.2.1.1. Выбор модуля. Первая задача, которую мы рассмотрим, — нахождение хороших значений параметров, определяющих линейную конгруэнтную последовательность. Сначала выясним, как правильно выбрать число m. Необходимо, чтобы m было довольно большим, так как период не может иметь больше m элементов. (Даже если мы намерены генерировать только случайные нули и единицы, *не* следует брать $m = 2$, ибо тогда последовательность в лучшем случае будет иметь вид $\ldots, 0, 1, 0, 1, 0, 1, \ldots$! Методы получения случайных нулей и единиц из линейной конгруэнтной последовательности обсуждаются в разделе 3.4.)

Другой фактор, который оказывает влияние на выбор m, — скорость генерирования: нужно подобрать значение m так, чтобы $(aX_n + c) \bmod m$ вычислялось быстро.

В качестве примера рассмотрим компьютер MIX. Можно вычислить $y \bmod m$, помещая y в регистры A и X и выполняя деление на m. Если y и m положительны, то $y \bmod m$ появится в регистре X. Но деление — сравнительно медленно протекающая операция, и этот недостаток можно компенсировать, если выбрать значение m таким, что особенно удобно, как *длина слова* нашего компьютера.

Пусть w будет длиной компьютерного слова, а именно — 2^e на e-разрядном двоичном компьютере или 10^e на e-цифровой десятичной вычислительной машине. (В настоящей книге мы часто будем употреблять букву e для обозначения произвольной целой степени. Несмотря на то что эта буква часто используется для обозначения основания натурального логарифма, мы надеемся, что читателю из контекста будет понятно, что она обозначает. Физики сталкиваются с подобными проблемами, когда используют e для обозначения заряда электрона.) Результат операции суммирования обычно дается по модулю w (но не на машинах, использующих процедуру единичного дополнения); умножение по модулю w также очень простое, поскольку затрагиваются только младшие разряды произведения. Таким образом, следующая программа эффективно вычисляет величину $(aX + c) \bmod w$.

```
LDA   A     rA ← a.
MUL   X     rAX ← (rA) · X.
SLAX  5     rA ← rAX mod w.                    (1)
ADD   C     rA ← (rA + c) mod w.   ∎
```

Результат появляется в регистре A. В конце программы возможно переполнение; если это нежелательно, то следует, допустим, команда "JOV *+1" — "выключить".

"Умная" техника, обычно менее известная, может использовать представление вычисления по модулю $w + 1$. По причинам, поясняемым ниже, как правило, требуется, чтобы $c = 0$, когда $m = w + 1$; тогда мы просто должны вычислить

$(aX) \bmod (w+1)$. Делает это следующая программа.

```
01  LDAN X         rA ← −X.
02  MUL  A         rAX ← (rA) · a.
03  STX  TEMP
04  SUB  TEMP      rA ← rA − rX.                    (2)
05  JANN *+3       Выход, если rA ≥ 0.
06  INCA 2         rA ← rA + 2.
07  ADD  =w−1=     rA ← rA + w − 1.   ▮
```

В регистре A сейчас содержится значение $(aX) \bmod (w+1)$. Конечно, оно может лежать где-нибудь между 0 и w включительно, так что читатель может законно удивиться, как можно представить так много значений в регистре A! (Обычно регистр не может хранить число, большее, чем $w-1$.) Ответом является то, что переполнение в программе (2) происходит тогда и только тогда, когда результат равен w (если предположить, что переполнение убрано в исходном положении). Можно отобразить w в виде нуля, так как программу (2) обычно нельзя использовать, когда $X = 0$; но более удобно просто отбросить значение w, если оно появляется в конгруэнтной последовательности по модулю $w+1$. Затем также можно избежать переполнения, просто заменив строки 05 и 06 в (2) строками "`JANN *+4; INCA 2; JAP *-5`".

Для доказательства того, что программа (2) действительно вычисляет $(aX) \bmod (w+1)$, заметим, что в строке 04 младшие разряды произведения вычитаются из старших разрядов. Переполнение не может произойти на этом шаге, и, если $aX = qw + r$ при $0 \le r < w$, получим значение $r - q$ в регистре A после строки 04. Сейчас

$$aX = q(w+1) + (r-q)$$

и мы имеем $-w < r - q < w$, так как $q < w$; следовательно, $(aX) \bmod (w+1)$ равно одному из двух значений ($r-q$ или $r-q+(w+1)$) в зависимости от того, $r - q \ge 0$ или $r - q < 0$.

Подобная техника может быть использована для получения произведения двух чисел по модулю $(w-1)$; см. упр. 8.

Для освоения следующих разделов требуется знать простые множители m, чтобы правильно выбрать a. В табл. 1 впервые дается полный список разложений на простые множители $w \pm 1$ почти для каждой известной длины компьютерного слова; при желании методы из раздела 4.5.4 можно использовать для расширения таблицы.

Читатель может поинтересоваться, почему здесь обсуждается использование $m = w \pm 1$, когда выбор $m = w$ так явно удобен. Причина в том, что, *когда $m = w$, цифры правой части X_n гораздо менее случайны, чем цифры левой части.* Если d является делителем m и если

$$Y_n = X_n \bmod d, \tag{3}$$

можно легко показать, что

$$Y_{n+1} = (aY_n + c) \bmod d. \tag{4}$$

(Пусть $X_{n+1} = aX_n + c - qm$, где q — некоторое целое число. Если обе части равенства взять по модулю d, можно потерять qm, когда d — множитель m.)

Для иллюстрации важности выражения (4) предположим, например, что имеется двоичный компьютер. Если $m = w = 2^e$, младшие четыре разряда X_n являются

Таблица 1

РАЗЛОЖЕНИЕ НА ПРОСТЫЕ МНОЖИТЕЛИ $w \pm 1$

$2^e - 1$	e	$2^e + 1$
$7 \cdot 31 \cdot 151$	15	$3^2 \cdot 11 \cdot 331$
$3 \cdot 5 \cdot 17 \cdot 257$	16	65537
131071	17	$3 \cdot 43691$
$3^3 \cdot 7 \cdot 19 \cdot 73$	18	$5 \cdot 13 \cdot 37 \cdot 109$
524287	19	$3 \cdot 174763$
$3 \cdot 5^2 \cdot 11 \cdot 31 \cdot 41$	20	$17 \cdot 61681$
$7^2 \cdot 127 \cdot 337$	21	$3^2 \cdot 43 \cdot 5419$
$3 \cdot 23 \cdot 89 \cdot 683$	22	$5 \cdot 397 \cdot 2113$
$47 \cdot 178481$	23	$3 \cdot 2796203$
$3^2 \cdot 5 \cdot 7 \cdot 13 \cdot 17 \cdot 241$	24	$97 \cdot 257 \cdot 673$
$31 \cdot 601 \cdot 1801$	25	$3 \cdot 11 \cdot 251 \cdot 4051$
$3 \cdot 2731 \cdot 8191$	26	$5 \cdot 53 \cdot 157 \cdot 1613$
$7 \cdot 73 \cdot 262657$	27	$3^4 \cdot 19 \cdot 87211$
$3 \cdot 5 \cdot 29 \cdot 43 \cdot 113 \cdot 127$	28	$17 \cdot 15790321$
$233 \cdot 1103 \cdot 2089$	29	$3 \cdot 59 \cdot 3033169$
$3^2 \cdot 7 \cdot 11 \cdot 31 \cdot 151 \cdot 331$	30	$5^2 \cdot 13 \cdot 41 \cdot 61 \cdot 1321$
2147483647	31	$3 \cdot 715827883$
$3 \cdot 5 \cdot 17 \cdot 257 \cdot 65537$	32	$641 \cdot 6700417$
$7 \cdot 23 \cdot 89 \cdot 599479$	33	$3^2 \cdot 67 \cdot 683 \cdot 20857$
$3 \cdot 43691 \cdot 131071$	34	$5 \cdot 137 \cdot 953 \cdot 26317$
$31 \cdot 71 \cdot 127 \cdot 122921$	35	$3 \cdot 11 \cdot 43 \cdot 281 \cdot 86171$
$3^3 \cdot 5 \cdot 7 \cdot 13 \cdot 19 \cdot 37 \cdot 73 \cdot 109$	36	$17 \cdot 241 \cdot 433 \cdot 38737$
$223 \cdot 616318177$	37	$3 \cdot 1777 \cdot 25781083$
$3 \cdot 174763 \cdot 524287$	38	$5 \cdot 229 \cdot 457 \cdot 525313$
$7 \cdot 79 \cdot 8191 \cdot 121369$	39	$3^2 \cdot 2731 \cdot 22366891$
$3 \cdot 5^2 \cdot 11 \cdot 17 \cdot 31 \cdot 41 \cdot 61681$	40	$257 \cdot 4278255361$
$13367 \cdot 164511353$	41	$3 \cdot 83 \cdot 8831418697$
$3^2 \cdot 7^2 \cdot 43 \cdot 127 \cdot 337 \cdot 5419$	42	$5 \cdot 13 \cdot 29 \cdot 113 \cdot 1429 \cdot 14449$
$431 \cdot 9719 \cdot 2099863$	43	$3 \cdot 2932031007403$
$3 \cdot 5 \cdot 23 \cdot 89 \cdot 397 \cdot 683 \cdot 2113$	44	$17 \cdot 353 \cdot 2931542417$
$7 \cdot 31 \cdot 73 \cdot 151 \cdot 631 \cdot 23311$	45	$3^3 \cdot 11 \cdot 19 \cdot 331 \cdot 18837001$
$3 \cdot 47 \cdot 178481 \cdot 2796203$	46	$5 \cdot 277 \cdot 1013 \cdot 1657 \cdot 30269$
$2351 \cdot 4513 \cdot 13264529$	47	$3 \cdot 283 \cdot 165768537521$
$3^2 \cdot 5 \cdot 7 \cdot 13 \cdot 17 \cdot 97 \cdot 241 \cdot 257 \cdot 673$	48	$193 \cdot 65537 \cdot 22253377$
$179951 \cdot 3203431780337$	59	$3 \cdot 2833 \cdot 37171 \cdot 1824726041$
$3^2 \cdot 5^2 \cdot 7 \cdot 11 \cdot 13 \cdot 31 \cdot 41 \cdot 61 \cdot 151 \cdot 331 \cdot 1321$	60	$17 \cdot 241 \cdot 61681 \cdot 4562284561$
$7^2 \cdot 73 \cdot 127 \cdot 337 \cdot 92737 \cdot 649657$	63	$3^3 \cdot 19 \cdot 43 \cdot 5419 \cdot 77158673929$
$3 \cdot 5 \cdot 17 \cdot 257 \cdot 641 \cdot 65537 \cdot 6700417$	64	$274177 \cdot 67280421310721$

$10^e - 1$	e	$10^e + 1$
$3^3 \cdot 7 \cdot 11 \cdot 13 \cdot 37$	6	$101 \cdot 9901$
$3^2 \cdot 239 \cdot 4649$	7	$11 \cdot 909091$
$3^2 \cdot 11 \cdot 73 \cdot 101 \cdot 137$	8	$17 \cdot 5882353$
$3^4 \cdot 37 \cdot 333667$	9	$7 \cdot 11 \cdot 13 \cdot 19 \cdot 52579$
$3^2 \cdot 11 \cdot 41 \cdot 271 \cdot 9091$	10	$101 \cdot 3541 \cdot 27961$
$3^2 \cdot 21649 \cdot 513239$	11	$11^2 \cdot 23 \cdot 4093 \cdot 8779$
$3^3 \cdot 7 \cdot 11 \cdot 13 \cdot 37 \cdot 101 \cdot 9901$	12	$73 \cdot 137 \cdot 99990001$
$3^2 \cdot 11 \cdot 17 \cdot 73 \cdot 101 \cdot 137 \cdot 5882353$	16	$353 \cdot 449 \cdot 641 \cdot 1409 \cdot 69857$

числами $Y_n = X_n \bmod 2^4$. Суть выражения (4) состоит в том, что младшие четыре разряда $\langle X_n \rangle$ формируют конгруэнтную последовательность с периодом 16 или меньше. Аналогично пять младших разрядов являются периодичными с периодом не более 32 и наименьший значащий разряд X_n является либо постоянным, либо строго периодичным.

Подобная ситуация не возникает, когда $m = w \pm 1$; в таком случае младшие разряды X_n ведут себя так же случайно, как и старшие. Например, при $w = 2^{35}$ и $m = 2^{35} - 1$ числа последовательности будут не очень случайны, если рассмотреть только их остатки по модулю 31, 71, 127 или 122921 (см. табл. 1); но младшие разряды, которые представляют числа последовательности, взятые по mod 2, будут достаточно случайны.

Альтернатива состоит в том, чтобы в качестве m взять наибольшее простое число, меньше, чем w. Это простое число можно найти, используя методы из раздела 4.5.4 и таблицы из того же раздела, в которых содержатся подходящие большие простые числа.

В большинстве случаев применения младшие разряды несущественны и выбор $m = w$ является совершенно удовлетворительным при условии, что программист, работающий со случайными числами, делает это сознательно.

Обсуждение до сих пор базировалось на использующих "величины со знаками" компьютерах типа MIX. Подобные идеи применяются в вычислительных машинах с дополнительной системой обозначений, хотя есть несколько полезных разновидностей. Например, компьютер DECsystem 20 имеет 36 бит с двоичным арифметическим дополнением; когда он вычисляет произведение двух неотрицательных чисел, младшие разряды содержат 35 бит со знаком "плюс". На этой вычислительной машине следовало бы полагать, что $w = 2^{35}$, но не 2^{36}. 32-битовое двоичное арифметическое дополнение на компьютерах IBM System/370 другое: младшие разряды операции умножения содержат 32 полных бита. Некоторые программисты считают, что это недостаток, так как младшие разряды могут быть отрицательными, когда исходное число положительно, и досадно корректировать это. На самом деле есть определенные *преимущества* с точки зрения генерирования случайных чисел, так как можно брать $m = 2^{32}$ вместо 2^{31} (см. упр. 4).

УПРАЖНЕНИЯ

1. [*M12*] В упр. 3.2.1–3 сделан вывод о том, что наилучший конгруэнтный генератор будет иметь множитель a, взаимно простой с m. Покажите, что, когда $m = w$, возможно лучшее вычисление $(aX + c) \bmod w$ точно в *трех* операциях MIX, чем в четырех операциях (1), с результатом, появляющимся в регистре X.

2. [*16*] Напишите подпрограмму на MIX, имеющую следующие характеристики.

Вызывающая последовательность:	`JMP RANDM`
Условия на входе:	Адрес ячейки `XRAND` содержит целое X
Условия на выходе:	$X \leftarrow \mathrm{rA} \leftarrow (aX + c) \bmod w$, $\mathrm{rX} \leftarrow 0$, переполнение выключено

(В результате обращения к этой подпрограмме можно получить следующее случайное число линейной конгруэнтной последовательности.)

▸ **3.** [*M25*] Многие компьютеры не имеют возможности делить числа из двух слов на числа из одного слова; они позволяют выполнять только операции над числами, из отдельных слов, такие как операция himult — himult $(x, y) = \lfloor xy/w \rfloor$ и операция lomult — lomult $(x, y) = xy \bmod w$, когда x и y — неотрицательные целые числа, меньшие, чем компьютерное слово w. Объясните, как вычислить $ax \bmod m$ в терминах himult и lomult, предполагая, что $0 \le a, x < m < w$ и $m \perp w$. Вы можете использовать заранее вычисленные константы, которые зависят от a, m и w.

▸ **4.** [*21*] Исследуйте вычисление линейной конгруэнтной последовательности с $m = 2^{32}$ на машинах с двоичным дополнением, таких, как компьютеры серии System/370.

5. [*20*] Дано, что m меньше, чем длина слова, и что x и y — неотрицательные целые числа, меньшие, чем m. Покажите, что разность $(x - y) \bmod m$ может быть вычислена точно четырьмя операциями без операции деления на машине MIX. Какая программа будет наилучшей для вычисления суммы $(x + y) \bmod m$?

▸ **6.** [*20*] Предыдущее упражнение наводит на мысль, что вычитание по модулю m — более простая операция, чем суммирование по модулю m. Обсудите последовательность, генерируемую по правилу

$$X_{n+1} = (aX_n - c) \bmod m.$$

Будет ли эта последовательность существенно отличаться от линейной конгруэнтной последовательности, определенной ранее? Будет ли она более эффективной при вычислениях?

7. [*M24*] Какие особенности можно заметить в табл. 1?

▸ **8.** [*20*] Напишите программу для вычисления $(aX) \bmod (w - 1)$ на компьютере MIX, аналогичную программе (2). Значения 0 и $w-1$ на входе и выходе вашей программы считаются эквивалентными.

▸ **9.** [*M25*] В большинстве языков программирования высокого уровня не предусмотрен хороший способ деления целого числа из двух слов на целое число из одного слова. Не предусматривается это и операцией himult из упр. 3. Цель данного упражнения — найти приемлемый способ преодоления таких ограничений, когда необходимо вычислить $ax \bmod m$ для переменной x и константы $0 < a < m$.

a) Докажите, что если $q = \lfloor m/a \rfloor$, то $a(x - (x \bmod q)) = \lfloor x/q \rfloor (m - (m \bmod a))$.

b) С помощью равенства (a) вычислите $ax \bmod m$, не оперируя числами, которые превосходят m по абсолютной величине, и предполагая, что $a^2 \le m$.

10. [*M26*] Решение упр. 9, (b) иногда применимо, когда $a^2 > m$. Определите точное число множителей a, для которых промежуточные результаты этого метода никогда не превосходят m для всех x между 0 и m.

11. [*M30*] Продолжая упр. 9, покажите, что можно оценить $ax \bmod m$, используя только следующие основные операции:

i) $u \times v$, где $u \ge 0$, $v \ge 0$ и $uv < m$;
ii) $\lfloor u/v \rfloor$, где $0 < v \le u < m$;
iii) $(u - v) \bmod m$, где $0 \le u, v < m$.

Действительно, это всегда возможно, если использовать максимум 12 операций типа (i) и (ii) и ограниченное число операций типа (iii), не считая предварительного вычисления констант, которые зависят от a и m. Например, объясните, как можно выполнить вычисления, когда a равно 62089911 и m равно $2^{31} - 1$. (Эти константы взяты из табл. 3.3.4–1.)

▸ **12.** [*M28*] Рассмотрите вычисления карандашом на бумаге или на счетах.

a) Найдите хороший метод умножения заданного десятизначного числа на 10 по модулю 9999998999.

b) Сделайте то же самое, но умножив не на 10, а на 999999900 (по модулю 9999998999).

c) Объясните, как вычислить степень $999999900^n \bmod 9999998999$ для $n = 1,2,3,\ldots$.

d) Выполните такие же вычисления с десятичным разложением числа $1/9999998999$.

e) Покажите, что эти идеи предоставляют возможность реализовать определенные виды линейных конгруэнтных генераторов, имеющих очень большие модули, с помощью лишь нескольких операций с генерируемым числом.

13. [*M24*] Повторите предыдущее упражнение, но с модулем 9999999001 и множителями 10 и 8999999101.

14. [*M25*] Обобщите идеи предыдущих двух упражнений для того, чтобы получить большое семейство линейных конгруэнтных генераторов с особенно большими модулями.

3.2.1.2. Выбор множителя. В этом разделе будут рассмотрены методы выбора множителя a для создания *периода максимальной длины*. Длинный период необходим для любой последовательности, используемой в качестве источника случайных чисел. Безусловно, мы ожидаем, что в периоде содержится значительно больше чисел, чем требуется для одноразового использования. Поэтому здесь внимание будет сосредоточено на длине периода. Читателю следовало бы помнить, однако, что длина периода — это только одно из требований к линейным конгруэнтным последовательностям, которые мы хотим использовать, как случайные последовательности. Например, когда $a = c = 1$, последовательность примет простой вид: $X_{n+1} = (X_n + 1) \bmod m$. Она, очевидно, имеет период длиной m, но несмотря на это в ней нет ничего случайного. Другие соображения, влияющие на выбор множителя, будут приведены ниже в этой главе. Так как возможны только m различных значений, длина периода, несомненно, не может быть больше m. Можно ли достичь максимальной длины периода — m? Пример, приведенный выше, показывает, что это всегда возможно, хотя выбор $a = c = 1$ не обеспечивает желаемые свойства последовательности. Исследуем *все* возможные значения a, c и X_0, которые дают период длиной m. Оказывается, что такие значения параметров могут быть охарактеризованы очень просто; когда m является произведением различных простых чисел, только значение $a = 1$ обеспечивает полный период, но когда m делится на простое число в большой степени, то существует значительная свобода в выборе a. Следующая теорема позволяет легко определить, возможно ли достижение периода максимальной длины.

Теорема A. *Линейная конгруэнтная последовательность, определенная числами m, a, c и X_0, имеет период длиной m тогда и только тогда, когда:*

i) *числа c и m взаимно простые;*

ii) *$b = a - 1$ кратно p для каждого простого p, являющегося делителем m;*

iii) *b кратно 4, если m кратно 4.*

Идеи, используемые при доказательстве этой теоремы, впервые возникли, по крайней мере, сто лет тому назад. Но первое ее доказательство в этой особой форме было предложено М. Гринбергером (M. Greenberger) для частного случая при $m = 2^e$ [см. *JACM* **8** (1961), 383–389]. Достаточность условий (i)–(iii) в общем случае была доказана Халлом (Hull) и Добеллом (Dobell) [см. *SIAM Review* **4**

(1962), 230–254]. Чтобы доказать теорему, мы сначала рассмотрим некоторые вспомогательные теоретико-числовые результаты, представляющие и самостоятельный интерес.

Лемма P. *Пусть p — простое число, а e — положительное целое число, такое, что $p^e > 2$. Если*

$$x \equiv 1 \text{ (по модулю } p^e), \qquad x \not\equiv 1 \text{ (по модулю } p^{e+1}), \tag{1}$$

то

$$x^p \equiv 1 \text{ (по модулю } p^{e+1}), \qquad x^p \not\equiv 1 \text{ (по модулю } p^{e+2}). \tag{2}$$

Доказательство. Если x не кратно p, то оно может быть представлено в виде $x = 1 + qp^e$ для некоторого целого q. По биномиальной формуле получаем

$$\begin{aligned} x^p &= 1 + \binom{p}{1} qp^e + \cdots + \binom{p}{p-1} q^{p-1} p^{(p-1)e} + q^p p^{pe} \\ &= 1 + qp^{e+1} \left(1 + \frac{1}{p}\binom{p}{2} qp^e + \frac{1}{p}\binom{p}{3} q^2 p^{2e} + \cdots + \frac{1}{p}\binom{p}{p} q^{p-1} p^{(p-1)e} \right) \end{aligned}$$

Величины в скобках являются целыми числами, и к тому же каждый член внутри скобок, за исключением первого, кратен p. Для таких k, что $1 < k < p$, биномиальные коэффициенты $\binom{p}{k}$ делятся на p (см. упр. 1.2.6–10); следовательно,

$$\frac{1}{p}\binom{p}{k} q^{k-1} p^{(k-1)e}$$

делится на $p^{(k-1)e}$. Последний член $q^{p-1}p^{(p-1)e-1}$ также делится на p, поскольку $(p-1)e > 1$, когда $p^e > 2$. Итак, $x^p \equiv 1 + qp^{e+1}$ (по модулю p^{e+2}), что и завершает доказательство. (*Замечание.* Обобщение этого результата приведено в упр. 3.2.2–11, (a).) ∎

Лемма Q. *Пусть число m допускает разложение на простые множители в виде*

$$m = p_1^{e_1} \ldots p_t^{e_t}. \tag{3}$$

Длина периода λ линейной конгруэнтной последовательности, определенной параметрами (X_0, a, c, m), является наименьшим общим кратным длин λ_j периодов линейных конгруэнтных последовательностей $(X_0 \bmod p_j^{e_j}, a \bmod p_j^{e_j}, c \bmod p_j^{e_j}, p_j^{e_j})$, $1 \le j \le t$.

Доказательство. Если использовать индукцию по t, то достаточно доказать, что если m_1 и m_2 — взаимно простые числа, то длина λ линейной конгруэнтной последовательности, определенной параметрами $(X_0, a, c, m_1 m_2)$, является наименьшим общим кратным длин λ_1 и λ_2 периодов последовательностей, определенных параметрами $(X_0 \bmod m_1, a \bmod m_1, c \bmod m_1, m_1)$ и $(X_0 \bmod m_2, a \bmod m_2, c \bmod m_2, m_2)$. В предыдущем разделе мы заметили (см. (4)), что если элементы этих трех последовательностей соответственно обозначены через X_n, Y_n и Z_n, то справедливо равенство

$$Y_n = X_n \bmod m_1 \qquad \text{и} \qquad Z_n = X_n \bmod m_2 \qquad \text{для всех } n \ge 0.$$

Поэтому по закону D из раздела 1.2.4 находим, что

$$X_n = X_k \qquad \text{тогда и только тогда, когда} \qquad Y_n = Y_k \quad \text{и} \quad Z_n = Z_k. \tag{4}$$

Пусть λ' — наименьшее общее кратное λ_1 и λ_2. Необходимо доказать, что $\lambda' = \lambda$. Так как $X_n = X_{n+\lambda}$ для всех достаточно больших n, $Y_n = Y_{n+\lambda}$ (следовательно, λ кратно λ_1) и $Z_n = Z_{n+\lambda}$ (следовательно, λ кратно λ_2). Таким образом, получим, что $\lambda \geq \lambda'$. Более того, известно, что $Y_n = Y_{n+\lambda'}$ и $Z_n = Z_{n+\lambda'}$ для всех достаточно больших n; поэтому из (4) следует, что $X_n = X_{n+\lambda'}$. Это доказывает, что $\lambda \leq \lambda'$. ▮

Сейчас мы готовы доказать теорему A. Из леммы Q следует, что теорему достаточно доказать для случая, когда m является степенью простого числа, поскольку

$$p_1^{e_1} \ldots p_t^{e_t} = \lambda = \operatorname{lcm}(\lambda_1, \ldots, \lambda_t) \leq \lambda_1 \ldots \lambda_t \leq p_1^{e_1} \ldots p_t^{e_t}$$

(lcm — наименьшее общее кратное. — *Прим. перев.*) выполняется тогда и только тогда, когда $\lambda_j = p_j^{e_j}$ для $1 \leq j \leq t$.

Предположим поэтому, что $m = p^e$, где p — простое число, а e — целое положительное число. Поскольку утверждение теоремы очевидно при $a = 1$, можно считать, что $a > 1$. Период может иметь длину m тогда и только тогда, когда каждое целое число x, такое, что $0 \leq x < m$, встречается в этом периоде. Действительно, никакое значение x в периоде не может встретиться более одного раза. Таким образом, период имеет длину m тогда и только тогда, когда период последовательности с начальным значением $X_0 = 0$ имеет период длиной m. Поэтому достаточно доказать теорему, когда $X_0 = 0$. Из формулы 3.2.1–(6) следует, что

$$X_n = \left(\frac{a^n - 1}{a - 1}\right) c \bmod m. \tag{5}$$

Если c и m не взаимно простые числа, то значение X_n никогда не может быть равно 1. Следовательно, условие (i) теоремы необходимо. Период имеет длину m тогда и только тогда, когда наименьшее положительное значение n, для которого $X_n = X_0 = 0$, равняется $n = m$. Из (5) и условия (i) следует, что доказательство нашей теоремы сводится к доказательству следующего утверждения.

Лемма R. *Предположим, что $1 < a < p^e$, где p — простое число. Если λ — наименьшее целое положительное число, для которого $(a^\lambda - 1)/(a - 1) \equiv 0$ (по модулю p^e), то*

$$\lambda = p^e \quad \textit{тогда и только тогда, когда} \quad \begin{cases} a \equiv 1 \text{ (по модулю } p), & \textit{где} \quad p > 2, \\ a \equiv 1 \text{ (по модулю } 4), & \textit{где} \quad p = 2. \end{cases}$$

Доказательство. Предположим, что $\lambda = p^e$. Если $a \not\equiv 1$ (по модулю p), то $(a^n - 1)/(a - 1) \equiv 0$ (по модулю p^e) тогда и только тогда, когда $a^n - 1 \equiv 0$ (по модулю p^e). Значит, условие $a^{p^e} - 1 \equiv 0$ (по модулю p^e) влечет равенство $a^{p^e} \equiv 1$ (по модулю p), но из теоремы 1.2.4F следует, что $a^{p^e} \equiv a$ (по модулю p). Таким образом, предположение, что $a \not\equiv 1$ (по модулю p), приводит к противоречию. Если $p = 2$ и $a \equiv 3$ (по модулю 4), то из упр. 8 следует

$$(a^{2^{e-1}} - 1)/(a - 1) \equiv 0 \text{ (по модулю } 2^e).$$

Эти рассуждения показывают, что в большинстве случаев необходимо, чтобы $a = 1 + qp^f$, где $p^f > 2$ и q не кратны p, всякий раз, когда $\lambda = p^e$.

Остается показать, что это условие *достаточно* для того, чтобы $\lambda = p^e$. Применив лемму P, находим, что для всех $g \geq 0$

$$a^{p^g} \equiv 1 \text{ (по модулю } p^{f+g}), \qquad a^{p^g} \not\equiv 1 \text{ (по модулю } p^{f+g+1});$$

следовательно,

$$\begin{aligned} (a^{p^g} - 1)/(a-1) &\equiv 0 \text{ (по модулю } p^g), \\ (a^{p^g} - 1)/(a-1) &\not\equiv 0 \text{ (по модулю } p^{g+1}). \end{aligned} \tag{6}$$

В частности, $(a^{p^e} - 1)/(a-1) \equiv 0$ (по модулю p^e). Сейчас для конгруэнтной последовательности, определяемой параметрами $(0, a, 1, p^e)$ X_n, справедливо $X_n = (a^n - 1)/(a-1) \bmod p^e$. Значит, ее период равен λ, т. е. $X_n = 0$ тогда и только тогда, когда n кратно λ. Следовательно, p^e кратно λ. Это может случиться, только если $\lambda = p^g$ для некоторых g и соотношения (6) означают, что $\lambda = p^e$. ▮

Итак, теорема A доказана. ▮

В завершение этого раздела рассмотрим специальный случай использования исключительно мультипликативных генераторов, когда $c = 0$. Несмотря на то что процесс генерирования случайных чисел является немного более быстрым в данном случае, теорема A показывает, что максимальный период не может быть достигнут. Действительно, это совершенно очевидно, так как последовательность удовлетворяет соотношению

$$X_{n+1} = aX_n \bmod m \tag{7}$$

и значение $X_n = 0$ может появиться, только если последовательность вырождается в нуль. Вообще, если d — любой делитель m и если X_n кратно d, все последующие элементы мультипликативной последовательности $X_{n+1}, X_{n+2}, \ldots$ будут кратны d. Так что когда $c = 0$, необходимо, чтобы X_n и m были взаимно простыми числами для всех n, что и ограничивает длину периода максимум до $\varphi(m)$ — числа целых взаимно простых чисел с m, лежащих между 0 и m.

Приемлемой длины периода можно достичь, даже если оговорить, что $c = 0$. Давайте сейчас попытаемся найти такие условия, которым удовлетворяет множитель, чтобы в этом специальном случае период стал настолько длинным, насколько это возможно.

Согласно лемме Q период последовательности зависит исключительно от периодов последовательностей при $m = p^e$. Рассмотрим эту ситуацию. Итак, $X_n = a^n X_0 \bmod p^e$ и ясно, что период будет иметь длину 1 (здесь можно только сказать, что длина периода не больше, чем e. — *Прим. ред.*), если a кратно p. Поэтому будем считать, что a и p взаимно простые. Тогда период будет наименьшим целым числом λ, таким, что $X_0 = a^\lambda X_0 \bmod p^e$. Если наибольшим общим делителем X_0 и p^e является p^f, то это условие эквивалентно условию

$$a^\lambda \equiv 1 \text{ (по модулю } p^{e-f}). \tag{8}$$

По теореме Эйлера (упр. 1.2.4–28) $a^{\varphi(p^{e-f})} \equiv 1$ (по модулю p^{e-f}); следовательно, λ является делителем

$$\varphi(p^{e-f}) = p^{e-f-1}(p-1).$$

Когда a и m — взаимно простые числа, наименьшее число λ, для которого $a^\lambda \equiv 1$ (по модулю m), принято называть *порядком a по модулю m*. Любое такое значение a, которое имеет *максимальный* возможный порядок по модулю m, называют *первообразным элементом* по модулю m.

Обозначим через $\lambda(m)$ порядок первообразного элемента, а именно — максимальный возможный порядок по модулю m. Из замечаний следует, что $\lambda(p^e)$ является делителем $p^{e-1}(p-1)$; достаточно легко (см. упр. 11–16, приведенные ниже) можно определить значения $\lambda(m)$ во всех следующих случаях:

$$\begin{aligned} \lambda(2) = 1, \qquad \lambda(4) = 2, \qquad \lambda(2^e) = 2^{e-2}, \quad \text{если} \quad e \ge 3; \\ \lambda(p^e) = p^{e-1}(p-1), \qquad \text{если} \quad p > 2; \\ \lambda(p_1^{e_1} \ldots p_t^{e_t}) = \operatorname{lcm}\big(\lambda(p_1^{e_1}), \ldots, \lambda(p_t^{e_t})\big). \end{aligned} \tag{9}$$

Все сказанное можно подытожить в следующей теореме.

Теорема B. [C. F. Gauss, *Disquisitiones Arithmeticæ* (1801), §90–92.] *Максимальным периодом, возможным, когда $c = 0$, является $\lambda(m)$, где $\lambda(m)$ определено в* (9). *Этот период достигается, если:*

i) *X_0 и m — взаимно простые числа;*

ii) *a является первообразным элементом по модулю m.* ▮

Заметим, что можно получить период длиной $m - 1$, если m — простое число; т. е. это всего на единицу меньше, чем максимальная длина периода. Так что для практических целей такой период может быть настолько длинным, насколько это необходимо.

Теперь возникает вопрос, как найти первообразные элементы по модулю m? В упражнениях, данных в конце раздела, приводится совершенно очевидный ответ на вопрос, когда m является простым числом или степенью простого числа. Результаты сформулированы в следующей теореме.

Теорема C. *Число a является первообразным элементом по модулю p^e тогда и только тогда, когда выполняется одно из следующих условий:*

i) *$p = 2$, $e = 1$ и a — нечетное число;*

ii) *$p = 2$, $e = 2$ и $a \bmod 4 = 3$;*

iii) *$p = 2$, $e = 3$ и $a \bmod 8 = 3$, 5 или 7;*

iv) *$p = 2$, $e \ge 4$ и $a \bmod 8 = 3$ или 5;*

v) *p — нечетное число, $e = 1$, $a \not\equiv 0$ (по модулю p) и $a^{(p-1)/q} \not\equiv 1$ (по модулю p) для любого простого делителя q числа $p - 1$;*

vi) *p — нечетное число, $e > 1$, a удовлетворяют условию* (v) *и $a^{p-1} \not\equiv 1$* (по модулю p^2). ▮

Условия (v) и (vi) теоремы легко проверяются на компьютере для больших p. Эффективные методы оценки степени, когда известны множители числа $p - 1$, обсуждаются в разделе 4.6.3.

Теорема C применима только к степеням простых чисел. Но если заданы значения a_j, являющиеся первообразными элементами по модулю $p_j^{e_j}$, то можно найти

единственное значение a, такое, что $a \equiv a_j$ (по модулю $p_j^{e_j}$) при $1 \le j \le t$, используя китайский алгоритм (алгоритм, построенный на основании китайской теоремы об остатках. — *Прим. перев.*), рассматриваемый в разделе 4.3.2. Число a будет первообразным элементом по модулю $p_1^{e_1} \ldots p_t^{e_t}$. Таким образом, существует приемлемый эффективный путь построения множителей, удовлетворяющих условию теоремы B, для любых модулей m умеренной размерности, хотя вычисления в общем случае могут быть весьма длинными.

В распространенном случае, когда $m = 2^e$, где $e \ge 4$, изложенные выше условия приводят к единственному требованию: $a \equiv 3$ или 5 (по модулю 8). В этой ситуации четвертая часть всех возможных множителей даст длину периода, равную $m/4$, а $m/4$ будет максимальной длиной периода, когда $c = 0$.

Существует второй, еще более распространенный случай, когда $m = 10^e$. Используя леммы P и Q, нетрудно получить необходимые и достаточные условия достижения максимального периода для десятичного компьютера (см. упр. 18).

Теорема D. *Если* $m = 10^e$, $e \ge 5$, $c = 0$ *и* X_0 *не кратно* 2 *или* 5, *то период линейной конгруэнтной последовательности равен* $5 \times 10^{e-2}$ *тогда и только тогда, когда* $a \bmod 200$ *равно одному из следующих* 32 *чисел:*

$$\begin{array}{l} 3,\ 11,\ 13,\ 19,\ 21,\ 27,\ 29,\ 37,\ 53,\ 59,\ 61,\ 67,\ 69,\ 77,\ 83,\ 91,\ 109,\ 117,\\ 123,\ 131,\ 133,\ 139,\ 141,\ 147,\ 163,\ 171,\ 173,\ 179,\ 181,\ 187,\ 189,\ 197. \quad \blacksquare \end{array} \tag{10}$$

УПРАЖНЕНИЯ

1. [*10*] Чему равна длина периода линейной конгруэнтной последовательности с параметрами $X_0 = 5772156648$, $a = 3141592621$, $c = 2718281829$ и $m = 10000000000$?

2. [*10*] Будут ли следующие два условия гарантировать максимальную длину периода, когда m является степенью 2? (i) c — нечетное число; (ii) $a \bmod 4 = 1$.

3. [*13*] Предположим, что $m = 10^e$, где $e \ge 2$, и пусть c — нечетное число, не кратное 5. Покажите, что линейная конгруэнтная последовательность будет иметь период максимальной длины тогда и только тогда, когда $a \bmod 20 = 1$.

4. [*M20*] Предположим, что $m = 2^e$ и $X_0 = 0$. Если числа a и c удовлетворяют условиям теоремы A, чему равно $X_{2^{e-1}}$?

5. [*14*] Найдите все множители a, удовлетворяющие условиям теоремы A, когда $m = 2^{35} + 1$. (Простые множители m можно найти в табл. 3.2.1.1–1.)

▸ **6.** [*20*] Найдите все множители a, удовлетворяющие условиям теоремы A, когда $m = 10^6 - 1$ (см. табл. 3.2.1.1–1).

▸ **7.** [*M23*] Период конгруэнтной последовательности не должен начинаться с X_0, но всегда можно найти индексы $\mu \ge 0$ и $\lambda > 0$, такие, что $X_{n+\lambda} = X_n$ при всех $n \ge \mu$, и для которых μ и λ являются наименьшими возможными значениями с этими свойствами (см. упр. 3.1–6 и 3.2.1–1). Если μ_j и λ_j — индексы, соответствующие последовательностям

$$(X_0 \bmod p_j^{e_j},\ a \bmod p_j^{e_j},\ c \bmod p_j^{e_j},\ p_j^{e_j}),$$

и если μ и λ соответствуют составной последовательности $(X_0, a, c, p_1^{e_1} \ldots p_t^{e_t})$, то согласно формулировке леммы Q λ является наименьшим общим кратным $\lambda_1, \ldots, \lambda_t$. Чему равно значение μ в терминах значений $\mu_1, \ldots, \mu_t$? Чему равно максимально возможное значение μ, получаемое путем варьирования X_0, a и c, когда $m = p_1^{e_1} \ldots p_t^{e_t}$ фиксировано?

8. [*M20*] Покажите, что если $a \bmod 4 = 3$, то $(a^{2^{e-1}} - 1)/(a - 1) \equiv 0$ (по модулю 2^e), когда $e > 1$. (Воспользуйтесь леммой P.)

▸ **9.** [*M22*] (В. Э. Томсон (W. E. Thomson).) Когда $c = 0$ и $m = 2^e \geq 16$, теоремы B и C утверждают, что период имеет длину 2^{e-2} тогда и только тогда, когда множитель a удовлетворяет равенствам $a \bmod 8 = 3$ или $a \bmod 8 = 5$. Покажите, что каждая такая последовательность, по существу, является линейной конгруэнтной последовательностью с $m = 2^{e-2}$, имеющей *полный* период, в следующем смысле:

a) если $X_{n+1} = (4c+1)X_n \bmod 2^e$ и $X_n = 4Y_n + 1$, то

$$Y_{n+1} = ((4c+1)Y_n + c) \bmod 2^{e-2};$$

b) если $X_{n+1} = (4c-1)X_n \bmod 2^e$ и $X_n = ((-1)^n(4Y_n+1)) \bmod 2^e$, то

$$Y_{n+1} = ((1-4c)Y_n - c) \bmod 2^{e-2}$$

[*Замечание.* В этих формулах c есть нечетное целое число. В литературе содержится достаточно утверждений о том, что последовательности с $c = 0$, удовлетворяющие теореме B, так или иначе являются более случайными, чем последовательности, удовлетворяющие условиям теоремы A, вопреки тому факту, что период — это только четверть длины периода, получаемого в условиях теоремы B. В данном упражнении такие утверждения опровергаются; в сущности, следует удалить два разряда длины слова, чтобы сохранить возможность прибавить c, когда m будет степенью 2.]

10. [*M21*] Для каких значений m справедливо $\lambda(m) = \varphi(m)$?

▸ **11.** [*M28*] Пусть x — нечетное целое число, большее, чем 1. (a) Покажите, что существует единственное целое число $f > 1$, такое, что $x \equiv 2^f \pm 1$ (по модулю 2^{f+1}). (b) Дано, что $1 < x < 2^e - 1$ и что f является соответствующим целым числом п. (a). Покажите, что порядок x по модулю 2^e равен 2^{e-f}. (c) В частности, это доказывает утверждения (i)–(iv) теоремы C.

12. [*M26*] Пусть p — простое нечетное число. Если $e > 1$, докажите, что a является первообразным элементом по модулю p^e тогда и только тогда, когда a — первообразный элемент по модулю p и $a^{p-1} \not\equiv 1$ (по модулю p^2). (Предположите, что $\lambda(p^e) = p^{e-1}(p-1)$. Этот факт доказан в упр. 14 и 16 ниже.)

13. [*M22*] Пусть p — простое число. Задано, что a не является первообразным элементом по модулю p. Покажите, что каждое a кратно p или $a^{(p-1)/q} \equiv 1$ (по модулю p) для некоторых простых чисел q, делящих $p-1$.

14. [*M18*] Предположим, что $e > 1$, p — нечетное простое число и a — первообразный элемент по модулю p. Докажите, что либо a, либо $a+p$ является первообразным элементом по модулю p^e. [*Указание.* См. упр. 12.]

15. [*M29*] (a) Пусть a_1 и a_2 взаимно просты с m и пусть их порядки по модулю m равны соответственно λ_1 и λ_2. Предположим, что λ является наименьшим общим кратным λ_1 и λ_2. Докажите, что $a_1^{\kappa_1}a_2^{\kappa_2}$ имеют порядок λ по модулю m для соответствующих целых чисел κ_1 и κ_2. [*Указание.* Рассмотрите сначала случай, когда λ_1 и λ_2 — взаимно простые числа.] (b) Пусть $\lambda(m)$ — максимальный порядок любого элемента по модулю m. Докажите, что $\lambda(m)$ кратно порядку каждого элемента по модулю m, т. е. что $a^{\lambda(m)} \equiv 1$ (по модулю m) всегда, когда a и m — взаимно простые числа. (Не используйте теорему B.)

▸ **16.** [*M24*] (*Существование первообразных корней.*) Пусть p — простое число.

a) Рассмотрим полином $f(x) = x^n + c_1x^{n-1} + \cdots + c_n$, где c_i — целые числа. Дано, что a — целое число, для которого $f(a) \equiv 0$ (по модулю p). Покажите, что существует полином

$$q(x) = x^{n-1} + q_1x^{n-2} + \cdots + q_{n-1}$$

с целыми коэффициентами, такой, что $f(x) \equiv (x-a)q(x)$ (по модулю p) для всех целых x.

b) Пусть $f(x)$ — такой же полином, как в (a). Покажите, что $f(x)$ имеет не более n различных "корней" по модулю p, т. е. существует не более n целых чисел a, $0 \le a < p$, таких, что $f(a) \equiv 0$ (по модулю p).

c) Так же, как и в упр. 15, (b), полином $f(x) = x^{\lambda(p)} - 1$ имеет $p - 1$ различных корней; следовательно, существует целое число a с порядком $p - 1$.

17. [*M26*] Не все значения, перечисленные в теореме D, можно получить, используя построения, приведенные в разделе, например 11 — не первообразный элемент по модулю 5^e. Как это возможно, если 11 *является* первообразным элементом по модулю 10^e согласно теореме D? Какие из чисел, перечисленных в теореме D, являются *одновременно* первообразными элементами по модулям 2^e и 5^e?

18. [*M25*] Докажите теорему D (см. предыдущее упражнение).

19. [*40*] Составьте таблицу нескольких подходящих множителей a для каждого из значений m, внесенных в табл. 3.2.1.1–1, предполагая, что $c = 0$.

▸ **20.** [*M24*] (Дж. Марсалья (G. Marsaglia).) Назначение упражнения состоит в изучении длины периода *произвольной* линейной конгруэнтной последовательности. Пусть $Y_n = 1 + a + \cdots + a^{n-1}$, так что $X_n = (AY_n + X_0) \bmod m$ для некоторой константы A из условия 3.2.1–(8).

a) Докажите, что длина периода $\langle X_n \rangle$ равна длине периода $\langle Y_n \bmod m' \rangle$, когда $m' = m/\gcd(A, m)$.

b) Докажите, что длина периода $\langle Y_n \bmod p^e \rangle$ удовлетворяет следующим требованиям, когда p — простое число. (i) Если $a \bmod p = 0$, длина периода равна 1. (ii) Если $a \bmod p = 1$, она равна p^e, за исключением случаев, когда $p = 2$, $e \ge 2$ и $a \bmod 4 = 3$. (iii) Если $p = 2$, $e \ge 2$ и $a \bmod 4 = 3$, она равна удвоенному порядку a по модулю p^e (см. упр. 11), за исключением случая, когда $a \equiv -1$ (по модулю 2^e), при котором она равна 2. (iv) Если $a \bmod p > 1$, длина периода равна порядку a по модулю p^e.

21. [*M25*] Пусть в линейной конгруэнтной последовательности с максимальным периодом $X_0 = 0$ s — наименьшее положительное целое число, такое, что $a^s \equiv 1$ (по модулю m). Докажите, что $\gcd(X_s, m) = s$ (gcd — наибольший общий делитель. — *Прим. перев.*).

▸ **22.** [*M25*] Обсудите проблему нахождения модулей $m = b^k \pm b^l \pm 1$ таким образом, чтобы генераторы, использующие вычитание с заимствованием и суммирование с переносом (см. упр. 3.2.1.1–14), имели очень длинные периоды.

3.2.1.3. Потенциал. В предыдущем разделе было показано, что максимальный период может быть достигнут, когда $b = a - 1$ кратно каждому простому делителю m, и b должно быть также кратно 4, если m кратно 4. Если z — основание системы счисления машины ($z = 2$ для бинарного компьютера и $z = 10$ для десятичного компьютера), m — длина слова в компьютере z^e, множитель

$$a = z^k + 1, \qquad 2 \le k < e, \tag{1}$$

удовлетворяет этим условиям. По теореме 3.2.1.2A можно брать $c = 1$. Рекуррентное соотношение теперь имеет вид

$$X_{n+1} = \big((z^k + 1)X_n + 1\big) \bmod z^e, \tag{2}$$

и это уравнение означает, что можно избежать умножения; просто достаточно перемещения и суммирования.

Например, пусть $a = B^2 + 1$, где B — размер байта компьютера MIX. Программа

```
LDA X;  SLA 2;  ADD X;  INCA 1
```
(3)

может использоваться вместо программы, приведенной в разделе 3.2.1.1, и время выполнения программы уменьшается от $16u$ до $7u$.

По этой причине множители, имеющие вид (1), широко обсуждались в литературе. Они действительно рекомендованы многими авторами. Однако первые несколько лет экспериментирования с этим методом убедительно показали, что *множителей, имеющих простой вид* (1), *следует избегать.* Сгенерированные числа просто недостаточно случайны.

Ниже в этой главе будет рассмотрена одна довольно сложная теория, связанная с недостатками всех известных плохих линейных конгруэнтных генераторов случайных чисел. Однако некоторые генераторы (такие, как (2)) настолько ужасны, что достаточно сравнительно простой теории, чтобы исключить их из рассмотрения. Эта простая теория связана с понятием "потенциал", которое мы сейчас обсудим.

Потенциал линейной конгруэнтной последовательности с максимальным периодом определяется как наименьшее целое число s, такое, что

$$b^s \equiv 0 \ (\text{по модулю } m). \tag{4}$$

(Целое число s всегда существует, когда множитель удовлетворяет условиям теоремы 3.2.1.2A, так как b кратно каждому простому делителю m.)

Можно анализировать случайность последовательности, положив $X_0 = 0$, так как 0 встречается в периоде. При этом предположении соотношение 3.2.1–(6) сводится к

$$X_n = \big((a^n - 1)c/b\big) \bmod m;$$

и, если разложить выражение $a^n - 1 = (b + 1)^n - 1$ по биномиальной формуле, получится

$$X_n = c\left(n + \binom{n}{2} b + \cdots + \binom{n}{s} b^{s-1}\right) \bmod m. \tag{5}$$

Все члены разложения b^s, b^{s+1} и т. д. можно исключить, так как они кратны m.

Уравнение (5) столь поучительно, что необходимо рассмотреть некоторые специальные случаи. Если $a = 1$, потенциал равен 1 и $X_n \equiv cn$ (по модулю m), как мы уже видели, так что последовательность наверняка не случайна. Если потенциал равен 2, то $X_n \equiv cn + cb\binom{n}{2}$, и снова последовательность не совсем случайна. Действительно, в этом случае

$$X_{n+1} - X_n \equiv c + cbn.$$

Таким образом, разность между последовательно генерируемыми числами выражена простой зависимостью от n. Точка (X_n, X_{n+1}, X_{n+2}) всегда лежит на одной из четырех плоскостей в трехмерном пространстве:

$$x - 2y + z = d + m, \qquad x - 2y + z = d - m,$$
$$x - 2y + z = d, \qquad x - 2y + z = d - 2m,$$

где $d = cb \bmod m$.

Если потенциал равен 3, то последовательность становится более или менее похожей на случайную, но все еще существует высокая степень зависимости между X_n, X_{n+1} и X_{n+2}. Тесты показывают, что последовательности с потенциалом 3 попрежнему недостаточно хороши. Сообщалось, что приемлемые результаты были получены в некоторых случаях при потенциале, равном 4, но это оспаривалось

другими исследователями. Кажется, что последовательности с потенциалом 5 и выше обладают достаточно хорошими случайными свойствами.

Предположим, например, что $m = 2^{35}$ и $a = 2^k + 1$. Тогда $b = 2^k$, так что величины $b^2 = 2^{2k}$ кратны m, когда $k \geq 18$: потенциал равен 2. Если $k = 17, 16, \ldots, 12$, то потенциал равен 3 и значение потенциала 4 достигается для $k = 11, 10, 9$. Таким образом, с точки зрения потенциала множители приемлемы при $k \leq 8$. Это означает, что $a \leq 257$, но, как мы увидим позже, *малых* множителей также следует избегать. Итак, все множители вида $2^k + 1$, когда $m = 2^{35}$, исключены.

Когда m равно $w \pm 1$, где w — длина слова компьютера, m, вообще говоря, не делится на высокие степени простых чисел и высокий потенциал невозможен (см. упр. 6). Таким образом, в этом случае *не* следует использовать метод максимального периода; лучше использовать метод чистого умножения со значением $c = 0$.

Нужно подчеркнуть, что высокий потенциал является необходимым, но недостаточным условием случайности; мы использовали понятие потенциала для того, чтобы исключить несостоятельные генераторы, но не для того, чтобы безусловно принимать все генераторы с высоким потенциалом. Линейные конгруэнтные последовательности должны пройти "спектральный тест", обсуждаемый в разделе 3.3.4, прежде чем они будут признаны случайными.

УПРАЖНЕНИЯ

1. [*M10*] Покажите, что независимо от размера байта B компьютера MIX программа (3) генерирует последовательность случайных чисел с максимальным периодом.

2. [*10*] Чему равен потенциал генератора, предложенного в программе (3) для компьютера MIX?

3. [*11*] Чему равен потенциал линейной конгруэнтной последовательности, когда $m = 2^{35}$ и $a = 3141592621$? Чему равен потенциал, если множитель равен $a = 2^{23} + 2^{13} + 2^2 + 1$?

4. [*15*] Покажите, что если $m = 2^e \geq 8$, то максимум потенциала достигается при $a \bmod 8 = 5$.

5. [*M20*] Дано, что $m = p_1^{e_1} \ldots p_t^{e_t}$ и $a = 1 + kp_1^{f_1} \ldots p_t^{f_t}$, где a удовлетворяет условиям теоремы 3.2.1.2A и k и m — взаимно простые числа. Покажите, что потенциал равен $\max(\lceil e_1/f_1 \rceil, \ldots, \lceil e_t/f_t \rceil)$.

▸ **6.** [*20*] Какие значения $m = w \pm 1$ из табл. 3.2.1.1–1 могут быть использованы в линейной конгруэнтной последовательности с максимальным периодом, чтобы ее потенциал был равен 4 или выше? (Воспользуйтесь результатом упр. 5.)

7. [*M20*] Когда a удовлетворяет условиям теоремы 3.2.1.2A, оно взаимно просто с m; поэтому существует число a', такое, что $aa' \equiv 1$ (по модулю m). Покажите, что a' может быть просто записано в терминах b.

▸ **8.** [*M26*] Генератор случайных чисел, определенный как $X_{n+1} = (2^{17} + 3)X_n \bmod 2^{35}$ и $X_0 = 1$, был протестирован следующим образом: пусть $Y_n = \lfloor 10X_n/2^{35} \rfloor$; тогда Y_n должен быть случайной цифрой между 0 и 9 и тройка цифр $(Y_{3n}, Y_{3n+1}, Y_{3n+2})$ должна принимать каждое из 1 000 возможных значений от (0, 0, 0) до (9, 9, 9) с приблизительно равными частотами. Но после того как было проверено 30 000 значений, оказалось, что одни тройки встречались крайне редко, а другие — чаще, чем должны были бы. Можно ли объяснить такой неудачный результат испытаний?

3.2.2. Другие методы

Конечно, линейные конгруэнтные последовательности — это не единственный источник случайных чисел, который предлагается для использования на компьютере. В этом разделе будет приведен обзор наиболее существенных альтернативных методов. Одни из них действительно важны, тогда как другие интересны главным образом потому, что они не так хороши, как хотелось бы.

В связи с получением случайных чисел возникает такая общепринятая ошибочная идея — достаточно взять хороший генератор случайных чисел и немного его модифицировать для того, чтобы получить "еще более случайную" последовательность. Часто это не так. Например, известно, что формула

$$X_{n+1} = (aX_n + c) \bmod m \tag{1}$$

позволяет получить более или менее хорошие случайные числа. Может ли последовательность, полученная из

$$X_{n+1} = \big((aX_n) \bmod (m+1) + c\big) \bmod m, \tag{2}$$

быть еще *более* случайной? Ответ: новая последовательность является *менее* случайной с большой долей вероятности. Таким образом, идея потерпела неудачу и при отсутствии какой-нибудь теории о поведении последовательности (2) мы попадаем в область генераторов типа $X_{n+1} = f(X_n)$ с выбранной наудачу функцией f. В упр. 3.1–11, 3.1–15 показано, что эти последовательности, вероятно, ведут себя намного хуже, чем последовательности, полученные при использовании более регулярных функций (1).

Давайте рассмотрим другой подход к попытке получить подлинно улучшенный вариант последовательности (1). Линейный конгруэнтный метод может быть обобщен, скажем, в квадратичный конгруэнтный метод:

$$X_{n+1} = (dX_n^2 + aX_n + c) \bmod m. \tag{3}$$

В упр. 8 обобщена теорема 3.2.1.2A, чтобы получить необходимые и достаточные условия для a, c и d, такие, чтобы последовательность, определенная соотношением (3), имела период максимальной длины m. В этом случае ограничения не более жесткие, чём в линейном методе.

Для m, которое является степенью 2, интересный квадратичный метод предложил Р. Р. Ковэю (R. R. Coveyou). Пусть

$$X_0 \bmod 4 = 2, \qquad X_{n+1} = X_n(X_n + 1) \bmod 2^e, \qquad n \geq 0. \tag{4}$$

Данная последовательность может быть вычислена приблизительно с той же эффективностью, что и (1), без любых беспокойств по поводу переполнения. Этот метод имеет интересную связь с подлинным методом средин квадратов фон Неймана (von Neumann). Если положить Y_n равным $2^e X_n$, так что Y_n является числом двойной точности, полученным в результате размещения справа e нулей у двоичного представления числа X_n, то Y_{n+1} точно совпадет со средними $2e$ цифрами $Y_n^2 + 2^e Y_n$! Другими словами, метод Ковэю в некоторой степени почти идентичен вырожденному методу средин квадратов двойной точности, однако он гарантирует получение длинного периода. Дополнительное свидетельство его случайности приведено в статье Ковэю, цитируемой в ответе к упр. 8.

Другие обобщения выражения (1) также очевидны. Например, можно попытаться увеличить длину периода последовательности. Период линейной конгруэнтной последовательности довольно длинный; когда m приблизительно равно длине слова компьютера, обычно получаются периоды порядка 10^9 или больше и в типичных вычислениях используется лишь маленькая часть последовательности. С другой стороны, при рассмотрении идеи "точности" в разделе 3.3.4 получается, что длина периода влияет на степень случайности последовательности. Значит, желательно получить длинный период и существует несколько подходящих для этого методов. Один из них состоит в том, чтобы сделать X_{n+1} зависящим от X_n и X_{n-1}, а не только от X_n. Тогда длина периода сможет достичь значения m^2, так как последовательность не станет повторяться, пока не будет получено равенство $(X_{n+\lambda}, X_{n+\lambda+1}) = (X_n, X_{n+1})$. Джон Мочли (John Mauchly) в неопубликованной работе, представленной на статистической конференции в 1949 году, расширил метод средин квадратов с помощью рекуррентного соотношения $X_n = \text{середина}\,(X_{n-1} \cdot X_{n-6})$.

Простейшая последовательность, в которой X_{n+1} зависит более чем от одного из предыдущих значений, — это последовательность чисел Фибоначчи

$$X_{n+1} = (X_n + X_{n-1}) \bmod m. \tag{5}$$

Данный генератор рассматривался в начале 50-х годов, и обычно он дает длину периода, бóльшую, чем m. Но тесты показывают, что числа, получаемые с помощью рекуррентного соотношения Фибоначчи, безусловно, *недостаточно* случайны, и, таким образом, выражение (5) нас, главным образом, интересует, как элегантный "плохой пример" источника случайных чисел. Можно также рассматривать генераторы вида

$$X_{n+1} = (X_n + X_{n-k}) \bmod m, \tag{6}$$

когда k — сравнительно большое число. Это рекуррентное соотношение было введено Грином, Смитом и Клем (Green, Smith, and Klem [*JACM* **6** (1959), 527–537]), сообщившими, что, когда $k \le 15$, тестирование последовательности с использованием критерия "интервалов", описанного в разделе 3.3.2, дала отрицательный результат, хотя при $k = 16$ результат был положительным.

Намного лучший аддитивный генератор был изобретен Дж. Ж. Митчелом (G. J. Mitchell) и Д. Ф. Муром (D. P. Moore) в 1958 году [не опубликовано]. Они предложили несколько необычную последовательность, определенную так:

$$X_n = (X_{n-24} + X_{n-55}) \bmod m, \qquad n \ge 55, \tag{7}$$

где m — четное число, а $X_0, \ldots, X_{54}$ — произвольные целые не все четные числа. Числа 24 и 55 в данном определении не были выбраны наудачу; эти специальные числа выбраны, оказывается, для того, чтобы определить такую последовательность, младшие значащие двоичные разряды $\langle X_n \bmod 2\rangle$ которой имеют длину периода $2^{55} - 1$. Очевидно, последовательность $\langle X_n\rangle$ должна иметь период по крайней мере такой же длины. В упр. 30 доказывается, что длина периода последовательности, определенной в выражении (7), точно равна $2^{e-1}(2^{55} - 1)$, когда $m = 2^e$.

На первый взгляд, рекуррентное соотношение (7) кажется не очень удобным для реализации на компьютере, но на самом деле существует эффективный путь генерирования этой последовательности с помощью циклической таблицы.

Алгоритм А (*Аддитивный генератор чисел*). В ячейки памяти $Y[1]$, $Y[2]$, ..., $Y[55]$ записано множество значений X_{54}, X_{53}, ..., X_0 соответственно; j вначале равно 24, а k равно 55. При реализации этого алгоритма на выходе последовательно получаем числа $X_{55}, X_{56}, \ldots$.

А1. [Суммирование.] (Если на выходе мы оказываемся в точке X_n, то $Y[j]$ равно X_{n-24}, а $Y[k]$ равно X_{n-55}.) Запишем $Y[k] \leftarrow (Y[k] + Y[j]) \bmod 2^e$, тогда на выходе получим $Y[k]$.

А2. [Продвижение.] Уменьшим j и k на 1. Если $j = 0$, то присвоим $j \leftarrow 55$; иначе, если $k = 0$, присвоить $k \leftarrow 55$. ▮

Этот алгоритм на компьютере MIX имеет следующий вид.

Программа А (*Аддитивный генератор чисел*). Если предположить, что индексные регистры 5 и 6, содержащие j и k, не затрагиваются частью программы, в которой эта подпрограмма размещена, то следующий текст программы реализует алгоритм А и заносит результат в регистр А.

```
LDA   Y,6   A1. Суммирование.
ADD   Y,5   Y_k + Y_j (возможно переполнение)
STA   Y,6       → Y_k.
DEC5  1     A2. Продвижение. j ← j − 1.
DEC6  1     k ← k − 1.
J5P   *+2
ENT5  55    Если j = 0, присвоить j ← 55.
J6P   *+2
ENT6  55    Если k = 0, присвоить k ← 55.   ▮
```

Этот генератор обычно работает быстрее других генераторов, обсуждавшихся ранее, так как он не требует никакого умножения. Кроме большой скорости выполнения этот алгоритм имеет самый длинный период из тех, которые встречались ранее, за исключением периода из упр. 3.2.1.2–22. Более того, как заметил Ричард Брент (Richard Brent), его можно реализовать в режиме работы с плавающей запятой, избегая преобразований целых чисел в дробные числа и наоборот (см. упр. 23). Поэтому можно доказать, что этот генератор является *наилучшим* источником случайных чисел для достижения практических целей. Основная причина, по которой трудно искренне рекомендовать последовательности, подобные (7), состоит в том, что получено очень мало теоретических результатов, основываясь на которых, можно проверить, имеет ли такая последовательность желаемые случайные свойства. Учитывая, как уже известно, что длинный период не всегда обеспечивает желаемые свойства, Джон Рейзер (John Reiser) (Ph. D. thesis, Stanford University, 1977) показал, однако, что аддитивные последовательности, подобные (7), будут в высокой степени хорошо распределены для больших размерностей, если обеспечить выполнение определенных приемлемых условий (см. упр. 26).

Числа 24 и 55 в (7) обычно называют *запаздыванием*, а числа X_n, определенные в (7), — *последовательностью Фибоначчи с запаздыванием*. Причины, по которым запаздывания, подобные (24, 55), работают хорошо, следуют из приведенных ниже теоретических результатов. Конечно, до некоторой степени легче использовать большие запаздывания, когда в приложениях случается применять, скажем, группы из 55 чисел одновременно. Среди чисел, генерируемых (7), никогда не найдется

Таблица 1

СМЕЩЕНИЯ, ПРИВОДЯЩИЕ К ДЛИННЫМ ПЕРИОДАМ ПО МОДУЛЮ 2

(24, 55)	(37, 100)	(83, 258)	(273, 607)	(576, 3217)	(7083, 19937)
(38, 89)	(30, 127)	(107, 378)	(1029, 2281)	(4187, 9689)	(9739, 23209)

Расширенные варианты этой таблицы приводятся в работах N. Zierler and J. Brillhart, *Information and Control* **13** (1968), 541–554, **14** (1969), 566–569, **15** (1969), 67–69; Y. Kurita and M. Matsumoto, *Math. Comp.* **56** (1991), 817–821; Heringa, Blöte, and Compagner, *Int. J. Mod. Phys.* **C3** (1992), 561–564.

значений X_n, лежащих строго между X_{n-24} и X_{n-55} (см. упр. 2). Ж.-М. Норманд (J.-M. Normand), Г. Й. Герман (H. J. Herrmann) и М. Хаджар (M. Hajjar) обнаружили небольшое смещение в числах, генерируемых (7), когда им понадобилось 10^{11} случайных чисел для проводимых с высокой точностью обширных исследований метода Монте-Карло [*J. Statistical Physics* **52** (1988), 441–446]; но при больших значениях k смещение уменьшалось. В табл. 1 приведено несколько пар чисел (l, k), для которых последовательность $X_n = (X_{n-l} + X_{n-k}) \bmod 2^e$ имеет период длиной $2^{e-1}(2^k - 1)$. Случая, когда $(l, k) = (30, 127)$, казалось бы, достаточно для большинства применений, особенно в сочетании с другими, увеличивающими случайность, методами, которые мы обсудим ниже.

Генератор случайных чисел во многом подобен сексу: когда он хорош — это прекрасно, когда он плох, все равно приятно (Джордж Марсалья, 1984).

Джордж Марсалья [*Comp. Sci. and Statistics: Symposium on the Interface* **16** (1984), 3–10] предложил заменить (7) на

$$X_n = (X_{n-24} \cdot X_{n-55}) \bmod m, \qquad n \geq 55, \tag{7'}$$

где m кратно 4, а все числа от X_0 до X_{54} нечетны, но сравнимы с 1 (по модулю 4). Тогда второстепенные младшие разряды имеют период $2^{55} - 1$, в то время как старшие двоичные разряды более тщательно перемешаны, чем раньше, так как они существенно зависят от всех разрядов X_{n-24} и X_{n-55}. В упр. 31 показано, что длина периода последовательности (7′) лишь незначительно меньше длины периода последовательности (7).

Генераторы последовательности Фибоначчи с запаздыванием успешно применялись во многих ситуациях с 1958 года. Таким образом, открытие в 90-е годы того, что они фактически провалились на крайне простом, незамысловатом критерии случайности, явилось шоком (см. упр. 3.3.2–31). Как избежать таких неприятностей, отбрасывая ненужные элементы последовательности, рассказывается в конце этого раздела.

Вместо рассмотрения исключительно аддитивных или исключительно мультипликативных последовательностей можно построить достаточно хороший генератор случайных чисел, используя всевозможные линейные комбинации $X_{n-1}, \ldots, X_{n-k}$ для малых k. В этом случае наилучший результат получается, когда модуль m является большим *простым числом*; например, m может быть выбрано так, чтобы оно было наибольшим простым числом, которое можно записать одним компьютерным словом (см. табл. 4.5.4–2). Когда $m = p$ — простое число, то по теории конечных полей можно найти множители $a_1, \ldots, a_k$, такие, что последовательность,

определенная соотношением

$$X_n = (a_1 X_{n-1} + \cdots + a_k X_{n-k}) \bmod p, \tag{8}$$

будет иметь период длиной $p^k - 1$. Здесь $X_0, \ldots, X_{k-1}$ могут быть выбраны произвольно, но так, чтобы все они не были нулями. (Частный случай, когда $k = 1$, соответствует мультипликативной конгруэнтной последовательности с уже известным простым модулем.) Константы $a_1, \ldots, a_k$ в (8) обладают подходящими свойствами тогда и только тогда, когда полином

$$f(x) = x^k - a_1 x^{k-1} - \cdots - a_k \tag{9}$$

является первообразным полиномом по модулю p, что выполняется тогда и только тогда, когда корень этого полинома есть первообразный элемент поля с p^k элементами (см. упр. 4.6.2–16).

Конечно, для достижения практических целей недостаточно простого факта *существования* подходящих констант a_1, ..., a_k, дающих период длиной $p^k - 1$. Необходимо быть в состоянии *найти* их, ведь нельзя проверить все p^k возможностей, так как p имеет порядок длины слова компьютера. К счастью, есть точно $\varphi(p^k - 1)/k$ подходящих наборов $(a_1, \ldots, a_k)$, поэтому в известной степени существует хороший шанс натолкнуться на один из них после нескольких случайных попыток. Но также следует уметь быстро определять, будет ли (9) первообразным полиномом по модулю p. Конечно, немыслимо генерировать до $p^k - 1$ элементов последовательности и ждать повторения! Методы проверки того, что полином будет первообразным по модулю p, обсуждались Аланеном (Alanen) и Кнутом (Knuth) в *Sankhyā* **A26** (1964), 305–328. Можно использовать следующий критерий. Пусть $r = (p^k - 1)/(p - 1)$.

i) $(-1)^{k-1} a_k$ должен быть первообразным корнем по модулю p (см. раздел 3.2.1.2).
ii) Полином x^r должен быть сравним с $(-1)^{k-1} a_k$ по модулям $f(x)$ и p.
iii) Степень $x^{r/q} \bmod f(x)$ (здесь используется арифметика полиномов по модулю p) должна быть положительной для каждого r — простого делителя q.

Эффективный способ вычисления полинома $x^n \bmod f(x)$ с использованием полиномиальной арифметики по модулю, заданному простым числом p, обсуждается в разделе 4.6.2.

Для того чтобы довести до конца тест, необходимо знать разложение на простые множители числа $r = (p^k - 1)/(p - 1)$, что и является ограничивающим фактором в вычислениях. r можно разложить на множители в приемлемый отрезок времени, когда $k = 2$, 3 и, возможно, 4, но большие значения k усложняют вычисления, когда p большое. Даже при $k = 2$ число "значащих случайных цифр", которое достигается при $k = 1$, по существу, удваивается, так что большие значения k вряд ли понадобятся.

Видоизмененный спектральный критерий (раздел 3.3.4) можно использовать для оценки последовательности чисел, генерируемых (8); см. упр. 3.3.4–24. Рассуждения, приведенные в этом разделе, показывают, что *не* следовало бы делать очевидный выбор ($a_1 = +1$ или -1), когда встречается такая форма первообразного полинома. Лучше выбрать большие, совершенно "случайные" значения $a_1, \ldots, a_k$, удовлетворяющие условиям, и проверить выбор с помощью спектрального критерия. Значительный объем вычислений приходится выполнять при возведении в

степень для нахождении $a_1, \ldots, a_k$. Но все известные доводы указывают на то, что результатом будет весьма удовлетворительный источник случайных чисел. Мы, по существу, добились случайности линейных конгруэнтных генераторов с k-кратной точностью, используя только операции с простой точностью.

Представляет интерес частный случай, когда $p = 2$. Иногда требуется генератор случайных чисел, вырабатывающий всего лишь случайную последовательность *двоичных разрядов* — нулей и единиц — вместо дробей, лежащих между нулем и единицей. Существует простой способ генерирования высокослучайных двоичных разрядов последовательности на бинарных компьютерах, основанный на манипулировании k-разрядными словами. Начать следует с произвольного ненулевого двоичного слова X. Затем необходимо вычислить следующий случайный бит последовательности, выполнив операции, которые приведены на языке компьютера MIX (см. упр. 16):

```
LDA  X      (Предположим, что переполнение сейчас "выключено".)
ADD  X      Перенести влево один разряд.
JNOV *+2    Перейти к другой команде, если исходное значение высшего
                разряда было нулем.                                    (10)
XOR  A      Иначе — установить число с "исключающим или".
STA  X
```

Четвертая операция "исключающее или" существует почти на всех двоичных компьютерах (см. упр. 2.5–28 и раздел 7.1). Она изменяет каждую позицию двоичного разряда rA, в ячейке A которой содержится "1". Содержимое ячейки A — это двоичная константа $(a_1 \ldots a_k)_2$, где $x^k - a_1 x^{k-1} - \cdots - a_k$ является первообразным полиномом по модулю 2, как упоминалось выше. После выполнения программы (10) следующим двоичным разрядом генерируемой последовательности можно взять младший двоичный разряд слова X. Или же можно последовательно выбирать старший двоичный разряд X, если он больше подходит.

Рассмотрим, например, рис. 1, на котором иллюстрируется генерируемая последовательность для $k = 4$ и СОДЕРЖИМОГО(A) $= (0011)_2$. Это, конечно, необычно малое значение k. В столбце показано, что последовательность двоичных разрядов последовательности, а именно — 1101011110001001..., повторяется с длиной периода $2^k - 1 = 15$. Эта последовательность совершенно случайна, если принять во внимание, что она была сгенерирована только с четырьмя двоичными разрядами памяти. Чтобы убедиться в этом, рассмотрим примыкающие множества четырех двоичных разрядов в периоде, а именно — 1101, 1010, 0101, 1011, 0111, 1111, 1110, 1100, 1000, 0001, 0010, 0100, 1001, 0011, 0110. Вообще говоря, каждое возможное примыкающее множество k двоичных разрядов однажды встречается в периоде, исключая множество всех нулей, так как длина периода равна $2^k - 1$. Таким образом, примыкающие множества k двоичных разрядов совершенно независимы. В разделе 3.5 показано, что это очень сильный критерий случайности, когда k равняется, скажем, 30 или больше. Теоретические результаты, иллюстрирующие случайность этой последовательности, приведены в статье Р. К. Таусворта (R. C. Tausworthe, *Math. Comp.* **19** (1965), 201–209).

1011
0101
1010
0111
1110
1111
1101
1001
0001
0010
0100
1000
0011
0110
1100
1011

Рис. 1. Последовательные значения компьютерного слова X в бинарном методе, если предположить, что $k = 4$ и СОДЕРЖИМОЕ(A) $= (0011)_2$.

Первообразный полином по модулю 2 степени ≤ 168 табулирован В. Станке (W. Stahnke, *Math. Comp.* **27** (1973), 977–980.) Когда $k = 35$, можем взять

$$\text{СОДЕРЖИМОЕ(A)} = (00000000000000000000000000000000101)_2,$$

но, принимая во внимание упр. 18 и 3.3.4–24, приходим к заключению, что лучше найти "случайные" константы, определяющие первообразный полином по модулю 2.

Предостережение. Кое-кто обманывается, полагая, что техника случайного поразрядного генерирования может использоваться для генерирования случайных дробей, которые занимают целое слово $(X_0X_1\ldots X_{k-1})_2$, $(X_kX_{k+1}\ldots X_{2k-1})_2$, ... Но на самом деле это скудный источник случайных дробей, даже несмотря на то, что отдельный двоичный разряд совершенно случаен. В упр. 18 объясняется, почему так происходит.

Аддитивный генератор (7) Митчела (Mitchell) и Мура (Moore) в основном базируется на концепции первообразных полиномов: полином $x^{55} + x^{24} + 1$ является первообразным, а табл. 1 — это, по существу, таблица определенных первообразных трехчленов по модулю 2. Почти идентичный генератор независимо от Митчела и Мура открыли в 1971 году Т. Ж. Левис (T. G. Lewis) и В. Г. Пэйн (W. H. Payne) [*JACM* **20** (1973), 456–468], но они использовали операцию "исключающее или" вместо суммирования. Это делает длину периода равной точно $2^{55} - 1$. Каждая позиция двоичного разряда в последовательности Левиса и Пэйна пробегает ту же периодическую последовательность, но имеет собственную начальную позицию. Опыт показал, что (7) дает лучшие результаты.

Мы сейчас увидим, что последовательности с $0 \le X_n < m$ и периодом $m^k - 1$ могут быть построены без больших усилий, где X_n — подходящая функция от $X_{n-1}, \ldots, X_{n-k}$ и m — простое число. Наибольший возможный период для *любой* последовательности, определенной соотношением вида

$$X_n = f(X_{n-1}, \ldots, X_{n-k}), \qquad 0 \le X_n < m, \tag{11}$$

как легко видеть, равен m^k. М. Г. Мартин (M. H. Martin) [*Bull. Amer. Math. Soc.* **40** (1934), 859–864] первым показал, что для всех m и k существуют функции, позволяющие достичь максимального периода. Его метод можно легко сформулировать (упр. 17) и достаточно эффективно запрограммировать (см. упр. 29), но он непригоден для генерирования случайных чисел, потому что изменение значений $X_{n-1} + \cdots + X_{n-k}$ — очень медленный процесс: все строки размерностью k встречаются, но в не совсем случайном порядке. Лучший класс функций f,

дающих максимальный период m^k, рассмотрен в упр. 21. Подобные программы, вообще говоря, не так эффективны для генератора случайных чисел, как другие уже описанные методы, но, когда речь идет о периоде в целом, они все-таки производят достаточно случайные последовательности.

Для генерирования случайных чисел предложено множество других схем. Наиболее интересным из этих альтернативных методов является, возможно, *обратная конгруэнтная последовательность*, предложенная Эйченауэром (Eichenauer) и Лехном (Lehn) [*Statistische Hefte* **27** (1986), 315–326]:

$$X_{n+1} = (aX_n^{-1} + c) \bmod p. \tag{12}$$

Здесь p — простое число, X_n принимает значения из множества $\{0, 1, \ldots, p-1, \infty\}$, а обращение определено как $0^{-1} = \infty$, $\infty^{-1} = 0$. В других случаях $X^{-1}X \equiv 1$ (по модулю p). Так как 0 всегда следует за ∞, а затем за c в этой последовательности, можно было бы просто определить $0^{-1} = 0$ для удобства реализации, но теория является цельной и естественной, когда $0^{-1} = \infty$. Существуют эффективные алгоритмы, которые можно реализовать на вычислительных машинах, пригодные для вычислений X^{-1} (по модулю p) (см., например, упр. 4.5.2–39). К несчастью, однако, операций, используемых в этих алгоритмах, в большинстве компьютеров нет. В упр. 35 показано, как много выборов a и c приводят к максимальному периоду длиной $p + 1$. В упр. 37 продемонстрированы наиболее важные свойства: обратная конгруэнтная последовательность совершенно лишена решетчатой структуры, что характерно для линейных конгруэнтных последовательностей.

Другой важный класс методов связан с *комбинацией* генераторов случайных чисел.

Всегда найдутся сторонники того, что линейные конгруэнтные, аддитивные и другие методы слишком просты для того, чтобы давать достаточно случайную последовательность, и невозможно будет *доказать*, что такой скептицизм необоснован. Может быть, они в самом деле правы, так что совершенно бесполезно обсуждать этот вопрос. Существует достаточно эффективный способ объединения двух последовательностей в третью, которая была бы настолько случайной, что удовлетворяла бы всех, кроме совершенно убежденных скептиков.

Допустим, имеются последовательности $X_0, X_1, \ldots$ и $Y_0, Y_1, \ldots$ случайных чисел, лежащих между 0 и $m - 1$ и предпочтительно сгенерированных двумя различными методами. Тогда можно, например, использовать одну случайную последовательность для изменения порядка элементов другой, как предложили М. Д. Мак-Ларен (M. D. MacLaren) и Дж. Марсалья (G. Marsaglia) [*JACM* **12** (1965), 83–89; см. также работу Марсалья и Брея (Bray), *CACM* **11** (1968), 757–759].

Алгоритм M (*Рандомизация перемешиванием*). Если заданы методы генерирования двух последовательностей $\langle X_n \rangle$ и $\langle Y_n \rangle$, этот алгоритм будет последовательно генерировать элементы "значительно более случайной" последовательности. Воспользуемся вспомогательной таблицей $V[0], V[1], \ldots, V[k-1]$, где k — некоторое число, для удобства обычно выбираемое приблизительно равным 100. Вначале V-таблица заполняется первыми k значениями X-последовательности.

M1. [Генерирование X, Y.] Положим X и Y равными следующим членам последовательностей $\langle X_n \rangle$ и $\langle Y_n \rangle$ соответственно.

M2. [Выбор j.] Присвоим $j \leftarrow \lfloor kY/m \rfloor$, где m — модуль, используемый в последовательности $\langle Y_n \rangle$, т. е. j — случайная величина, определяемая Y, $0 \le j < k$.

M3. [Замена.] Выведем $V[j]$, а затем присвоим $V[j] \leftarrow X$. ▮

Предположим, например, что алгоритм M применяется к таким двум последовательностям при $k = 64$:

$$\begin{aligned} X_0 &= 5772156649, \qquad X_{n+1} = (3141592653X_n + 2718281829) \bmod 2^{35}; \\ Y_0 &= 1781072418, \qquad Y_{n+1} = (2718281829Y_n + 3141592653) \bmod 2^{35}. \end{aligned} \tag{13}$$

Интуиция подсказывает, что последовательность, полученная в результате реализации алгоритма M (13), удовлетворяет *любым* требованиям случайности, которые предъявляются к генерируемым на компьютере последовательностям, поскольку зависимость между соседними выходными элементами почти полностью исключена. Более того, время генерирования этой последовательности лишь незначительно превышает удвоенное время, необходимое для генерирования последовательности $\langle X_n \rangle$.

В упр. 15 показано, что в ситуациях, представляющих наибольший практический интерес, длина периода последовательности, которая получается при реализации алгоритма M, равна наименьшему общему кратному длин периодов $\langle X_n \rangle$ и $\langle Y_n \rangle$. В частности, если отбросить значение 0, когда оно встречается в Y-последовательности, так что $\langle Y_n \rangle$ имеет период длиной $2^{35} - 1$, то числа, генерируемые алгоритмом M из рекуррентных соотношений (13), будут иметь период длиной $2^{70} - 2^{35}$. (См. работу Дж. Артура Гринвуда [J. Arthur Greenwood, *Comp. Sci. and Statistics: Symp. on the Interface* **9** (1976), 222].)

Однако существует еще лучший путь перемешивания элементов последовательности, открытый Картером Бейсом (Carter Bays) и С. Д. Дархамом (S. D. Durham) [*ACM Trans. Math. Software* **2** (1976), 59–64]. Хотя их подход и появился для того, чтобы несколько упростить алгоритм M, неожиданно оказалось, что он может дать лучшие результаты, чем алгоритм M, даже несмотря на то, что на входе он требует только одну последовательность $\langle X_n \rangle$ вместо двух.

Алгоритм B (*Рандомизация перемешиванием*). Если задан метод генерирования последовательности $\langle X_n \rangle$, этот алгоритм будет последовательно выводить элементы "значительно более случайной" последовательности, используя вспомогательную таблицу $V[0], V[1], \ldots, V[k-1]$, как и в алгоритме M. Вначале V-таблица заполняется первыми k значениями X-последовательности, а вспомогательную переменную Y положим равной $k+1$-му значению.

B1. [Выбор j.] Присвоим $j \leftarrow \lfloor kY/m \rfloor$, где m — модуль, используемый в последовательности $\langle X_n \rangle$; т. е. j — это случайная величина, определяемая Y, $0 \le j < k$.

B2. [Замена.] Выведем $V[j]$, присвоим $V[j] \leftarrow X$, выведем $V[j]$ и установим $V[j]$ следующим членом последовательности $\langle X_n \rangle$. ▮

Желание почувствовать различие между алгоритмами M и B побудит читателя заняться упр. 3 и 5.

На компьютере MIX можно реализовать алгоритм B, взяв k равным размеру байта и выполнив вычисления в соответствии со следующей простой программой

генерирования случайных чисел.

```
LD6  Y(1:1)   j ← старший разряд байта Y.
LDA  X        rA ← X_n.
INCA 1        (См. упр. 3.2.1.1–1.)
MUL  A        rX ← X_{n+1}.
STX  X        "n ← n + 1."
LDA  V,6                                   (14)
STA  Y        Y ← V[j].
STX  V,6      V[j] ← X_n.  ▮
```

Выход появляется в регистре A. Заметим, что алгоритм B требует всего четыре дополнительные операции для генерирования числа.

Ф. Гебхардт (F. Gebhardt) [*Math. Comp.* **21** (1967), 708–709] нашел, что удовлетворительная случайная последовательность порождается алгоритмом M, даже когда он применяется к такой неслучайной последовательности, как последовательность Фибоначчи, с $X_n = F_{2n} \bmod m$ и $Y_n = F_{2n+1} \bmod m$. Однако для алгоритма M также возможно получение последовательности, *менее* случайной, чем исходная последовательность, если $\langle X_n \rangle$ и $\langle Y_n \rangle$ строго зависимы, как в упр. 3. Такие проблемы, кажется, не возникают с алгоритмом B. Поскольку алгоритм B не делает никакую последовательность менее случайной и очень мала цена увеличения случайности, он может быть рекомендован к использованию с любым другим генератором случайных чисел.

Однако методы перемешивания имеют "врожденный дефект" Они изменяют порядок следования генерируемых чисел, но не сами числа. В большинстве случаев порядок следования является решающим фактором, но, если генератор случайных чисел не удовлетворяет "критерию промежутков между днями рождений", обсуждаемому в разделе 3.3.2, или критерию случайных блужданий из упр. 3.3.2–31, то положение после перемешивания не улучшится. Перемешивание имеет еще одно сравнительное неудобство, заключающееся в том, что оно не позволяет стартовать с заданного места в периоде либо быстро перемещаться из X_n в X_{n+k} при больших k.

Многие поэтому советуют объединять последовательности $\langle X_n \rangle$ и $\langle Y_n \rangle$ более простым способом, лишенным дефектов перемешивания. Например, можно использовать объединение вида

$$Z_n = (X_n - Y_n) \bmod m, \tag{15}$$

где $0 \le X_n < m$ и $0 \le Y_n < m' \le m$. В упр. 13 и 14 обсуждается длина периода таких последовательностей; в упр. 3.3.2–23 показано, что (15) имеет тенденцию к увеличению случайности, если начальные значения X_0 и Y_0 выбираются независимо.

Простой метод устранения структурных смещений арифметически генерируемых чисел был предложен на заре программирования Дж. Тоддом (J. Todd) и О. Таусски Тодд (O. Taussky Todd) [*Symp. on Monte Carlo Methods* (Wiley, 1956), 15–28]. Мы можем просто выбросить несколько чисел последовательности. Их предложение мало использовалось в линейных конгруэнтных генераторах, но оно стало использоваться сейчас в связи с появлением генераторов, подобных (7), имеющих очень длинный период, потому что есть сколько угодно чисел, которые можно отбросить.

Простейшим путем улучшения случайности (7) является использование только каждого j-го элемента для некоторого малого j. Но лучшим способом, возможно,

еще более простым, является применение (7) для получения, скажем, массива из 500 случайных чисел и использование только первых 55 чисел. После этого таким же методом генерируются следующие 500 чисел. Эта идея была предложена Мартином Люшером (Martin Lüscher) [*Computer Physics Communications* **79** (1994), 100–110]. Толчком послужила теория хаоса в динамических системах. Можно рассматривать (7) как процесс преобразования 55 значений $(X_{n-55}, \ldots, X_{n-1})$ в другой вектор из 55 значений $(X_{n+t-55}, \ldots, X_{n+t-1})$. Предположим, что генерируется $t \geq 55$ значений, а используются первые 55. Тогда, если $t = 55$, новый вектор значений в некоторой степени близок старому; но, если $t \approx 500$, старый и новый векторы всегда не коррелируют между собой (см. упр. 33). Для аналогичного случая генераторов суммирования с переносом или вычитания с заимствованием (упр. 3.2.1.1–14), как известно, векторы будут представлениями чисел в b-ичной системе счисления в линейном конгруэнтном генераторе, а подходящим множителем, когда генерируется сразу t чисел, будет b^{-t}. Теория Люшера в этом случае, следовательно, может быть подтверждена спектральным критерием из раздела 3.3.4. Портативный генератор случайных чисел, основанный на последовательности Фибоначчи с запаздыванием, усиленный методом Люшера, рассматривается в разделе 3.6, в котором приведены и комментарии.

Генераторы случайных чисел, как правило, выполняют лишь незначительное число умножений и/или суммирований при переходе от одного члена последовательности к другому. Когда такие генераторы комбинируются, как рассказывалось выше, здравый смысл говорит нам, что полученная последовательность не должна отличаться от настоящей случайной последовательности. Однако интуиция не может заменить строгое математическое доказательство. Если поработать дольше (скажем, 1 000 или 1 000 000 часов), можно получить последовательности, для которых, по существу, имеются лучшие теоретические гарантии случайности.

Например, рассмотрим последовательность двоичных разрядов $B_1, B_2, \ldots$, генерируемую соотношением

$$X_{n+1} = X_n^2 \bmod M, \qquad B_n = X_n \bmod 2 \tag{16}$$

(см. работу Blum, Blum, and Shub, *SICOMP* **15** (1986), 364–383), или более сложную последовательность, генерируемую соотношением

$$X_{n+1} = X_n^2 \bmod M, \qquad B_n = X_n \cdot Z \bmod 2, \tag{17}$$

где скалярное произведение r-разрядных двоичных чисел $(x_{r-1} \ldots x_0)_2$ и $(z_{r-1} \ldots z_0)_2$ равно $x_{r-1}z_{r-1} + \cdots + x_0 z_0$. Здесь Z — r-разрядная "маска", а r — число двоичных разрядов в M. Модуль M может быть произведением двух больших простых чисел вида $4k+3$, а начальное значение X_0 — взаимно простым числом с M. Правило (17), предложенное Леонидом Левиным, является обобщением метода средин квадратов фон Неймана; мы будем называть его *смешанно-квадратичным методом*, потому что он перемешивает квадраты двоичных разрядов. Правило (16), конечно, является частным случаем для $Z = 1$. В разделе 3.5F содержится доказательство того, что, когда X_0, Z и M выбраны наудачу, последовательность, сгенерированная соотношениями (16) и (17), удовлетворяет всем статистическим критериям случайности; на генерирование такой последовательности требуется усилий не больше, чем на умножение больших чисел. Другими словами, двоичные разряды неотличимы от

действительно случайных чисел для любых вычислений, длящихся менее ста лет, на современных быстродействующих компьютерах, когда M достаточно велико. Если это не так, то можно найти множители нетривиальных частей таких чисел намного быстрее, чем известно сейчас. Формула (16) проще, чем (17), но модуль M в (16) должен быть несколько больше, чем в (17), если необходимо получить те же статистические гарантии.

УПРАЖНЕНИЯ

▶ **1.** [*12*] На практике случайные числа формируются с помощью $X_{n+1} = (aX_n + c) \bmod m$, где X_n — *целые числа*, которые впоследствии рассматриваются как *дроби* $U_n = X_n/m$. Рекуррентное соотношение для U_n на самом деле имеет вид

$$U_{n+1} = (aU_n + c/m) \bmod 1.$$

Объясните, как генерируется случайная последовательность с помощью этого соотношения, *непосредственно* используя арифметику с плавающей точкой компьютера.

▶ **2.** [*M20*] В хорошем источнике случайных чисел неравенства $X_{n-1} < X_{n+1} < X_n$ будут встречаться примерно один раз из шести, так как каждое из шести возможных отношений порядка X_{n-1}, X_n и X_{n+1} должно иметь одну и ту же вероятность. Покажите, однако, что приведенный выше порядок *никогда* не возникнет, если использовать последовательность Фибоначчи (5).

3. [*23*] (a) Какую последовательность генерирует алгоритм M, если

$$X_0 = 0, \quad X_{n+1} = (5X_n + 3) \bmod 8, \qquad Y_0 = 0, \quad Y_{n+1} = (5Y_n + 1) \bmod 8$$

и $k = 4$? (Заметим, что потенциал равен двум, т. е. $\langle X_n \rangle$ и $\langle Y_n \rangle$ не настолько случайны, чтобы стоило их использовать.) (b) Что случится, если алгоритм B применить к этой же последовательности $\langle X_n \rangle$ с $k = 4$?

4. [*00*] Почему наибольший значащий байт используется в первой строке программы (14) вместо других?

▶ **5.** [*20*] Обсудите, стоит ли использовать $X_n = Y_n$ в алгоритме M для того, чтобы повысить скорость генерирования. Будет ли результат аналогичен результату, полученному с использованием алгоритма B?

6. [*10*] В бинарном методе (10) младший разряд `X` случаен, если программа многократно повторяется. Почему все *слово* `X` не случайно?

7. [*20*] Покажите, что можно получить полную последовательность длиной 2^e (т. е. последовательность, в которой каждое из 2^e возможных множеств e примыкающих разрядов встречается только один раз за период), если программу (10) изменить следующим образом.

```
LDA  X          LDA  A          JNOV *+3        XOR  A
JANZ *+2        ADD  X          JAZ  *+2        STA  X
```

8. [*M39*] Докажите, что квадратичная конгруэнтная последовательность (3) имеет период длиной m тогда и только тогда, когда выполняются следующие условия:

i) *c и m — взаимно простые числа;*
ii) *d и $a - 1$ кратны p для всех p — нечетных простых делителей m;*
iii) *d четно и $d \equiv a - 1$* (по модулю 4), *если m кратно* 4;
$d \equiv a - 1$ (по модулю 2), *если m кратно* 2;
iv) $d \not\equiv 3c$ (по модулю 9), *если m кратно* 9.

[*Указание.* Последовательность, определенная как $X_0 = 0$, $X_{n+1} = dX_n^2 + aX_n + c$, по модулю m имеет период длиной m тогда и только тогда, когда эта же последовательность по модулю r имеет период длиной r, где r — делитель m.]

▸ **9.** [*M24*] (P. R. Ковэю (R. R. Coveyou).) Используйте результаты упр. 8 для доказательства того, что длина периода модифицированного метода средин квадратов (4) равна 2^{e-2}.

10. [*M29*] Покажите, что если X_0 и X_1 — такие простые числа, что по крайней мере одно из них нечетно, и $m = 2^e$, то период последовательности Фибоначчи (5) равен $3 \cdot 2^{e-1}$.

11. [*M36*] Назначение этого упражнения состоит в анализе определенных свойств последовательности целых чисел, удовлетворяющих рекуррентному соотношению

$$X_n = a_1 X_{n-1} + \cdots + a_k X_{n-k}, \qquad n \geq k.$$

Если можно вычислить длину периода данной последовательности по модулю $m = p^e$, когда p — простое число, то длина периода этой последовательности по отношению к произвольному модулю m равна наименьшему общему кратному длин периодов последовательностей, для которых модуль равен степеням простых сомножителей m.

a) Если $f(z)$, $a(z)$, $b(z)$ — это полиномы с целыми коэффициентами, то запишем $a(z) \equiv b(z)$ по модулям $f(z)$ и m, если $a(z) = b(z) + f(z)u(z) + mv(z)$ для некоторых полиномов $u(z)$ и $v(z)$ с целыми коэффициентами. Докажите, что имеет место следующее утверждение, когда $f(0) = 1$ и $p^e > 2$: если $z^\lambda \equiv 1$ по модулям $f(z)$ и p^e и $z^\lambda \not\equiv 1$ по модулям $f(z)$ и p^{e+1}, то $z^{p\lambda} \equiv 1$ по модулям $f(z)$ и p^{e+1} и $z^{p\lambda} \not\equiv 1$ по модулям $f(z)$ и p^{e+2}.

b) Пусть $f(z) = 1 - a_1 z - \cdots - a_k z^k$ и

$$G(z) = 1/f(z) = A_0 + A_1 z + A_2 z^2 + \cdots.$$

Пусть $\lambda(m)$ обозначает длину периода $\langle A_n \bmod m \rangle$. Докажите, что $\lambda(m)$ — наименьшее положительное λ, такое, что $z^\lambda \equiv 1$ по модулям $f(z)$ и m.

c) Дано: p — простое число, $p^e > 2$ и $\lambda(p^e) \neq \lambda(p^{e+1})$. Докажите, что $\lambda(p^{e+r}) = p^r \lambda(p^e)$ для всех $r \geq 0$. (Таким образом, чтобы найти длину периода последовательности $\langle A_n \bmod 2^e \rangle$, можно подсчитывать $\lambda(4)$, $\lambda(8)$, $\lambda(16)$, ..., пока не будет найдено наименьшее $e \geq 3$, такое, что $\lambda(2^e) \neq \lambda(4)$. Тогда длина периода будет определена по $\bmod 2^e$ для всех e. В упр. 4.6.3–26 объясняется, как вычислить X_n для больших n за $O(\log n)$ операций.)

d) Покажите, что любая последовательность целых чисел, которая удовлетворяет рекуррентному соотношению, сформулированному в начале этого упражнения, имеет производящую функцию $g(z)/f(z)$ для некоторого полинома $g(z)$ с целыми коэффициентами.

e) Дано, что полиномы $f(z)$ и $g(z)$ в п. (d) взаимно просты по модулю p (см. раздел 4.6.1). Докажите, что последовательность $\langle X_n \bmod p^e \rangle$ имеет ту же длину периода, что и специальная последовательность $\langle A_n \bmod p^e \rangle$ в (b). (Нельзя увеличить длину периода путем выбора *любых* $X_0, \ldots, X_{k-1}$, так как общая последовательность является линейной комбинацией "сдвигов" специальной последовательности.) [*Указание.* Из упр. 4.6.2–22 (лемма Хенселя) следует, что существуют полиномы, такие, что $a(z)f(z) + b(z)g(z) \equiv 1$ (по модулю p^e).]

▸ **12.** [*M28*] Найдите целые числа X_0, X_1, a, b и c, такие, что последовательность

$$X_{n+1} = (aX_n + bX_{n-1} + c) \bmod 2^e, \qquad n \geq 1,$$

имеет самый длинный период среди всех последовательностей этого типа. [*Указание.* Можно показать, что $X_{n+2} = ((a+1)X_{n+1} + (b-a)X_n - bX_{n-1}) \bmod 2^e$; см. упр. 11, (c).]

13. [*M20*] Пусть $\langle X_n \rangle$ и $\langle Y_n \rangle$ — последовательности целых чисел по $\bmod m$ с периодами длиной λ_1 и λ_2. Объединим их, положив $Z_n = (X_n + Y_n) \bmod m$. Покажите, что если λ_1 и λ_2 — взаимно простые числа, то последовательность $\langle Z_n \rangle$ имеет период длиной $\lambda_1 \lambda_2$.

14. [*M24*] Пусть X_n, Y_n, Z_n, λ_1 и λ_2 определены так же, как в предыдущем упражнении. Предположим, что разложение λ_1 на простые множители имеет вид $2^{e_2}3^{e_3}5^{e_5}\ldots$ и аналогично $\lambda_2 = 2^{f_2}3^{f_3}5^{f_5}\ldots$. Пусть $g_p = (\max(e_p, f_p)$, если $e_p \neq f_p$, в других случаях — 0) и пусть $\lambda_0 = 2^{g_2}3^{g_3}5^{g_5}\ldots$. Покажите, что длина периода λ' последовательности $\langle Z_n\rangle$ кратна λ_0 и является делителем $\lambda = \operatorname{lcm}(\lambda_1, \lambda_2)$. В частности, $\lambda' = \lambda$, если ($e_p \neq f_p$ или $e_p = f_p = 0$) для каждого простого p.

15. [*M27*] Пусть последовательность $\langle X_n\rangle$ в алгоритме M имеет длину периода λ_1 и все элементы в этом периоде различны. Пусть $q_n = \min\{r \mid r > 0$ и $\lfloor kY_{n-r}/m\rfloor = \lfloor kY_n/m\rfloor\}$. Предположим, что $q_n < \frac{1}{2}\lambda_1$ для всех $n \geq n_0$ и что последовательность $\langle q_n\rangle$ имеет длину периода λ_2. Пусть λ — наименьшее общее кратное λ_1 и λ_2. Докажите, что последовательность на выходе $\langle Z_n\rangle$, полученная с помощью алгоритма M, имеет длину периода λ.

▶ **16.** [*M28*] Пусть в методе (10) СОДЕРЖИМОЕ(A) равно $(a_1a_2\ldots a_k)_2$ в бинарной записи. Покажите, что последовательность, генерируемая младшими разрядами X_0, X_1, ..., удовлетворяет соотношению

$$X_n = (a_1X_{n-1} + a_2X_{n-2} + \cdots + a_kX_{n-k}) \bmod 2.$$

[Это можно рассматривать как другой способ определения последовательности, хотя связь между данным соотношением и программой (10), на первый взгляд, не заметна!]

17. [*M33*] (М. Х. Мартин (M. H. Martin), 1934.) Пусть m и k — положительные числа и пусть $X_1 = X_2 = \cdots = X_k = 0$. Для всех $n > 0$ положим X_{n+k} равным наибольшему неотрицательному числу $y < m$, такому, что k-мерная строка $(X_{n+1}, \ldots, X_{n+k-1}, y)$ еще не появилась в последовательности. Другими словами, $(X_{n+1}, \ldots, X_{n+k-1}, y)$ должна отличаться от $(X_{r+1}, \ldots, X_{r+k})$ для $0 \leq r < n$. При этом каждая возможная k-мерная строка встретится по крайней мере один раз в последовательности. В конце концов процесс закончится, когда будет достигнуто такое значение n, при котором строка $(X_{n+1}, \ldots, X_{n+k-1}, y)$ уже появлялась в последовательности для всех неотрицательных $y < m$. Например, если $m = k = 3$, такой последовательностью будет 000222122021121020120011101 00 и процесс закончится в этой точке. (a) Докажите, что, когда последовательность закончится, будут выполняться равенства $X_{n+1} = \cdots = X_{n+k-1} = 0$. (b) Докажите, что *каждая* k-мерная строка элементов $(a_1, a_2, \ldots, a_k)$, таких, что $0 \leq a_j < m$, появляется в последовательности. Поэтому последовательность окончится при $n = m^k$. [*Указание.* Докажите, что k-мерная строка $(a_1, \ldots, a_s, 0, \ldots, 0)$ появляется, когда $a_s \neq 0$, используя индукцию по s.] Заметим, что если сейчас определить $f(X_n, \ldots, X_{n+k-1}) = X_{n+k}$ для $1 \leq n \leq m^k$, полагая $X_{m^k+k} = 0$, то получится функция с максимальным возможным периодом.

18. [*M22*] Пусть $\langle X_n\rangle$ — это последовательность двоичных разрядов, генерируемая методом (10), с $k = 35$ и СОДЕРЖИМОЕ(A) $= (00000000000000000000000000000101)_2$.

Пусть U_n — бинарная дробь $(.X_{nk}X_{nk+1}\ldots X_{nk+k-1})_2$. Покажите, что эта последовательность $\langle U_n\rangle$ не удовлетворяет сериальному критерию пар (раздел 3.3.2B), когда $d = 8$.

19. [*M41*] Для каждого простого числа p из первого столбца табл. 2 раздела 4.5.4 найдите подходящие константы a_1 и a_2, как предлагается в разделе, чтобы длина периода (8) при $k = 2$ равнялась $p^2 - 1$. (Например, см. рекуррентное соотношение 3.3.4–(39).)

20. [*M40*] Вычислите подходящие константы для использования в качестве содержимого регистра A (СОДЕРЖИМОЕ(A)) в методе (10), имеющие приблизительно то же количество нулей, что и константы для $2 \leq k \leq 64$.

21. [*M35*] (Д. Рис (D. Rees).) В разделе объясняется, как найти функции f, такие, что последовательность (11) имеет длину периода $m^k - 1$, если m — простое число и не все $X_0, \ldots, X_{k-1}$ равны нулю. Покажите, что такие функции могут быть изменены для получения последовательности наподобие (11) с длиной периода m^k для *всех* целых m.

[*Указание.* Рассмотрите лемму 3.2.1.2Q, трюк из упр. 7 и последовательность вида $\langle pX_{2n} + X_{2n+1}\rangle$.]

▸ **22.** [*M24*] В разделе нет обширных обсуждений способов расширения линейной последовательности (8) до случая, когда m является простым числом. Докажите, что достаточно длинный период может быть получен, когда m "свободно от квадратов", т. е. является произведением различных простых чисел. (Из табл. 3.2.1.1–1 ясно, что $m = w \pm 1$ часто удовлетворяет этим предположениям. Многие из результатов, приведенных в настоящем разделе, могут поэтому быть распространены на случай, который в некоторой степени более удобен для вычислений.)

▸ **23.** [*20*] Рассмотрите последовательность $X_n = (X_{n-55} - X_{n-24}) \bmod m$ как альтернативу последовательности (7).

24. [*M20*] Пусть $0 < l < k$. Докажите, что последовательность двоичных разрядов, определенная рекуррентным соотношением $X_n = (X_{n-k+l} + X_{n-k}) \bmod 2$, имеет длину периода $2^k - 1$ всякий раз, когда такой же период имеет последовательность, определенная соотношением $Y_n = (Y_{n-l} + Y_{n-k}) \bmod 2$.

25. [*26*] Рассмотрите альтернативу для программы A, состоящую в том, что все 55 входов в таблице Y-в заменяются 55 раз требуемыми случайными числами.

26. [*M48*] (Дж. Ф. Рейзер (J. F. Reiser).) Пусть p — простое число и k — положительное число. Пусть заданы целые числа $a_1, \ldots, a_k$ и $x_1, \ldots, x_k$, пусть λ_α — период последовательности $\langle X_n\rangle$, заданной рекуррентным соотношением

$$X_n = x_n \bmod p^\alpha, \quad 0 \le n < k; \qquad X_n = (a_1 X_{n-1} + \cdots + a_k X_{n-k}) \bmod p^\alpha, \quad n \ge k,$$

и пусть N_α равно числу нулей, которые встречаются в периоде (числу индексов j, таких, что $\mu_\alpha \le j < \mu_\alpha + \lambda_\alpha$ и $X_j = 0$). Докажите или опровергните следующее утверждение: существует константа c (зависящая, возможно, от p, k и $a_1, \ldots, a_k$), такая, что $N_\alpha \le cp^{\alpha(k-2)/(k-1)}$ для всех α и всех $x_1, \ldots, x_k$.

[*Замечание.* Рейзер доказал, что если рекуррентная последовательность имеет максимальную длину периода по модулю p (т. е. если $\lambda_1 = p^k - 1$) и если утверждение имеет место, то k-мерное расхождение $\langle X_n\rangle$ будет равно $O(\alpha^k p^{-\alpha/(k-1)})$ при $\alpha \to \infty$. Таким образом, аддитивный генератор, подобный (7), был бы распределен в 55 измерениях, когда $m = 2^e$ и рассматривается полный период. (См. раздел 3.3.4, в котором определено понятие расхождения в k измерениях.) Утверждение является слабым условием, если $\langle X_n\rangle$ принимает каждое значение примерно одинаково часто и если $\lambda_\alpha = p^{\alpha-1}(p^k - 1)$. Величина $N_\alpha \approx (p^k - 1)/p$ не увеличивается, вообще говоря, когда α возрастает. Рейзер проверил это утверждение для $k = 3$. С другой стороны, он показал, что можно найти необычайно плохие (зависящие от α) начальные значения $x_1, \ldots, x_k$, такие, что $N_{2\alpha} \ge p^\alpha$, обеспечивающие $\lambda_\alpha = p^{\alpha-1}(p^k - 1)$, $k \ge 3$, α достаточно большое.]

27. [*M30*] Предположим, что алгоритм B применяется к последовательности $\langle X_n\rangle$ с длиной периода λ, где $\lambda \gg k$. Покажите, что для фиксированного k и всех достаточно больших λ последовательность на выходе будет периодичной с той же самой длиной периода λ, если $\langle X_n\rangle$ не является слишком случайной. [*Указание.* Найдите структуру последовательных значений $\lfloor kX_n/m\rfloor$, которая обеспечивает "синхронизацию" дальнейшего поведения алгоритма B.]

28. [*40*] (А. Дж. Вотерман (A. G. Waterman).) Исследуйте линейную конгруэнтную последовательность с m, равным квадрату или кубу длины компьютерного слова, в то время как a и c заданы с обычной точностью.

▸ **29.** [*40*] Найдите хороший метод вычисления функции $f(x_1, \ldots, x_k)$, определенной последовательностью Мартина (Martin) в упр. 17, если задана только строка $(x_1, \ldots, x_k)$ размерности k.

30. [*M37*] (P. П. Брент (R. P. Brent).) Пусть $f(x) = x^k - a_1x^{k-1} - \cdots - a_k$ — первообразный полином по модулю 2, и предположим, что $X_0, \ldots, X_{k-1}$ — целые числа, не все четные.

a) Докажите, что длина периода последовательности, заданной рекуррентным соотношением $X_n = (a_1X_{n-1} + \cdots + a_kX_{n-k}) \bmod 2^e$, равна $2^{e-1}(2^k - 1)$ для всех $e \geq 1$ тогда и только тогда, когда $f(x)^2 + f(-x)^2 \not\equiv 2f(x^2)$ и $f(x)^2 + f(-x)^2 \not\equiv 2(-1)^k f(-x^2)$ (по модулю 8). [*Указание.* Равенство $x^{2^k} \equiv -x$ (по модулям 4 и $f(x)$) справедливо тогда и только тогда, когда $f(x)^2 + f(-x)^2 \equiv 2f(x^2)$ (по модулю 8).]

b) Докажите, что это условие всегда выполняется, когда полином $f(x) = x^k \pm x^l \pm 1$ является первообразным полиномом по модулю 2 и $k > 2$.

31. [*M30*] (Дж. Марсалья (G. Marsaglia).) Какова длина периода последовательности (7′), когда $m = 2^e \geq 8$? Предположите, что не все $X_0, \ldots, X_{54} \equiv \pm 1$ (по модулю 8).

32. [*M21*] Каким рекуррентным соотношениям удовлетворяют элементы подпоследовательностей $\langle X_{2n} \rangle$ и $\langle X_{3n} \rangle$, когда $X_n = (X_{n-24} + X_{n-55}) \bmod m$?

▶ **33.** [*M23*] (a) Пусть $g_n(z) = X_{n+30} + X_{n+29}z + \cdots + X_nz^{30} + X_{n+54}z^{31} + \cdots + X_{n+31}z^{54}$, где X_n удовлетворяют рекуррентному соотношению Фибоначчи с запаздыванием (7). Найдите простое соотношение между $g_n(z)$ и $g_{n+t}(z)$. (b) Выразите X_{500} в терминах $X_0, \ldots, X_{54}$.

34. [*M25*] Докажите, что обратная рекуррентная последовательность (12) имеет период $p + 1$ тогда и только тогда, когда полином $f(x) = x^2 - cx - a$ обладает следующими двумя свойствами: (i) $x^{p+1} \bmod f(x)$ равняется отличной от нуля константе, если вычислять, используя полиномиальную арифметику по модулю p; (ii) $x^{(p+1)/q} \bmod f(x)$ имеет степень 1 для каждого простого q, делящего $p + 1$. [*Указание.* Рассмотрите степени матрицы $\left(\begin{smallmatrix} 0 & 1 \\ a & c \end{smallmatrix}\right)$.]

35. [*HM35*] Как много пар (a, c) удовлетворяют условиям упр. 34?

36. [*M25*] Докажите, что обратная конгруэнтная последовательность $X_{n+1} = (aX_n^{-1} + c) \bmod 2^e$, $X_0 = 1$, $e \geq 3$, имеет период длиной 2^{e-1} всякий раз, когда $a \bmod 4 = 1$ и $c \bmod 4 = 2$.

▶ **37.** [*HM32*] Пусть p — простое число, и предположим, что $X_{n+1} = (aX_n^{-1} + c) \bmod p$ определяет обратную конгруэнтную последовательность с периодом $p + 1$. К тому же пусть $0 \leq b_1 < \cdots < b_d \leq p$. Рассмотрим множество

$$V = \{(X_{n+b_1}, X_{n+b_2}, \ldots, X_{n+b_d}) \mid 0 \leq n \leq p \text{ и } X_{n+b_j} \neq \infty \text{ для } 1 \leq j \leq d\}$$

В нем содержится $p + 1 - d$ векторов; любые d из них лежат в некоторой $(d - 1)$-мерной гиперплоскости $H = \{(v_1, \ldots, v_d) \mid r_1v_1 + \cdots + r_dv_d \equiv r_0 \text{ (по модулю } p)\}$, где $(r_1, \ldots, r_d) \not\equiv (0, \ldots, 0)$. Докажите, что никакие $d + 1$ векторов из V не лежат в одной и той же гиперплоскости.

3.3. СТАТИСТИЧЕСКИЕ КРИТЕРИИ

НАША ОСНОВНАЯ ЦЕЛЬ — получить последовательность, которая ведет себя так, как будто она является случайной. Выше рассказывалось, как сделать период последовательности настолько длинным, что при практических применениях он никогда не будет повторяться. Это важный критерий, но он не дает гарантии, что последовательность будет использоваться в приложениях. Как решить, достаточно ли случайной будет последовательность?

Если дать наудачу выбранному человеку карандаш и бумагу и попросить его написать 100 десятичных цифр, то очень мало шансов, что будет получен удовлетворительный результат. Люди стремятся избегать действий, приводящих к результатам, которые кажутся неслучайными, таким, например, как появление пары равных смежных цифр (хотя приблизительно одна из 10 цифр должна равняться предыдущей). И, если показать тому же человеку таблицу настоящих случайных чисел, он, вероятно, скажет, что эти числа не случайны. Он заметит много кажущихся закономерностей.

В соответствии с высказыванием доктора И. Дж. Матрикса (I. J. Matrix), который цитировал Мартина Гарднера (Martin Gardner) в *Scientific American*, January, 1965, "Математики рассматривают десятичное разложение числа π как случайную последовательность, но современный специалист по магическим свойствам чисел найдет для себя множество интересных примеров". Матрикс указал, например, что первым повторяющимся двузначным числом в разложении числа π является 26, а второй раз оно появляется как раз посередине любопытных повторений пар чисел:

$$3.14159265358979323846264338327950. \qquad (1)$$

Составив список дюжины или более подобных свойств этих чисел, он заметил, что разложение числа π, если его правильно интерпретировать, может рассказать обо всей истории человечества!

Все мы замечаем закономерности в наших телефонных номерах, номерах водительских прав и т. д., чтобы их запомнить. Наша основная мысль состоит в том, что нельзя быть уверенным в том, что данная последовательность является случайной. Для этого нужно применять какой-нибудь беспристрастный критерий.

Теоретическая статистика предоставляет некоторые количественные меры случайности. Существует буквально бесконечное число критериев, которые можно использовать для проверки того, будет ли последовательность случайной. Обсудим критерии, с нашей точки зрения, наиболее полезные, наиболее поучительные и наиболее приспособленные к вычислениям на компьютерах.

Если критерии $T_1, T_2, \ldots, T_n$ подтверждают, что последовательность ведет себя случайным образом, это еще не *означает*, вообще говоря, что проверка с помощью T_{n+1}-го критерия будет успешной. Однако каждая успешная проверка дает все больше и больше уверенности в случайности последовательности. Обычно к последовательности применяется около полудюжины статистических критериев, и если она удовлетворяет этим критериям, то последовательность считается случайной (это презумпция невиновности до доказательства вины).

Каждую последовательность, которая будет широко использоваться, необходимо тщательно проверить. В следующих разделах объясняется, как правильно применять критерии. Различаются два вида критериев: *эмпирические критерии*, при использовании которых компьютер манипулирует группами чисел последовательности и вычисляет определенные статистики, и *теоретические критерии*, для которых характеристики последовательности определяются с помощью теоретико-числовых методов, основанных на рекуррентных правилах, которые используются для образования последовательности.

Если информации, содержащейся в этой книге, будет недостаточно, можете обратиться к книге Даррелла Хаффа за техническими указаниями (*How to Lie With Statistics* Darrell Huff (Norton, 1954)). Хафф Д. Б. (Huff, Darrell Burton)

3.3.1. Основные критерии проверки случайных наблюдений

А. Критерий "хи-квадрат". Критерий "хи-квадрат" (χ^2-критерий), возможно, самый известный из всех статистических критериев. Он является основным методом, используемым в сочетании с другими критериями. Прежде чем рассматривать идею в целом, проанализируем частный пример применения χ^2-критерия к бросанию игральной кости. Используем две "правильные" игральные кости (каждая из которых независимо допускает выпадение значений 1, 2, 3, 4, 5 или 6 с равной вероятностью). В следующей таблице дана вероятность получения определенной суммы s при одном бросании игральных костей.

$$\begin{array}{rccccccccccc} \text{Значение } s = & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ \text{Вероятность } p_s = & \frac{1}{36} & \frac{1}{18} & \frac{1}{12} & \frac{1}{9} & \frac{5}{36} & \frac{1}{6} & \frac{5}{36} & \frac{1}{9} & \frac{1}{12} & \frac{1}{18} & \frac{1}{36} \end{array} \qquad (1)$$

Например, величина 4 может быть получена тремя способами: $1+3$, $2+2$, $3+1$; это составляет $\frac{3}{36} = \frac{1}{12} = p_4$ из 36 возможных результатов.

Если бросать игральную кость n раз, то в среднем мы получим величину s примерно np_s раз. Например, при 144 бросаниях величина 4 выпадает около 12 раз. В следующей таблице показано, какие результаты *в действительности* получены при 144 бросаниях игральных костей.

$$\begin{array}{rccccccccccc} \text{Величина } s = & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\ \text{Наблюдаемое число, } Y_s = & 2 & 4 & 10 & 12 & 22 & 29 & 21 & 15 & 14 & 9 & 6 \\ \text{Ожидаемое число, } np_s = & 4 & 8 & 12 & 16 & 20 & 24 & 20 & 16 & 12 & 8 & 4 \end{array} \qquad (2)$$

Заметим, что во всех случаях наблюдаемое число отличалось от ожидаемого числа. Действительно, результаты случайного бросания игральной кости вряд ли всегда будут появляться *именно* с правильной частотой. Существует 36^{144} возможных последовательностей 144 бросаний и все они равновероятны. Одна из таких последовательностей состоит из всех двоек ("змеиные глаза"), и всякий, кто выбросил 144 змеиных глаза подряд, будет убежден, что кости утяжелены. Несмотря на это последовательность всех двоек является такой же вероятной, как и любая другая последовательность, если точно определить результат каждого бросания каждой игральной кости.

Принимая во внимание все сказанное, как проверить, утяжелена ли данная пара игральных костей? На этот вопрос нельзя дать ответ "да" или "нет", но

можно дать *вероятностный* ответ, т. е. сказать, насколько вероятно или не вероятно происшедшее событие.

В приведенном выше примере совершенно естественно рассмотреть квадраты разностей между наблюдаемыми числами Y_s и ожидаемыми числами np_s. Можно сложить их, получив

$$V = (Y_2 - np_2)^2 + (Y_3 - np_3)^2 + \cdots + (Y_{12} - np_{12})^2. \tag{3}$$

Плохой набор игральных костей привел бы к относительно большому значению V, а для данного значения V можно сказать следующее: "Чему равна вероятность таких больших значений V, если использовать «правильные» игральные кости?". Если эта вероятность очень мала, скажем, $\frac{1}{100}$, мы будем знать, что только около одного раза из ста "правильные" игральные кости будут давать результаты, настолько далекие от ожидаемых значений, что возникнут определенные основания для подозрений. (Помним, тем не менее, что те же самые *хорошие* игральные кости будут давать такое большое значение V приблизительно в одном случае из ста, так что предусмотрительным лицам придется повторять эксперимент, когда большие значения V являются частыми.)

В статистике V в (3) слагаемым $(Y_7 - np_7)^2$ и $(Y_2 - np_2)^2$ приписываются равные веса, несмотря на то что $(Y_7 - np_7)^2$, вероятно, будет больше, чем $(Y_2 - np_2)^2$, так как 7 появляются приблизительно в шесть раз чаще, чем 2. Оказывается, что "правильная" статистика, по крайней мере статистика, которая, как доказано, наиболее важна, будет приписывать $(Y_7 - np_7)^2$ только $\frac{1}{6}$ веса, приписываемого $(Y_2 - np_2)^2$, и необходимо изменить (3) следующим образом:

$$V = \frac{(Y_2 - np_2)^2}{np_2} + \frac{(Y_3 - np_3)^2}{np_3} + \cdots + \frac{(Y_{12} - np_{12})^2}{np_{12}}. \tag{4}$$

Эта статистика называется статистикой "хи-квадрат" наблюдаемых значений $Y_2, \ldots, Y_{12}$ при бросании игральных костей. Для данных из таблицы (2) получим, что

$$V = \frac{(2-4)^2}{4} + \frac{(4-8)^2}{8} + \cdots + \frac{(9-8)^2}{8} + \frac{(6-4)^2}{4} = 7\frac{7}{48}. \tag{5}$$

Теперь возникает важный вопрос: "Будет ли $7\frac{7}{48}$ невероятно большим значением для V при наших предположениях?". Прежде чем ответить на него, рассмотрим, как применяется метод "хи-квадрат" в общих ситуациях.

Предположим, что каждое наблюдение может принадлежать одной из k категорий. Проводим n *независимых наблюдений*. Это означает, что исход одного наблюдения абсолютно не влияет на исход других наблюдений. Пусть p_s — вероятность того, что каждое наблюдение относится к категории s, и пусть Y_s — число наблюдений, которые действительно *относятся* к категории s. Образуем статистику

$$V = \sum_{s=1}^{k} \frac{(Y_s - np_s)^2}{np_s}. \tag{6}$$

В примере, приведенном выше, существует одиннадцать возможных исходов каждого бросания игральных костей, т. е. $k = 11$. (Формула (6) немного изменила обозначения формулы (4), так как нумеруются возможности 1–k вместо 2–12.)

Возводя в квадрат $(Y_s - np_s)^2 = Y_s^2 - 2np_sY_s + n^2p_s^2$ в (6) и учитывая тот факт, что

$$\begin{aligned} Y_1 + Y_2 + \cdots + Y_k &= n, \\ p_1 + p_2 + \cdots + p_k &= 1, \end{aligned} \tag{7}$$

получаем формулу

$$V = \frac{1}{n}\sum_{s=1}^{k}\left(\frac{Y_s^2}{p_s}\right) - n, \tag{8}$$

которая часто упрощает вычисление V.

Возвратимся к вопросу "Чему равно приемлемое значение V?". Его можно определить с помощью таких таблиц, как табл. 1, которая дает значения "χ^2-распределения с ν степенями свободы" для различных значений ν. Используем строку таблицы с $\nu = k-1$, так как *число "степеней свободы" равно $k-1$, что на единицу меньше, чем число категорий.* (Интуитивно это означает, что $Y_1, Y_2, \ldots, Y_k$ не являются полностью независимыми, так как формула (7) показывает, что Y_k может быть вычислено, если $Y_1, \ldots, Y_{k-1}$ известны. Поэтому нужно считать, что число степеней свободы равно $k-1$. Эти аргументы не строги, но они подтверждаются теоретически.)

Если в таблице выбрать число x, стоящее на ν-й строке и в столбце p, то "вероятность того, что значение V в (8) будет меньше либо равно x, приближенно равна p, если n достаточно велико". Например, 95-процентное значение в строке 10 равно 18.31; значения, такие, что $V > 18.31$, будут появляться приблизительно в 5% случаев.

Допустим, что наш эксперимент с бросанием игральных костей был промоделирован на компьютере с помощью некоторой последовательности чисел, предположительно случайных, со следующими результатами.

Значение $s =$	2	3	4	5	6	7	8	9	10	11	12
Эксперимент 1, $Y_s =$	4	10	10	13	20	18	18	11	13	14	13
Эксперимент 2, $Y_s =$	3	7	11	15	19	24	21	17	13	9	5

(9)

Можно подсчитать χ^2-статистику в первом случае, $V_1 = 29\frac{59}{120}$, и во втором случае, $V_2 = 1\frac{17}{120}$. Сравнивая эти величины со значениями таблицы при 10 степенях свободы, мы видим, что V_1 *намного больше*; V будет больше 23.21 только в 1% случаев! (Используя более полные таблицы, можно обнаружить, что V будет так же велико, как и V_1, только в 0.1% случаев.) Поэтому эксперимент 1 демонстрирует значительное отклонение от случайного поведения. (Автор часто употребляет термин "отклонение от случайного поведения" и подобные ему термины в том смысле, что наблюдаемые реализации случайной величины маловероятны при предполагаемом распределении этой случайной величины. — *Прим. ред.*)

С другой стороны, V_2 достаточно мало́, так как наблюдаемые значения Y_s в эксперименте 2 достаточно близки к ожидаемым значениям np_s в (2). Из χ^2-таблицы на самом деле ясно, что V_2 *слишком мало́*. Наблюдаемые значения настолько близки к ожидаемым, что нельзя рассматривать результаты как случайные! (В самом деле, если воспользоваться другими таблицами, можно увидеть, что такие маленькие значения V встречаются только в 0.03% случаев, когда имеем χ^2-распределение с 10 степенями свободы.) Наконец, значение $V = 7\frac{7}{48}$, полученное в (5), также

Таблица 1

НЕКОТОРЫЕ ПРОЦЕНТНЫЕ ТОЧКИ χ^2-РАСПРЕДЕЛЕНИЯ

	$p = 1\%$	$p = 5\%$	$p = 25\%$	$p = 50\%$	$p = 75\%$	$p = 95\%$	$p = 99\%$
$\nu = 1$	0.00016	0.00393	0.1015	0.4549	1.323	3.841	6.635
$\nu = 2$	0.02010	0.1026	0.5754	1.386	2.773	5.991	9.210
$\nu = 3$	0.1148	0.3518	1.213	2.366	4.108	7.815	11.34
$\nu = 4$	0.2971	0.7107	1.923	3.357	5.385	9.488	13.28
$\nu = 5$	0.5543	1.1455	2.675	4.351	6.626	11.07	15.09
$\nu = 6$	0.8721	1.635	3.455	5.348	7.841	12.59	16.81
$\nu = 7$	1.239	2.167	4.255	6.346	9.037	14.07	18.48
$\nu = 8$	1.646	2.733	5.071	7.344	10.22	15.51	20.09
$\nu = 9$	2.088	3.325	5.899	8.343	11.39	16.92	21.67
$\nu = 10$	2.558	3.940	6.737	9.342	12.55	18.31	23.21
$\nu = 11$	3.053	4.575	7.584	10.34	13.70	19.68	24.72
$\nu = 12$	3.571	5.226	8.438	11.34	14.85	21.03	26.22
$\nu = 15$	5.229	7.261	11.04	14.34	18.25	25.00	30.58
$\nu = 20$	8.260	10.85	15.45	19.34	23.83	31.41	37.57
$\nu = 30$	14.95	18.49	24.48	29.34	34.80	43.77	50.89
$\nu = 50$	29.71	34.76	42.94	49.33	56.33	67.50	76.15
$\nu > 30$	$\nu + \sqrt{2\nu}x_p + \frac{2}{3}x_p^2 - \frac{2}{3} + O(1/\sqrt{\nu})$						
$x_p =$	-2.33	-1.64	$-.674$	0.00	0.674	1.64	2.33

Другие значения можно найти в книге *Handbook of Mathematical Functions*, вышедшей под редакцией М. Абрамовича (M. Abramowitz) и И. А. Стегун (I. A. Stegun) (Washington, D.C.: U.S. Government Printing Office, 1964); табл. 26.8. См. также (22) и упр. 16.

может быть проверено по табл. 1. Оно находится между 25- и 50-процентной точками, поэтому рассматривать это значение как значимо большое либо значимо малое нельзя. Таким образом, наблюдения в (2) являются удовлетворительно случайными по отношению к этому критерию. (Имеется в виду, что данные не опровергают гипотезу о распределении этой случайной величины. — *Прим. ред.*) В некоторой степени замечательно, что для использования таблиц не имеет значения, чему равны n и вероятность p_s. Только число $\nu = k-1$ влияет на результаты. Следует отметить, однако, что значения табл. 1 — это только приближенные значения: *дело в том, что в ней приведены значения χ^2-распределения, которое является предельным распределением случайной величины V в формуле* (6). Поэтому табличные значения близки к реальным только при больших n. Насколько большими должны быть n? Эмпирическое правило гласит: нужно взять n настолько большим, чтобы все значения величин np_s были больше или равны пяти. Однако лучше брать существенно

большие n, чтобы получить более надежный критерий. В приведенном выше примере $n = 144$, np_2 равнялось только 4 и эмпирическое правило было нарушено. Это объясняется только тем, что автору просто надоело бросать игральные кости; это привело к тому, что значения табл. 1 оказались менее подходящими. Эксперимент, проведенный на компьютере при $n = 1000$ или 10 000, или даже 100 000, был бы намного лучше рассмотренного нами. Мы могли также объединить данные при $s = 2$ и $s = 12$; этот критерий имел бы только 9 степеней свободы, но аппроксимация χ^2 была бы более точной.

Можно пояснить, насколько груба аппроксимация, если рассмотреть случай только двух категорий, имеющих вероятности p_1 и p_2. Предположим, $p_1 = \frac{1}{4}$ и $p_2 = \frac{3}{4}$. В соответствии со сформулированным эмпирическим правилом необходимо провести более двадцати наблюдений, $n \geq 20$, чтобы иметь удовлетворительную точность. Давайте это проверим. Когда $n = 20$, возможные значения V будут такими: $(Y_1 - 5)^2/5 + (5 - Y_1)^2/15 = \frac{4}{15}r^2$ для $-5 \leq r \leq 15$. Теперь посмотрим, насколько точно в первой строке ($\nu = 1$) табл. 1 описывается распределение V. χ^2-распределение непрерывно, в то время как распределение V имеет довольно большие скачки, поэтому нужно сделать несколько замечаний, прежде чем представить точное распределение. Если различные возможные исходы эксперимента приводят к величинам $V_0 \leq V_1 \leq \cdots \leq V_n$ с соответствующими вероятностями $\pi_0, \pi_1, \ldots, \pi_n$, то предположим, что заданная вероятность p попадает в интервал $\pi_0 + \cdots + \pi_{j-1} \leq p < \pi_0 + \cdots + \pi_{j-1} + \pi_j$. Найдем такую "процентную точку" x, где V меньше x с вероятностью $\leq p$ и V больше x с вероятностью $\leq 1 - p$. Нетрудно видеть, что существует только одно такое число, а именно — $x = V_j$. В нашем примере для $n = 20$ и $\nu = 1$ оказывается, что процентные точки для точного распределения, соответственно аппроксимации в табл. 1 для $p = 1\%, 5\%, 25\%, 50\%, 75\%, 95\%$ и 99%, равны

$$0, \quad 0, \quad .27, \quad .27, \quad 1.07, \quad 4.27, \quad 6.67$$

(с точностью до двух десятичных знаков). Например, процентная точка для $p = 95\%$ равна 4.27, тогда как приближенное значение в табл. 1 равно 3.841, что существенно меньше. Поэтому, если пользоваться таблицей, следует отнести значение $V = 4.27$ за 95%-й уровень, на самом же деле вероятность того, что $V \geq 4.27$, больше 6.5%. Когда $n = 21$, ситуация меняется мало, поскольку средние значения $np_1 = 5.25$ и $np_2 = 15.75$ могут никогда не достигаться точно. Процентные точки для $n = 21$ равны

$$.02, \quad .02, \quad .14, \quad .40, \quad 1.29, \quad 3.57, \quad 5.73.$$

Можно было бы ожидать, что значения из табл. 1 дадут лучшее приближение при $n = 50$, но соответствующая таблица, оказывается, в некоторых аспектах еще больше отличается от табл. 1, чем при $n = 20$:

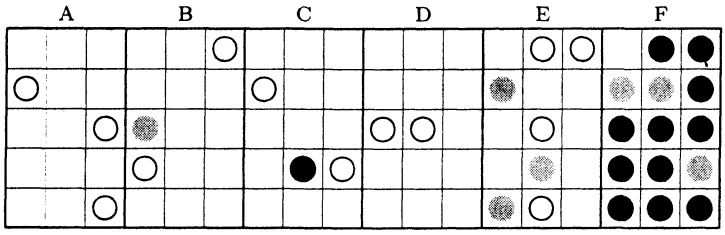
$$.03, \quad .03, \quad .03, \quad .67, \quad 1.31, \quad 3.23, \quad 6.$$

Приведем значения при $n = 300$:

$$0, \quad 0, \quad .07, \quad .44, \quad 1.44, \quad 4, \quad 6.42.$$

Даже в этом случае, когда $np_s \geq 75$ для каждой категории, значения в табл. 1 хороши только относительно одной значащей цифры.

Вопрос о правильном выборе n достаточно сложен. Если игральные кости действительно несимметричны, то это будет проявляться все больше и больше при



Диапазон V	Указание	Обозначение
0–1%, 99–100%	Отбросить	●
1–5% , 95–99%	Подозрительный	●
5–10%, 90–95%	Почти подозрительный	○

Рис. 2. Указания "значимости" отклонения χ^2-критерия при $h = 90$ (см. также рис. 5).

возрастании n (см. упр. 12). Но при больших значениях n имеет место тенденция к сглаживанию *локального* неслучайного поведения, когда блоки чисел со строгим смещением следуют за блоками чисел с противоположным смещением. При реальном бросании игральных костей сглаживания локального неслучайного поведения можно не опасаться, так как одни и те же игральные кости используются во время всего эксперимента, но последовательность случайных чисел, генерируемых компьютером, может очень часто демонстрировать такие аномалии. Возможно, χ^2-критерий следовало бы применять для нескольких разных значений n. Во всяком случае, значения n должны быть по возможности большими.

Теперь можно окончательно описать χ^2-критерий следующим образом. Выполняется достаточно большое число n независимых наблюдений. (Важно избегать использования χ^2-критерия при зависимых наблюдениях. См., например, упр. 10, в котором рассмотрен случай, когда одна половина наблюдений зависит от другой.) Подсчитываем число наблюдений, относящихся к каждой из k категорий, и величину V, приведенную в формулах (6) и (8). Затем V сравниваем с числами из табл. 1 при $\nu = k - 1$. Если V меньше 1%-й точки или больше 99%-й точки, отбрасываем эти числа как недостаточно случайные. (Если быть более точными, то отбрасываем следующую гипотезу: вероятности того, что наблюдения относятся к категории s, равны p_s. — *Прим. ред.*) Если V лежит между 1%- и 5%-й точками или между 95%- и 99%-й точками, то эти числа "подозрительны"; если (интерполируя таблицу) V лежит между 5%- и 10%-й точками или 90%- и 95%-й точками, числа можно считать "почти подозрительными". Проверка по χ^2-критерию часто производится три раза (и более) с разными данными. Если по крайней мере два из трех результатов оказываются подозрительными, то числа рассматриваются как недостаточно случайные.

Например, на рис. 2 схематично показаны результаты применения пяти различных типов χ^2-критерия для каждой из шести последовательностей случайных чисел. Каждой проверке подвергались три различных блока чисел последовательности. Генератор A — это метод Мак-Ларена-Марсальи (MacLaren-Marsaglia) (алгоритм 3.2.2M, примененный к последовательности в 3.2.2–(13)). Генератор E — это метод

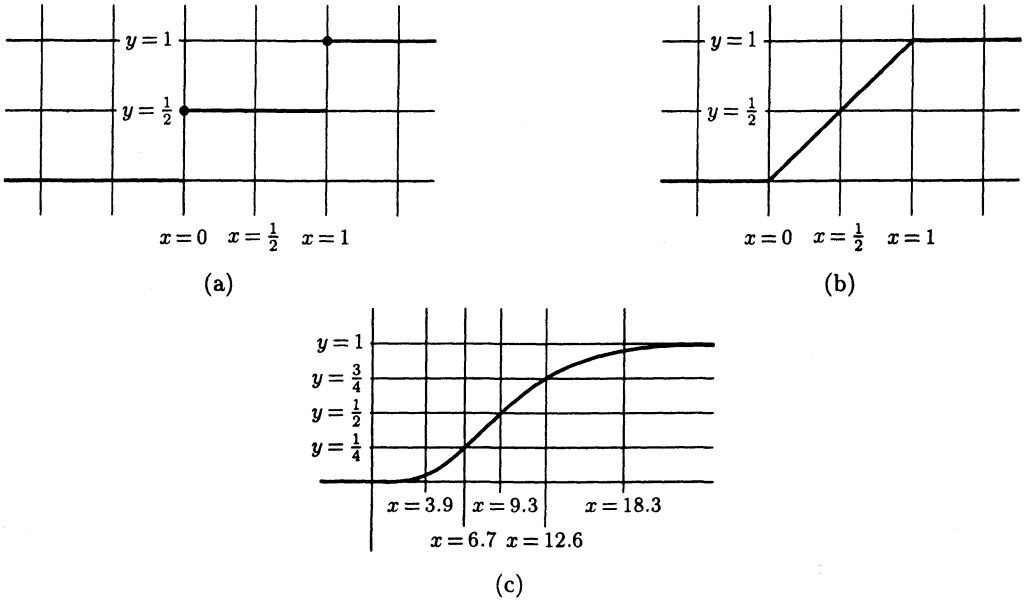


Рис. 3. Примеры функций распределения.

Фибоначчи (Fibonacci), 3.2.2–(5), а другие генераторы — это линейные конгруэнтные последовательности со следующими параметрами.

Генератор B: $X_0 = 0, \quad a = 3141592653, \quad c = 2718281829, \quad m = 2^{35}.$

Генератор C: $X_0 = 0, \quad a = 2^7 + 1, \quad c = 1, \quad m = 2^{35}.$

Генератор D: $X_0 = 47594118, \quad a = 23, \quad c = 0, \quad m = 10^8 + 1.$

Генератор F: $X_0 = 314159265, \quad a = 2^{18} + 1, \quad c = 1, \quad m = 2^{35}.$

Из рис. 2 заключаем, что (как следует из результатов проверки) генераторы A, B, D удовлетворительны, генератор C находится на границе и его следовало бы отбросить, генераторы E и F, безусловно, неудовлетворительны. Генератор F имеет, конечно, низкий потенциал; генераторы C и D уже обсуждались в литературе, но их множители слишком малы. (Генератор D — оригинальный мультипликативный генератор, предложенный Лехмером (Lehmer) в 1948 году; генератор C — оригинальный линейный конгруэнтный генератор с $c \neq 0$, предложенный Ротенбергом (Rotenberg) в 1960 году.)

Вместо терминов "подозрительный", "почти подозрительный" и т. д. для описания результатов применения χ^2-критерия можно, *кстати*, использовать процедуру, обсуждаемую ниже в этом разделе.

B. Критерий Колмогорова-Смирнова. Как мы уже видели, χ^2-критерий применяется в ситуациях, когда наблюдения могут относиться только к конечному числу категорий. Однако совершенно небесполезно рассматривать случайные величины, которые принимают бесконечное множество значений, такие как случайные дроби (случайные действительные числа между 0 и 1). Хотя только конечное множество

действительных чисел может быть представлено в компьютере, мы хотим, чтобы они вели себя подобно реальным числам в интервале $[0\,..\,1)$.

В теории вероятностей и математической статистике одни и те же обозначения используются, когда случайная величина принимает конечное либо бесконечное число значений. Чтобы задать распределение значений случайной величины X, следует сделать это в терминах *функции распределения* $F(x)$, где

$$F(x) = \Pr(X \le x) = \text{вероятности, что } (X \le x).$$

На рис. 3 приведены три примера. Сначала (рис. 3, (a)) представлена функция распределения *случайного двоичного разряда*, когда X принимает только значения 0 и 1, каждое с вероятностью $\frac{1}{2}$. Затем (рис. 3, (b)) показана функция распределения *равномерно распределенных случайных действительных чисел*, лежащих между 0 и 1. Здесь вероятность того, что $X \le x$, просто равна x, когда $0 \le x \le 1$. Например, вероятность, что $X \le \frac{2}{3}$, равна, естественно, $\frac{2}{3}$. И наконец, на рис. 3, (c) представлено предельное распределение случайной величины V в χ^2-критерии (здесь приведен случай с 10 степенями свободы). Это распределение в другом виде приводилось в табл. 1. Заметим, что $F(x)$ всегда возрастает от 0 до 1, когда x возрастает от $-\infty$ до $+\infty$.

Если выполнить n независимых наблюдений случайной величины X и получить значения X_1, X_2, ..., X_n, то можно будет построить *эмпирическую функцию распределения* $F_n(x)$, где

$$F_n(x) = \frac{\text{число } X_1, X_2, \ldots, X_n, \text{ таких, что они} \le x}{n}. \tag{10}$$

На рис. 4 показаны три эмпирические функции распределения (они изображены зигзагообразными линиями, хотя, строго говоря, вертикальные линии не являются частью графика $F_n(x)$). На них наложены графики предполагаемых настоящих функций распределения $F(x)$. Когда n становится больше, $F_n(x)$ более точно приближает $F(x)$.

Критерий Колмогорова-Смирнова (КС-критерий) может использоваться, когда функция $F(x)$ непрерывна (не имеет скачков). Он основан на *разности между* $F(x)$ *и* $F_n(x)$. Плохой источник случайных чисел (автор имеет в виду источник случайных чисел, распределение которых не соответствует требуемому. — *Прим. ред.*) будет давать такую эмпирическую функцию, которая плохо приближает предполагаемую функцию распределения $F(x)$. На рис. 4, (b) приведен пример случайной величины, для которой почти все X_i расположены слишком высоко, а это приводит к тому, что эмпирическая функция распределения находится слишком низко. На рис. 4, (c) приведен еще худший пример. Ясно, что такое большое расхождение между $F_n(x)$ и $F(x)$ совершенно невероятно, и КС-критерий может только показать, насколько именно.

Чтобы построить КС-критерий, образуем такие статистики:

$$\begin{aligned} K_n^+ &= \sqrt{n} \max_{-\infty<x<+\infty} \big(F_n(x) - F(x)\big); \\ K_n^- &= \sqrt{n} \max_{-\infty<x<+\infty} \big(F(x) - F_n(x)\big). \end{aligned} \tag{11}$$

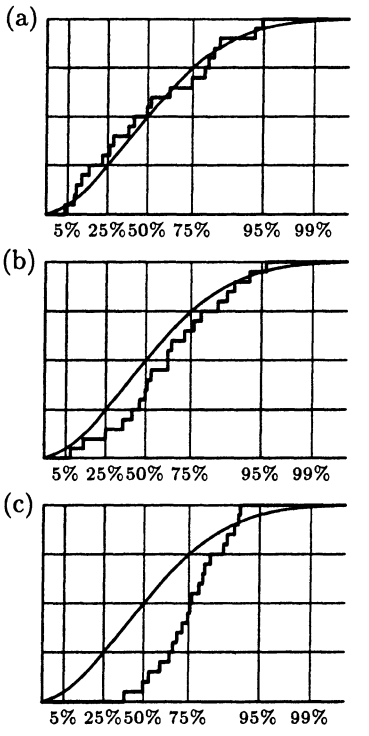


Рис. 4. Примеры эмпирических распределений.

Здесь K_n^+ определяет наибольшее из отклонений, когда F_n больше F, а K_n^- — F_n меньше F. Эти статистики для примеров, приведенных на рис. 4, таковы.

	Рис. 4, (a)	Рис. 4, (b)	Рис. 4, (c)	
K_{20}^+	0.492	0.134	0.313	(12)
K_{20}^-	0.536	1.027	2.101	

(*Замечание.* Множитель $\sqrt{n}$, который появляется в формуле (11), на первый взгляд, сбивает с толку. В упр. 6 показано, что для фиксированного x стандартное отклонение $F_n(x)$ пропорционально $1/\sqrt{n}$; следовательно, множитель $\sqrt{n}$ увеличивает статистики K_n^+ и K_n^- таким образом, что это стандартное отклонение не зависит от n.)

Точно так же, как и для χ^2-критерия, можно сравнить значения K_n^+ и K_n^- с процентной таблицей и определить, будут ли они значимо выше или ниже. Табл. 2 можно использовать одновременно для K_n^+ и K_n^-. Например, вероятность, что $K_{20}^- \le 0.7975$, равна 75% (автор имеет в виду здесь и далее, что вероятность равна 0.75. — *Прим. ред.*). Отличие от χ^2-критерия состоит в том, что в таблице представлены *не* просто приближенные значения, которые справедливы для достаточно больших n; табл. 2 дает точные значения (за исключением, конечно, ошибки округления), и КС-критерий может надежно использоваться для любого значения n.

Формулы (11) в том виде, в котором они записаны, не совсем подходят для вычислений на компьютере, так как формально нужно находить максимум по бесконечному множеству значений x. Но, учитывая тот факт, что $F(x)$ возрастает, и

Таблица 2

НЕКОТОРЫЕ ПРОЦЕНТНЫЕ ТОЧКИ РАСПРЕДЕЛЕНИЙ K_n^+ И K_n^-

	$p = 1\%$	$p = 5\%$	$p = 25\%$	$p = 50\%$	$p = 75\%$	$p = 95\%$	$p = 99\%$
$n = 1$	0.01000	0.05000	0.2500	0.5000	0.7500	0.9500	0.9900
$n = 2$	0.01400	0.06749	0.2929	0.5176	0.7071	1.0980	1.2728
$n = 3$	0.01699	0.07919	0.3112	0.5147	0.7539	1.1017	1.3589
$n = 4$	0.01943	0.08789	0.3202	0.5110	0.7642	1.1304	1.3777
$n = 5$	0.02152	0.09471	0.3249	0.5245	0.7674	1.1392	1.4024
$n = 6$	0.02336	0.1002	0.3272	0.5319	0.7703	1.1463	1.4144
$n = 7$	0.02501	0.1048	0.3280	0.5364	0.7755	1.1537	1.4246
$n = 8$	0.02650	0.1086	0.3280	0.5392	0.7797	1.1586	1.4327
$n = 9$	0.02786	0.1119	0.3274	0.5411	0.7825	1.1624	1.4388
$n = 10$	0.02912	0.1147	0.3297	0.5426	0.7845	1.1658	1.4440
$n = 11$	0.03028	0.1172	0.3330	0.5439	0.7863	1.1688	1.4484
$n = 12$	0.03137	0.1193	0.3357	0.5453	0.7880	1.1714	1.4521
$n = 15$	0.03424	0.1244	0.3412	0.5500	0.7926	1.1773	1.4606
$n = 20$	0.03807	0.1298	0.3461	0.5547	0.7975	1.1839	1.4698
$n = 30$	0.04354	0.1351	0.3509	0.5605	0.8036	1.1916	1.4801
$n > 30$	$y_p - \frac{1}{6}n^{-1/2} + O(1/n)$, где $y_p^2 = \frac{1}{2}\ln(1/(1-p))$						
$y_p =$	0.07089	0.1601	0.3793	0.5887	0,8326	1.2239	1.5174

Для расширения этой таблицы воспользуйтесь формулами (25) и (26), а также ответом к упр. 20.

то, что $F_n(x)$ возрастают только в конечном множестве точек, можно предложить простую процедуру для подсчета статистик K_n^+ и K_n^-.

Шаг 1. Получить независимые наблюдения $X_1, X_2, \ldots, X_n$.

Шаг 2. Упорядочить наблюдения так, чтобы они располагались в порядке возрастания (построить вариационный ряд): $X_1 \le X_2 \le \cdots \le X_n$. (Эффективный алгоритм сортировки приведен в главе 5. Но в этом случае сортировки можно избежать, как показано в упр. 23.)

Шаг 3. Нужные статистики сейчас задаются формулами

$$K_n^+ = \sqrt{n} \max_{1\le j\le n}\left(\frac{j}{n} - F(X_j)\right);$$
$$K_n^- = \sqrt{n} \max_{1\le j\le n}\left(F(X_j) - \frac{j-1}{n}\right). \tag{13}$$

Выбрать подходящее число наблюдений n немного легче для этого критерия, чем для χ^2-критерия, хотя некоторые из рассуждений похожи. Если случайные

величины X_j действительно имеют вероятностное распределение $G(x)$, в то время как предполагается, что они имеют распределение, определяемое функцией $F(x)$, n должно быть сравнительно большим для того, чтобы отбросить гипотезу о равенстве $G(x) = F(x)$. Для этого n должно быть настолько большим, чтобы ожидаемые эмпирические функции распределения $G_n(x)$ и $F_n(x)$ заметно различались. С другой стороны, большие значения n приводят к усреднению локального неслучайного поведения, и такое нежелательное поведение приводит к значительным неприятностям в большинстве компьютерных применений случайных чисел; это может служить причиной для выбора *небольших* значений n. Хороший компромисс будет достигнут, если взять n равным, скажем, 1 000 и вычислить достаточно много K^+_{1000} для различных частей случайной последовательности. Тем самым получим значения

$$K^+_{1000}(1), \qquad K^+_{1000}(2), \qquad \ldots, \qquad K^+_{1000}(r). \tag{14}$$

Затем можно применить КС-критерий *снова* к *этим* результатам. Пусть сейчас $F(x)$ — это функция распределения случайной величины K^+_{1000}. Найдем эмпирическую функцию распределения $F_r(x)$ для наблюдений случайной величины в (14). К счастью, функция $F(x)$ в этом случае имеет очень простой вид. Для больших n, таких, как $n = 1000$, функция распределения K^+_n примерно равна

$$F_\infty(x) = 1 - e^{-2x^2}, \qquad x \geq 0. \tag{15}$$

Данное замечание применимо к K^-_n, поскольку K^+_n и K^-_n имеют одно и то же ожидаемое поведение. *Можно использовать КС-критерий следующим образом: сначала критерий применяется несколько раз при умеренно больших n, а затем наблюдения объединяют и снова применяют КС-критерий. Это позволяет обнаружить как локальные, так и глобальные отклонения от гипотетического распределения.*

Например, автор провел следующий простой эксперимент во время написания этой главы. Критерий "максимум-5", описанный в следующем разделе, был применен к множеству 1 000 равномерно распределенных чисел. Было выполнено 200 наблюдений X_1, X_2, ..., X_{200}, которые, как предполагалось, имели функцию распределения $F(x) = x^5$ при $0 \leq x \leq 1$. Наблюдения были разделены на 20 групп по 10 наблюдений в каждой, и статистика K^+_{10} была вычислена для каждой группы. По 20 значениям K^+_{10}, полученным таким образом, построена эмпирическая функция, показанная на рис. 4. Гладкая кривая, изображенная на каждом из рис. 4, — это истинная функция распределения статистики K^+_{10}. На рис. 4, (a) изображено эмпирическое распределение K^+_{10}, полученное из последовательности

$$Y_{n+1} = (3141592653Y_n + 2718281829) \bmod 2^{35}, \qquad U_n = Y_n/2^{35}$$

и это распределение удовлетворительно приближает истинную функцию распределения, т. е. последовательность можно считать случайной. На рис. 4, (b) приведена такая же эмпирическая функция распределения, построенная по последовательности Фибоначчи. Эта последовательность имеет *глобально* неслучайное поведение, т. е. можно показать, используя критерий "максимум-5", что наблюдения X_n не имеют на самом деле распределения $F(x) = x^5$. На рис. 4, (c) изображена эмпирическая функция распределения, полученная с использованием заведомо плохой, с малым потенциалом, линейной конгруэнтной последовательности $Y_{n+1} = \big((2^{18} + 1)Y_n + 1\big) \bmod 2^{35}, U_n = Y_n/2^{35}$. Результаты применения КС-критерия к этой

эмпирической функции приведены в (12). Используя табл. 2 для $n = 20$, получаем, что значения K_{20}^+ и K_{20}^- для рис. 4, (b) почти подозрительны (они лежат около 5- и 88-процентного уровней), но не совсем плохи для того, чтобы отбросить их полностью. Значение K_{20}^- для рис. 4, (c), конечно, не подходит, поэтому критерий "максимум-5" показывает явную несостоятельность этого генератора случайных чисел.

Можно было бы ожидать, что КС-критерию в этом эксперименте труднее замечать "глобальные неслучайности", чем локальные, так как основные наблюдения для рис. 4 были сделаны только по 10 раз. Если взять 20 групп по 1 000 чисел в каждой, то на рис. 4, (b) будет показано намного более значимое отклонение. Чтобы проиллюстрировать это, *единственный* КС-критерий был применен ко всем 200 наблюдениям, которые привели к появлению рис. 4. Были получены следующие результаты.

	Рис. 4, (a)	Рис. 4, (b)	Рис. 4, (c)
K_{200}^+	0.477	1.537	2.819
K_{200}^-	0.817	0.194	0.058

(16)

"Глобальная неслучайность" генератора Фибоначчи здесь проявляется вполне определенно.

Теперь можно окончательно описать критерий Колмогорова-Смирнова. Дано *n независимых наблюдений* $X_1, \ldots, X_n$ из некоторого распределения, заданного *непрерывной* функцией распределения $F(x)$. Иначе говоря, $F(x)$ должна быть функцией, похожей на функции, которые показаны на рис. 3, (b) и 3, (c), и не имеющей таких скачков, как скачки функции на рис. 3, (a). Затем процедура, описанная перед формулой (13), применяется к этим наблюдениям, и получаются статистики K_n^+ и K_n^-. Они должны быть распределены в соответствии с табл. 2.

Сравним критерии КС и χ^2. Вначале заметим, что критерий КС можно использовать совместно с χ^2-критерием, чтобы получить *лучшую* процедуру, чем метод, который, между прочим, упоминался при окончательном описании χ^2-критерия. (Это означает, что существует лучший способ, чем выполнение трех проверок и определение количества "подозрительных".) Предположим, что уже сделано, скажем, 10 независимых χ^2-проверок различных частей случайной последовательности и получены значения $V_1, V_2, \ldots, V_{10}$. Это не совсем хороший метод определения, сколько значений V подозрительны в большей или меньшей степени. Данная процедура будет работать в крайних случаях, и очень большие или очень малые значения могут означать, что последовательность имеет также много локальных "неслучайностей". Но лучший общий метод состоит в составлении эмпирической функции этих 10 значений и ее сравнении с истинным распределением, полученным из табл. 1. Эмпирическое распределение дает ясную картину результатов использования критерия χ^2. И действительно статистики K_{10}^+ и K_{10}^- могут быть определены из эмпирических χ^2-значений, таким образом указывая на успех или неудачу проверки. Имея только 10 значений или целых 100, это легко проверить вручную с помощью графических методов. Если значений V много, понадобится программа для вычисления χ^2-распределения на компьютере. Заметим, что *все 20 наблюдений на рис. 4, (c) находятся между 5- и 95-процентным уровнями*, поэтому

не нужно рассматривать *любое* из них, как подозрительное, индивидуально. Однако совместно эмпирические распределения показывают, что эти наблюдения не проходят проверку. Важная разница между КС-критерием и χ^2-критерием состоит в том, что КС-критерий применяется к распределению, заданному функцией $F(x)$, которая не имеет скачков, в то время как χ^2-критерий применяется к распределениям, имеющим только скачки (поскольку все наблюдения делятся на k категорий). Таким образом, эти два критерия предназначены для разных видов распределений. Однако критерий χ^2 можно применять даже тогда, когда $F(x)$ непрерывна, если разделить область определения $F(x)$ на k частей и не обращать внимания на то, как случайные величины распределены в каждой из них. Например, чтобы проверить, можно ли рассматривать величины U_1, U_2, ..., U_n как значения случайной величины, имеющей равномерное распределение между нулем и единицей, нужно проверить гипотезу, что эта случайная величина имеет распределение $F(x) = x$ при $0 \le x \le 1$. Для этого естественно использовать КС-критерий. Но можно разделить интервал между нулем и единицей на $k = 100$ равных частей, подсчитать, сколько значений U_k попадет в каждую часть, и применить χ^2-критерий с 99 степенями свободы. Сейчас существует много теоретических результатов сравнения эффективности КС-критерия с χ^2-критерием. Автор нашел несколько примеров, в которых КС-критерий более чувствителен к неслучайности, чем χ^2-критерий. С другой стороны, в некоторых примерах χ^2-критерий дал более значимые результаты. Если, например, 100 категорий, о которых говорилось выше, пронумеровать числами 0, 1, ..., 99 и если отклонение от ожидаемых величин положительно по сравнению с числами от 0 до 49 и отрицательно по сравнению с числами от 50 до 99, то эмпирическая функция распределения будет больше отличаться от $F(x)$, чем показывает значение χ^2. Но если отклонение от ожидаемых величин положительно по сравнению с 0, 2, ..., 98 и отрицательно по сравнению с 1, 3, ..., 99, то эмпирическая функция распределения будет примыкать к $F(x)$ намного теснее. Различные виды отклонений измеряются по-разному. χ^2-критерий был применен к 200 наблюдениям, что привело к ситуации, изображенной на рис. 4 (с $k = 10$). Соответствующие значения V были равны 9.4, 17.7 и 39.3, поэтому в частном случае они хорошо сопоставлялись со значениями КС-критерия, приведенными в (16). Так как χ^2-критерий по своей сути менее точный и требует сравнительно больших значений n, то КС-критерий имеет некоторые преимущества, когда проверяются непрерывные распределения.

Следующий пример также интересен. Наблюдения, изображенные на рис. 2, — это χ^2-статистики, которые основаны на $n = 200$ наблюдениях критерия "максимум-t" для $1 \le t \le 5$ с областью значений, разделенной на 10 равновероятных частей. КС-статистики K_{200}^+ и K_{200}^- могут быть вычислены по тому же множеству в 200 наблюдений, и результаты могут быть табулированы так же, как на рис. 2 (можно показать, какие КС-значения выходят за 99-процентный уровень, и т. д.). Результаты для такого случая показаны на рис. 5. Заметим, что генератор D (оригинальный метод Лехмера) выглядит очень плохо на рис. 5, в то время как χ^2-критерий, примененный *к тем же данным*, наоборот, на рис. 2 выглядит хорошо. Генератор E (метод Фибоначчи) на рис. 5 выглядит неплохо. Хорошие генераторы A и B проходят все испытания удовлетворительно. Причина расхождений между рис. 2 и 5 состоит, в том, что (а) число наблюдений 200 на самом деле не достаточно

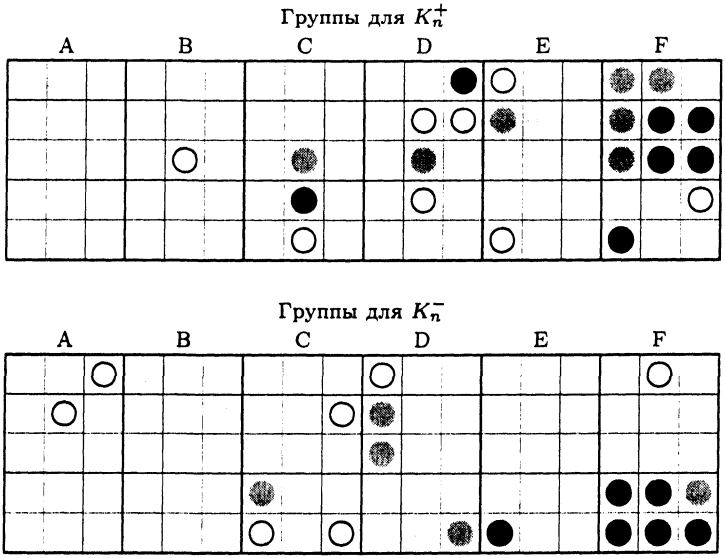


Рис. 5. КС-критерии, примененные к тем же данным, которые представлены на рис. 2.

большое для строгой проверки и (b) слова "отбросить", "подозрительные", "почти подозрительные", упорядочивающие критерий, сами выглядят подозрительно.

(Несправедливо обвинять Лехмера, что он использовал "плохой" генератор случайных чисел в 40-х годах, так как он применял этот генератор D совершенно правильно. Компьютер ENIAC был машиной параллельного действия, и программы заносились в нее штепсельным коммутатором. Лехмер сделал так, что его сумматоры постоянно умножали собственное содержимое на 23 (по модулю $10^8 + 1$), поставляя новые значения каждые несколько микросекунд. Поскольку множитель 23 очень мал и известно, что каждое значение, полученное таким образом, также зависит от предыдущего значения, его нельзя рассматривать в качестве достаточно случайного. Но время между действительным *использованием* величин в специальных сумматорах с помощью сопутствующих программ было достаточно большим и постоянно менялось. Поэтому на самом деле множитель был равен 23^k для достаточно больших, но *меняющихся* значений k.)

C. История, библиография и теория. χ^2-критерий был предложен Карлом Пирсоном (Pearson) в 1900 году [*Philosophical Magazine*, Series 5, **50**, 157–175]. Его замечательная работа рассматривается как фундамент современной математической статистики. Предшественники Пирсона просто строили графики экспериментальных результатов и утверждали, что они правильны. В своей статье Пирсон привел несколько интересных примеров злоупотреблений статистикой. Он также доказал, что некоторые результаты наблюдений за рулеткой (на которой он проводил эксперименты в течение двух недель в Монте-Карло в 1892 году) были так далеки от ожидаемых частот, что шансы получить их снова при предположении, что рулетка устроена добросовестно, равны одному из 10^{29}! Общее обсуждение χ^2-критерия и

обширную библиографию можно найти в обзорной работе Вильяма Дж. Кокрена (William G. Cochran, *Annals Math. Stat.* **23** (1952), 315–345).

Рассмотрим вкратце теорию, лежащую в основе χ^2-критерия. Легко увидеть, что точные вероятности того, что $Y_1 = y_1, \ldots, Y_k = y_k$, равны

$$\frac{n!}{y_1! \ldots y_k!} p_1^{y_1} \ldots p_k^{y_k}. \tag{17}$$

Если предположить, что Y_s принимает значение y_s с вероятностью Пуассона

$$\frac{e^{-np_s}(np_s)^{y_s}}{y_s!}$$

и что Y_i независимы, то $(Y_1, \ldots, Y_k)$ будет равен $(y_1, \ldots, y_k)$ с вероятностью

$$\prod_{s=1}^{k} \frac{e^{-np_s}(np_s)^{y_s}}{y_s!}$$

и $Y_1 + \cdots + Y_k$ будет равно n с вероятностью

$$\sum_{\substack{y_1+\cdots+y_k=n \\ y_1,\ldots,y_k \geq 0}} \prod_{s=1}^{k} \frac{e^{-np_s}(np_s)^{y_s}}{y_s!} = \frac{e^{-n}n^n}{n!}.$$

Если предположить, что они независимы, *кроме того*, что выполняется условие $Y_1 + \cdots + Y_k = n$ (т. е. $Y_1, \ldots, Y_{k-1}$ независимы, а Y_k выражается через $Y_1, \ldots, Y_k$ с помощью этого равенства. — *Прим. ред.*), вероятность, что $(Y_1, \ldots, Y_k) = (y_1, \ldots, y_k)$, равна отношению

$$\left(\prod_{s=1}^{k} \frac{e^{-np_s}(np_s)^{y_s}}{y_s!} \right) \Bigg/ \left(\frac{e^{-n}n^n}{n!} \right),$$

которое, в свою очередь, равно (17). *Поэтому можно рассматривать Y_s, $s = 1, \ldots, k-1$ как независимые случайные величины, имеющие распределение Пуассона и такие, что* $\sum_{s=1}^{n} Y_s = n$.

Теперь удобно сделать замену переменных,

$$Z_s = \frac{Y_s - np_s}{\sqrt{np_s}}, \tag{18}$$

поэтому $V = Z_1^2 + \cdots + Z_k^2$. Условие $Y_1 + \cdots + Y_k = n$ эквивалентно требованию

$$\sqrt{p_1}\, Z_1 + \cdots + \sqrt{p_k}\, Z_k = 0. \tag{19}$$

Рассмотрим теперь $(k-1)$-мерное пространство S векторов $(Z_1, \ldots, Z_k)$, таких, что выполняется (19). Для больших значений n каждое Z_s имеет приближенно нормальное распределение (см. упр. 1.2.10–15). Поэтому вероятность попасть в дифференциальный объем $dz_2 \ldots dz_k$ области S *приближенно* пропорциональна величине $\exp\left(-(z_1^2 + \cdots + z_k^2)/2\right)$. (Именно из-за этого χ^2-критерий можно использо-

вать только при больших n.) Вероятность, что $V \le v$, равна

$$\frac{\int_{(z_1,\ldots,z_k)\text{ в }S\text{ и }z_1^2+\cdots+z_k^2\le v} \exp\left(-(z_1^2+\cdots+z_k^2)/2\right)\, dz_2\ldots dz_k}{\int_{(z_1,\ldots,z_k)\text{ в }S} \exp\left(-(z_1^2+\cdots+z_k^2)/2\right)\, dz_2\ldots dz_k}. \tag{20}$$

Так как гиперплоскость (19) проходит через начало координат k-мерного пространства, в числителе (20) интегрируем по $(k-1)$-мерной гиперсфере с центром в начале координат. Перейдя к обобщенным полярным координатам с радиусом χ и углами $\omega_1,\ldots,\omega_{k-2}$, преобразуем (20) в

$$\frac{\int_{\chi^2\le v} e^{-\chi^2/2}\chi^{k-2} f(\omega_1,\ldots,\omega_{k-2})\, d\chi\, d\omega_1\ldots d\omega_{k-2}}{\int e^{-\chi^2/2}\chi^{k-2} f(\omega_1,\ldots,\omega_{k-2})\, d\chi\, d\omega_1\ldots d\omega_{k-2}},$$

где f — некоторая известная функция (см. упр. 15). Интегрирование по переменным $\omega_1,\ldots,\omega_{k-2}$ дает постоянный множитель в числителе и знаменателе, который сокращается. Окончательно для приближенной вероятности того, что $V \le v$, получаем формулу

$$\frac{\int_0^{\sqrt{v}} e^{-\chi^2/2}\chi^{k-2}\, d\chi}{\int_0^{\infty} e^{-\chi^2/2}\chi^{k-2}\, d\chi}. \tag{21}$$

В (21) для переменной интегрирования используем символ "χ" так же, как Пирсон в своей основополагающей работе. Вот откуда происходит название χ^2-критерия. Если заменить переменную в интеграле $t = \chi^2/2$, то интегралы можно выразить в терминах неполной гамма-функции, сведения о которой содержатся в разделе 1.2.11.3. Наконец, приведем определение χ^2-распределения с $k-1$ степенью свободы

$$\lim_{n\to\infty} \Pr(V \le v) = \gamma\Big(\frac{k-1}{2}, \frac{v}{2}\Big)\Big/\Gamma\Big(\frac{k-1}{2}\Big). \tag{22}$$

Сейчас вернемся к КС-критерию. В 1933 году А. Н. Колмогоров предложил критерий, основанный на статистике

$$K_n = \sqrt{n} \max_{-\infty<x<+\infty} \big|F_n(x) - F(x)\big| = \max(K_n^+, K_n^-). \tag{23}$$

Н. В. Смирнов рассмотрел несколько модификаций этого критерия в 1939 году, предлагая, кстати, проверять отдельно статистики K_n^+ и K_n^-, как мы это делали выше. Существует большая группа модификаций критерия Колмогорова, но статистики K_n^+ и K_n^-, кажется, лучше всего подходят для использования на компьютере. Обширный обзор литературы, посвященной КС-критериям и их обобщениям, можно найти в монографии Дж. Дурбина (J. Durbin, *Regional Conf. Series on Applied Math.* **9** (SIAM, 1973)).

Для того чтобы изучить распределения K_n^+ и K_n^-, начнем со следующего основного факта. *Если X — случайная величина с непрерывной функцией распределения $F(x)$, то $F(X)$ — это случайная величина, равномерно распределенная между* 0 *и* 1. Чтобы доказать это, достаточно проверить, что если $0 \le y \le 1$, то $F(X) \le y$ с вероятностью y. Так как F непрерывна, существует такое x_0, что $F(x_0) = y$. Поэтому вероятность, что $F(X) \le y$, равна вероятности, что $X \le x_0$. По определению x_0 последняя вероятность равна $F(x_0)$, а это число, в свою очередь, равно y.

Пусть $Y_j = nF(X_j)$ для $1 \le j \le n$, где X_n должны быть рассортированы, как на шаге 2 перед формулой (13). Поэтому случайные величины Y_j независимы и имеют одно и то же распределение, а именно — равномерное распределение между 0 и n. Они рассортированы в порядке неубывания $Y_1 \le Y_2 \le \cdots \le Y_n$, и первое равенство в (13) может быть преобразовано следующим образом:

$$K_n^+ = \frac{1}{\sqrt{n}} \max(1 - Y_1, 2 - Y_2, \ldots, n - Y_n).$$

Если $0 \le t \le n$, вероятность, что $K_n^+ \le t/\sqrt{n}$, равна, следовательно, вероятности того, что $Y_j \ge j - t$ для всех $1 \le j \le n$. Это легко выразить в терминах n-мерных интегралов:

$$\frac{\int_{\alpha_n}^{n} dy_n \int_{\alpha_{n-1}}^{y_n} dy_{n-1} \cdots \int_{\alpha_1}^{y_2} dy_1}{\int_0^n dy_n \int_0^{y_n} dy_{n-1} \cdots \int_0^{y_2} dy_1}, \qquad \text{где} \qquad \alpha_j = \max(j - t, 0). \tag{24}$$

Знаменатель здесь вычисляется моментально; он равен $n^n/n!$. Это выражение имеет смысл, так как гиперкуб всех векторов $(y_1, y_2, \ldots, y_n)$, таких, что $0 \le y_j < n$, имеет объем n^n и может быть разделен на $n!$ равных частей, каждая из которых соответствует одному из возможных способов упорядочения y_i. Найти интеграл, стоящий в числителе, немного труднее. Но его можно вычислить методом, предложенным в упр. 17, и получить общую формулу:

$$\Pr\left(K_n^+ \le \frac{t}{\sqrt{n}}\right) = \frac{t}{n^n} \sum_{0 \le k \le t} \binom{n}{k} (k - t)^k (t + n - k)^{n-k-1} \tag{25}$$

$$= 1 - \frac{t}{n^n} \sum_{t < k \le n} \binom{n}{k} (k - t)^k (t + n - k)^{n-k-1} \tag{26}$$

Распределение K_n^- точно такое же. Равенство (26) впервые получено Н. В. Смирновым [*Успехи мат. наук* **10** (1944), 176–206] (см. также работу З. В. Бирнбаума и Фреда Х. Тинги (Z. W. Birnbaum and Fred H. Tingey, *Annals Math. Stat.* **22** (1951), 592–596)). Смирнов вывел также асимптотическую формулу

$$\Pr(K_n^+ \le s) = 1 - e^{-2s^2}\left(1 - \frac{2}{3}s/\sqrt{n} + O(1/n)\right) \tag{27}$$

для всех фиксированных $s \ge 0$, что привело к получению приближенных значений для больших n, которые приведены в табл. 2.

Биномиальная теорема Абеля, равенство 1.2.6–(16), показывает, что равенства (25) и (26) эквивалентны. Можно расширить табл. 2, используя ту либо другую формулу. Существует интересный компромисс: хотя сумма в (25) имеет лишь около $s\sqrt{n}$ членов, когда $s = t/\sqrt{n}$, она должна вычисляться с помощью арифметики с многократной точностью, поскольку члены большие и их главные цифры сокращаются. Но такая проблема не возникает в (26), так как члены этой формулы положительны, но (26) имеет $n - s\sqrt{n}$ членов.

УПРАЖНЕНИЯ

1. [*00*] Какую строку χ^2-таблицы следовало бы использовать, чтобы проверить, будет ли величина $V = 7\frac{7}{48}$ формулы (5) невероятно большой?

2. [*20*] Пусть две игральные кости "устроены" так, что на одной из них 1 будет выпадать вдвое чаще, чем любое другое значение, а на другой 6 будет выпадать вдвое чаще, чем любое другое значение. Найдите вероятность p_s того, что сумма показаний на двух игральных костях равна точно s, $2 \le s \le 12$.

▶ **3.** [*23*] Игральные кости устроены так, как описано в предыдущем упражнении. Они были брошены 144 раза, и получились следующие значения.

Значение $s =$	2	3	4	5	6	7	8	9	10	11	12
Число наблюдений $Y_s =$	2	6	10	16	18	32	20	13	16	9	2

Примените χ^2-критерий к *этим* значениям, используя вероятности из (1) и считая, что игральные кости на самом деле не поддельные. Определит ли χ^2-критерий, что кости плохие? Если нет, объясните, почему.

▶ **4.** [*23*] На самом деле автор получил данные эксперимента 1 из (9), моделируя игральные кости, одна из которых была нормальна, а другая всегда давала только значения 1 или 6. (Причем обозначения появлялись с равными вероятностями.) Подсчитайте вероятности, которыми можно заменить (1) в этом случае, и, используя χ^2-критерий, решите, соответствуют ли результаты эксперимента такому устройству игральных костей.

5. [*22*] Пусть $F(x)$ — равномерное распределение (см. рис. 3, (b)). Найдите K_{20}^+ и K_{20}^- для следующих 20 наблюдений:

0.414, 0.732, 0.236, 0.162, 0.259, 0.442, 0.189, 0.693, 0.098, 0.302,
0.442, 0.434, 0.141, 0.017, 0.318, 0.869, 0.772, 0.678, 0.354, 0.718.

Проверьте, будут ли наблюдения значимо отличаться от ожидаемого поведения по отношению к каждой из двух проверок.

6. [*M20*] Пусть $F_n(x)$ задано формулой (10) для фиксированного x. Чему равна вероятность, что $F_n(x) = s/n$, для заданного целого s? Чему равно среднее значение $F_n(x)$? Чему равно стандартное отклонение?

7. [*M15*] Покажите, что K_n^+ и K_n^- никогда не могут быть отрицательными. Какое наиболее возможное значение может иметь K_n^+?

8. [*00*] В разделе описывается эксперимент, в котором 20 значений статистики K_{10}^+ были получены при изучении случайной последовательности. Эти значения были нанесены на график, чтобы получить рис. 4. КС-статистика была подсчитана с помощью этого графика. Почему для изучения полученной статистики вместо таблицы при $n = 10$ использовались табличные значения для $n = 20$?

▶ **9.** [*20*] Описанный в разделе эксперимент состоит в том, что 20 значений K_{10}^+, вычисленных с помощью критерия "максимум-5", который применялся к различным частям случайной последовательности, наносятся на график. Мы подсчитали также соответствующие 20 значений K_{10}^-, поскольку K_{10}^- так же распределены, как и K_{10}^+. Теперь можно объединить 40 значений, полученных таким образом (т. е. 20 значений K_{10}^+ и 20 значений K_{10}^-), опять применить КС-критерий и получить новые значения K_{40}^+ и K_{40}^-. Обсудите ценность этой идеи.

▶ **10.** [*20*] Предположим, что χ^2-статистика подсчитана по результатам n наблюдений, т. е. получено значение V. Повторим подсчет статистики, используя те же n наблюдений (конечно, получится такой же результат). Затем суммируем данные обоих испытаний, рассматривая их как единственный χ^2-критерий с $2n$ наблюдениями. (Эта процедура нарушает требование независимости всех наблюдений, которое было выдвинуто в разделе: все наблюдения должны быть независимыми.) Каково соотношение между этими двумя значениями V?

11. [*10*] Выполните упр. 10, заменив χ^2-критерий КС-критерием.

12. [*M28*] Пусть подсчет χ^2-критерия основан на множестве, состоящем из n наблюдений, в предположении, что p_s — вероятность того, что каждое наблюдение соответствует категории s. Предположим, что на самом деле вероятность отнесения наблюдения к категории s равна $q_s \neq p_s$ (см. упр. 3). Хотелось бы, конечно, чтобы χ^2-критерий обнаружил тот факт, что предположения p_s неверны. Покажите, что это *произойдет*, если n достаточно велико. Докажите аналогичный результат и для КС-критерия.

13. [*M24*] Докажите, что равенства (13) эквивалентны равенствам (11).

▶ **14.** [*HM26*] Пусть Z_s задано равенством (18). Покажите, непосредственно используя формулу Стирлинга, что полиномиальные вероятности

$$n! p_1^{Y_1} \dots p_k^{Y_k} / Y_1! \dots Y_k! = e^{-V/2} / \sqrt{(2n\pi)^{k-1} p_1 \dots p_k} + O(n^{-k/2}),$$

если $Z_1, Z_2, \dots; Z_k$ ограничены при $n \to \infty$. (Эта идея ведет к обоснованию χ^2-критерия; она ближе к "основным принципам" и требует меньше усилий, чем доказательство, приведенное в разделе.)

15. [*HM24*] Полярные координаты в двумерном пространстве определяются равенствами $x = r\cos\theta$ и $y = r\sin\theta$. При интегрировании справедливо равенство $dx\,dy = r\,dr\,d\theta$. Для более общего n-мерного пространства можно положить

$$x_k = r\sin\theta_1 \dots \sin\theta_{k-1}\cos\theta_k, \quad 1 \le k < n, \qquad \text{и} \qquad x_n = r\sin\theta_1 \dots \sin\theta_{n-1}.$$

Покажите, что в этом случае

$$dx_1\,dx_2 \dots dx_n = |r^{n-1}\sin^{n-2}\theta_1 \dots \sin\theta_{n-2}\,dr\,d\theta_1 \dots d\theta_{n-1}|.$$

▶ **16.** [*HM35*] Обобщите теорему 1.2.11.3A, чтобы найти значение

$$\gamma(x+1,\, x + z\sqrt{2x} + y)/\Gamma(x+1)$$

для больших x и фиксированных y, z. Опустите члены ответа, имеющие порядок $O(1/x)$. Используйте этот результат, чтобы найти приближенное решение t уравнения

$$\gamma\Big(\frac{\nu}{2}, \frac{t}{2}\Big) \Big/ \Gamma\Big(\frac{\nu}{2}\Big) = p$$

для больших ν и фиксированных p, таким образом получив асимптотические формулы, приведенные в табл. 1. [*Указание.* См. упр. 1.2.11.3–8.]

17. [*HM26*] Пусть t — фиксированное число. Для $0 \le k \le n$ положим

$$P_{nk}(x) = \int_{n-t}^{x} dx_n \int_{n-1-t}^{x_n} dx_{n-1} \dots \int_{k+1-t}^{x_{k+2}} dx_{k+1} \int_{0}^{x_{k+1}} dx_k \dots \int_{0}^{x_2} dx_1;$$

по определению пусть $P_{00}(x) = 1$. Докажите следующие соотношения.

a) $P_{nk}(x) = \displaystyle\int_{n}^{x+t} dx_n \int_{n-1}^{x_n} dx_{n-1} \dots \int_{k+1}^{x_{k+2}} dx_{k+1} \int_{t}^{x_{k+1}} dx_k \dots \int_{t}^{x_2} dx_1.$

b) $P_{n0}(x) = (x+t)^n/n! - (x+t)^{n-1}/(n-1)!.$

c) $P_{nk}(x) - P_{n(k-1)}(x) = \dfrac{(k-t)^k}{k!} P_{(n-k)0}(x-k)$, если $1 \le k \le n$.

d) Кроме того, получите общую формулу для $P_{nk}(x)$ и примените ее к вычислению (24).

18. [*M20*] Найдите "простое" объяснение, почему K_n^- имеет то же распределение, что и K_n^+.

19. [*HM48*] Предложите критерий, аналогичный критерию Колмогорова-Смирнова, для применения к многомерным распределениям $F(x_1,\ldots,x_r) = \Pr(X_1 \le x_1,\ldots,X_r \le x_r)$. (Можете использовать такие процедуры, как, например, "критерий серий" (см. следующий раздел).)

20. [*HM41*] Получите следующие члены асимптотического разложения для КС-распределения, продолжив (27).

21. [*M40*] Хотя в разделе говорится, что КС-критерий может применяться только тогда, когда $F(x)$ — непрерывная функция распределения, конечно, можно попытаться вычислить K_n^+ и K_n^-, когда распределение имеет скачки. Проанализируйте возможное поведение K_n^+ и K_n^- для различных разрывных распределений $F(x)$. Сравните эффективность полученных статистических критериев с χ^2-критерием на нескольких выборках случайных чисел.

22. [*HM46*] Исследуйте "подправленный" КС-критерий, предложенный в ответе к упр. 6.

23. [*M22*] (Т. Гонсалес (T. Gonzalez), С. Сахни (S. Sahni) и В. Р. Франта (W. R. Franta).) (a) Предположим, что максимальное значение в формуле (13) для КС-статистики K_n^+ принимается для данного индекса j, когда $\lfloor nF(X_j)\rfloor = k$. Докажите, что $F(X_j) = \max_{1\le i\le n}\{F(X_i) \mid \lfloor nF(X_i)\rfloor = k\}$. (b) Напишите алгоритм для вычисления K_n^+ и K_n^- за $O(n)$ шагов (без сортировки).

▶ **24.** [*40*] Проведите опыты с различными вероятностными распределениями (p,q,r) трех категорий, где $p+q+r=1$, вычисляя точное распределение χ^2-статистики V для различных n и тем самым определяя, насколько точным является приближение χ^2-распределения с двумя степенями свободы.

25. [*HM26*] Предположим, что $Y_i = \sum_{j=1}^n a_{ij}X_j + \mu_i$ для $1 \le i \le m$, где X_1, ..., X_n — независимые случайные величины со средним, равным нулю, и единичной дисперсией. Матрица $A = (a_{ij})$ имеет ранг n.

a) Выразите ковариационную матрицу $C = (c_{ij})$, где $c_{ij} = \mathrm{E}(Y_i - \mu_i)(Y_j - \mu_j)$, в терминах матрицы A.

b) Докажите следующее: если $\bar{C} = (\bar{c}_{ij})$ — любая матрица, такая, что $C\bar{C}C = C$, то статистика

$$W = \sum_{i=1}^m \sum_{j=1}^m (Y_i - \mu_i)(Y_j - \mu_j)\bar{c}_{ij}$$

равна $X_1^2 + \cdots + X_n^2$. [Следовательно, если X_j имеют нормальное распределение, W имеет χ^2-распределение с n степенями свободы.]

Устойчивость среднего при бросании монеты
определяется законом, ... который гарантирует,
что вы не разоритесь сами, слишком много теряя, и
не разорите своих оппонентов, слишком много выигрывая.

— ТОМ СТОППАРД (TOM STOPPARD),
Розенкранц и Гилденстерн мертвы (1966)

3.3.2. Эмпирические критерии

В этом разделе рассматриваются одиннадцать специфических критериев, которые традиционно применяются для проверки, будет ли последовательность случайной. Обсуждение каждого критерия разбивается на две части: (a) "краткое" описание

способов применения; (b) теоретическое обоснование. (Читатель, не привыкший к математическим рассуждениям, может пропустить теоретические выкладки. С другой стороны, математически подготовленный читатель может найти приведенную теорию весьма интересной, даже если он никогда не собирается проверять генераторы случайных чисел, так как здесь вводятся некоторые понятия комбинаторики. Действительно, в этом разделе вводится несколько важных понятий, представляющих для нас интерес в связи с совершенно иными вопросами.)

Каждый критерий применяется к последовательности

$$\langle U_n \rangle = U_0, U_1, U_2, \ldots \tag{1}$$

действительных чисел, которые, как предполагается, независимы и равномерно распределены между 0 и 1. Некоторые из этих критериев предназначены непосредственно для целочисленных последовательностей, а не для последовательностей действительных чисел (1). В таком случае вместо нее используется вспомогательная последовательность

$$\langle Y_n \rangle = Y_0, Y_1, Y_2, \ldots, \tag{2}$$

определенная правилом

$$Y_n = \lfloor dU_n \rfloor. \tag{3}$$

Это последовательность целых чисел, которые, как предполагается, независимы и одинаково распределены между 0 и $d-1$. (Иначе говоря, вероятность, что случайная величина примет значение k, $k = 0, \ldots, d-1$, равна $1/d$. — *Прим. ред.*) Число d выбирается таким, чтобы его было удобно использовать в том либо ином смысле. Например, можно выбрать $d = 64 = 2^6$ на бинарном компьютере так, что Y_n представляет шесть старших двоичных разрядов двоичного представления числа U_n. Значение d должно быть достаточно большим, чтобы критерий был значимым, но не настолько большим, чтобы критерий стал практически неприменимым.

Введенные выше обозначения U_n, Y_n и d будут использоваться в этом разделе, хотя значение d будет, вероятно, изменяться в различных критериях.

A. Критерий равномерности (критерий частот). Первое требование, предъявляемое к последовательности (1), состоит в том, что ее члены — это числа, равномерно распределенные между 0 и 1. Существует два способа проверить это. (a) Использовать критерий Колмогорова-Смирнова с $F(x) = x$ для $0 \le x \le 1$. (b) Использовать последовательность (2) вместо (1), где d — подходящее число, например 100 на десятичном либо 64 или 128 — на бинарном компьютере. Для каждого r, $0 \le r < d$, подсчитаем число случаев, когда $Y_j = r$ для $0 \le j < n$, а затем применим χ^2-критерий, принимая $k = d$ и вероятности $p_s = 1/d$ для каждой категории.

Описание и обоснование этих критериев приведено в разделе 3.3.1.

B. Критерий серий. Более общее требование к последовательности состоит в том, чтобы пары последовательных чисел были равномерно распределены независимым образом. В конечном счете Солнце восходит так же часто, как и заходит, но это не делает его движение случайным.

В критерии серий просто подсчитываем число случаев, когда пара $(Y_{2j}, Y_{2j+1}) = (q, r)$ для $0 \le j < n$. Такая операция осуществляется для каждой пары целых

чисел (q, r), таких, что $0 \le q, r < d$. Затем применяем χ^2-критерий к этим $k = d^2$ категориям, где $1/d^2$ — вероятность отнесения пары чисел к каждой из категорий. Как и для критерия равномерности, d — подходящее число, но оно должно быть немного меньшим, чем значения, предложенные выше, так как значимый χ^2-критерий должен иметь n сравнительно большим по сравнению с k (скажем, по крайней мере $n \ge 5d^2$).

Ясно, что можно обобщить этот критерий для троек, четверок и т. д. вместо пар (см. упр. 2). Тогда значение d необходимо существенно уменьшить для того, чтобы число категорий не получилось слишком большим. Поэтому при рассмотрении четверок и больших серий чисел используются менее точные критерии, такие как покер-критерий и максимум-критерий, описанные ниже.

Заметим, что $2n$ чисел последовательности (2) использовались в этом критерии для того, чтобы сделать n наблюдений. Было бы ошибкой применять критерий серий к парам (Y_0, Y_1), (Y_1, Y_2), ..., (Y_{n-1}, Y_n). Может ли читатель сказать, почему? Но можно применить критерий серий еще и к парам (Y_{2j+1}, Y_{2j+2}), ожидая, что наша последовательность удовлетворит этим двум проверкам. Однако нужно помнить, что эти проверки на самом деле взаимозависимы. С другой стороны, Джордж Марсалья (George Marsaglia) доказал, что если использовать пары (Y_0, Y_1), (Y_1, Y_2), ..., (Y_{n-1}, Y_n) и применить обычный χ^2-критерий для вычисления обеих статистик (V_2 для критерия серий и V_1 — для критерия частот по $Y_0, \ldots, Y_{n-1}$) с тем же значением d, то случайная величина $V_2 - V_1$ будет иметь χ^2-распределение с $d(d-1)$ степенями свободы, когда n достаточно большое (см. упр. 24).

C. Критерий интервалов. Этот критерий используется для проверки длины "интервалов" между появлением U_j на определенном отрезке. Если α и β — два действительных числа, таких, что $0 \le \alpha < \beta \le 1$, то рассмотрим длины подпоследовательностей U_j, U_{j+1}, ..., U_{j+r}, в которых U_{j+r} лежит между α и β, а другие U_s не лежат между этими числами. (Эту подпоследовательность, состоящую из $r+1$ числа, будем называть интервалом длиной r.)

Алгоритм G (*Данные для критерия интервалов*). Следующий алгоритм (рис. 6), примененный к последовательности (1) для любых значений α и β, подсчитывает число интервалов длиной 0, 1, ..., $t-1$ и число интервалов длиной $\ge t$, пока не получится n интервалов.

G1. [Инициализация.] Присвоить $j \leftarrow -1$, $s \leftarrow 0$ и присвоить COUNT[r] $\leftarrow 0$ для $0 \le r \le t$.

G2. [Присвоение r значения 0.] Присвоить $r \leftarrow 0$.

G3. [$\alpha \le U_j < \beta$?] Увеличить j на 1. Если $U_j \ge \alpha$ и $U_j < \beta$, то перейти к шагу G5.

G4. [Увеличение r.] Увеличить r на единицу и возвратиться к шагу G3.

G5. [Регистрация длины интервала.] (Интервал длиной r только что найден.) Если $r \ge t$, то увеличить COUNT[t] на 1, иначе — увеличить COUNT[r] на 1.

G6. [Найдены ли n интервалов?] Увеличить s на 1. Если $s < n$, то вернуться к шагу G2. ▮

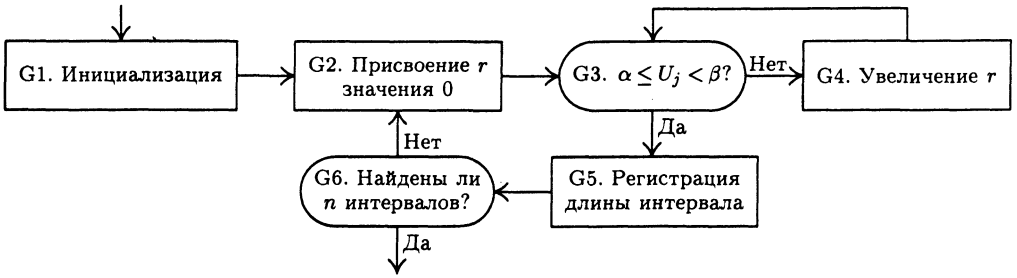


Рис. 6. Сбор данных для критерия интервалов. (Алгоритмы для критериев собирания купонов и монотонности подобны этому.)

После реализации алгоритма G χ^2-критерий применяется при $k = t + 1$ к значениям COUNT[0], COUNT[1], ..., COUNT[t] с использованием следующих вероятностей:

$$p_r = p(1-p)^r \quad \text{для } 0 \le r \le t-1; \qquad p_t = (1-p)^t. \tag{4}$$

Здесь $p = \beta - \alpha$ — вероятность того, что $\alpha \le U_j < \beta$. Значения n и t выбираются, как обычно, чтобы ожидаемое значение COUNT[r] равнялось 5 или больше (желательно больше).

Критерий интервалов часто применяется при $\alpha = 0$ или $\beta = 1$ для того, чтобы на шаге G3 обойтись без одного сравнения. Особые случаи $(\alpha, \beta) = (0, \frac{1}{2})$ и $(\frac{1}{2}, 1)$ иногда называются "отклонение выше среднего" и "отклонение ниже среднего" соответственно.

Вероятности в (4) получить легко, и мы это оставляем читателю. Заметим, что для критерия интервалов, описанного выше, необходимо получить n *интервалов* определенной длины. Однако таких интервалов в *количестве* n может не оказаться. Если последовательность $\langle U_n \rangle$ недостаточно случайна, то алгоритм G может не закончиться. Поэтому можно предложить другой критерий интервалов, требующий фиксированное число значений U_j (см. упр. 5).

D. Покер-критерий (критерий разбиений). "Классический" покер-критерий рассматривает n групп по пять последовательных целых чисел $\{Y_{5j}, Y_{5j+1}, Y_{5j+2}, Y_{5j+3}, Y_{5j+4}\}$ для $0 \le j < n$ и проверяет, какие из следующих семи комбинаций соответствуют таким пятеркам чисел (порядок не имеет значения).

Все числа разные:	*abcde*
Одна пара:	*aabcd*
Две пары:	*aabbc*
Три числа одного вида:	*aaabc*
Полный набор:	*aaabb*
Четыре числа одного вида:	*aaaab*
Пять чисел одного вида:	*aaaaa*

χ^2-критерий основан на подсчете числа пятерок в каждой категории.

Уместно рассмотреть какую-нибудь упрощенную версию этого критерия, для которой можно использовать более простые программы. Хорошим компромиссом

будет критерий, использующий более простой подсчет *различных* значений в множестве пятерок. В этом случае можно выделить только пять категорий:

5 значений = все разные;

4 значения = одна пара;

3 значения = две пары или три числа одного вида;

2 значения = полный набор или четыре числа одного вида;

1 значение = пять чисел одного вида.

При такой схеме упрощаются подсчеты и критерий остается почти таким же хорошим.

В общем случае можно рассматривать n групп k последовательных чисел и подсчитывать число групп из k чисел с r различными числами. Затем применяется χ^2-критерий, в котором используются вероятности того, что в группе r различных чисел

$$p_r = \frac{d(d-1)\ldots(d-r+1)}{d^k} \left\{ {k \atop r} \right\}. \tag{5}$$

(Числа Стирлинга $\left\{ {k \atop r} \right\}$ определены в разделе 1.2.6 и могут быть подсчитаны по приведенным в нем формулам.) Так как вероятности p_r очень малы, когда $r = 1$ или 2, следует, вообще говоря, перед применением χ^2-критерия объединить несколько категорий, имеющих малые вероятности, в одну.

Чтобы получить формулу для p_r, следует подсчитать, сколько d^k групп из k чисел, расположенных между 0 и $d-1$, имеют точно r различных элементов, и разделить это число на d^k. Так как $d(d-1)\ldots(d-r+1)$ — это число упорядоченных наборов из r элементов множества, содержащего d элементов, достаточно показать, что $\left\{ {k \atop r} \right\}$ — это число способов разбиения множества из k элементов на точно r частей. Поэтому в упр. 1.2.6–64 завершается доказательство равенства (5).

E. Критерий собирания купонов. Следующий критерий соотносится с покер-критерием, так как критерий интервалов соотносится с критерием частот. Используется последовательность $Y_0, Y_1, \ldots$ и находятся длины отрезков $Y_{j+1}, Y_{j+2}, \ldots, Y_{j+r}$, содержащие "полный набор" целых чисел от 0 до $d-1$. Алгоритм C описывает эту процедуру.

Алгоритм C (*Данные для критерия собирания купонов*). Если дана последовательность целых чисел $Y_0, Y_1, \ldots$, таких, что $0 \le Y_j < d$, то алгоритм подсчитывает длины n последовательных "собравших купоны" отрезков. После реализации алгоритма COUNT[r] — это число отрезков длиной r для $d \le r < t$, а COUNT[t] — это число отрезков длиной $\ge t$.

C1. [Инициализация.] Присвоить $j \leftarrow -1$, $s \leftarrow 0$ и присвоить COUNT[r] $\leftarrow 0$ для $d \le r \le t$.

C2. [Присвоить q и r значение 0.] Присвоить $q \leftarrow r \leftarrow 0$ и присвоить OCCURS[k] $\leftarrow 0$ для $0 \le k < d$.

C3. [Следующее наблюдение.] Увеличить r и j на 1. Если OCCURS[Y_j] $\neq 0$, повторить этот шаг.

C4. [Полное множество?] Присвоить $\texttt{OCCURS}[Y_j] \leftarrow 1$ и $q \leftarrow q+1$. (В наблюдаемой подпоследовательности содержится q различных значений; если $q = d$, то множество полное.) Если $q < d$, возвратиться к шагу C3.

C5. [Регистрация длины.] Если $r \geq t$, увеличить $\texttt{COUNT}[t]$ на 1, иначе — увеличить $\texttt{COUNT}[r]$ на 1.

C6. [n найдено?] Увеличить s на 1. Если $s < n$, возвратиться к шагу C2. ▮

Пример использования этого алгоритма приведен в упр. 7. Представим себе мальчика, который собирает купоны d видов, равномерно распределенные по его сверткам с завтраком. Он ест завтраки до тех пор, пока не получит по крайней мере по одному купону каждого типа.

После того как алгоритм C вычислит n длин, нужно применить χ^2-критерий к $\texttt{COUNT}[d]$, $\texttt{COUNT}[d+1]$, ..., $\texttt{COUNT}[t]$ с $k = t - d + 1$. Соответствующие вероятности равны

$$p_r = \frac{d!}{d^r}\left\{{r-1 \atop d-1}\right\}, \qquad d \leq r < t; \qquad p_t = 1 - \frac{d!}{d^{t-1}}\left\{{t-1 \atop d}\right\}. \tag{6}$$

Чтобы найти их, просто заметим, что если q_r равна вероятности того, что подпоследовательность длиной r *не полна*, то

$$q_r = 1 - \frac{d!}{d^r}\left\{{r \atop d}\right\}$$

согласно (5), т. е. это вероятность того, что набор из r элементов не имеет d различных значений. Поэтому (6) следует из соотношений $p_t = q_{t-1}$ и

$$p_r = q_{r-1} - q_r \qquad \text{для } d \leq r < t.$$

В упр. 9 и 10, а также в работах Джорджа Пойа (Полиа) (George Pólya, *Zeitschrift für angewandte Math. und Mech.* **10** (1930), 96–97) и Германа фон Шеллинга (Hermann von Schelling, *AMM* **61** (1954), 306–311) приведены формулы, возникающие в связи с *обобщением* критерия собирания купонов.

F. Критерий перестановок. Разделим последовательность на выходе на n групп по t элементов в каждой, т. е. $(U_{jt}, U_{jt+1}, \ldots, U_{jt+t-1})$ для $0 \leq j < n$. Элементы в каждой группе можно упорядочивать $t!$ различными способами. Подсчитывается число групп с любым возможным порядком и применяется χ^2-критерий с $k = t!$ возможными категориями и вероятностью $1/t!$ для каждой категории.

Например, если $t = 3$, существует шесть возможных категорий, а именно $U_{3j} < U_{3j+1} < U_{3j+2}$, или $U_{3j} < U_{3j+2} < U_{3j+1}$, или $\cdots$, или $U_{3j+2} < U_{3j+1} < U_{3j}$. В этом критерии предполагается, что U_s не могут быть равны между собой, т. е. вероятность того, что $U_s = U_j$ при $s \neq j$ равна нулю.

Чтобы реализовать этот критерий на компьютере, удобно использовать следующий сам по себе интересный алгоритм.

Алгоритм P (*Анализ перестановок*). Если задана последовательность различных элементов $(U_1, \ldots, U_t)$, вычислим целое число $f(U_1, \ldots, U_t)$, такое, что

$$0 \leq f(U_1, \ldots, U_t) < t!$$

и $f(U_1, \ldots, U_t) = f(V_1, \ldots, V_t)$ тогда и только тогда, когда $(U_1, \ldots, U_t)$ и $(V_1, \ldots, V_t)$ имеют тот же относительный порядок.

P1. [Инициализация.] Присвоить $r \leftarrow t$, $f \leftarrow 0$. (В этом алгоритме всегда будет выполняться неравенство $0 \le f < t!/r!$.)

P2. [Найти максимум.] Найти максимум $\{U_1, \ldots, U_r\}$. Если U_s — это максимум, присвоить $f \leftarrow r \cdot f + s - 1$.

P3. [Замена.] Заменить $U_r \leftrightarrow U_s$.

P4. [Уменьшить r.] Уменьшить r на 1. Если $r > 1$, то вернуться к шагу P2. ▮

Последовательность $(U_1, \ldots, U_t)$ будет расположена в порядке возрастания, когда алгоритм остановит работу. Чтобы доказать, что функция f единственным образом *характеризуется* начальным порядком $(U_1, \ldots, U_t)$, заметим, что алгоритм P может быть обращен.

Для $r = 2, 3, \ldots, t$
присвоить $s \leftarrow f \bmod r$, $f \leftarrow \lfloor f/r \rfloor$
и заменить $U_r \leftrightarrow U_{s+1}$.

Легко видеть, что это разрушит эффект шагов P2–P4. Следовательно, не существует двух перестановок, которые могут иметь одно и то же значение f, и алгоритм P обоснован.

Главной идеей, которая лежит в основе алгоритма P, является представление со смешанным основанием, называемым "факториальной числовой системой": каждое целое число на отрезке $0 \le f < t!$ может быть единственным образом записано в виде

$$\begin{aligned} f &= \big(\ldots(c_{t-1} \times (t-1) + c_{t-2}) \times (t-2) + \cdots + c_2\big) \times 2 + c_1 \\ &= (t-1)!\,c_{t-1} + (t-2)!\,c_{t-2} + \cdots + 2!\,c_2 + 1!\,c_1, \end{aligned} \tag{7}$$

где "цифры" c_j — это целые числа, удовлетворяющие неравенствам

$$0 \le c_j \le j \qquad \text{для } 1 \le j < t. \tag{8}$$

В алгоритме P $c_{r-1} = s - 1$, когда шаг P2 выполнен для заданного значения r.

G. Критерий монотонности. В последовательности можно проверять распределение восходящих и нисходящих серий. Имеется в виду проверка распределений длин *монотонных* частей заданной последовательности (отрезки возрастания или убывания).

В качестве примера точного определения серии рассмотрим последовательность цифр "1298536704". Проводя вертикальные линии слева, справа, а также между X_j и X_{j+1} всякий раз, когда $X_j > X_{j+1}$, получим

$$\big|1\ \ 2\ \ 9\big|8\big|5\big|3\ \ 6\ \ 7\big|0\ \ 4\big|. \tag{9}$$

Таким образом выделяются восходящие серии: сначала — серия длиной 3, затем — две серии длиной 1, затем — снова серия длиной 3 и, наконец, серия длиной 2. В алгоритме из упр. 12 показано, как табулировать длину восходящей серии.

Мы не будем применять χ^2-критерий к подсчету серий, как в критерии интервалов и критерии собирания купонов (которые во многих других отношениях

подобны этому критерию), поскольку смежные интервалы *не* являются независимыми. Длинные серии имеют тенденцию следовать за короткими и наоборот. Такого отсутствия независимости достаточно, чтобы применение χ^2-критерия было неправомочным. Вместо этого можно подсчитать следующую статистику для случая, когда длины серий определены так, как в упр. 12:

$$V = \frac{1}{n-6} \sum_{1\le i,j\le 6} (\texttt{COUNT}[i] - nb_i)(\texttt{COUNT}[j] - nb_j)a_{ij}, \tag{10}$$

где n — длина последовательности и матрицы коэффициентов $A = (a_{ij})_{1\le i,j\le 6}$ и $B = (b_i)_{1\le i\le 6}$ заданы в следующем виде:

$$A = \begin{pmatrix} 4529.4 & 9044.9 & 13568 & 18091 & 22615 & 27892 \\ 9044.9 & 18097 & 27139 & 36187 & 45234 & 55789 \\ 13568 & 27139 & 40721 & 54281 & 67852 & 83685 \\ 18091 & 36187 & 54281 & 72414 & 90470 & 111580 \\ 22615 & 45234 & 67852 & 90470 & 113262 & 139476 \\ 27892 & 55789 & 83685 & 111580 & 139476 & 172860 \end{pmatrix}, \quad B = \begin{pmatrix} \frac{1}{6} \\ \frac{5}{24} \\ \frac{11}{120} \\ \frac{19}{720} \\ \frac{29}{5040} \\ \frac{1}{840} \end{pmatrix}. \tag{11}$$

(Значения a_{ij}, приведенные здесь, только приблизительны; точные значения могут быть получены из формул, приведенных ниже.) *Статистика V в* (10) *должна иметь χ^2-распределение с шестью, а не пятью, степенями свободы*, когда n большое. Значение n должно, скажем, равняться 4 000 или больше. Тот же критерий можно применить к нисходящей серии.

Значительно более простой и более практичный критерий монотонности приведен в упр. 14, так что читатель, интересующийся только проверкой генераторов случайных чисел, может пропустить несколько страниц и перейти к критерию "максимум-t" после выполнения этого упражнения. С другой стороны, с математической точки зрения поучительно увидеть, как можно изучить сложный критерий монотонности со взаимно независимыми сериями. Так что на некоторое время мы отклонимся в сторону.

Пусть задана перестановка n элементов. Пусть $Z_{pi} = 1$, если положение i является началом возрастающей серии длиной p или больше, и пусть $Z_{pi} = 0$ в других случаях. Например, рассмотрим перестановку (9) с $n = 10$. Легко видеть, что

$$Z_{11} = Z_{21} = Z_{31} = Z_{14} = Z_{15} = Z_{16} = Z_{26} = Z_{36} = Z_{19} = Z_{29} = 1,$$

а все остальные Z_{ij} равны нулю. В этих обозначениях

$$R'_p = Z_{p1} + Z_{p2} + \cdots + Z_{pn} \tag{12}$$

представляет собой число серий длиной $\ge p$ и

$$R_p = R'_p - R'_{p+1} \tag{13}$$

является числом серий, длина которых точно равна p. Наша цель — подсчитать среднее значение (mean) R_p, а также *ковариацию* (covar)

$$\operatorname{covar}(R_p, R_q) = \operatorname{mean}\big(\big(R_p - \operatorname{mean}(R_p)\big)\big(R_q - \operatorname{mean}(R_q)\big)\big),$$

являющуюся мерой зависимости R_p и R_q. Средние значения должны быть подсчитаны как среднее по множеству всех $n!$ перестановок.

Равенства (12) и (13) показывают, что можно выразить в терминах средних значений Z_{pi} и $Z_{pi}Z_{qj}$, поэтому сначала получим следующий результат (предполагая, что $i < j$):

$$\frac{1}{n!}\sum Z_{pi} = \begin{cases} \dfrac{p+\delta_{i1}}{(p+1)!}, & \text{если } i \le n-p+1; \\ 0 & \text{в других случаях.} \end{cases}$$

$$\frac{1}{n!}\sum Z_{pi}Z_{qj} = \begin{cases} \dfrac{(p+\delta_{i1})q}{(p+1)!\,(q+1)!}, & \text{если } i+p < j \le n-q+1; \\ \dfrac{p+\delta_{i1}}{(p+1)!\,q!} - \dfrac{p+q+\delta_{i1}}{(p+q+1)!}, & \text{если } i+p = j \le n-q+1; \\ 0 & \text{в других случаях.} \end{cases} \tag{14}$$

Суммирование ($\sum$) производится по всем возможным перестановкам. Для иллюстрации вычислений рассмотрим наиболее трудный случай, когда $i+p = j \le n-q+1$, а $i > 1$. Величина $Z_{pi}Z_{qj}$ равна либо нулю, либо единице, поэтому суммирование сводится к подсчету числа перестановок $U_1U_2\ldots U_n$, для которых $Z_{pi} = Z_{qj} = 1$, т. е. всех таких перестановок, что

$$U_{i-1} > U_i < \cdots < U_{i+p-1} > U_{i+p} < \cdots < U_{i+p+q-1}. \tag{15}$$

Количество таких перестановок может быть подсчитано следующим образом. Существует $\binom{n}{p+q+1}$ возможностей выбора элементов для положения, изображенного в (15), существует

$$(p+q+1)\binom{p+q}{p} - \binom{p+q+1}{p+1} - \binom{p+q+1}{1} + 1 \tag{16}$$

способов их расположения в таком порядке, как в (15) (см. упр. 13), и существует $(n-p-q-1)!$ способов упорядочения оставшихся элементов. Следовательно, выражение в (16) нужно умножить на $\binom{n}{p+q+1}(n-p-q-1)!$ и разделить на $n!$, чтобы получить искомую формулу.

Из соотношения (14) и несколько громоздких вычислений получим

$$\operatorname{mean}(R'_p) = (n+1)p/(p+1)! - (p-1)/p!, \qquad 1 \le p \le n; \tag{17}$$

$$\begin{aligned} \operatorname{covar}(R'_p, R'_q) &= \operatorname{mean}(R'_p R'_q) - \operatorname{mean}(R'_p)\operatorname{mean}(R'_q) \\ &= \sum_{1\le i,j\le n} \frac{1}{n!}\sum Z_{pi}Z_{qj} - \operatorname{mean}(R'_p)\operatorname{mean}(R'_q) \\ &= \begin{cases} \operatorname{mean}(R'_t) + f(p,q,n), & \text{если } p+q \le n, \\ \operatorname{mean}(R'_t) - \operatorname{mean}(R'_p)\operatorname{mean}(R'_q), & \text{если } p+q > n, \end{cases} \end{aligned} \tag{18}$$

где $t = \max(p, q)$, $s = p + q$ и

$$f(p,q,n) = (n+1)\left(\frac{s(1-pq)+pq}{(p+1)!\,(q+1)!} - \frac{2s}{(s+1)!}\right) + 2\left(\frac{s-1}{s!}\right) + \frac{(s^2-s-2)pq - s^2 - p^2q^2 + 1}{(p+1)!\,(q+1)!} \tag{19}$$

Это выражение для ковариации, к сожалению, является довольно сложным, но без него нельзя обойтись. Из этих формул легко вычислить

$$\begin{aligned} \operatorname{mean}(R_p) &= \operatorname{mean}(R'_p) - \operatorname{mean}(R'_{p+1}), \\ \operatorname{covar}(R_p, R'_q) &= \operatorname{covar}(R'_p, R'_q) - \operatorname{covar}(R'_{p+1}, R'_q), \\ \operatorname{covar}(R_p, R_q) &= \operatorname{covar}(R_p, R'_q) - \operatorname{covar}(R_p, R'_{q+1}). \end{aligned} \tag{20}$$

В работе *Annals Math. Stat.* **15** (1944), 163–165, Я. Вольфовиц (J. Wolfowitz) показал, что величины R_1, R_2, ..., R_{t-1}, R'_t становятся нормально распределенными при $n \to \infty$ со средним и ковариацией, приведенными выше. Отсюда следует, что имеет место такой критерий серий. Если задана последовательность из n случайных чисел, то нужно подсчитать число серий R_p длиной p для $1 \le p < t$, а также число серий R'_t длиной t или больше. Пусть

$$\begin{aligned} Q_1 = R_1 - \operatorname{mean}(R_1), \quad \ldots, \quad Q_{t-1} = R_{t-1} - \operatorname{mean}(R_{t-1}), \\ Q_t = R'_t - \operatorname{mean}(R'_t). \end{aligned} \tag{21}$$

Построим матрицу C ковариаций R'_t, например $C_{13} = \operatorname{covar}(R_1, R_3)$, в то время как $C_{1t} = \operatorname{covar}(R_1, R'_t)$. Когда $t = 6$, то

$$C = nC_1 + C_2, \tag{22}$$

где

$$C_1 = \begin{pmatrix} \frac{23}{180} & \frac{-7}{360} & \frac{-5}{336} & \frac{-433}{60480} & \frac{-13}{5670} & \frac{-121}{181440} \\ \frac{-7}{360} & \frac{2843}{20160} & \frac{-989}{20160} & \frac{-7159}{362880} & \frac{-10019}{1814400} & \frac{-1303}{907200} \\ \frac{-5}{336} & \frac{-989}{20160} & \frac{54563}{907200} & \frac{-21311}{1814400} & \frac{-62369}{19958400} & \frac{-7783}{9979200} \\ \frac{-433}{60480} & \frac{-7159}{362880} & \frac{-21311}{1814400} & \frac{886657}{39916800} & \frac{-257699}{239500800} & \frac{-62611}{239500800} \\ \frac{-13}{5670} & \frac{-10019}{1814400} & \frac{-62369}{19958400} & \frac{-257699}{239500800} & \frac{29874811}{5448643200} & \frac{-1407179}{21794572800} \\ \frac{-121}{181440} & \frac{-1303}{907200} & \frac{-7783}{9979200} & \frac{-62611}{239500800} & \frac{-1407179}{21794572800} & \frac{2134697}{1816214400} \end{pmatrix},$$

$$C_2 = \begin{pmatrix} \frac{83}{180} & \frac{-29}{180} & \frac{-11}{210} & \frac{-41}{12096} & \frac{91}{25920} & \frac{41}{18144} \\ \frac{-29}{180} & \frac{-305}{4032} & \frac{319}{20160} & \frac{2557}{72576} & \frac{10177}{604800} & \frac{413}{64800} \\ \frac{-11}{210} & \frac{319}{20160} & \frac{-58747}{907200} & \frac{19703}{604800} & \frac{239471}{19958400} & \frac{39517}{9979200} \\ \frac{-41}{12096} & \frac{2557}{72576} & \frac{19703}{604800} & \frac{-220837}{4435200} & \frac{1196401}{239500800} & \frac{360989}{239500800} \\ \frac{91}{25920} & \frac{10177}{604800} & \frac{239471}{19958400} & \frac{1196401}{239500800} & \frac{-139126639}{7264857600} & \frac{4577641}{10897286400} \\ \frac{41}{18144} & \frac{413}{64800} & \frac{39517}{9979200} & \frac{360989}{239500800} & \frac{4577641}{10897286400} & \frac{-122953057}{21794572800} \end{pmatrix},$$

если $n \geq 12$. Сейчас образуем матрицу $A = (a_{ij})$, обратную к матрице C, и вычислим $\sum_{i,j=1}^{t} Q_i Q_j a_{ij}$. Результат для больших n должен иметь приближенно χ^2-распределение с t степенями свободы.

Матрица A, заданная ранее в (11), равняется матрице, обратной к C_1, с точностью до пяти значащих цифр. Настоящая обратная матрица A равна $n^{-1}C_1^{-1} - n^{-2}C_1^{-1}C_2C_1^{-1} + n^{-3}C_1^{-1}C_2C_1^{-1}C_2C_1^{-1} - \cdots$, и это приводит к тому, что $C_1^{-1}C_2C_1^{-1}$ очень приближенно равна $-6C^{-1}$. Поэтому $V \approx Q^T C_1^{-1} Q/(n-6)$.

H. Критерий "максимум-t". Обозначим $V_j = \max(U_{tj}, U_{tj+1}, \ldots, U_{tj+t-1})$ для $0 \leq j < n$. Применим критерий Колмогорова-Смирнова к последовательности V_0, $V_1, \ldots, V_{n-1}$. Таким образом проверим гипотезу о том, что функция распределения V_j равна $F(x) = x^t$, $0 \leq x \leq 1$. Можно также применить критерий Колмогорова-Смирнова к последовательности $V_0^t, V_1^t, \ldots, V_{n-1}^t$, проверяя гипотезу о равномерном распределении.

Для обоснования критерия необходимо показать, что функция распределения V_j равна $F(x) = x^t$. Вероятность того, что $\max(U_1, U_2, \ldots, U_t) \leq x$, равна вероятности того, что одновременно $U_1 \leq x$, $U_2 \leq x$, ..., $U_t \leq x$, которая, в свою очередь, равна произведению вероятностей $U_k \leq x$ при $k = 1, \ldots, t$, а именно — $xx \ldots x = x^t$.

I. Критерий конфликтов. χ^2-критерий можно применять только тогда, когда ненулевое число элементов ожидается в каждой категории. Но существует критерий другого вида, который можно использовать, когда число категорий намного больше числа наблюдений. Этот критерий имеет отношение к рандомизации — важному методу информационного поиска; он будет рассматриваться в разделе 6.4.

Предположим, что имеется m урн, и поместим в них наудачу n шаров, причем m намного больше n. Большинство шаров попадет в пустые урны, но если шар попадет в урну, в которой уже содержится хотя бы один шар, то будем говорить, что произошел "конфликт". Критерий конфликтов подсчитывает число конфликтов, и генератор удовлетворяет этому критерию, если не возникает слишком много или слишком мало конфликтов.

Для определенности предположим, что $m = 2^{20}$ и $n = 2^{14}$. Тогда в среднем на 64 урны приходится только один шар. Вероятность того, что в конкретную урну попадет ровно k шаров, равна $p_k = \binom{n}{k} m^{-k}(1-m^{-1})^{n-k}$, поэтому среднее число конфликтов в урне равно

$$\sum_{k\geq 1}(k-1)p_k = \sum_{k\geq 0} kp_k - \sum_{k\geq 1} p_k = \frac{n}{m} - 1 + p_0.$$

Так как $p_0 = (1-m^{-1})^n = 1 - nm^{-1} + \binom{n}{2}m^{-2}$ — маленькое число, получим, что общее среднее число конфликтов во всех m урнах намного меньше $n^2/(2m) = 128$. (Истинное значение равно ≈ 127.33.)

Критерий конфликтов можно использовать для того, чтобы оценивать генератор случайных чисел для строк больших размерностей. Например, когда $m = 2^{20}$ и $n = 2^{14}$, можно проверять 20-мерную случайность генератора случайных чисел, положив $d = 2$ и сформировав 20-мерный вектор $V_j = (Y_{20j}, Y_{20j+1}, \ldots, Y_{20j+19})$ для $0 \leq j < n$. Мы храним таблицу из $m = 2^{20}$ двоичных разрядов для определения конфликтов и один двоичный разряд для каждого возможного значения координаты вектора V_j; на компьютере с 32 двоичными разрядами в слове это дает 2^{15} слов.

Сначала во все 2^{20} двоичных разрядов таблицы занесем 0; затем для каждого V_j, если в соответствующий двоичный разряд уже занесена 1, регистрируем конфликт, в противном случае заносим в двоичный разряд 1. Этот критерий можно использовать для размерности 10 при $d = 4$ и т. д.

Чтобы решить, проходит ли последовательность это испытание, можно использовать следующую таблицу процентных точек, когда $m = 2^{20}$ и $n = 2^{14}$

Конфликты $\le$	101	108	119	126	134	145	153
С вероятностью	.009	.043	.244	.476	.742	.946	.989

Для определения этих вероятностей используется та же теория, что и для покер-критерия (5); вероятность, что произойдет c конфликтов, равна вероятности того, что будут заняты $n - c$ урн, а именно

$$\frac{m(m-1)\dots(m-n+c+1)}{m^n} \left\{ {n \atop n-c} \right\}.$$

Хотя m и n очень большие, не составляет труда вычислить эти вероятности, используя следующий метод.

Алгоритм S (*Процентные точки для критерия конфликтов*). Пусть заданы m и n. Этот алгоритм определяет распределение числа конфликтов, происходящих, когда n шаров рассеиваются по m урнам. Для вычислений используется вспомогательная таблица $A[0]$, $A[1]$, ..., $A[n]$ чисел с плавающей запятой; фактически $A[j]$ будет не равным 0 только для $j_0 \le j \le j_1$ и $j_1 - j_0$ будет иметь порядок, не больший, чем $\log n$, поэтому их можно получить с использованием значительно меньшего объема памяти.

S1. [Инициализация.] Присвоить $A[j] \leftarrow 0$ для $0 \le j \le n$; затем присвоить $A[1] \leftarrow 1$ и $j_0 \leftarrow j_1 \leftarrow 1$. Повторить шаг S2 ровно $n - 1$ раз и перейти к шагу S3.

S2. [Корректировка вероятностей.] (Этот шаг соответствует бросанию шара в урну; $A[j]$ — вероятность того, что заняты точно j урн.) Присвоить $j_1 \leftarrow j_1 + 1$. Затем для $j \leftarrow j_1, j_1 - 1, \dots, j_0$ (в таком порядке) присвоить $A[j] \leftarrow (j/m)A[j] + \big((1 + 1/m) - (j/m)\big)A[j-1]$. Если $A[j]$ стало очень малым в результате вычислений, скажем, $A[j] < 10^{-20}$, присвоить $A[j] \leftarrow 0$; в таком случае уменьшить j_1 на 1, если $j = j_1$, или увеличить j_0 на 1, если $j = j_0$.

S3. [Вычисление ответа.] На этом шаге будет использоваться вспомогательная таблица $(T_1, T_2, \dots, T_{\text{tmax}}) = (.01, .05, .25, .50, .75, .95, .99, 1.00)$, содержащая точно определенные интересующие нас процентные точки. Присвоить $p \leftarrow 0$, $t \leftarrow 1$ и $j \leftarrow j_0 - 1$. Выполнять следующие итерации, пока не будет достигнут $t = \text{tmax}$: увеличить j на 1 и присвоить $p \leftarrow p + A[j]$. Затем, если $p > T_t$, выход $n - j - 1$ и $1 - p$ (имеется в виду, что с вероятностью $1 - p$ существует по крайней мере $n - j - 1$ конфликтов); иначе — увеличиваем t на 1 до тех пор, пока $p \le T_t$. ▮

J. Критерий промежутков между днями рождений. Джордж Марсалья в 1984 году ввел новый критерий. Поместим n шаров в m урн, как и в критерии конфликтов, но под урнами подразумеваем дни года, а под шарами — дни рождения. Предположим, что $(Y_1, \dots, Y_n)$ — это дни рождения, где $0 \le Y_k < m$. Расположим их в порядке неубывания $Y_{(1)} \le \dots \le Y_{(n)}$, определим n "промежутков" $S_1 = Y_{(2)} - Y_{(1)}$, ..., $S_{n-1} = Y_{(n)} - Y_{(n-1)}$, $S_n = Y_{(1)} + m - Y_{(n)}$ и, наконец, расположим промежутки

в таком порядке: $S_{(1)} \le \cdots \le S_{(n)}$. Пусть R — число равных промежутков, а именно — число индексов j, таких, что $1 < j \le n$ и $S_{(j)} = S_{(j-1)}$. Когда $m = 2^{25}$ и $n = 512$, должно получиться следующее.

$R =$	0	1	2	3 или больше
С вероятностью	.368801577	.369035243	.183471182	.078691997

(Среднее число равных промежутков для выбранных m и n должно быть равным приблизительно 1.) Примените этот критерий, скажем, 1 000 раз и воспользуйтесь χ^2-критерием с тремя степенями свободы, чтобы сравнить эмпирические значения R_i с правильным распределением. Так можно выяснить, будет ли генератор вырабатывать приемлемые случайные промежутки между днями рождений. В упр. 28–30 развивается теория, связанная с этим критерием, и приводятся формулы для других значений m и n.

Такой критерий, как критерий промежутков между днями рождений, в первую очередь, важен в связи с тем замечательным фактом, что генератор Фибоначчи с запаздыванием *не удовлетворяет* этому критерию, хотя он вполне удовлетворяет другим традиционным критериям. [Драматические примеры таких неудач приведены Марсальи, Земаном и Тсангом (Marsaglia, Zaman, and Tsang, *Stat. and Prob. Letters* **8** (1990), 35–39).] Рассмотрим, например, последовательность

$$X_n = (X_{n-24} + X_{n-55}) \bmod m$$

из 3.2.2–(7). Числа этой последовательности удовлетворяют соотношению

$$X_n + X_{n-86} \equiv X_{n-24} + X_{n-31} \quad (\text{по модулю } m),$$

так как обе стороны конгруэнтны $X_{n-24} + X_{n-55} + X_{n-86}$. Поэтому две пары разностей равны

$$X_n - X_{n-24} \equiv X_{n-31} - X_{n-86}$$

и

$$X_n - X_{n-31} \equiv X_{n-24} - X_{n-86}.$$

Всякий раз, когда X_n приемлемо близко к X_{n-24} или X_{n-31} (как должно было бы произойти в настоящей случайной последовательности), одна и та же разность имеет хороший шанс появиться в двух промежутках. Получится значительно больше случаев равенств (обычно со средним $R \approx 2$, а не 1). Но если исключить из R любой равный промежуток, возникающий из условий конгруэнтности, то полученная статистика R' будет удовлетворять критерию дней рождений. (Один из способов избежать неудачи состоит в том, чтобы отбрасывать определенные элементы последовательности, используя, например, только $X_0, X_2, X_4, \ldots$ как случайные числа. Тогда мы никогда не получим все четыре элемента множества $\{X_n, X_{n-24}, X_{n-31}, X_{n-86}\}$ и в критерии промежутков между днями рождений исчезнут проблемы. Существует даже лучший способ избавиться от проблем — отбросить последовательные группы чисел, как предложил Люшер (Lüscher); см. раздел 3.2.2. Подобные замечания можно использовать для генераторов вычитания с заимствованием и суммирования с переносом из упр. 3.2.1.1–14.)

K. Критерий сериальной корреляции. Можно подсчитать следующую статистику:

$$C = \frac{n(U_0U_1+U_1U_2+\cdots+U_{n-2}U_{n-1}+U_{n-1}U_0)-(U_0+U_1+\cdots+U_{n-1})^2}{n(U_0^2+U_1^2+\cdots+U_{n-1}^2)-(U_0+U_1+\cdots+U_{n-1})^2}. \tag{23}$$

Это коэффициенты сериальной корреляции, мера зависимости U_{j+1} от U_j.

Коэффициенты корреляции часто появляются в статистических работах. Если задано n величин U_0, U_1, ..., U_{n-1} и n других величин V_0, V_1, ..., V_{n-1}, то коэффициент корреляции между ними определяется следующим образом:

$$C = \frac{n\sum(U_jV_j)-(\sum U_j)(\sum V_j)}{\sqrt{\left(n\sum U_j^2-(\sum U_j)^2\right)\left(n\sum V_j^2-(\sum V_j)^2\right)}}. \tag{24}$$

Все суммирования в этой формуле производятся по области $0 \le j < n$; формула (23) — это частный случай последней формулы для $V_j = U_{(j+1) \bmod n}$. Знаменатель в (24) равен 0, когда $U_0 = U_1 = \cdots = U_{n-1}$ или $V_0 = V_1 = \cdots = V_{n-1}$, поэтому мы исключаем из рассмотрения такой случай.

Коэффициент корреляции всегда лежит между -1 и $+1$. Когда он равен 0 или очень мал, значит, величины U_j и V_j независимы одна от другой (говоря более точно, между ними нет линейной зависимости. — *Прим. ред.*); если же значение коэффициента корреляции равно ±1, это означает полную линейную зависимость. Действительно, в таком случае $V_j = \alpha \pm \beta U_j$ для всех j и некоторых констант α и β (см. упр. 17).

Следовательно, C в (23) должно быть как можно ближе к 0. На самом деле, поскольку U_0U_1 не является полностью независимым от U_1U_2, нельзя ожидать, что сериальный коэффициент корреляции будет *точно* равен 0 (см. упр. 18). "Хорошим" значением C будет значение, находящееся между $\mu_n - 2\sigma_n$ и $\mu_n + 2\sigma_n$, где

$$\mu_n = \frac{-1}{n-1}, \qquad \sigma_n^2 = \frac{n^2}{(n-1)^2(n-2)}, \qquad n > 2. \tag{25}$$

Ожидается, что C находится между этими значениями в 95% случаев.

Формула для σ_n^2 в (25) — это верхняя граница сериальных корреляций между произвольно распределенными независимыми переменными. Когда U_i равномерно распределены, то истинная дисперсия равна $\frac{24}{5}n^{-2} + O(n^{-7/3}\log n)$ (см. упр. 20).

Вместо обычного коэффициента корреляции между наблюдениями $(U_0, U_1, \ldots, U_{n-1})$ и их соседями $(U_1, \ldots, U_{n-1}, U_0)$ можно вычислить коэффициент корреляции между $(U_0, U_1, \ldots, U_{n-1})$ и любой циклически смещенной последовательностью $(U_q, \ldots, U_{n-1}, U_0, \ldots, U_{q-1})$. Циклическая корреляция должна быть небольшой для $0 < q < n$. Непосредственное вычисление по формуле (24) для всех q потребует около n^2 операций умножения, однако на самом деле можно подсчитать все корреляции всего за $O(n \log n)$ шагов, если использовать быстрое преобразование Фурье. (См. раздел 4.6.4, а также работу L. P. Schmid, *CACM* **8** (1965), 115.)

L. Критерий подпоследовательностей. Внешние программы часто вызывают случайные числа пакетами. Например, если программа работает со случайными переменными X, Y и Z, она может потребовать одновременного генерирования трех

случайных чисел. В таких ситуациях важно, чтобы подпоследовательность, образующая каждую *тройку* членов первоначальной последовательности, была случайной. Если программа требует сразу q чисел, то последовательность

$$U_0, U_q, U_{2q}, \ldots; \quad U_1, U_{q+1}, U_{2q+1}, \ldots; \quad \ldots; \quad U_{q-1}, U_{2q-1}, U_{3q-1}, \ldots$$

может быть проверена с помощью описанных выше критериев для первоначальной последовательности U_0, U_1, U_2, ...

Опыты с линейными конгруэнтными последовательностями показали, что поведение этих порожденных последовательностей редко бывает менее случайным, чем поведение первоначальной последовательности, если только q не имеет большого множителя, общего с длиной периода. На бинарном компьютере с m, равным длине компьютерного слова, например, критерий подпоследовательностей для $q = 8$ даст результаты, худшие среди всех $q < 16$, а на десятичном компьютере значение $q = 10$ приведет к получению наиболее неудовлетворительных подпоследовательностей. (Это можно объяснить, в некоторой степени основываясь на потенциале, так как подобные значения q имеют тенденцию к уменьшению потенциала. В упр. 3.2.1.2–20 приведено более подробное объяснение.)

М. Исторические замечания и дальнейшее обсуждение. Статистические критерии, естественно, возникли в связи с потребностью доказать или опровергнуть гипотезы относительно различных наблюдений. Хорошо известны две ранние работы, посвященные проверке случайности искусственно генерируемых чисел. Это две статьи М. Дж. Кендалла и Б. Бабингтон-Смита (M. G. Kendall and B. Babington-Smith, *Journal of the Royal Statistical Society* **101** (1938), 147–166; также в этом журнале см. **6** (1939), 51–61). В приведенных работах внимание было сосредоточено в большей степени на проверке случайности цифр между 0 и 9, чем на случайности реальных чисел; с этой целью авторы обсуждали критерий частот, критерий серий, критерий интервалов и покер-критерий, однако они неудачно применяли критерий серий. Кендалл и Бабингтон-Смит использовали также вариант критерия собирания купонов. Метод, описанный в этом разделе, был введен Р. Е. Гринвудом (R. E. Greenwood, *Math. Comp.* **9** (1955), 1–5.)

Критерий монотонности имеет довольно интересную историю. Первоначально он рассматривался для восходящих и нисходящих серий одновременно. Восходящие серии следовали за нисходящими, затем — восходящие серии и т. д. Заметим, что критерий монотонности и критерий перестановок зависят не от того, имеет ли U_n равномерное распределение, а от того факта, что $U_i = U_j$ появляется с вероятностью 0, когда $i \neq j$. Поэтому данные критерии могут применяться ко многим типам случайных последовательностей. Критерий монотонности в примитивной форме введен И. Ж. Бьенэйме (J. Bienaymé, *Comptes Rendus Acad. Sci. Paris* **81** (1875), 417–423). Примерно через шестьдесят лет В. О. Кермак и А. Г. Мак-Кендрик опубликовали две обширные статьи на эту тему (W. O. Kermack and A. G. McKendrick, *Proc. Royal Society Edinburgh* **57** (1937), 228–240, 332–376). В качестве примера они установили, что количество выпавших в Эдинбурге осадков между 1785 и 1930 годами носило "всецело случайный характер" по отношению к критерию монотонности (хотя они проверяли только среднее и стандартное отклонения длин серий). Другие исследователи также начали использовать этот критерий, но только в 1944 году было показано, что использовать χ^2-метод в этом критерии неправомочно.

В работе Х. Левена и Я. Вольфовица (H. Levene and J. Wolfowitz, *Annals Math. Stat.* **15** (1944), 58–69) введен правильный критерий монотонности (для чередующихся восходящих и нисходящих серий). В ней же обсуждались ошибки при использовании этого критерия ранее. Отдельные критерии для восходящих и нисходящих серий, предложенные выше в настоящем разделе, больше подходят для использования на компьютере, поэтому мы не приводим сложные формулы для чередования восходящих и нисходящих серий. (См. обзорную работу Д. Э. Бартона и К. Л. Маллоу [D. E. Barton and C. L. Mallows, *Annals Math. Stat.* **36** (1965), 236–260].)

Среди всех рассмотренных выше критериев критерий частот и критерий сериальной корреляции кажутся более слабыми в том смысле, что почти все генераторы случайных чисел им удовлетворяют. Теоретические обоснования такого явления кратко обсуждаются в разделе 3.5 (см. упр. 3.5–26). С другой стороны, критерий монотонности является более строгим. Результаты упр. 3.3.3–23 и 24 показывают, что линейные конгруэнтные генераторы стремятся иметь серии, в некоторой степени более длинные, чем нормальные, если множитель недостаточно большой. Поэтому рекомендуется применять критерий монотонности, приведенный в упр. 14.

Критерий конфликтов также заслуживает самых наилучших рекомендаций, так как он специально предназначен для определения недостатков многих, к сожалению, широко распространенных генераторов. Основанный на идеях Х. Делгаса Христиансена (H. Delgas Christiansen) [Inst. Math. Stat. and Oper. Res., Tech. Univ. Denmark (October, 1975); не опубликовано], этот критерий получил распространение с появлением компьютеров; он предназначен для использования на компьютере и не подходит для вычислений вручную.

Читатель, вероятно, удивлен: *"Зачем применять такое количество критериев?"*. Может показаться, что на проверку случайных чисел затрачивается больше компьютерного времени, чем на их использование! Это неверно, хотя можно переусердствовать в проверке.

Необходимость проверки последовательности с помощью нескольких критериев неоднократно отмечалась в литературе. Исследователи проверили, например, что некоторые генераторы чисел наподобие метода средин квадратов удовлетворяли критерию частот, критерию интервалов и покер-критерию, однако не удовлетворяли критерию серий. Линейные конгруэнтные последовательности с малым множителем, как известно, были проверены многими критериями, однако не выдержали проверку критерием монотонности, так как у них слишком мало серий единичной длины. Критерий "максимум-t" можно также использовать для поиска плохих генераторов, которые без проверки этим критерием кажутся вполне респектабельными. Генератор "вычитания с заимствованием" не проходит проверку критерием интервалов, когда максимальная длина интервала превышает самое большое запаздывание; см. работу Ваттулайнена, Канкаала, Сааринена и Ала-Ниссила (Vattulainen, Kankaala, Saarinen, and Ala-Nissila, *Computer Physics Communications* **86** (1995), 209–226), в которой идет речь о многих других критериях. Генератор Фибоначчи с запаздыванием, для которого теоретически гарантировано, что он имеет равнораспределенные младшие значащие разряды, тем не менее не удовлетворяет простым вариантам одноразрядного критерия равномерности (см. упр. 31 и 35, а также 3.6–14).

Возможно, основной смысл экстенсивной проверки генераторов случайных чисел состоит в том, что людям, неправильно применяющим генератор случайных чисел господина X, будет тяжело допустить, что на самом деле неправильна их собственная программа. Они будут обвинять генератор до тех пор, пока господин X не *докажет* им, что его числа достаточно случайны. С другой стороны, если источник случайных чисел предназначен только для личного использования господином X, последний может не беспокоиться о его проверке, так как с большой вероятностью технических рекомендаций, приведенных в настоящей главе, достаточно для проверки этого датчика.

Чем больше используются компьютеры, тем больше необходимо случайных чисел, и генераторы случайных чисел, которые раньше считались вполне удовлетворительными, теперь недостаточно хороши для применения в физике, комбинаторике, стохастической геометрии и т. д. Джордж Марсалья в связи с этим ввел *строгие критерии*, которые превосходят классические методы, подобные критерию интервалов и покер-критерию, в том смысле, что они по сравнению с этими критериями замечают другие отклонения. Например, он нашел, что последовательность $X_{n+1} = (62605X_n + 113218009) \bmod 2^{29}$ имеет значительное отклонение в следующем эксперименте. Было получено 2^{21} случайных чисел X_n и выделено 10 их старших двоичных разрядов $Y_n = \lfloor X_n/2^{19} \rfloor$. Подсчитано, сколько из 2^{20} возможных пар (y, y') 10-разрядных чисел *не* появятся среди (Y_1, Y_2), (Y_2, Y_3), ..., $(Y_{2^{21}-1}, Y_{2^{21}})$. Должно быть примерно 141909.33 отсутствующих пар со стандартным отклонением ≈ 290.46 (см. упр. 34). Но в шести последовательных испытаниях, начиная с $X_1 = 1234567$, выполнили расчеты и получили значение стандартного отклонения между 1.5 и 3.5, которое получилось слишком низким. Распределение оказалось слишком "плоским" для того, чтобы быть случайным, поскольку, вероятно, из 2^{21} чисел значимыми дробями являются только 1/256 на весь период. Подобный генератор с множителем 69069 и модулем 2^{30}, как показано, является лучшим. Марсалья и Земан (Zaman) назвали эту процедуру "обезьяний критерий", так как она позволяет подсчитать количество двухсимвольных комбинаций, которые обезьяна пропустит после печатанья на клавиатуре с 1 024 клавишами (см. *Computers and Math.* **26**, 9 (November, 1993), 1–10, где приведен анализ нескольких "обезьяньих критериев").

УПРАЖНЕНИЯ

1. [*10*] Почему критерий серий, приведенный в п. B, следует применять к (Y_0, Y_1), (Y_2, Y_3), ..., (Y_{2n-2}, Y_{2n-1}) вместо пар (Y_0, Y_1), (Y_1, Y_2), ..., (Y_{n-1}, Y_n)?

2. [*10*] Представьте приблизительный путь обобщения критерия серий для троек, четверок и т. д. вместо пар.

▸ **3.** [*M20*] Сколько U_j нужно проверить в критерии интервалов (алгоритм G), прежде чем будет найдено в среднем n интервалов, если предположить, что последовательность случайна? Чему равно стандартное отклонение этой величины?

4. [*M12*] Докажите, что вероятности в (4) верны и для критерия интервалов.

5. [*M23*] "Классический" критерий интервалов Кендалла и Бабингтон-Смита рассматривает числа $U_0, U_1, \ldots, U_{N-1}$ как циклическую последовательность с U_{N+j}, совпадающую с U_j. Здесь N — фиксированное число U_j, определяемое критерием. Если n — это число $U_0, \ldots, U_{N-1}$, попадающих в интервал $\alpha \le U_j < \beta$, то существует n интервалов в циклической последовательности. Пусть Z_r — число интервалов длиной r для $0 \le r < t$ и пусть

Z_t — число интервалов длиной $\geq t$. Покажите, что величина $V = \sum_{0\leq r\leq t}(Z_r - np_r)^2/np_r$ должна иметь предельное χ^2-распределение с t степенями свободы, когда N стремится к бесконечности, а p_r заданы в (4).

6. [*40*] (Х. Гейрингер (H. Geiringer).) Подсчитав частоты первых 2 000 десятичных чисел в представлении $e = 2.71828\ldots$, мы получили χ^2-значение 1.06, указывающее, что действительные частоты цифр 0, 1, ..., 9 намного ближе к их ожидаемым значениям, чем при случайном распределении. (На самом деле $\chi^2 \geq 1.15$ с вероятностью 99.9%.) Этот же критерий, примененный к первым 10 000 цифр e, дает приемлемое значение $\chi^2 = 8.61$; но тот факт, что первые 2 000 цифр так равномерно распределены, достоин удивления. Будет ли происходить то же самое, если записать e в системе счисления с другим основанием? [См. *AMM* **72** (1965), 483–500.]

7. [*08*] Примените процедуру критерия собирания купонов (алгоритм C) с $d = 3$ и $n = 7$ к последовательности 11012210221202020012122010102011 21. Чему равны длины семи последовательностей?

▶ **8.** [*M22*] Сколько в среднем U_j нужно проверить в критерии собирания купонов, прежде чем с помощью алгоритма C будет найдено n полных множеств при предположении, что последовательность случайна? Чему равно стандартное отклонение? [*Указание.* См. формулу 1.2.9–(28).]

9. [*M21*] Обобщите критерий собирания купонов, чтобы поиск прекращался, как только будет найдено w различных значений, где w — фиксированное положительное целое число, меньшее или равное d. Какие вероятности следует использовать вместо (6)?

10. [*M23*] Выполните упр. 8 для более общего критерия собирания купонов, описанного в упр. 9.

11. [*00*] Восходящие серии в конкретной перестановке показаны в (9). Каковы нисходящие серии в этой перестановке?

12. [*20*] Пусть U_0, U_1, ..., U_{n-1} — n различных чисел. Напишите алгоритм, определяющий длины всех восходящих серий в этой последовательности. Когда ваш алгоритм завершит работу, `COUNT`$[r]$ должен быть равен числу серий длиной r для $1 \leq r \leq 5$, а `COUNT`$[6]$ — числу серий длиной 6 или больше.

13. [*M23*] Покажите, что (16) — это число перестановок из $p+q+1$ различных элементов, имеющих структуру (15).

▶ **14.** [*M15*] Если "выбросить" элемент, который следует непосредственно за серией, то, когда X_j больше X_{j+1}, начнем следующую серию с X_{j+2}. Длины серий независимы, и можно использовать обычный χ^2-критерий (вместо ужасно сложного метода, описанного в разделе). Чему приближенно равны вероятности для длин серий этого простого критерия монотонности?

15. [*M10*] Почему в критерии "максимум-t" предполагается, что величины V_0^t, V_1^t, ..., V_{n-1}^t равномерно распределены между нулем и единицей?

▶ **16.** [*15*] Господин Дж. Г. Квик ("Студент") хотел использовать критерий "максимум-t" для нескольких различных значений t.

a) Положив $Z_{jt} = \max(U_j, U_{j+1}, \ldots, U_{j+t-1})$, он нашел простой путь перехода от последовательности $Z_{0(t-1)}$, $Z_{1(t-1)}$, ... к последовательности Z_{0t}, Z_{1t}, ..., для которой требуется очень мало времени и места. В чем состояла его блестящая идея?

b) Он решил модифицировать метод "максимум-t" так, чтобы j-м наблюдением было $\max(U_j, \ldots, U_{j+t-1})$; другими словами, он взял $V_j = Z_{jt}$ вместо $V_j = Z_{(tj)t}$, как предлагается в разделе. Квик считал, что *все* Z_j должны иметь одно и то же распределение, поэтому критерий будет даже сильнее, если использовать каждое Z_{jt}, $0 \leq j < n$,

вместо каждого t-го члена. Но когда он применил χ^2-критерий в случае одинаковых распределений к величинам V_j^t, получились чрезмерно высокие значения статистики V, которые увеличивались при возрастании t. Почему так произошло?

17. [*M25*] Даны любые числа $U_0, \ldots, U_{n-1}, V_0, \ldots, V_{n-1}$. Пусть их средние значения равны

$$\bar{u} = \frac{1}{n}\sum_{0\le k<n} U_k, \qquad \bar{v} = \frac{1}{n}\sum_{0\le k<n} V_k.$$

a) Пусть $U_k' = U_k - \bar{u}$, $V_k' = V_k - \bar{v}$. Покажите, что коэффициент корреляции C, определенный в (24), равен

$$\sum_{0\le k<n} U_k'V_k' \Big/ \sqrt{\sum_{0\le k<n} {U_k'}^2}\sqrt{\sum_{0\le k<n} {V_k'}^2}.$$

b) Пусть $C = N/D$, где N и D — числитель и знаменатель выражения из п. (a). Покажите, что $N^2 \le D^2$, отсюда $-1 \le C \le 1$. Получите формулы для разности $D^2 - N^2$. [*Указание.* См. упр. 1.2.3–30.]

c) Если $C = \pm 1$, покажите, что $\alpha U_k + \beta V_k = \tau$, $0 \le k < n$, для некоторых не всех равных нулю постоянных α, β и τ.

18. [*M20*] (a) Покажите, что, если $n = 2$, сериальный коэффициент корреляции (23) всегда равен -1 (если знаменатель не равен нулю). (b) Аналогично покажите, что, когда $n = 3$, сериальный коэффициент корреляции всегда равен $-\frac{1}{2}$. (c) Покажите, что знаменатель в (23) равен нулю тогда и только тогда, когда $U_0 = U_1 = \cdots = U_{n-1}$.

19. [*M30*] (Дж. П. Батлер (J. P. Butler).) Пусть $U_0, \ldots, U_{n-1}$ — независимые одинаково распределенные случайные величины. Докажите, что ожидаемое значение сериального коэффициента корреляции (23), среднее по всем случаям с ненулевым знаменателем, равно $-1/(n-1)$.

20. [*HM41*] Продолжая предыдущее упражнение, докажите, что дисперсия (23) равна $n^2/(n-1)^2(n-2) - n^3\,\mathrm{E}((U_0-U_1)^4/D^2)/2(n-2)$, где D — знаменатель из (23), а E — ожидаемое значение по всем случаям, когда $D \ne 0$. Чему равно асимптотическое значение $\mathrm{E}((U_0-U_1)^4/D^2)$, когда каждое U_j равномерно распределено?

21. [*19*] Какие значения f будут вычислены алгоритмом P, если применить его к перестановкам $(1,2,9,8,5,3,6,7,0,4)$?

22. [*18*] Для какой перестановки $\{0,1,2,3,4,5,6,7,8,9\}$ алгоритм P выдаст значение $f = 1024$?

23. [*M22*] Пусть $\langle Y_n\rangle$ и $\langle Y_n'\rangle$ — последовательности целых чисел, имеющие периоды длиной λ и λ' соответственно с $0 \le Y_n, Y_n' < d$. Пусть также $Z_n = (Y_n + Y'_{n+r}) \bmod d$, где r выбрано наудачу между 0 и $\lambda' - 1$. Покажите, что $\langle Z_n\rangle$ удовлетворяет t-мерному критерию серий по крайней мере так же, как $\langle Y_n\rangle$, в следующем смысле. Пусть $P(x_1,\ldots,x_t)$ и $Q(x_1,\ldots,x_t)$ — вероятности того, что t-мерная строка $(x_1,\ldots,x_t)$ появляется в $\langle Y_n\rangle$ и $\langle Z_n\rangle$:

$$P(x_1,\ldots,x_t) = \frac{1}{\lambda}\sum_{n=0}^{\lambda-1}[(Y_n,\ldots,Y_{n+t-1}) = (x_1,\ldots,x_t)];$$

$$Q(x_1,\ldots,x_t) = \frac{1}{\lambda\lambda'}\sum_{n=0}^{\lambda-1}\sum_{r=0}^{\lambda'-1}[(Z_n,\ldots,Z_{n+t-1}) = (x_1,\ldots,x_t)].$$

Тогда $\displaystyle\sum_{(x_1,\ldots,x_t)} (Q(x_1,\ldots,x_t) - d^{-t})^2 \le \sum_{(x_1,\ldots,x_t)} (P(x_1,\ldots,x_t) - d^{-t})^2.$

24. [*HM35*] (Дж. Марсалья (G. Marsaglia).) Покажите, что критерий серий для n пересекающихся t-мерных строк $(Y_1, Y_2, \ldots, Y_t)$, $(Y_2, Y_3, \ldots, Y_{t+1})$, $\ldots$, $(Y_n, Y_{n+1}, \ldots, Y_{t+n-1})$ может быть осуществлен следующим образом. Для каждой цепочки $\alpha = a_1 \ldots a_m$ с $0 \le a_i < d$ положим $N(\alpha)$ равным числу случаев, когда α появляется как подцепочка $Y_1, Y_2, \ldots, Y_n$, $Y_{n+1}, \ldots, Y_{n+m-1}$, и пусть $P(\alpha) = P(a_1) \ldots P(a_m)$ — вероятность того, что α появляется в любом заданном положении. Разные цифры могут появляться с различными вероятностями $P(0)$, $P(1)$, $\ldots$, $P(d-1)$. Подсчитаем статистику

$$V = \frac{1}{n} \sum_{|\alpha|=t} \frac{N(\alpha)}{P(\alpha)} - \frac{1}{n} \sum_{|\alpha|=t-1} \frac{N(\alpha)}{P(\alpha)}.$$

Тогда V должно иметь χ^2-распределение с $d^t - d^{t-1}$ степенями свободы, когда n большое. [*Указание.* Используйте упр. 3.3.1–25.]

25. [*M46*] Почему выполняется приближенное равенство $C_1^{-1} C_2 C_1^{-1} \approx -6C_1^{-1}$, где C_1 и C_2 — матрицы, определенные в (22)?

26. [*HM30*] Пусть $U_1, U_2, \ldots, U_n$ независимы и равномерно распределены в $[0\,..\,1)$ и пусть $U_{(1)} \le U_{(2)} \le \cdots \le U_{(n)}$ — их значения после сортировки. Определите разности $S_1 = U_{(2)} - U_{(1)}, \ldots, S_{n-1} = U_{(n)} - U_{(n-1)}$, $S_n = U_{(1)} + 1 - U_{(n)}$ и рассортируйте их $S_{(1)} \le \cdots \le S_{(n)}$, как в критерии промежутков между днями рождений. В следующих вычислениях удобно использовать обозначение x_+^n в качестве сокращенной записи выражения $x^n[x \ge 0]$.

a) Пусть даны любые действительные числа $s_1, s_2, \ldots, s_n$. Докажите, что вероятность того, что неравенства $S_1 \ge s_1$ и $S_2 \ge s_2$, $\ldots$, $S_n \ge s_n$ выполняются одновременно, равна $(1 - s_1 - s_2 - \cdots - s_n)_+^{n-1}$.

b) Докажите, следовательно, что наименьшая разность $S_{(1)}$ будет $\le s$ с вероятностью $1 - (1 - ns)_+^{n-1}$.

c) Какова функция распределения $F_k(s) = \Pr(S_{(k)} \le s)$ для $1 \le k \le n$?

d) Вычислите среднее и дисперсию каждого $S_{(k)}$.

▶ **27.** [*HM26*] (*Итерированные разности.*) В обозначениях предыдущего упражнения покажите, что числа $S_1' = nS_{(1)}$, $S_2' = (n-1)(S_{(2)} - S_{(1)})$,..., $S_n' = 1(S_{(n)} - S_{(n-1)})$ имеют то же самое совместное вероятностное распределение, что и первоначальные разности $S_1, \ldots, S_n$ n равномерно распределенных случайных величин. Можно упорядочить их в виде $S_{(1)}' \le \cdots \le S_{(n)}'$ и повторить это преобразование, чтобы получить еще одно множество случайных разностей $S_1'', \ldots, S_n''$ и т. д. Каждое последующее множество разностей $S_1^{(k)}, \ldots, S_n^{(k)}$ может быть также проверено по критерию Колмогорова-Смирнова, где

$$K_{n-1}^+ = \sqrt{n-1} \max_{1 \le j < n} \left(\frac{j}{n-1} - S_1^{(k)} - \cdots - S_j^{(k)} \right),$$

$$K_{n-1}^- = \sqrt{n-1} \max_{1 \le j < n} \left(S_1^{(k)} + \cdots + S_j^{(k)} - \frac{j-1}{n-1} \right).$$

Детально проверьте преобразование $(S_1, \ldots, S_n)$ в $(S_1', \ldots, S_n')$ для $n = 2$ и $n = 3$. Объясните, почему постоянное повторение этого процесса в конечном счете приведет к неудаче, если его применять к компьютерному генератору чисел с ограниченной точностью. (Один из методов сравнения генераторов случайных чисел состоит в наблюдении, как долго они проживут при таких мучительных испытаниях.)

28. [*M26*] Пусть $b_{nrs}(m)$ — число n-мерных строк $(y_1, \ldots, y_n)$ с $0 \le y_j < m$, имеющих точно r равных разностей и s нулевых разностей. Таким образом, вероятность того, что $R = r$, в критерии промежутков между днями рождений равна $\sum_{s=0}^{r+1} b_{nrs}(m)/m^n$. Также пусть $p_n(m)$ — число разбиений m на не более чем n частей (упр. 5.1.1–15). (a) Выразите $b_{n00}(m)$ в терминах разбиений. [*Указание.* Рассмотрите случай с малыми m и n.]

(b) Покажите, что существует простое соотношение между $b_{nrs}(m)$ и $b_{(n-s)(r+1-s)0}(m)$, когда $s > 0$. (c) Выведите точную формулу вероятности того, что разности равны.

29. [*M35*] Продолжая упр. 28, найдите простое выражение для производящих функций $b_{nr}(z) = \sum_{m\geq 0} b_{nr0}(m)z^m/m$, когда $r = 0$, 1 и 2.

30. [*HM41*] Продолжая предыдущее упражнение, докажите, что если $m = n^3/\alpha$, то

$$p_n(m) = \frac{m^{n-1}e^{\alpha/4}}{n!\,(n-1)!}\Big(1 - \frac{13\alpha^2}{288n} + \frac{169\alpha^4 + 2016\alpha^3 - 1728\alpha^2 - 41472\alpha}{165888n^2} + O(n^{-3})\Big)$$

для фиксированного α при $n \to \infty$. Найдите аналогичную формулу для $q_n(m)$ — числа разбиений m на n *различных* положительных частей. Выведите с точностью до $O(1/n)$ асимптотические вероятности того, что в критерии промежутков между днями рождений R равно 0, 1 и 2.

▸ **31.** [*M21*] Рекуррентное соотношение $Y_n = (Y_{n-24} + Y_{n-55}) \bmod 2$, описывающее младший значимый разряд генератора Фибоначчи с запаздыванием 3.2.2–(7), как и второй младший значащий разряд 3.2.2–(7′), как известно, имеет период длиной $2^{55} - 1$; следовательно, каждая возможная не равная нулю конфигурация двоичных разрядов $(Y_n, Y_{n+1}, \ldots, Y_{n+54})$ появляется одинаково часто. Тем не менее докажите, что если генерировать 79 последовательных двоичных разрядов $Y_n, \ldots, Y_{n+78}$, начиная со случайной точки в периоде, вероятность, что там будет больше единиц, чем нулей, больше 51%. Если использовать такие двоичные разряды для определения "случайного блуждания", состоящего в том, что точка движется вправо, когда двоичный разряд содержит 1, и влево, когда двоичный разряд содержит 0, то в большинстве случаев будем заканчивать блуждание справа от нашей начальной точки. [*Указание.* Найдите производящую функцию $\sum_{k=0}^{79} \Pr(Y_n + \cdots + Y_{n+78} = k)\, z^k$.]

32. [*M20*] Определите, верно ли следующее: если X и Y — независимые одинаково распределенные случайные величины со средним 0 и если для них более вероятно быть положительными, чем отрицательными, то $X + Y$ более вероятно будет положительным, чем отрицательным.

33. [*HM32*] Найдите асимптотическое значение вероятности того, что $k + l$ последовательных двоичных разрядов, генерируемых рекуррентным соотношением $Y_n = (Y_{n-l} + Y_{n-k}) \bmod 2$, имеют больше единиц, чем нулей, когда $k > 2l$ и длина периода этой рекуррентной последовательности равна $2^k - 1$. Предполагается, что k большое.

34. [*HM29*] Объясните, как оценить среднее и дисперсию числа двухбуквенных комбинаций, не появляющихся последовательно в случайной строке длиной n в m-буквенном алфавите. Предположим, что m большое и $n \approx 2m^2$.

▸ **35.** [*HM32*] (Дж. Н. Линдхолм (J. H. Lindholm), 1968.) Предположим, что случайные двоичные разряды $\langle Y_n \rangle$ генерируются с помощью рекуррентного соотношения

$$Y_n = (a_1 Y_{n-1} + a_2 Y_{n-2} + \cdots + a_k Y_{n-k}) \bmod 2$$

для некоторого набора $a_1, \ldots, a_k$ с периодом длиной $2^k - 1$. Начнем со значений $Y_0 = 1$ и $Y_1 = \cdots = Y_{k-1} = 0$. Пусть $Z_n = (-1)^{Y_n+1} = 2Y_n - 1$ — случайный знак. Рассмотрим статистику $S_m = Z_n + Z_{n+1} + \cdots + Z_{n+m-1}$, где n — случайная точка в периоде.

a) Докажите, что $\mathrm{E}\, S_m = m/N$, где $N = 2^k - 1$.

b) Чему равно $\mathrm{E}\, S_m^2$? Предположите, что $m \leq N$. [*Указание.* См. упр. 3.2.2–16.]

c) Чему равнялись бы $\mathrm{E}\, S_m$ и $\mathrm{E}\, S_m^2$, будь Z_j действительно случайными?

d) Предполагая, что $m \leq N$, докажите равенство $\mathrm{E}\, S_m^3 = m^3/N - 6B(N+1)/N$, где

$$B = \sum_{0<i<j<m} [(Y_{i+1}Y_{i+2}\ldots Y_{i+k-1})_2 = (Y_{j+1}Y_{j+2}\ldots Y_{j+k-1})_2]\,(m-j)\,.$$

е) Оцените B в частном случае, рассмотренном в упр. 31: $m = 79$ и $Y_n = (Y_{n-24} + Y_{n-55}) \bmod 2$.

*3.3.3. Теоретические критерии

Несмотря на то что каждый генератор случайных чисел можно проверить с помощью критериев, приведенных в предыдущем разделе, всегда лучше иметь *априорный* критерий, а именно — теоретический результат, с помощью которого можно заранее сказать, насколько хороши будут проверки генератора. Такие теоретические результаты помогают понять методы генерирования намного лучше, чем эмпирические методы проб и ошибок. В этом разделе мы подробнее изучим линейные конгруэнтные последовательности. Если можно предугадать результаты определенных проверок заранее, прежде чем генерировать случайные числа, то имеется больше возможностей должным образом выбрать a, m и c.

Развитие теории такого типа весьма сложно, хотя достигнут определенный прогресс. Получены пока далекие от общих результаты для *статистических критериев, которые можно применять к полным периодам.* Тем не менее статистические критерии приобретают смысл, когда они применяются ко всему периоду; например, критерий равномерности будет давать отличные результаты и без этого требования, но критерий серий, критерий интервалов, критерий перестановок, максимум-критерий и другие можно плодотворно проанализировать на всем периоде. Такие исследования будут определять *глобальную* "неслучайность" последовательности, т. е. неправильное поведение очень больших выборок.

Теория, которая будет здесь рассматриваться, вполне всеобъемлюща, однако она не исключает необходимости проверять локальные "неслучайности" методами, приведенными в разделе 3.3.2. Действительно, любая проверка с использованием только коротких последовательностей очень сложна. Известно лишь несколько теоретических результатов, касающихся поведения линейной конгруэнтной последовательности на промежутке, меньшем, чем целый период (они обсуждаются в конце раздела 3.3.4; см. также упр. 18).

Начнем с доказательства простого *априорного* закона для наименее сложного случая — для критерия перестановок. Суть нашей первой теоремы состоит в том, что для последовательности с большим потенциалом примерно в половине случаев выполняется неравенство $X_{n+1} < X_n$.

Теорема P. *Пусть a, c и m образуют линейную конгруэнтную последовательность с максимальным периодом. Пусть $b = a - 1$ и d — наибольший общий делитель m и b. Вероятность того, что $X_{n+1} < X_n$, равна $\frac{1}{2} + r$, где*

$$r = \big(2(c \bmod d) - d\big)/2m; \tag{1}$$

следовательно, $|r| < d/2m$.

Доказательство. Для доказательства этой теоремы используются методы, интересные сами по себе. Сначала определим

$$s(x) = (ax + c) \bmod m. \tag{2}$$

Таким образом, $X_{n+1} = s(X_n)$ и теорема сводится к подсчету количества целых x, таких, что $0 \le x < m$, а $s(x) < x$, поскольку такое число встречается где-то в

периоде. Необходимо показать, что это число равно

$$\tfrac{1}{2}\big(m + 2(c \bmod d) - d\big). \tag{3}$$

Функция $\lceil (x - s(x))/m \rceil$ равна 1, когда $x > s(x)$, и 0 — в противном случае. Следовательно, искомое число, которое мы хотим подсчитать, можно записать следующим образом:

$$\begin{aligned}\sum_{0\le x<m} \left\lceil \frac{x-s(x)}{m} \right\rceil &= \sum_{0\le x<m} \left\lceil \frac{x}{m} - \left(\frac{ax+c}{m} - \left\lfloor \frac{ax+c}{m} \right\rfloor \right) \right\rceil \\ &= \sum_{0\le x<m} \left(\left\lfloor \frac{ax+c}{m} \right\rfloor - \left\lfloor \frac{bx+c}{m} \right\rfloor \right). \end{aligned} \tag{4}$$

(Вспомним, что $\lceil -y \rceil = -\lfloor y \rfloor$ и $b = a-1$.) Такие суммы можно подсчитать, используя методы из упр. 1.2.4–37, в котором было доказано, что

$$\sum_{0\le j<k} \left\lfloor \frac{hj+c}{k} \right\rfloor = \frac{(h-1)(k-1)}{2} + \frac{g-1}{2} + g\lfloor c/g \rfloor, \qquad g = \gcd(h,k), \tag{5}$$

для любых целых h и k, $k > 0$ (gcd — наибольший общий делитель). Так как a и m — взаимно простые числа, эта формула дает

$$\sum_{0\le x<m} \left\lfloor \frac{ax+c}{m} \right\rfloor = \frac{(a-1)(m-1)}{2} + c,$$

$$\sum_{0\le x<m} \left\lfloor \frac{bx+c}{m} \right\rfloor = \frac{(b-1)(m-1)}{2} + \frac{d-1}{2} + c - (c \bmod d)$$

и (3) следует немедленно. ∎

Доказательство теоремы P указывает, что *априорные* критерии можно исследовать, если уметь удовлетворительно обращаться с суммами, содержащими функции $\lfloor\ \rfloor$ и $\lceil\ \rceil$. Во многих случаях наиболее мощной техникой оперирования функциями "пол" и "потолок" является их замена двумя отчасти более симметричными операциями:

$$\delta(x) = \lfloor x \rfloor + 1 - \lceil x \rceil = [x\text{–целое число}]; \tag{6}$$

$$((x)) = x - \lfloor x \rfloor - \tfrac{1}{2} + \tfrac{1}{2}\delta(x) = x - \lceil x \rceil + \tfrac{1}{2} - \tfrac{1}{2}\delta(x) = x - \tfrac{1}{2}\big(\lfloor x \rfloor + \lceil x \rceil\big). \tag{7}$$

Последняя функция — это "пилообразная" функция, встречающаяся в теории рядов Фурье. На рис. 7 приведен ее график. Причина того, что мы предпочитаем работать с функцией $((x))$, а не с $\lfloor x \rfloor$ или $\lceil x \rceil$, состоит в том, что $((x))$ обладает несколькими очень полезными свойствами:

$$((-x)) = -((x)); \tag{8}$$

$$((x+n)) = ((x)) \quad \text{для целого } n; \tag{9}$$

$$((nx)) = ((x)) + \left(\left(x + \frac{1}{n}\right)\right) + \cdots + \left(\left(x + \frac{n-1}{n}\right)\right) \quad \text{для целого } n \ge 1 \tag{10}$$

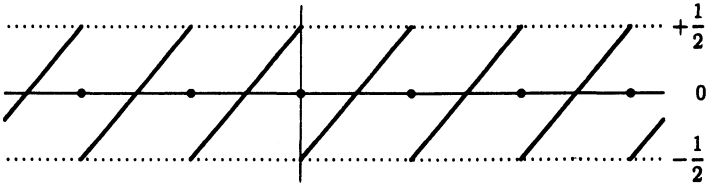


Рис. 7. Пилообразная функция $((x))$.

(см. упр. 2).

Для приобретения практических навыков работы с этими функциями докажем теорему P снова, на этот раз, не используя упр. 1.2.4–37. С помощью равенств (7)–(9) можно показать, что

$$\begin{aligned}\left\lceil\frac{x-s(x)}{m}\right\rceil &= \frac{x-s(x)}{m} - \left(\!\left(\frac{x-s(x)}{m}\right)\!\right) + \frac{1}{2} - \frac{1}{2}\delta\left(\frac{x-s(x)}{m}\right)\\ &= \frac{x-s(x)}{m} - \left(\!\left(\frac{x-(ax+c)}{m}\right)\!\right) + \frac{1}{2}\\ &= \frac{x-s(x)}{m} + \left(\!\left(\frac{bx+c}{m}\right)\!\right) + \frac{1}{2},\end{aligned} \tag{11}$$

так как $\big(x-s(x)\big)/m$ никогда не будет целым числом. Легко видеть, что

$$\sum_{0\le x<m} \frac{x-s(x)}{m} = 0,$$

поскольку и x, и $s(x)$ принимают значение из $\{0,1,\ldots,m-1\}$ точно один раз. Следовательно, (11) влечет

$$\sum_{0\le x<m}\left\lceil\frac{x-s(x)}{m}\right\rceil = \sum_{0\le x<m}\left(\!\left(\frac{bx+c}{m}\right)\!\right) + \frac{m}{2}. \tag{12}$$

Пусть $b=b_0d$, $m=m_0d$, где b_0 и m_0 — взаимно простые числа. Нам известно, что $(b_0x) \bmod m_0$ принимает значения $\{0,1,\ldots,m_0-1\}$ в некотором порядке, когда x изменяется от 0 до m_0-1. Из (9), (10) и равенства

$$\left(\!\left(\frac{b(x+m_0)+c}{m}\right)\!\right) = \left(\!\left(\frac{bx+c}{m}\right)\!\right)$$

получим

$$\begin{aligned}\sum_{0\le x<m}\left(\!\left(\frac{bx+c}{m}\right)\!\right) &= d\sum_{0\le x<m_0}\left(\!\left(\frac{bx+c}{m}\right)\!\right)\\ &= d\sum_{0\le x<m_0}\left(\!\left(\frac{c}{m}+\frac{b_0x}{m_0}\right)\!\right) = d\left(\!\left(\frac{c}{d}\right)\!\right).\end{aligned} \tag{13}$$

Теорема P немедленно следует из (12) и (13).

Одно из следствий теоремы P таково: практически при *любом* выборе a и c можно получить приемлемые вероятности того, что $X_{n+1} < X_n$, по крайней мере

при полном периоде, за исключением больших d. Большие значения d соответствуют низкому потенциалу. (Уже известно, что генераторы с низким потенциалом нежелательны).

Следующая теорема дает более точные условия выбора a и c. Рассмотрим *критерий сериальной корреляции*, применяемый на полном периоде. Величина C, определенная в разделе 3.3.2 (формула (23)), равна

$$C = \left(m \sum_{0\le x<m} x s(x) - \left(\sum_{0\le x<m} x\right)^2\right) \Big/ \left(m \sum_{0\le x<m} x^2 - \left(\sum_{0\le x<m} x\right)^2\right). \tag{14}$$

Пусть x' — такой элемент, что $s(x') = 0$. Тогда

$$s(x) = m\left(\left(\frac{ax+c}{m}\right)\right) + \frac{m}{2}[x \ne x']. \tag{15}$$

Формулы, которые необходимо получить, лучше всего выразить с помощью суммы

$$\sigma(h,k,c) = 12 \sum_{0\le j<k} \left(\left(\frac{j}{k}\right)\right)\left(\left(\frac{hj+c}{k}\right)\right), \tag{16}$$

важной функции, возникающей в некоторых математических задачах. Ее называют *обобщенной суммой Дедекинда*, так как Ричард Дедекинд (Dedekind) ввел функцию $\sigma(h,k,0)$ в 1876 году, когда комментировал неоконченную рукопись Римана. [См. работу B. Riemann, *Gesammelte math. Werke*, 2nd ed. (1892), 466–478.]

Используя достаточно известные формулы

$$\sum_{0\le x<m} x = \frac{m(m-1)}{2} \qquad \text{и} \qquad \sum_{0\le x<m} x^2 = \frac{m(m-\frac{1}{2})(m-1)}{3},$$

можно легко преобразовать (14) в

$$C = \frac{m\sigma(a,m,c) - 3 + 6(m - x' - c)}{m^2 - 1} \tag{17}$$

(см. упр. 5). Так как m обычно очень большое, можно отбросить члены порядка $1/m$ и получить приближенную формулу

$$C \approx \sigma(a,m,c)/m \tag{18}$$

с погрешностью по абсолютной величине, меньшей, чем $6/m$.

Критерий сериальной корреляции сейчас сводится к определению значения суммы Дедекинда $\sigma(a,m,c)$. Вычислять $\sigma(a,m,c)$ непосредственно из определения (16) не легче непосредственного вычисления коэффициента корреляции, но, к счастью, существуют простые методы быстрого подсчета сумм Дедекинда.

Лемма B (*"Закон взаимности" для сумм Дедекинда*). *Пусть h, k и c — целые числа. Если $0 \le c < k$, $0 < h \le k$ и если h и k — взаимно простые числа, то*

$$\sigma(h,k,c) + \sigma(k,h,c) = \frac{h}{k} + \frac{k}{h} + \frac{1}{hk} + \frac{6c^2}{hk} - 6\left\lfloor \frac{c}{h} \right\rfloor - 3e(h,c), \tag{19}$$

где

$$e(h,c) = [c=0] + [c \bmod h \ne 0]. \tag{20}$$

Доказательство. Оставляем читателю доказательство того, что при наших предположениях имеет место равенство

$$\sigma(h,k,c)+\sigma(k,h,c)=\sigma(h,k,0)+\sigma(k,h,0)+\frac{6c^2}{hk}-6\left\lfloor\frac{c}{h}\right\rfloor-3e(h,c)+3 \tag{21}$$

(см. упр. 6). Теперь докажем лемму только для $c=0$.

Приведенное ниже доказательство, в котором используются комплексные корни из единицы, по существу, принадлежит Л. Карлицу (L. Carlitz). Это действительно более простое доказательство, чем доказательство с использованием элементарных преобразований сумм (см. упр. 7), но для следующего метода необходимо больше математических понятий, чем требуется обычно для задач такого типа, поэтому он более поучителен.

Пусть $f(x)$ и $g(x)$ — полиномы, определенные следующим образом:

$$\begin{aligned} f(x)&=1+x+\cdots+x^{k-1}=(x^k-1)/(x-1),\\ g(x)&=x+2x^2+\cdots+(k-1)x^{k-1}\\ &=xf'(x)=kx^k/(x-1)-x(x^k-1)/(x-1)^2. \end{aligned} \tag{22}$$

Если ω — комплексный k-й корень из единицы $e^{2\pi i/k}$, то из формул 1.2.9–(13) получим

$$\frac{1}{k}\sum_{0\le j<k}\omega^{-jr}g(\omega^j x)=rx^r,\qquad\text{если } 0\le r<k. \tag{23}$$

Положим $x=1$, тогда $g(\omega^j x)=k/(\omega^j-1)$, если $j\neq 0$, иначе это выражение равно $k(k-1)/2$. Поэтому

$$r \bmod k=\sum_{0<j<k}\frac{\omega^{-jr}}{\omega^j-1}+\tfrac{1}{2}(k-1),\qquad\text{если } r \text{ целое.}$$

(Равенство (23) показывает, что правая часть равна r, когда $0\le r<k$, и она не изменяется при прибавлении к r числа, кратного k.) Следовательно,

$$\left(\!\left(\frac{r}{k}\right)\!\right)=\frac{1}{k}\sum_{0<j<k}\frac{\omega^{-jr}}{\omega^j-1}-\frac{1}{2k}+\frac{1}{2}\delta\left(\frac{r}{k}\right). \tag{24}$$

Эта важная формула, которая справедлива для всех целых r, позволяет свести многие вычисления, включающие $((r/k))$, к суммам, содержащим k-й корень единицы. Она предоставляет новые возможности для вычислений. В частности, получим следующую формулу, когда $h\perp k$:

$$\sigma(h,k,0)+\frac{3(k-1)}{k^2}=\frac{12}{k^2}\sum_{0<r<k}\sum_{0<i<k}\sum_{0<j<k}\frac{\omega^{-ir}}{\omega^i-1}\frac{\omega^{-jhr}}{\omega^j-1}. \tag{25}$$

Правая ее часть может быть упрощена, если просуммировать ее по r. Получим $\sum_{0\le r<k}\omega^{rs}=f(\omega^s)=0$, если $s \bmod k\neq 0$. Равенство (25) сведется к

$$\sigma(h,k,0)+\frac{3(k-1)}{k}=\frac{12}{k}\sum_{0<j<k}\frac{1}{(\omega^{-jh}-1)(\omega^j-1)}. \tag{26}$$

Аналогичная формула получена для $\sigma(k,h,0)$ с $\zeta = e^{2\pi i/h}$ вместо ω.

Не ясно, что делать с суммой (26), однако существует элегантный метод ее преобразования, основанный на том факте, что каждый член суммы является функцией от ω^j, где $0 < j < k$. Следовательно, суммы, в сущности, берутся по k-м корням из единицы, отличным от 1. Когда x_1, x_2, ..., x_n — различные комплексные числа, получаем равенство

$$\sum_{j=1}^{n} \frac{1}{(x_j-x_1)\dots(x_j-x_{j-1})(x-x_j)(x_j-x_{j+1})\dots(x_j-x_n)} = \frac{1}{(x-x_1)\dots(x-x_n)}, \tag{27}$$

которое можно вывести в результате применения обычного метода разложения на элементарные дроби в правой части этого равенства. Кроме того, если $q(x) = (x-y_1)(x-y_2)\dots(x-y_m)$, справедливо

$$q'(y_j) = (y_j-y_1)\dots(y_j-y_{j-1})(y_j-y_{j+1})\dots(y_j-y_m); \tag{28}$$

это тождество часто используется для упрощения выражений, подобных левой части (27). Когда h и k — взаимно простые, все числа ω, ω^2, ..., ω^{k-1}, ζ, ζ^2, ..., ζ^{h-1} различны. Поэтому можно рассмотреть формулу (27) в частном случае для полинома $(x-\omega)\dots(x-\omega^{k-1})(x-\zeta)\dots(x-\zeta^{h-1}) = (x^k-1)(x^h-1)/(x-1)^2$, получив следующее тождество для x:

$$\frac{1}{h}\sum_{0<j<h} \frac{\zeta^j(\zeta^j-1)^2}{(\zeta^{jk}-1)(x-\zeta^j)} + \frac{1}{k}\sum_{0<j<k} \frac{\omega^j(\omega^j-1)^2}{(\omega^{jh}-1)(x-\omega^j)} = \frac{(x-1)^2}{(x^h-1)(x^k-1)} \tag{29}$$

Оно имеет много интересных следствий и позволяет получить многочисленные формулы для сумм наподобие (26). Например, если (29) дважды продифференцировать по x и устремить $x \to 1$, получится

$$\frac{2}{h}\sum_{0<j<h} \frac{\zeta^j(\zeta^j-1)^2}{(\zeta^{jk}-1)(1-\zeta^j)^3} + \frac{2}{k}\sum_{0<j<k} \frac{\omega^j(\omega^j-1)^2}{(\omega^{jh}-1)(1-\omega^j)^3} = \frac{1}{6}\Big(\frac{h}{k}+\frac{k}{h}+\frac{1}{hk}\Big) + \frac{1}{2} - \frac{1}{2h} - \frac{1}{2k}.$$

Заменим j на $h-j$ и на $k-j$ в этой сумме и используем (26), чтобы получить равенство

$$\frac{1}{6}\left(\sigma(k,h,0) + \frac{3(h-1)}{h}\right) + \frac{1}{6}\left(\sigma(h,k,0) + \frac{3(k-1)}{k}\right) = \frac{1}{6}\Big(\frac{h}{k}+\frac{k}{h}+\frac{1}{hk}\Big) + \frac{1}{2} - \frac{1}{2h} - \frac{1}{2k},$$

эквивалентное желаемому результату. ▮

Лемма B дает явно заданную функцию $f(h,k,c)$, такую, что

$$\sigma(h,k,c) = f(h,k,c) - \sigma(k,h,c), \tag{30}$$

какими бы ни были взаимно простые числа h и k, такие, что $0 < h \le k$, $0 \le c < k$. Из определения (16) следует, что

$$\sigma(k,h,c) = \sigma(k \bmod h,\ h,\ c \bmod h). \tag{31}$$

Поэтому можно повторно использовать (30) для оценки $\sigma(h,k,c)$ путем уменьшения параметров, как в алгоритме Евклида.

Дальнейшие упрощения произойдут, когда мы более подробно исследуем эту итеративную процедуру. Положим, что $m_1 = k$, $m_2 = h$, $c_1 = c$, и построим следующую таблицу.

$$\begin{aligned} m_1 &= a_1 m_2 + m_3 & \qquad c_1 &= b_1 m_2 + c_2 \\ m_2 &= a_2 m_3 + m_4 & \qquad c_2 &= b_2 m_3 + c_3 \\ m_3 &= a_3 m_4 + m_5 & \qquad c_3 &= b_3 m_4 + c_4 \\ m_4 &= a_4 m_5 & \qquad c_4 &= b_4 m_5 + c_5 \end{aligned} \tag{32}$$

Здесь

$$\begin{aligned} a_j &= \lfloor m_j / m_{j+1} \rfloor, & \qquad b_j &= \lfloor c_j / m_{j+1} \rfloor, \\ m_{j+2} &= m_j \bmod m_{j+1}, & \qquad c_{j+1} &= c_j \bmod m_{j+1}. \end{aligned} \tag{33}$$

Отсюда следует, что

$$0 \le m_{j+1} < m_j, \qquad 0 \le c_j < m_j. \tag{34}$$

Для удобства предположим, что алгоритм Евклида закончится в (32) после четырех итераций. Это предположение будет иметь структуру, подобную той, которая наблюдается в общем случае. Так как h и k — взаимно простые числа, для начала в (32) следует положить $m_5 = 1$ и $c_5 = 0$.

Допустим также, что $c_3 \neq 0$, но $c_4 = 0$ для того, чтобы получить эффект от применения рекуррентной процедуры. Равенства (30) и (31) дают

$$\begin{aligned} \sigma(h,k,c) &= \sigma(m_2, m_1, c_1) \\ &= f(m_2, m_1, c_1) - \sigma(m_3, m_2, c_2) \\ &= \cdots \\ &= f(m_2, m_1, c_1) - f(m_3, m_2, c_2) + f(m_4, m_3, c_3) - f(m_5, m_4, c_4). \end{aligned}$$

Первая часть $h/k + k/h$ формулы для $f(h,k,c)$ в (19) приводит к добавлению

$$\frac{m_2}{m_1} + \frac{m_1}{m_2} - \frac{m_3}{m_2} - \frac{m_2}{m_3} + \frac{m_4}{m_3} + \frac{m_3}{m_4} - \frac{m_5}{m_4} - \frac{m_4}{m_5}$$

к общей формуле, которая сводится к

$$\frac{h}{k} + \frac{m_1 - m_3}{m_2} - \frac{m_2 - m_4}{m_3} + \frac{m_3 - m_5}{m_4} - \frac{m_4}{m_5} = \frac{h}{k} + a_1 - a_2 + a_3 - a_4.$$

Следующая часть (19), $1/hk$, также приводит к простому добавлению; в соответствии с 4.5.3–(9) и другими формулами раздела 4.5.3 получим

$$\frac{1}{m_1 m_2} - \frac{1}{m_2 m_3} + \frac{1}{m_3 m_4} - \frac{1}{m_4 m_5} = \frac{h'}{k} - 1, \tag{35}$$

где h' — единственное целое число, удовлетворяющее

$$h'h \equiv 1 \ (\text{по модулю } k), \quad 0 < h' \le k. \tag{36}$$

Добавляя все эти слагаемые, вспомним о предположении, что $c_4 = 0$ (так что $e(m_4, c_3) = 0$, см. (20)). Находим, что

$$\sigma(h,k,c) = \frac{h+h'}{k} + (a_1 - a_2 + a_3 - a_4) - 6(b_1 - b_2 + b_3 - b_4) + 6\left(\frac{c_1^2}{m_1 m_2} - \frac{c_2^2}{m_2 m_3} + \frac{c_3^2}{m_3 m_4} - \frac{c_4^2}{m_4 m_5}\right) + 2$$

в терминах таблицы (32). Подобный результат имеет место в общем случае.

Теорема D. *Пусть h, k и c — целые числа с $0 < h \le k$, $0 \le c < k$ и h, взаимно простым с k. Образуем "таблицу Евклида", как определено в* (32), *и предположим, что процесс остановится после t шагов с $m_{t+1} = 1$. Пусть s — наименьший индекс, для которого $c_s = 0$, и пусть число h' определено в* (36). *Тогда*

$$\sigma(h,k,c) = \frac{h+h'}{k} + \sum_{1\le j\le t} (-1)^{j+1}\left(a_j - 6b_j + 6\frac{c_j^2}{m_j m_{j+1}}\right) + 3\big((-1)^s + \delta_{s1}\big) - 2 + (-1)^t. \quad \blacksquare$$

Алгоритм Евклида тщательно анализируется в разделе 4.5.3; величины a_1, a_2, ..., a_t называются *частичными отношениями* h/k. Теорема 4.5.3F указывает, что число итераций t никогда не превосходит $\log_\phi k$; следовательно, суммы Дедекинда могут быть быстро вычислены. Члены $c_j^2/m_j m_{j+1}$ можно упростить еще больше; эффективный алгоритм для оценки $\sigma(h,k,c)$ можно найти в упр. 17.

Изучив обобщенные суммы Дедекинда, применим полученные знания для вычисления сериального коэффициента корреляции.

Пример 1. *Найти сериальный коэффициент, когда $m = 2^{35}$, $a = 2^{34} + 1$, $c = 1$.*

Решение. Из (17) следует, что справедливо соотношение

$$C = \big(2^{35}\sigma(2^{34}+1,\, 2^{35},\, 1) - 3 + 6(2^{35} - (2^{34} - 1) - 1)\big)/(2^{70} - 1).$$

Чтобы вычислить $\sigma(2^{34}+1,\, 2^{35}, 1)$, можно образовать следующую таблицу.

$m_1 = 2^{35}$		$c_1 = 1$	
$m_2 = 2^{34}+1$	$a_1 = 1$	$c_2 = 1$	$b_1 = 0$
$m_3 = 2^{34}-1$	$a_2 = 1$	$c_3 = 1$	$b_2 = 0$
$m_4 = 2$	$a_3 = 2^{33}-1$	$c_4 = 1$	$b_3 = 0$
$m_5 = 1$	$a_4 = 2$	$c_5 = 0$	$b_4 = 1$

Так как $h' = 2^{34} + 1$, это значение в соответствии с теоремой D становится равным $2^{33} - 3 + 2^{-32}$. Таким образом,

$$C = (2^{68} + 5)/(2^{70} - 1) = \tfrac{1}{4} + \epsilon, \qquad |\epsilon| < 2^{-67}. \tag{37}$$

Подобная корреляция намного превышает ту, которая должна быть у случайной последовательности. Конечно, этот генератор имеет очень низкий потенциал и его можно было бы отбросить и раньше, как неслучайный.

Пример 2. *Найти приближенное значение коэффициента сериальной корреляции, когда* $m = 10^{10}$, $a = 10001$, $c = 2113248653$.

Решение. Справедливо равенство $C \approx \sigma(a, m, c)/m$. Вычисления производим следующим образом:

$$\begin{array}{llllll} m_1 = 10000000000 & & & c_1 = 2113248653 & & \\ m_2 = 10001 & a_1 = 999900 & & c_2 = 7350 & b_1 = 211303 & \\ m_3 = 100 & a_2 = 100 & & c_3 = 50 & b_2 = 73 & \\ m_4 = 1 & a_3 = 100 & & c_4 = 0 & b_3 = 50 & \end{array}$$

$$\sigma(m_2, m_1, c_1) = -31.6926653544; \qquad C \approx -3 \cdot 10^{-9}. \tag{38}$$

На самом деле это очень приемлемое значение C. Но потенциал генератора равен только 3, *так что реально это не очень хороший датчик случайных чисел, несмотря на то что у него низкая сериальная корреляция.* Таким образом, необходимо (но не достаточно) иметь низкий сериальный коэффициент корреляции.

Пример 3. *Оценить сериальную корреляцию для произвольных* a, m *и* c.

Решение. Если применить только один раз соотношение (30), то получится

$$\sigma(a, m, c) \approx \frac{m}{a} + 6\frac{c^2}{am} - 6\frac{c}{a} - \sigma(m, a, c).$$

Поскольку из упр. 12 следует, что $|\sigma(m, a, c)| < a$, справедливы соотношения

$$C \approx \frac{\sigma(a, m, c)}{m} \approx \frac{1}{a}\left(1 - 6\frac{c}{m} + 6\left(\frac{c}{m}\right)^2\right). \tag{39}$$

Ошибка этой аппроксимации по абсолютной величине меньше $(a + 6)/m$.

Оценка в (39) — это первый известный теоретический результат, касающийся случайности конгруэнтных последовательностей (генераторов). Р. Р. Ковэю (R. R. Coveyou) [*JACM* **7** (1960), 72–74] получил его методом усреднения по всем *действительным* числам x, лежащим между 0 и m, вместо того, чтобы рассматривать только целые числа (см. упр. 21). Затем Мартин Гринбергер (Martin Greenberger) [*Math. Comp.* **15** (1961), 383–389] нашел точное отклонение, включая и оценку ошибки.

Так началась одна из самых грустных глав в истории компьютерной науки! Хотя приведенная выше аппроксимация была вполне корректной, она была совершенно неприемлемой на практике. Люди отбросили хорошие генераторы и заменили их ужасными генераторами, казавшимися хорошими с точки зрения критерия (39). Более чем на десятилетие репутация наиболее используемых генераторов случайных чисел была серьезно подорвана только в связи с прогрессом теории.

Небольшие познания — опасная вещь.

— АЛЕКСАНДР ПОП (ALEXANDER POPE),
Эссе о критике, 215 (1711)

Если серьезно проанализировать ошибки, то можно лучше понять, как были допущены злоупотребления в (39). Во-первых, кое-кто некритично предполагал, что маленькая сериальная корреляция по целому периоду является хорошей гарантией

случайности. На самом же деле она не может обеспечить даже маленькую сериальную корреляцию для 1 000 последовательных элементов последовательности (см. упр. 14).

Во-вторых, (39) и точность приближения обеспечивают относительно малое значение C только при $a \approx \sqrt{m}$. Поэтому предлагалось выбирать множитель, равный примерно $\sqrt{m}$. На самом деле почти все множители дают значения C, постоянно меньшие, чем $1/\sqrt{m}$, поэтому (39) не является очень хорошей аппроксимацией истинного поведения оценки. Минимизация грубой верхней границы C не минимизирует само значение C.

В-третьих, было замечено, что (39) дает свои самые лучшие оценки, когда

$$c/m \approx \tfrac{1}{2} \pm \tfrac{1}{6}\sqrt{3}, \tag{40}$$

так как эти значения являются корнями уравнения $1 - 6x + 6x^2 = 0$. "При отсутствии любого другого критерия выбора c можно использовать и этот." Последнее утверждение имеет смысл, но оно вводит в лучшем случае в заблуждение, так как эксперимент показал, что значение c оказывает большое влияние на истинное значение сериального коэффициента корреляции, когда a — хороший множитель. Выбор (40) в значительной степени снижает значение C только в случаях, аналогичных примеру 2. И мы обманываем себя, так как плохой множитель продемонстрирует свои недостатки другим путем.

Ясно, что необходима лучшая оценка, чем (39). Такая оценка сейчас доступна благодаря теореме D, в основных чертах доказанной в работе Ульриха Дитера (Ulrich Dieter) [*Math. Comp.* **25** (1971), 855–883]. В соответствии с теоремой D $\sigma(a, m, c)$ *будет малым, если частичные отношения* a/m *малы.* Кроме того, детальнее анализируя обобщенные суммы Дедекинда, можно получить вполне точные оценки.

Теорема К. *При предположениях теоремы D всегда выполняется*

$$-\frac{1}{2}\sum_{\substack{1\le j\le t\\ j\text{ odd}}} a_j - \sum_{\substack{1\le j\le t\\ j\text{ even}}} a_j \le \sigma(h,k,c) \le \sum_{\substack{1\le j\le t\\ j\text{ odd}}} a_j + \frac{1}{2}\sum_{\substack{1\le j\le t\\ j\text{ even}}} a_j - \frac{1}{2}. \tag{41}$$

Доказательство. Обратитесь к работе Д. Э. Кнута (D. E. Knuth, *Acta Arithmetica* **33** (1978), 297–325), в которой, кроме всего прочего, показано, что эти границы, по существу, — наилучшие возможные границы, когда частичные отношения большие. ▮

Пример 4. *Оцените сериальную корреляцию для* $a = 3141592621$, $m = 2^{35}$ *и нечетного* c.

Решение. Частичные отношения a/m равны 10, 1, 14, 1, 7, 1, 1, 1, 3, 3, 3, 5, 2, 1, 8, 7, 1, 4, 1, 2, 4, 2. Таким образом, по теореме К

$$-55 \le \sigma(a, m, c) \le 67.5$$

и сериальная корреляция гарантированно будет крайне низкой для всех c.

Заметим, что эта граница значительно лучше, чем можно было получить из (39), так как ошибка в (39) имеет порядок a/m. Наш "случайный" множитель не может быть настолько лучше специально выбранного множителя, чтобы хорошо выглядеть

при использовании (39). На самом деле можно показать, что *среднее* значение $\sum_{j=1}^{t} a_j$, взятое по всем множителям a, относительно простым с m, равно

$$\frac{6}{\pi^2}(\ln m)^2 + O\big((\log m)(\log\log m)^4\big)$$

(см. упр. 4.5.3–35). Поэтому вероятность того, что случайный множитель имеет большую сумму $\sum_{j=1}^{t} a_j$, скажем, больше $(\log m)^{2+\epsilon}$ для некоторого фиксированного $\epsilon > 0$, стремится к нулю при $m \to \infty$. Это доказывает эмпирически очевидный момент, что почти все линейные конгруэнтные последовательности имеют чрезвычайно низкую сериальную корреляцию по целому периоду.

В упражнениях, приведенных ниже, показано, что другие *априорные* критерии, такие как критерий серий по целому периоду, могут быть выражены и в терминах небольшого обобщения сумм Дедекинда. Из теоремы К следует, что линейная конгруэнтная последовательность будет удовлетворять этим критериям тогда и только тогда, когда определенные дроби (зависящие от a и m, но не от c) имеют малые частичные отношения. В частности, из результатов упр. 19 следует, что *проверка по критерию серий для пар удовлетворительна тогда и только тогда, когда* a/m *имеет небольшие частичные отношения.*

В книге *Суммы Дедекинда* Ганса Радемахера и Эмиля Гросвальда (Hans Rademacher and Emil Grosswald, Math. Assoc. of America, Carus Monograph No. 16, 1972) обсуждается история и свойства сумм Дедекинда и их обобщений. Другие теоретические критерии, в том числе критерий серий для больших размерностей, рассматриваются в разделе 3.3.4.

УПРАЖНЕНИЯ (часть 1)

1. [*M10*] Выразите $x \bmod y$ в терминах "пилообразной" и δ-функций.

2. [*M20*] Докажите "реплективный закон", равенство (10).

3. [*HM22*] Найдите разложение в ряд Фурье (по синусам и косинусам) функции $((x))$.

▶ **4.** [*M19*] Если $m = 10^{10}$, то какое максимальное значение возможно для d (в обозначениях теоремы P), если дано, что потенциал генератора равен 10?

5. [*M21*] Получите формулу (17).

6. [*M27*] Предположим, что $hh' + kk' = 1$.

a) Покажите без использования леммы B, что

$$\sigma(h,k,c) = \sigma(h,k,0) + 12\sum_{0<j<c}\left(\left(\frac{h'j}{k}\right)\right) + 6\left(\left(\frac{h'c}{k}\right)\right)$$

для всех целых $c \geq 0$.

b) Покажите, что $\left(\left(\frac{h'j}{k}\right)\right) + \left(\left(\frac{k'j}{h}\right)\right) = \frac{j}{hk} - \frac{1}{2}\delta\left(\frac{j}{h}\right)$, если $0 < j < k$.

c) Основываясь на предположениях леммы B, докажите равенство (21).

▶ **7.** [*M24*] Докажите закон взаимности (19), когда $c = 0$, используя обобщенный закон взаимности из упр. 1.2.4–45.

▶ **8.** [*M34*] (Л. Карлиц (L. Carlitz).) Пусть

$$\rho(p,q,r) = 12\sum_{0\leq j<r}\left(\left(\frac{jp}{r}\right)\right)\left(\left(\frac{jq}{r}\right)\right).$$

Обобщив метод доказательства, использованный в лемме B, докажите следующее прекрасное тождество Г. Радемахера (H. Rademacher). Если каждые из чисел p, q, r взаимно просты одно относительно другого, то

$$\rho(p,q,r)+\rho(q,r,p)+\rho(r,p,q)=\frac{p}{qr}+\frac{q}{rp}+\frac{r}{pq}-3.$$

(Закон взаимности для сумм Дедекинда при $c=0$ является частным случаем при $r=1$.)

9. [*M40*] Существует ли простое доказательство тождества Радемахера (упр. 8) с использованием в частном случае метода доказательства упр. 7?

10. [*M20*] Покажите, что когда $0<h<k$, то можно легко выразить $\sigma(k-h,k,c)$ и $\sigma(h,k,-c)$ в терминах $\sigma(h,k,c)$.

11. [*M30*] Формулы, приведенные в разделе, показывают, как оценить $\sigma(h,k,c)$, когда h и k — взаимно простые числа и c — целое число. В общем случае докажите, что

a) $\sigma(dh,dk,dc)=\sigma(h,k,c)$ для целых $d>0$;

b) $\sigma(h,k,c+\theta)=\sigma(h,k,c)+6((h'c/k))$ для целых c, действительных $0<\theta<1$, $h\perp k$ и $hh'\equiv 1$ (по модулю k).

12. [*M24*] Покажите, что если h и k — взаимно простые числа и c — целое число, то $|\sigma(h,k,c)|\le(k-1)(k-2)/k$.

13. [*M24*] Обобщите равенство (26) так, чтобы получить выражение для $\sigma(h,k,c)$.

▶ **14.** [*M20*] Линейный конгруэнтный генератор, для которого $m=2^{35}$, $a=2^{18}+1$, $c=1$, был подвергнут сериальному корреляционному критерию для трех групп по 1 000 последовательных чисел. В результате этого была получена очень высокая корреляция, лежащая между 0.2 и 0.3 в каждом случае. Чему равна сериальная корреляция данного генератора, взятая по всем 2^{35} числам периода?

15. [*M21*] Обобщите лемму B так, чтобы ее можно было применять к *действительным* числам c, $0\le c<k$.

16. [*M24*] Задана таблица Евклида, определенная в (32). Пусть $p_0=1$, $p_1=a_1$ и $p_j=a_jp_{j-1}+p_{j-2}$ для $1<j\le t$. Покажите, что сложные части сумм в теореме D могут быть переписаны следующим образом, позволяющим выполнять вычисления с нецелыми числами:

$$\sum_{1\le j\le t}(-1)^{j+1}\frac{c_j^2}{m_jm_{j+1}}=\frac{1}{m_1}\sum_{1\le j\le t}(-1)^{j+1}b_j(c_j+c_{j+1})p_{j-1}.$$

[*Указание.* Докажите тождество $\sum_{1\le j\le r}(-1)^{j+1}/m_jm_{j+1}=(-1)^{r+1}p_{r-1}/m_1m_{r+1}$ для $1\le r\le t$.]

17. [*M22*] Напишите алгоритм вычисления $\sigma(h,k,c)$ для целых h, k, c, удовлетворяющих предположениям теоремы D. Он должен использовать только арифметику целых чисел (с неограниченной точностью), и ответ должен быть записан в виде $A+B/k$, где A и B — целые числа (см. упр. 16.) По возможности используйте только конечное число переменных для временного запоминания вместо того, чтобы вводить такой массив, как $a_1, a_2, \ldots, a_t$.

▶ **18.** [*M23*] (У. Дитер (U. Dieter).) Даны положительные целые числа h, k, z. Пусть

$$S(h,k,c,z)=\sum_{0\le j<z}\left(\!\!\left(\frac{hj+c}{k}\right)\!\!\right).$$

Покажите, что сумма может быть вычислена в приближенной форме в терминах обобщенных сумм Дедекинда и пилообразной функции. [*Указание.* Когда $z\le k$, величина $\lfloor j/k\rfloor-\lfloor(j-z)/k\rfloor$ равна 1 для $0\le j<z$ и равна 0 для $z\le j<k$, поэтому можно включить данный множитель и выполнить суммирование по $0\le j<k$.]

▸ **19.** [*M23*] Покажите, что *критерий серий* можно проанализировать по полному периоду в терминах обобщенных сумм Дедекинда, если найти формулу вероятности того, что $\alpha \le X_n < \beta$ и $\alpha' \le X_{n+1} < \beta'$, когда α, β, α', β' — заданные целые числа, причем $0 \le \alpha < \beta \le m$ и $0 \le \alpha' < \beta' \le m$. [*Указание.* Рассмотрите величину $\lfloor (x-\alpha)/m \rfloor - \lfloor (x-\beta)/m \rfloor$.]

20. [*M29*] (У. Дитер.) Обобщите теорему P, чтобы получить формулу для вероятности того, что $X_n > X_{n+1} > X_{n+2}$, в терминах обобщенных сумм Дедекинда.

УПРАЖНЕНИЯ (часть 2)

Во многих случаях точные вычисления с целыми числами достаточно трудно осуществить, но можно попытаться изучить возникающие вероятности, если усреднить по всем действительным величинам x вместо того, чтобы ограничить вычисление целыми числами. Хотя эти результаты будут только приближенными, они прольют немного света на проблему.

Удобно использовать числа U_n, лежащие между нулем и единицей. Для линейной конгруэнтной последовательности $U_n = X_n/m$ получим, что $U_{n+1} = \{aU_n + \theta\}$, где $\theta = c/m$ и $\{x\}$ определено как $x \bmod 1$. Например, формула для сериальной корреляции примет вид

$$C = \left(\int_0^1 x\{ax+\theta\}\,dx - \left(\int_0^1 x\,dx\right)^2 \right) \Big/ \left(\int_0^1 x^2\,dx - \left(\int_0^1 x\,dx\right)^2 \right).$$

▸ **21.** [*HM23*] (Р. Р. Ковэю (R. R. Coveyou).) Чему равно значение C в только что приведенной формуле?

▸ **22.** [*M22*] Пусть a — целое число и пусть $0 \le \theta < 1$. Если x — действительное число, принимающее значения между 0 и 1, и если $s(x) = \{ax + \theta\}$, чему равна вероятность, что $s(x) < x$? (Это аналог теоремы P для "действительных чисел".)

23. [*M28*] В предыдущем упражнении дана вероятность того, что $U_{n+1} < U_n$. Чему равна вероятность $U_{n+2} < U_{n+1} < U_n$ в предположении, что U_n — случайное действительное число, лежащее между нулем и единицей?

24. [*M29*] Учитывая предположения из предыдущего упражнения и исключая случай для $\theta = 0$, покажите, что $U_n > U_{n+1} > \cdots > U_{n+t-1}$ происходит с вероятностью

$$\frac{1}{t!}\left(1+\frac{1}{a}\right)\cdots\left(1+\frac{t-2}{a}\right).$$

Чему равна средняя длина нисходящих серий, начиная с U_n, где U_n выбрано наудачу между нулем и единицей?

▸ **25.** [*M25*] Пусть α, β, α', β' — действительные числа, $0 \le \alpha < \beta \le 1$, $0 \le \alpha' < \beta' \le 1$. Учитывая предположения из упр. 22, выясните, чему равна вероятность того, что $\alpha \le x < \beta$ и $\alpha' \le s(x) < \beta'$? (Это аналог упр. 19 для "действительных чисел".)

26. [*M21*] Рассмотрите генератор Фибоначчи, где $U_{n+1} = \{U_n + U_{n-1}\}$. Предполагая, что U_1 и U_2 независимо наудачу выбраны между 0 и 1, найдите вероятность того, что $U_1 < U_2 < U_3$, $U_1 < U_3 < U_2$, $U_2 < U_1 < U_3$ и т. д. [*Указание.* Разделите единичный квадрат $\{(x,y) \mid 0 \le x, y < 1\}$ на шесть частей, зависящих от относительного порядка x, y и $\{x+y\}$, и определите площадь каждой части.]

27. [*M32*] В генераторе Фибоначчи из предыдущего упражнения будем считать, что U_0 и U_1 независимо выбраны в единичном квадрате, однако исключается следующее неравенство: $U_0 > U_1$. *Определите вероятность, что U_1 является началом возрастающей серии длиной k*, так что $U_0 > U_1 < \cdots < U_k > U_{k+1}$. Сравните с соответствующими вероятностями для случайной последовательности.

28. [*M35*] В соответствии с формулой 3.2.1.3–(5) линейный конгруэнтный генератор с потенциалом 2 удовлетворяет условию $X_{n-1} - 2X_n + X_{n+1} \equiv (a-1)c$ (по модулю m). Рассмотрим генератор, который обобщает предыдущий. Пусть $U_{n+1} = \{\alpha + 2U_n - U_{n-1}\}$. Как в упр. 26, разделите единичный квадрат на части, которые указывают относительный порядок U_1, U_2 и U_3 для каждой пары (U_1, U_2). Существует ли какое-нибудь значение α, для которого все шесть возможных упорядочений имеют вероятность $\frac{1}{6}$, если предположить, что U_1 и U_2 выбраны наудачу в единичном квадрате?

3.3.4. Спектральный критерий

В этом разделе рассматривается особенно важный метод проверки качества линейных конгруэнтных генераторов случайных чисел. Все хорошие генераторы проходят проверку спектральным критерием; все генераторы, известные сейчас как плохие, фактически *провалились* при этой проверке. Таким образом, спектральный критерий является наиболее мощным известным до сих пор критерием и заслуживает особого внимания. В процессе обсуждения будут выяснены те ограничения на степени случайности, которые не могут быть преодолены при использовании линейных конгруэнтных последовательностей и их обобщений.

Спектральный критерий обладает свойствами как теоретических, так и эмпирических критериев, рассмотренных в предыдущих разделах. Он похож и на теоретические критерии, поскольку проверяет свойства последовательности на полном периоде, и на эмпирические критерии, поскольку для получения результата требует вычислений на компьютере.

А. Идеи, служащие обоснованием критерия. Наиболее важные испытания для проверки, насколько случайной будет последовательность, связаны со свойствами совместных распределений t последовательных элементов последовательности, и спектральный критерий как раз и используется для проверки гипотез об этих распределениях. Если задана последовательность $\langle U_n \rangle$ с периодом m, то для построения критерия необходимо проанализировать множество всех m точек

$$\{\,(U_n, U_{n+1}, \ldots, U_{n+t-1}) \mid 0 \le n < m\,\} \tag{1}$$

в t-мерном пространстве.

Для простоты предположим, что задана линейная конгруэнтная последовательность (X_0, a, c, m) с максимальным периодом длиной m (так что $c \neq 0$) или что m — простое число и $c = 0$, а период имеет длину $m - 1$. В последнем случае прибавим точку $(0, 0, \ldots, 0)$ к множеству (1), чтобы всегда было ровно m точек. Эта дополнительная точка почти не влияет на общую ситуацию, когда m большое, и делает теорию более простой. При таком предположении (1) можно переписать следующим образом:

$$\left\{ \frac{1}{m}\big(x, s(x), s(s(x)), \ldots, s^{[t-1]}(x)\big) \;\middle|\; 0 \le x < m \right\}, \tag{2}$$

где

$$s(x) = (ax + c) \bmod m \tag{3}$$

представляет собой элемент, следующий за x. Здесь рассматривается лишь множество всех таких точек в t-мерном пространстве, а не порядок, в котором они на самом деле генерируются. Но порядок генерирования отражен в зависимости между

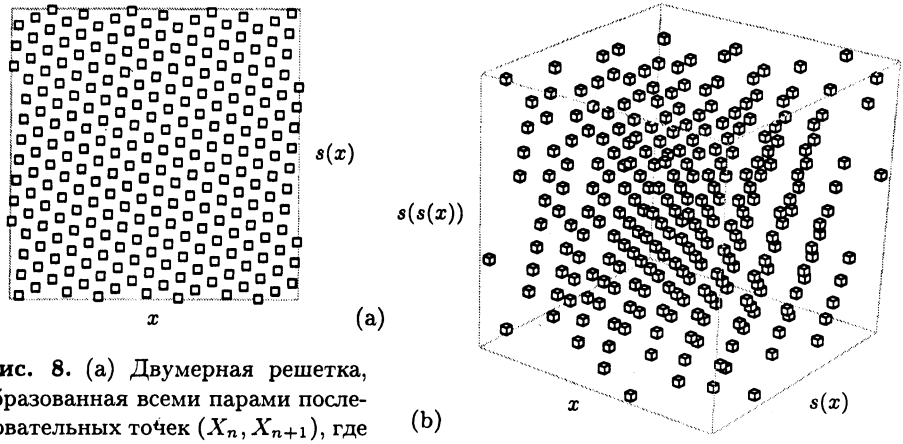


Рис. 8. (a) Двумерная решетка, образованная всеми парами последовательных точек (X_n, X_{n+1}), где $X_{n+1} = (137X_n + 187) \bmod 256$.
(b) Трехмерная решетка трехмерных строк (X_n, X_{n+1}, X_{n+2}).

компонентами векторов, и спектральный критерий изучает такую зависимость для различных размерностей t со всеми точками вида (2).

Например, на рис. 8 показано множество точек в типичных случаях малых размерностей 2 и 3 для генератора с

$$s(x) = (137x + 187) \bmod 256. \tag{4}$$

Конечно, генератор с периодом длиной 256 едва ли будет случайным, но 256 достаточно мало́, чтобы можно было начертить диаграммы и достичь некоторого понимания ситуации, прежде чем перейти к большим m, представляющим практический интерес.

Возможно, наиболее поразительным в схеме коробочек на рис. 8, (a) есть то, что их можно покрыть достаточно малым числом параллельных линий. На самом деле существует много различных семей параллельных линий, проходящих через все точки. Например, семейство, содержащее 20 приблизительно вертикальных линий, пройдет через все точки; семейство, содержащее 21 параллельную линию, наклоненную примерно под углом 30°, также пройдет через все эти точки. Вообще говоря, подобное явление наблюдается в старых садах, посаженных по некоторой системе.

Тот же генератор, рассмотренный в трех измерениях, дает 256 точек в кубе, полученных путем присоединения компонентов "второго порядка" $s(s(x))$ к каждой из 256 точек $\big(x, s(x)\big)$ на плоскости рис. 8, (a), как показано на рис. 8, (b). Представим себе, что эта 3-D-кристаллическая структура построена, как физическая модель, а именно — как куб, который мы можем вертеть в руках, как хотим; при этом можно заметить, что существуют различные семейства параллельных плоскостей, которые проходят через все данные точки. По словам Уоллеса Живена (Wallace Givens), случайность чисел проявляется "главным образом, на плоскостях".

На первый взгляд может показаться, что такое систематическое поведение является настолько неслучайным, что конгруэнтные генераторы совершенно несостоятельны. Однако, если задуматься и вспомнить, что на практике m будет существенно больше, можно прийти к лучшему пониманию этого явления. Регулярную

структуру на рис. 8, а именно "зернистость", можно увидеть, если понаблюдать за случайными числами через мощный микроскоп. Если рассмотреть настоящие случайные числа между 0 и 1 и так округлить либо урезать их с ограниченной точностью, чтобы каждое из них было равно целому числу, умноженному на $1/\nu$ для некоторого заданного числа ν, то t-мерные точки, полученные по правилу (1), будут иметь весьма регулярный характер (если смотреть на них под микроскопом).

Пусть $1/\nu_2$ — максимальное расстояние между линиями всех семейств параллельных линий, которые проходят через двумерные точки $\{(x/m, s(x)/m)\}$. Назовем ν_2 двумерной *точностью* генератора случайных чисел, так как пары последовательных чисел имеют хорошую структуру, которая особенно хороша относительно ν_2. Аналогично пусть $1/\nu_3$ — максимальное расстояние между плоскостями из семейств параллельных плоскостей, проходящих через все точки $\{(x/m, s(x)/m, s(s(x))/m)\}$; назовем ν_3 точностью в трех измерениях. t-мерная точность ν_t равна величине, обратной минимальному расстоянию между гиперплоскостями из семейств параллельных $(t-1)$-мерных гиперплоскостей, проходящих через все точки $\{(x/m, s(x)/m, \ldots, s^{[t-1]}(x)/m)\}$.

Существенная разница между периодическими последовательностями и настоящими последовательностями, члены которых "урезаны" до кратных $1/\nu$ чисел, состоит в том, что точность настоящих случайных последовательностей одна и та же во всех размерностях, а точность периодических последовательностей убывает, когда t растет. На самом деле, так как в t-мерном кубе находится только m точек, когда m — длина периода, мы не можем получить t-мерную точность, бóльшую, чем примерно $m^{1/t}$.

Когда нас интересует, будут ли t последовательных значений независимы, компьютерный генератор случайных чисел будет вести себя, в сущности, так, как будто это настоящие случайные числа, и будет урезать их до $\lg \nu_t$ двоичных разрядов, где ν_t убывает с возрастанием t. На практике подобное изменение точности нас вполне устраивает. Не будем настаивать на том, чтобы 10-мерная точность была равна 2^{32} в том смысле, что все $(2^{32})^{10}$ возможные 10-мерные строки $(U_n, U_{n+1}, \ldots, U_{n+9})$ должны быть равновероятны на 32-разрядной машине. Для таких больших значений t необходимо только, чтобы несколько старших разрядов $(U_n, U_{n+1}, \ldots, U_{n+t-1})$ вели себя так, как если бы они были независимыми случайными величинами.

С другой стороны, когда для приложений нужен генератор случайных чисел, обеспечивающий получение последовательности, очень близкой к случайной, простые конгруэнтные генераторы для этого не подходят. Вместо них нужно использовать генератор с длинным периодом, даже если на самом деле необходимо генерировать только малую часть периода. Если длину периода возвести в квадрат, то, по существу, будет возведена в квадрат и точность в больших измерениях, т. е. эффективное число точных разрядов удвоится.

Спектральный критерий основан на значении ν_t для малых t, скажем, $2 \le t \le 6$. Размерности 2, 3 и 4, кажется, адекватны для определения важных недостатков в последовательности. Но так как здесь рассматривается целый период, разумно в некоторой степени быть осторожными и перейти к другому измерению (или двум). С другой стороны, значения ν_t при $t \ge 10$, кажется, не имеют практического значения. (И это хорошо, поскольку было бы очень трудно вычислить точность ν_t, когда $t \ge 10$.)

Существует не вполне ясная зависимость между спектральным критерием и критерием серий. Например, частный случай критерия серий для целого периода, рассмотренный в упр. 3.3.3–19, подсчитывает число ячеек в каждом из 64 подквадратов (см. рис. 8, (а)). Основная разница состоит в том, что спектральный критерий вращает точки до тех пор, пока не определит наименее благоприятную ориентацию. Ниже в этом разделе мы еще возвратимся к критерию серий.

Может показаться, что достаточно применить спектральный критерий только для одного довольно большого значения t. Если генератор пройдет проверку критерием с тремя измерениями, то кажется правдоподобным, что он пройдет проверку и 2-D-критерием; следовательно, эту проверку можно не делать. Такие рассуждения ошибочны, поскольку не учтено то, что мы требуем более жестких ограничений при более низких размерностях. Подобная ситуация наблюдается при использовании критерия серий. Рассмотрим генератор, у которого почти то же количество точек попадает в каждый подкуб единичного куба, когда единичный куб разделен на 64 подкуба размера $\frac{1}{4} \times \frac{1}{4} \times \frac{1}{4}$. Тот же генератор может дать полностью *пустой* подквадрат единичного квадрата, когда единичный квадрат делится на 64 подквадрата размера $\frac{1}{8} \times \frac{1}{8}$. Таким образом, средние значения при низких измерениях увеличиваются и для каждого измерения требуется отдельная проверка.

Не всегда выполняется неравенство $\nu_t \le m^{1/t}$, хотя $m^{1/t}$ будет верхней гранью для прямоугольной решетки. Например, оказывается, что $\nu_2 = \sqrt{274} > \sqrt{256}$ на рис. 8, поскольку приблизительно шестиугольная структура объединяет m точек так, что возможно строго прямоугольное упорядочение.

Чтобы построить алгоритм, который эффективно подсчитывает ν_t, следует более глубоко разобраться в необходимой для этого математической теории. Поэтому читатель, которого не интересуют математические обоснования, может перейти сразу к части D этого раздела, в которой спектральный критерий будет представлен как "общий" метод, сопровождаемый некоторыми примерами. Однако для математического обоснования спектрального критерия необходимо знать только элементарные преобразования векторов.

Некоторые авторы предлагают использовать минимальное число N_t параллельных линий или гиперплоскостей, проходящих через все точки, вместо максимального расстояния $1/\nu_t$ между ними. Однако число N_t не кажется таким важным, как понятие точности, определенное выше, поскольку оно имеет смещение, зависящее от величины отклонения этих линий или гиперплоскостей от координатных осей куба. Например, приблизительно 20 вертикальных линий, проходящих через все точки на рис. 8, (а), дают действительно $1/\sqrt{328}$ единиц в соответствии с формулой (14), которая приводилась ниже: $(u_1, u_2) = (18, -2)$. Поэтому можно ошибочно принять, что точность равна $\sqrt{328}$ или, возможно, даже 20. Истинная точность, равная $\sqrt{274}$, реализуется только для большого семейства из 21 линии с наклоном $7/15$; другое семейство из 24 линий с наклоном $-11/13$ также имеет бóльшие расстояния между линиями, чем семейство из 20 линий, поскольку $1/\sqrt{290} > 1/\sqrt{328}$. Способ, с помощью которого семейства линий располагаются относительно границ единичного гиперкуба, кажется, не является каким-либо особенно "чистым" или значащим критерием. Однако те, кто предпочитают считать гиперплоскости, могут подсчитывать N_t, используя метод, весьма похожий на тот, с помощью которого мы будем подсчитывать ν_t (см. упр. 16).

***B. Дальнейшее исследование критерия.** Анализ основного множества (2) начнем с наблюдения

$$\frac{1}{m}s^{[j]}(x) = \left(\frac{a^j x + (1 + a + \cdots + a^{j-1})c}{m}\right) \bmod 1. \tag{5}$$

От "модуля 1" можно избавиться, выполнив операцию, с помощью которой множество удлиняется периодически, и сделав бесконечное множество копий исходного t-размерного гиперкуба во всех направлениях. Это дает множество

$$\begin{aligned} L &= \left\{ \left(\frac{x}{m} + k_1, \frac{s(x)}{m} + k_2, \ldots, \frac{s^{[t-1]}(x)}{m} + k_t\right) \;\middle|\; \text{целое } x, k_1, k_2, \ldots, k_t \right\} \\ &= \left\{ V_0 + \left(\frac{x}{m} + k_1, \frac{ax}{m} + k_2, \ldots, \frac{a^{t-1}x}{m} + k_t\right) \;\middle|\; \text{целое } x, k_1, k_2, \ldots, k_t \right\}, \end{aligned}$$

где

$$V_0 = \frac{1}{m}\left(0,\ c,\ (1+a)c,\ \ldots,\ (1 + a + \cdots + a^{t-2})c\right) \tag{6}$$

является вектором констант. Переменная k_1 избыточна в этом представлении L, потому что можно менять $(x, k_1, k_2, \ldots, k_t)$ на $(x + k_1 m,\ 0,\ k_2 - a k_1,\ \ldots,\ k_t - a^{t-1}k_1)$, доводя k_1 до нуля без потери общности. Поэтому получим сравнительно простую формулу:

$$L = \{V_0 + y_1 V_1 + y_2 V_2 + \cdots + y_t V_t \mid \text{целые } y_1, y_2, \ldots, y_t\}, \tag{7}$$

где

$$V_1 = \frac{1}{m}(1, a, a^2, \ldots, a^{t-1}); \tag{8}$$

$$V_2 = (0,1,0,\ldots,0), \quad V_3 = (0,0,1,\ldots,0), \quad \ldots, \quad V_t = (0,0,0,\ldots,1). \tag{9}$$

Точки $(x_1, x_2, \ldots, x_t)$ L, удовлетворяющие неравенству $0 \le x_j < 1$ для всех j, являются именно m точками исходного множества (2).

Заметим, что приращение c появляется только в V_0 и влияние V_0 заключается в сдвиге всех элементов L без изменения их относительных расстояний. Следовательно, *с никоим образом не влияет на спектральный критерий* и в качестве хорошего предположения можно взять $V_0 = (0,0,\ldots,0)$ при вычислении ν_t. Когда V_0 — нулевой вектор, имеем точечную *структуру*

$$L_0 = \{y_1 V_1 + y_2 V_2 + \cdots + y_t V_t \mid \text{целые } y_1, y_2, \ldots, y_t\} \tag{10}$$

и нашей целью является изучение расстояний между смежными $(t-1)$-мерными гиперплоскостями в семействе параллельных гиперплоскостей, покрывающих все точки L_0.

Семейство параллельных $(t-1)$-мерных гиперплоскостей можно определить следующим образом. Пусть ненулевой вектор $U = (u_1, \ldots, u_t)$ перпендикулярен всем гиперплоскостям; тогда множеством точек на определенной гиперплоскости является множество

$$\{(x_1, \ldots, x_t) \mid x_1 u_1 + \cdots + x_t u_t = q\}, \tag{11}$$

где q — различные константы для каждой гиперплоскости семейства. Другими словами, каждая гиперплоскость — это множество всех векторов X, для которых

скалярное произведение $X \cdot U$ имеет данное значение q. В нашем случае все гиперплоскости разделяются фиксированными расстояниями и в одной из них содержится $(0, 0, \ldots, 0)$. Следовательно, можно так установить значение U, что множество всех *целых* значений q даст все гиперплоскости в семействе. Тогда расстояние между соседними гиперплоскостями будет равно минимальному расстоянию от $(0, 0, \ldots, 0)$ до гиперплоскости с $q = 1$, а именно:

$$\min_{\text{действительные } x_1,\ldots,x_t} \left\{ \sqrt{x_1^2 + \cdots + x_t^2} \;\middle|\; x_1u_1 + \cdots + x_tu_t = 1 \right\}. \tag{12}$$

В соответствии с неравенством Коши (см. упр. 1.2.3–30) имеем

$$(x_1u_1 + \cdots + x_tu_t)^2 \le (x_1^2 + \cdots + x_t^2)\,(u_1^2 + \cdots + u_t^2), \tag{13}$$

следовательно, минимум в (12) достигается, когда каждое $x_j = u_j/(u_1^2 + \cdots + u_t^2)$. Это означает, что расстояние между соседними гиперплоскостями равно

$$1/\sqrt{u_1^2 + \cdots + u_t^2} = 1/\text{длина}(U). \tag{14}$$

Другими словами, искомая величина ν_t точно равна длине кратчайшего вектора U, определяющего семейство гиперплоскостей $\{X \cdot U = q \mid \text{целое } q\}$, в которых содержатся все элементы L_0.

Такой вектор $U = (u_1, \ldots, u_t)$ должен быть ненулевым и удовлетворять требованию $V \cdot U =$ целое для всех V в L_0. В частности, так как все точки $(1, 0, \ldots, 0)$, $(0, 1, \ldots, 0)$, $\ldots$, $(0, 0, \ldots, 1)$ принадлежат L_0, все u_j должны быть целыми. Кроме того, так как V_1 принадлежит L_0, получим, что $\frac{1}{m}(u_1 + au_2 + \cdots + a^{t-1}u_t)$ — целые, т. е.

$$u_1 + au_2 + \cdots + a^{t-1}u_t \equiv 0 \pmod{m}. \tag{15}$$

И наоборот, любой ненулевой целый вектор $U = (u_1, \ldots, u_t)$, удовлетворяющий (15), определяет семейство гиперплоскостей с необходимыми свойствами, так как будут покрыты все L_0. Скалярное произведение $(y_1V_1 + \cdots + y_tV_t) \cdot U$ будет целым для всех целых $y_1, \ldots, y_t$. Мы доказали, что

$$\begin{aligned} \nu_t^2 &= \min_{(u_1,\ldots,u_t)\neq(0,\ldots,0)} \{u_1^2 + \cdots + u_t^2 \mid u_1 + au_2 + \cdots + a^{t-1}u_t \equiv 0 \text{ (по модулю } m)\} \\ &= \min_{(x_1,\ldots,x_t)\neq(0,\ldots,0)} \left((mx_1 - ax_2 - a^2x_3 - \cdots - a^{t-1}x_t)^2 + x_2^2 + x_3^2 + \cdots + x_t^2\right). \end{aligned} \tag{16}$$

C. Обоснование вычислительных методов. Сведем спектральный критерий к задаче нахождения минимального значения (16). Но как можно найти минимальное значение за разумный отрезок времени? Грубое силовое исследование не входит в наши планы, так как m — очень большое в случаях, представляющих практический интерес.

Будет интересно и, возможно, более полезно разработать вычислительные методы решений даже более общей проблемы: *найти минимальное значение величины*

$$f(x_1, \ldots, x_t) = (u_{11}x_1 + \cdots + u_{t1}x_t)^2 + \cdots + (u_{1t}x_1 + \cdots + u_{tt}x_t)^2 \tag{17}$$

по всем ненулевым целым векторам $(x_1, \ldots, x_t)$ для любой невырожденной матрицы с коэффициентами $U = (u_{ij})$. Выражение (17) назовем положительно определенной квадратичной формой от t переменных. Так как U — невырожденная матрица, (17) не может быть нулем, если не все x_j равны нулю.

Запишем как $U_1, \ldots, U_t$ все строки U. Тогда (17) можно записать как

$$f(x_1, \ldots, x_t) = (x_1U_1 + \cdots + x_tU_t) \cdot (x_1U_1 + \cdots + x_tU_t), \tag{18}$$

квадрат длины вектора $x_1U_1 + \cdots + x_tU_t$. Невырожденная матрица U имеет обратную матрицу, а это означает, что можно найти единственным образом определенные векторы $V_1, \ldots, V_t$, такие, что

$$U_i \cdot V_j = \delta_{ij}, \qquad 1 \le i, j \le t. \tag{19}$$

Например, в частном случае (16), возникающем в связи со спектральным критерием, получим

$$\begin{aligned}
U_1 &= (\ \ m, 0, 0, \ldots, 0), & V_1 &= \tfrac{1}{m}(1, a, a^2, \ldots, a^{t-1}),\\
U_2 &= (\ -a, 1, 0, \ldots, 0), & V_2 &= \ \ (0, 1,\ 0, \ldots,\ \ 0),\\
U_3 &= (\ -a^2, 0, 1, \ldots, 0), & V_3 &= \ \ (0, 0,\ 1, \ldots,\ \ 0),\\
&\ \ \ldots\ldots\ldots\ldots\ldots & &\ \ \ldots\ldots\ldots\ldots\ldots\\
U_t &= (-a^{t-1}, 0, 0, \ldots, 1), & V_t &= \ \ (0, 0,\ 0, \ldots,\ \ 1).
\end{aligned} \tag{20}$$

Эти V_j точно равны векторам (8) и (9), которые использовались для определения начальной решетки L_0. Как и подозревает читатель, здесь нет совпадения: действительно, мы начинали с произвольной решетки L_0, определенной любым множеством линейно независимых векторов $V_1, \ldots, V_t$. Доводы, используемые нами выше, могут быть обобщены, чтобы показать, что максимальное расстояние между гиперплоскостями в покрывающем все точки семействе равно минимуму (17), где коэффициенты u_{ij} определены в (19) (см. упр. 2).

Первый шаг в минимизации (18) — ее сведение к конечной задаче: необходимо показать, что для нахождения минимума нет необходимости в бесконечном множестве векторов $(x_1, \ldots, x_t)$. Удобно выразить x_k через векторы $V_1, \ldots, V_t$, а именно

$$x_k = (x_1U_1 + \cdots + x_tU_t) \cdot V_k,$$

и из неравенства Коши получить неравенство

$$\big((x_1U_1 + \cdots + x_tU_t) \cdot V_k\big)^2 \le f(x_1, \ldots, x_t)(V_k \cdot V_k).$$

Итак, получена удобная верхняя граница каждой координаты x_k.

Лемма А. *Пусть $(x_1, \ldots, x_t)$ — ненулевой вектор, минимизирующий* (18), *и пусть $(y_1, \ldots, y_t)$ — любой ненулевой целочисленный вектор. Тогда*

$$x_k^2 \le f(y_1, \ldots, y_t)(V_k \cdot V_k) \qquad \text{для } 1 \le k \le t. \tag{21}$$

В частности, полагая $y_i = \delta_{ij}$ для всех i, получим

$$x_k^2 \le (U_j \cdot U_j)(V_k \cdot V_k) \qquad \text{для } 1 \le j, k \le t. \quad \blacksquare \tag{22}$$

Лемма А сводит задачу к нахождению минимума по конечному множеству, но правая часть (21) обычно является слишком большой для того, чтобы можно было перебрать все значения (по крайней мере, нужна еще одна добавочная идея). В таком случае помогает старый афоризм: “Если вы не можете решить задачу в том виде, в каком она сформулирована, упростите ее, чтобы получить

такой же ответ". Например, алгоритм Евклида имеет такой вид, что если наибольший общий делитель чисел на входе неизвестен, придется выбирать его среди меньших чисел, имеющих тот же наибольший общий делитель. (На самом деле в основе открытия всех подобных алгоритмов, возможно, лежит несколько более общий подход: "Если вы не можете непосредственно решить задачу, сведите ее к одной из более простых задач, решив которые, вы сможете справиться с исходной задачей".)

В рассматриваемом здесь случае более простой проблемой является проблема, требующая меньшего количества проверок, потому что правая часть в (22) уменьшилась. Ключевой используемой идеей является возможность заменить одну квадратичную форму другой, которая представляет собой эквивалент для достижения всех практических целей. Пусть j — любой фиксированный индекс, $1 \le j \le t$, и пусть $(q_1, \ldots, q_{j-1}, q_{j+1}, \ldots, q_t)$ — любая последовательность $t-1$ целых чисел. Рассмотрим следующее преобразование векторов:

$$\begin{aligned} V_i' &= V_i - q_i V_j, \qquad & x_i' &= x_i - q_i x_j, \qquad & U_i' &= U_i \qquad \text{для } i \ne j; \\ V_j' &= V_j, & x_j' &= x_j, & U_j' &= U_j + \textstyle\sum_{i \ne j} q_i U_i. \end{aligned} \tag{23}$$

Легко видеть, что новые векторы $U_1', \ldots, U_t'$ определяют квадратичную форму f', для которой $f'(x_1', \ldots, x_t') = f(x_1, \ldots, x_t)$. Кроме того, основное условие ортогональности (19) сохраняется, так как легко проверить, что $U_i' \cdot V_j' = \delta_{ij}$. Когда $(x_1, \ldots, x_t)$ пробегает множество всех не равных нулю целочисленных векторов, то $(x_1', \ldots, x_t')$ пробегает это же множество; следовательно, новая форма f' имеет тот же минимум, что и f.

Нашей целью является использование преобразования (23) и замена U_i на U_i' и V_i на V_i' для всех i, чтобы сделать правую часть (22) малой (правая часть (22) будет малой, когда $U_j \cdot U_j$ и $V_k \cdot V_k$ будут малы). Следовательно, вполне естественно задать два следующих вопроса относительно преобразования (23).

a) *Какие значения q_i делают $V_i' \cdot V_i'$ настолько малыми, насколько это возможно?*

b) *Какие значения q_1, ..., q_{j-1}, q_{j+1}, ..., q_t делают $U_j' \cdot U_j'$ настолько малыми, насколько это возможно?*

Легче всего ответить на эти вопросы сначала для *действительных* значений q_i. Вопрос (a) совершенно простой, так как

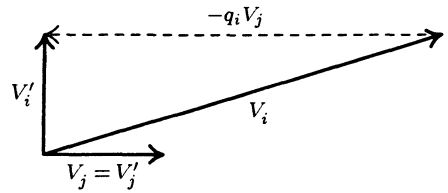
$$\begin{aligned} (V_i - q_i V_j) \cdot (V_i - q_i V_j) &= V_i \cdot V_i - 2q_i\, V_i \cdot V_j + q_i^2\, V_j \cdot V_j \\ &= (V_j \cdot V_j)\left(q_i - (V_i \cdot V_j / V_j \cdot V_j)\right)^2 + V_i \cdot V_i - (V_i \cdot V_j)^2 / V_j \cdot V_j \end{aligned}$$

и минимум достигается, когда

$$q_i = V_i \cdot V_j \,/\, V_j \cdot V_j. \tag{24}$$

С геометрической точки зрения мы спрашиваем, сколько раз можно вычесть V_j из V_i, чтобы получить вектор минимальной длины V_i', и отвечаем: нужно выбрать такое q_i, чтобы V_i' было перпендикулярно V_j (т. е. сделать так, чтобы выполнялось

равенство $V_i' \cdot V_j = 0$). Это изображено на следующем графике.



(25)

Обратимся к вопросу (b). Необходимо выбрать такое q_i, чтобы $U_j + \sum_{i \neq j} q_i U_i$ имели минимальную длину. С геометрической точки зрения следует начать с U_j и прибавить некоторый вектор в $(t-1)$-мерной гиперплоскости, равный сумме кратных $\{U_i \mid i \neq j\}$. Снова лучшим решением является такой выбор, при котором U_j' является перпендикулярным к гиперплоскости, т. е. $U_j' \cdot U_k = 0$ для всех $k \neq j$:

$$U_j \cdot U_k + \sum_{i \neq j} q_i (U_i \cdot U_k) = 0, \qquad 1 \leq k \leq t, \qquad k \neq j. \tag{26}$$

(В упр. 12 приводится строгое доказательство того, что решение (b) должно удовлетворять этим $t-1$ уравнениям.)

Ответив на вопросы (a) и (b), мы оказались в двойном затруднении: можно ли выбрать q_i соответственно (24) так, чтобы $V_i' \cdot V_i'$ было минимальным, или согласно (26) так, чтобы $U_j' \cdot U_j'$ было минимальным? Каждая из этих альтернатив приводит к уменьшению части (22), поэтому сразу не ясно, какому выбору отдать предпочтение. К счастью, существует очень простое решение этой дилеммы: условия (24) и (26) те же самые! (См. упр. 7.) Следовательно, ответы на вопросы (a) и (b) совпадают. Получается, что длину обоих векторов U_i и V_j можно уменьшить одновременно. На самом деле мы только что снова открыли *процесс ортогонализации Грама-Шмидта (Gram-Schmidt)* [см. *Crelle* **94** (1883), 41–73].

Нашу радость омрачает понимание того, что вопросы (a) и (b) рассматривались только для *действительных* значений q_i. Для решения поставленной задачи следует ограничиться целыми значениями, в связи с чем провести V_i' точно перпендикулярно V_j нельзя. Лучшее, что можно сделать в (a), — это положить q_i *наиболее близким целым* к $V_i \cdot V_j \,/\, V_j \cdot V_j$ (см. (25)). Оказывается, что это *не* всегда лучшее решение вопроса (b); на самом деле U_j' иногда может быть длиннее U_j. Однако граница (21) никогда не растет, поэтому мы можем запомнить наименьшее значение $f(y_1, \ldots, y_t)$, найденное до сих пор. Этот выбор q_i, основанный исключительно на вопросе (a), является совершенно удовлетворительным.

Если преобразование (23) повторно применить таким образом, чтобы ни один из векторов V_i не стал длиннее и по крайней мере один стал короче, мы никогда не попадем в петлю, т. е. мы никогда не будем рассматривать ту же квадратичную форму после ряда нетривиальных преобразований подобного вида. В конце концов, мы застрянем: ни одно из преобразований (23) для $1 \leq j \leq t$ не будет в состоянии укоротить любой из векторов $V_1, \ldots, V_t$. В этой точке можно возвращаться к исчерпывающему исследованию, используя границы леммы A, которые будут вполне малы в большинстве случаев. Изредка этих границ (21) недостаточно, и другой вид преобразования обычно дает алгоритм выхода из положения, в котором мы застряли, и уменьшения границы (см. упр. 18). Тем не менее доказано, что преобра-

зование (23) в самого себя вполне отвечает требованиям спектрального критерия; к тому же доказано, что оно поразительно мощное, когда вычисления осуществляются так, как в алгоритме, обсуждаемом ниже.

***D. Как выполнить спектральный критерий.** В этом разделе будет приведена эффективная вычислительная процедура. Госпер (Gosper) и Дитер (Dieter) заметили, что можно использовать результаты для более низкой размерности, чтобы значительно быстрее получить спектральный критерий в высокой размерности. Это усовершенствование включено в следующий алгоритм вместе с упрощением Гаусса (Gauss) для двумерного случая (упр. 5).

Алгоритм S (*Спектральный критерий*). Этот алгоритм определяет значение

$$\nu_t = \min\left\{ \sqrt{x_1^2 + \cdots + x_t^2} \;\middle|\; x_1 + ax_2 + \cdots + a^{t-1}x_t \equiv 0 \ (\text{по модулю } m) \right\} \tag{27}$$

для $2 \le t \le T$, заданных a, m и T, где $0 < a < m$ и a и m — взаимно простые числа. (Минимум взят по всем ненулевым целочисленным векторам $(x_1, \ldots, x_t)$, а число ν_t является t-мерной точностью генератора случайных чисел, как обсуждалось выше.) Вся арифметика в пределах алгоритма дана в целых числах, размеры которых редко либо никогда не превышают m^2, исключая шаг S7. К тому же почти все целые переменные будут меньше m по абсолютной величине на протяжении вычислений.

Когда ν_t вычисляется для $t \ge 3$, алгоритм работает с двумя $t \times t$-матрицами U и V, векторы-строки которых обозначены через $U_i = (u_{i1}, \ldots, u_{it})$ и $V_i = (v_{i1}, \ldots, v_{it})$ для $1 \le i \le t$. Эти векторы удовлетворяют условиям

$$u_{i1} + au_{i2} + \cdots + a^{t-1}u_{it} \equiv 0 \ (\text{по модулю } m), \qquad 1 \le i \le t; \tag{28}$$

$$U_i \cdot V_j = m\delta_{ij}, \qquad 1 \le i, j \le t. \tag{29}$$

(Таким образом, V_j из предыдущих обсуждений умножаются на m, чтобы их компоненты были целыми числами.) Существует три дополнительных вектора: $X = (x_1, \ldots, x_t)$, $Y = (y_1, \ldots, y_t)$ и $Z = (z_1, \ldots, z_t)$. В алгоритме r будет обозначать $a^{t-1} \bmod m$, а s — наименьшую верхнюю грань ν_t^2, которая была найдена ранее.

S1. [Инициализация.] Присвоить $t \leftarrow 2$, $h \leftarrow a$, $h' \leftarrow m$, $p \leftarrow 1$, $p' \leftarrow 0$, $r \leftarrow a$, $s \leftarrow 1 + a^2$. (Первые шаги алгоритма — это специальный метод, примененный к случаю $t = 2$, который очень похож на алгоритм Евклида. Получим

$$h - ap \equiv h' - ap' \equiv 0 \ (\text{по модулю } m) \qquad \text{и} \qquad hp' - h'p = \pm m \tag{30}$$

на протяжении этого этапа вычислений.)

S2. [Шаг Евклида.] Присвоить $q \leftarrow \lfloor h'/h \rfloor$, $u \leftarrow h' - qh$, $v \leftarrow p' - qp$. Если $u^2 + v^2 < s$, присвоить $s \leftarrow u^2 + v^2$, $h' \leftarrow h$, $h \leftarrow u$, $p' \leftarrow p$, $p \leftarrow v$ и повторить шаг S2.

S3. [Вычисление ν_2.] Присвоить $u \leftarrow u - h$, $v \leftarrow v - p$, а если $u^2 + v^2 < s$, присвоить $s \leftarrow u^2 + v^2$, $h' \leftarrow u$, $p' \leftarrow v$. Затем выход $\sqrt{s} = \nu_2$. (Справедливость этих вычислений для двумерного случая доказана в упр. 5. Подготовим матрицы U и V, удовлетворяющие соответственно (28) и (29), для вычислений в больших размерностях.) Присвоить

$$U \leftarrow \begin{pmatrix} -h & p \\ -h' & p' \end{pmatrix}, \qquad V \leftarrow \pm \begin{pmatrix} p' & h' \\ -p & -h \end{pmatrix},$$

где знак "минус" выбирается для V, только если $p' > 0$.

S4. [Опережение t.] Если $t = T$, алгоритм завершается. (Иначе нужно увеличить t на 1. В этой точке U и V являются матрицами размера $t \times t$, удовлетворяющими (28) и (29), и их необходимо увеличить, присоединяя подходящие новые столбцы и строки.) Присвоить $t \leftarrow t+1$ и $r \leftarrow (ar) \bmod m$. Присвоить U_t новую строку $(-r, 0, 0, \ldots, 0, 1)$ из t элементов и присвоить $u_{it} \leftarrow 0$ для $1 \le i < t$. Присвоить V_t новую строку $(0, 0, 0, \ldots, 0, m)$. Окончательно для $1 \le i < t$ присвоить $q \leftarrow$ округленное$(v_{i1}r/m)$, $v_{it} \leftarrow v_{i1}r - qm$ и $U_t \leftarrow U_t + qU_i$. (Здесь "округленное (x)" определено как ближайшее целое к x, например $\lfloor x + 1/2 \rfloor$. Мы, по существу, присваиваем $v_{it} \leftarrow v_{i1}r$ и непосредственно применяем преобразование (23) с $j = t$, потому что числа $|v_{i1}r|$ настолько велики, что их нужно сразу уменьшить.) Окончательно присвоить $s \leftarrow \min(s, U_t \cdot U_t)$, $k \leftarrow t$ и $j \leftarrow 1$. (На следующем шаге j обозначает индекс текущей строки для преобразования (23), а k — последний такой индекс, когда преобразование укоротит по крайней мере один из векторов V_i.)

S5. [Преобразовать.] Для $1 \le i \le t$ выполнить следующие операции: если $i \ne j$ и $2|V_i \cdot V_j| > V_j \cdot V_j$, то присвоить $q \leftarrow$ округление$(V_i \cdot V_j \,/\, V_j \cdot V_j)$, $V_i \leftarrow V_i - qV_j$, $U_j \leftarrow U_j + qU_i$, $s \leftarrow \min(s, U_j \cdot U_j)$ и $k \leftarrow j$. (Пропускаем преобразование, когда $2|V_i \cdot V_j|$ точно равно $V_j \cdot V_j$; в упр. 19 показано, что эта предосторожность удерживает алгоритм от зацикливания.

S6. [Опережение j.] Если $j = t$, присвоить $j \leftarrow 1$; иначе присвоить $j \leftarrow j+1$. Если $j \ne k$, вернуться к шагу S5. (Если $j = k$, проходим $t-1$ последовательных циклов без преобразований, так как процесс преобразования "застрял".)

S7. [Подготовка поиска.] (Сейчас абсолютный минимум будет установлен путем исчерпывающего поиска по всем $(x_1, \ldots, x_t)$, удовлетворяющим условию (21) леммы A.) Присвоить $X \leftarrow Y \leftarrow (0, \ldots, 0)$, присвоить $k \leftarrow t$ и присвоить

$$z_j \leftarrow \left\lfloor \sqrt{\lfloor (V_j \cdot V_j)s/m^2 \rfloor} \right\rfloor \qquad \text{для } 1 \le j \le t. \tag{31}$$

(Проверим все $X = (x_1, \ldots, x_t)$ с $|x_j| \le z_j$ для $1 \le j \le t$. Обычно $|z_j| \le 1$, но Л. Киллингбек (L. C. Killingbeck) заметил в 1999 году, что большие значения появляются около 0.00001 раз для всех множителей, когда $m = 2^{64}$. Во время перебора вектор Y всегда будет равен $x_1U_1 + \cdots + x_tU_t$, поскольку $f(x_1, \ldots, x_t) = Y \cdot Y$. Так как $f(-x_1, \ldots, -x_t) = f(x_1, \ldots, x_t)$, будем проверять только векторы, первая ненулевая компонента которых положительна. Метод, по существу, состоит в том, что при подсчетах $(x_1, \ldots, x_t)$ рассматривается как цифры в изменяющейся системе счисления со смешанным основанием $(2z_1 + 1, \ldots, 2z_t + 1)$; см. раздел 4.1.)

S8. [Опережение x_k.] Если $x_k = z_k$, перейти к шагу S10. Иначе — увеличить x_k на 1 и присвоить $Y \leftarrow Y + U_k$.

S9. [Опережение k.] Присвоить $k \leftarrow k+1$. Затем, если $k \le t$, присвоить $x_k \leftarrow -z_k$, $Y \leftarrow Y - 2z_kU_k$ и повторить шаг S9. Но, если $k > t$, присвоить $s \leftarrow \min(s, Y \cdot Y)$.

S10. [Уменьшение k.] Присвоить $k \leftarrow k-1$. Если $k \ge 1$, возвратиться к шагу S8. Иначе — выход $\nu_t = \sqrt{s}$; перебор заканчивается, и возврат к шагу S4. ▮

Практически алгоритм S применяется, скажем, для $T = 5$ или 6. Обычно он работает достаточно хорошо, когда $T = 7$ или 8, но может быть ужасно медленным

при $T \geq 9$, поскольку при переборе время его работы увеличивается как 3^T. (Если минимальное значение ν_t достигается в различных точках, то в результате перебора будут найдены все точки. Следовательно, обычно находим, что все $z_k = 1$ для большого t. Как замечено выше, значения ν_t вообще непригодны для практического применения, когда t большое.)

Приведенный ниже пример поможет лучше понять алгоритм S. Рассмотрим линейную конгруэнтную последовательность, определенную таким образом:

$$m = 10^{10}, \qquad a = 3141592621, \qquad c = 1, \qquad X_0 = 0. \tag{32}$$

Шести циклов алгоритма Евклида на шагах S2 и S3 достаточно для доказательства того, что минимальное ненулевое значение $x_1^2 + x_2^2$ с

$$x_1 + 3141592621x_2 \equiv 0 \pmod{10^{10}}$$

достигается при $x_1 = 67654$, $x_2 = 226$. Следовательно, двумерная точность этого генератора равна

$$\nu_2 = \sqrt{67654^2 + 226^2} \approx 67654.37748.$$

Переходя к размерности 3, найдем минимальное ненулевое значение $x_1^2 + x_2^2 + x_3^2$, такое, что

$$x_1 + 3141592621x_2 + 3141592621^2x_3 \equiv 0 \pmod{10^{10}}. \tag{33}$$

На шаге S4 увеличиваются матрицы

$$U = \begin{pmatrix} -67654 & -226 & 0 \\ -44190611 & 191 & 0 \\ 5793866 & 33 & 1 \end{pmatrix}, \quad V = \begin{pmatrix} -191 & -44190611 & 2564918569 \\ -226 & 67654 & 1307181134 \\ 0 & 0 & 10000000000 \end{pmatrix}.$$

Первая итерация шага S5 с $q = 1$ для $i = 2$ и $q = 4$ для $i = 3$ заменяет их матрицами

$$U = \begin{pmatrix} -21082801 & 97 & 4 \\ -44190611 & 191 & 0 \\ 5793866 & 33 & 1 \end{pmatrix}, \quad V = \begin{pmatrix} -191 & -44190611 & 2564918569 \\ 35 & 44258265 & 1257737435 \\ 764 & 176762444 & -259674276 \end{pmatrix}.$$

(Первая строка U_1 на самом деле получается длиннее в этом преобразовании, несмотря на то что строки матрицы U должны стать короче.)

Следующие 14 итераций шага S5 дают $(j, q_1, q_2, q_3) = (2, -2, *, 0)$, $(3, 0, 3, *)$, $(1, *, -10, -1)$, $(2, -1, *, -6)$, $(3, -1, 0, *)$, $(1, *, 0, 2)$, $(2, 0, *, -1)$, $(3, 3, 4, *)$, $(1, *, 0, 0)$, $(2, -5, *, 0)$, $(3, 1, 0, *)$, $(1, *, -3, -1)$, $(2, 0, *, 0)$, $(3, 0, 0, *)$. Сейчас процесс преобразования окончен, но строки матриц стали значительно короче:

$$U = \begin{pmatrix} -1479 & 616 & -2777 \\ -3022 & 104 & 918 \\ -227 & -983 & -130 \end{pmatrix}, \quad V = \begin{pmatrix} -888874 & 601246 & -2994234 \\ -2809871 & 438109 & 1593689 \\ -854296 & -9749816 & -1707736 \end{pmatrix}. \tag{34}$$

Найденные пределы (z_1, z_2, z_3) на шаге S7 оказываются равными $(0, 0, 1)$, поэтому U_3 будет самым коротким решением (33). В результате получим

$$\nu_3 = \sqrt{227^2 + 983^2 + 130^2} \approx 1017.21089.$$

Для того чтобы найти эту величину, достаточно было лишь нескольких итераций, хотя условие (33) выглядит, на первый взгляд, устрашающим. Наши вычисления показали, что все точки (U_n, U_{n+1}, U_{n+2}), произведенные генератором случайных

чисел (32) лежат около семейства параллельных плоскостей, отклоняясь на 0.001 единиц, но нет такой системы параллельных плоскостей, которые отличались бы больше чем на 0.001 единиц.

Перебор значений на шагах S8–S10 редко приводит к значению s. Один такой случай, найденный в 1982 году Р. Карлиным (R. Carling) и Е. Левин (E. Levine), произошел при $a = 464680339$, $m = 2^{29}$ и $t = 5$; другой случай произошел, когда автор вычислял ν_6^2 для строки 21 табл. 1, приведенной ниже в этом разделе.

Е. Рейтинги различных генераторов. До сих пор нельзя сказать, будет ли данный генератор случайных чисел на самом деле удовлетворять спектральному критерию. Успех этого испытания зависит от приложений, так как одни приложения предъявляют более высокие требования, чем другие. Если мы получим, что $\nu_t \geq 2^{30/t}$ для $2 \leq t \leq 6$, то будет ли этого достаточно для большинства случаев (хотя автор должен признаться, что выбрал критерий отчасти потому, что 30 делится на 2, 3, 5 и 6).

В некоторых случаях хорошо бы иметь правило для определения, удовлетворяет ли датчик критерию, относительно независимое от m, чтобы можно было сказать, что некоторый множитель хорош либо плох по отношению к другим множителям для заданного m, не проверяя остальных. Разумной мерой, заслуживающей быть показателем для определения, насколько хорош заданный генератор, мог бы быть объем эллипсоида в t-мерном пространстве, определенный соотношением

$$(x_1 m - x_2 a - \cdots - x_t a^{t-1})^2 + x_2^2 + \cdots + x_t^2 \leq \nu_t^2,$$

так как этот объем стремится к вероятности того, что точка с ненулевыми *целыми* координатами $(x_1, \ldots, x_t)$, которая соответствует решению (15), находится в эллипсоиде. Поэтому предлагаем вычислить этот объем, а именно — показать, что он равен

$$\mu_t = \frac{\pi^{t/2}\, \nu_t^t}{(t/2)!\, m}, \tag{35}$$

и рассматривать его как меру эффективности множителя a для заданного m. В этой формуле

$$\left(\frac{t}{2}\right)! = \left(\frac{t}{2}\right)\left(\frac{t}{2} - 1\right) \ldots \left(\frac{1}{2}\right)\sqrt{\pi} \qquad \text{для нечетных } t. \tag{36}$$

Таким образом, для размерностей, меньших или равных шести, эта мера имеет следующие значения:

$$\mu_2 = \pi\nu_2^2/m, \qquad \mu_3 = \tfrac{4}{3}\pi\nu_3^3/m, \qquad \mu_4 = \tfrac{1}{2}\pi^2\nu_4^4/m,$$
$$\mu_5 = \tfrac{8}{15}\pi^2\nu_5^5/m, \qquad \mu_6 = \tfrac{1}{6}\pi^3\nu_6^6/m.$$

Можно сказать, что множитель a удовлетворяет спектральному критерию, если μ_t равно 0.1 или больше для $2 \leq t \leq 6$, и "проходит его, как на зеленый свет", если $\mu_t \geq 1$ для всех этих t. Малое значение μ_t указывает, что, вероятно, перед нами — весьма неудачный множитель, так как очень мало решеток будут иметь точки с целыми координатами столь близко к началу координат. С другой стороны, большое значение μ_t показывает, что найден необычайно хороший множитель для данного m, но это не означает, что случайные числа являются необходимо очень хорошими,

так как m может быть слишком малым. Только истинное значение ν_t определяет степень случайности.

В табл. 1 показано, какие величины могут появляться в обычных последовательностях. В каждой строке таблицы рассматривается какой-либо фиксированный генератор, и для него приведены значения ν_t^2, μ_t и "число двоичных разрядов точности" $\lg \nu_t$. В строках 1–4 перечислены генераторы, изображенные на рис. 2 и 5 из раздела 3.3.1. Генераторы в строках 1 и 2 имеют слишком малый множитель; диаграмма, подобная изображенной на рис. 8, при малых a будет близка к вертикальной "полосе". Ужасный генератор в строке 3 имеет хорошее μ_2, но очень плохие μ_3 и μ_4. Подобно всем генераторам с потенциалом 2, он имеет $\nu_3 = \sqrt{6}$ и $\nu_4 = 2$ (см. упр. 3). В строке 4 приведен "случайный" множитель. Этот генератор достаточно хорошо удовлетворяет эмпирическим критериям случайности, но его значения $\mu_2, \ldots, \mu_6$ не очень большие. На самом деле значение μ_5 указывает, что генератор не удовлетворяет нашему критерию.

В строке 5 приведен генератор, представленный на рис. 8. Он удовлетворяет спектральному критерию с высокой степенью надежности, когда рассматривать $\mu_2 - \mu_6$, но, конечно, m настолько мало́, что числа с трудом могут быть названы случайными; значения ν_t ужасно малы.

В строке 6 приведен генератор, обсуждавшийся выше, в (32). В строке 7 представлен подобный пример, имеющий ненормально малое значение μ_3. В строке 8 приведен неслучайный множитель для того же модуля m; все его частичные отношения равны 1, 2 или 3. Такие множители были предложены И. Борошем (I. Borosh) и Г. Нейдерейтером (H. Niederreiter), поскольку суммы Дедекинда в этом случае особенно малы и дают хорошие результаты при проверке двумерным критерием серий (см. раздел 3.3.3 и упр. 30). Для генератора строки 8 имеется только одно частичное отношение '3'; не существует множителей, конгруэнтных 1 по модулю 20, частичные отношения которых относительно 10^{10} равны только́ 1 или 2. Генератор, приведенный в строке 9, имеет другой множитель, который специально выбран с плохими намерениями, как предложил Вотерман (Waterman). Он гарантирует достаточно большое значение μ_2 (см. упр. 11). Строка 10 интересна, поскольку указанный в ней генератор имеет большое значение μ_3 при очень малом значении μ_2 (см. упр. 8).

Строка 11 табл. 1 — это воспоминание о старых добрых временах: данный генератор когда-то широко использовался, после того как его предложила О. Таусски (O. Taussky) в начале 50-х годов. Но компьютеры, для которых модуль равнялся примерно 2^{35}, начали утрачивать популярность в конце 60-х и почти полностью пропали в 80-е годы, когда получили распространение машины с 32-разрядной арифметикой. Так, сравнительно малые размеры компьютерного слова были заменены сравнительно большими заботами. В строке 12 содержится, увы, генератор, который действительно использовался на таких машинах в последнее десятилетие (90-е годы) в научных компьютерных центрах мира. Его истинное имя — RANDU (похоже на "random" — "случайный". — *Прим. ред.*), и этого достаточно, чтобы вызвать испуг в глазах и спазмы в желудке у многих ученых, специализирующихся на компьютерах! Действительно, генератор определен следующим образом:

$$X_0 \text{ нечетное}, \qquad X_{n+1} = (65539X_n) \bmod 2^{31}. \tag{37}$$

Таблица 1

ВЫБОРОЧНЫЕ РЕЗУЛЬТАТЫ ПРИМЕНЕНИЯ СПЕКТРАЛЬНОГО КРИТЕРИЯ

Строка	a	m	ν_2^2	ν_3^2	ν_4^2	ν_5^2	ν_6^2
1	23	10^8+1	530	530	530	530	447
2	2^7+1	2^{35}	16642	16642	16642	15602	252
3	$2^{18}+1$	2^{35}	34359738368	6	4	4	4
4	3141592653	2^{35}	2997222016	1026050	27822	1118	1118
5	137	256	274	30	14	6	4
6	3141592621	10^{10}	4577114792	1034718	62454	1776	542
7	3141592221	10^{10}	4293881050	276266	97450	3366	2382
8	4219755981	10^{10}	10721093248	2595578	49362	5868	820
9	4160984121	10^{10}	9183801602	4615650	16686	6840	1344
10	$2^{24}+2^{13}+5$	2^{35}	8364058	8364058	21476	16712	1496
11	5^{13}	2^{35}	33161885770	2925242	113374	13070	2256
12	$2^{16}+3$	2^{29}	536936458	118	116	116	116
13	1812433253	2^{32}	4326934538	1462856	15082	4866	906
14	1566083941	2^{32}	4659748970	2079590	44902	4652	662
15	69069	2^{32}	4243209856	2072544	52804	6990	242
16	1664525	2^{32}	4938916874	2322494	63712	4092	1038
17	314159269	$2^{31}-1$	1432232969	899290	36985	3427	1144
18	62089911	$2^{31}-1$	1977289717	1662317	48191	6101	1462
19	16807	$2^{31}-1$	282475250	408197	21682	4439	895
20	48271	$2^{31}-1$	1990735345	1433881	47418	4404	1402
21	40692	$2^{31}-249$	1655838865	1403422	42475	6507	1438
22	44485709377909	2^{46}	5.6×10^{13}	1180915002	1882426	279928	26230
23	31167285	2^{48}	3.2×10^{14}	4111841446	17341510	306326	59278
24	См. (38)	См. (38)	2.4×10^{18}	4.7×10^{11}	1.9×10^{9}	3194548	1611610
25	См. (39)	См. (39)	$(2^{31}-1)^2$	1.4×10^{12}	643578623	12930027	837632
26	См. текст	2^{64}	8.8×10^{18}	6.4×10^{12}	4.1×10^{9}	45662836	1846368
27	См. текст	$\approx 2^{78}$	$2^{62}+1$	4281084902	2.2×10^{9}	1.8×10^{9}	1862407
28	$2^{-24\cdot389}$	$\approx 2^{576}$	1.8×10^{173}	3.5×10^{115}	4.4×10^{86}	2×10^{69}	5×10^{57}
29	$(2^{32}-5)^{-400}$	$\approx 2^{1376}$	1.6×10^{414}	8.6×10^{275}	1×10^{207}	2×10^{165}	8×10^{137}

В упр. 20 показано, что 2^{29} — подходящий модуль для спектрального критерия. Так как $9X_n - 6X_{n+1} + X_{n+2} \equiv 0$ (по модулю 2^{31}), генератор не удовлетворяет большинству трехмерных критериев случайности и его никогда не следовало бы использовать. Почти любой множитель $\equiv 5$ (по модулю 8) был бы лучше. (Любопытный факт относительно RANDU, отмеченный Госпером (Gosper), состоит в том, что $\nu_4 = \nu_5 = \nu_6 = \nu_7 = \nu_8 = \nu_9 = \sqrt{116}$. Следовательно, μ_9 равно эффективному значению 11.98.) Строки 13 и 14 — это множители Бороша-Нидеррейтера (Borosh-Niederreiter) и Вотермана (Waterman) для модуля 2^{32}. В строках 16 и 23 расположены генераторы Лаво (Lavaux) и Йенсена (Janssens); параметры этих генераторов были найдены на компьютере, чтобы получить хороший множитель в смысле спектрального критерия, для которого μ_2 принимает очень большое значение. В строке 22 находится генератор с множителем, используемым при $c = 0$ и $m = 2^{48}$; он содержится в библиотеке компьютера Cray X-MP. В строке 26 содержится генератор, предложенный Хайнесом (Haynes) (его превосходный множитель 6364136223846793005 настолько велик, что не поместился в столбце таблицы!). Генератор строки 15 предложен Дж. Марсалья (G. Marsaglia) в качестве "кандидата на наилучший множитель",

($\epsilon = \frac{1}{10}$)

$\lg \nu_2$	$\lg \nu_3$	$\lg \nu_4$	$\lg \nu_5$	$\lg \nu_6$	μ_2	μ_3	μ_4	μ_5	μ_6	Строка
4.5	4.5	4.5	4.5	4.4	$2\epsilon^5$	$5\epsilon^4$	0.01	0.34	4.62	1
7.0	7.0	7.0	7.0	4.0	$2\epsilon^6$	$3\epsilon^4$	0.04	4.66	$2\epsilon^3$	2
17.5	1.3	1.0	1.0	1.0	3.14	$2\epsilon^9$	$2\epsilon^9$	$5\epsilon^9$	ϵ^8	3
15.7	10.0	7.4	5.1	5.1	0.27	0.13	0.11	0.01	0.21	4
4.0	2.5	1.9	1.3	1.0	3.36	2.69	3.78	1.81	1.29	5
16.0	10.0	8.0	5.4	4.5	1.44	0.44	1.92	0.07	0.08	6
16.0	9.0	8.3	5.9	5.6	1.35	0.06	4.69	0.35	6.98	7
16.7	10.7	7.8	6.3	4.8	3.37	1.75	1.20	1.39	0.28	8
16.5	11.1	7.0	6.4	5.2	2.89	4.15	0.14	2.04	1.25	9
11.5	11.5	7.2	7.0	5.3	$8\epsilon^4$	2.95	0.07	5.53	0.50	10
17.5	10.7	8.4	6.8	5.6	3.03	0.61	1.85	2.99	1.73	11
14.5	3.4	3.4	3.4	3.4	3.14	ϵ^5	ϵ^4	ϵ^3	0.02	12
16.0	10.2	6.9	6.1	4.9	3.16	1.73	0.26	2.02	0.89	13
16.1	10.5	7.7	6.1	4.7	3.41	2.92	2.32	1.81	0.35	14
16.0	10.5	7.8	6.4	4.0	3.10	2.91	3.20	5.01	0.02	15
16.1	10.6	8.0	6.0	5.0	3.61	3.45	4.66	1.31	1.35	16
15.2	9.9	7.6	5.9	5.1	2.10	1.66	3.14	1.69	3.60	17
15.4	10.3	7.8	6.3	5.3	2.89	4.18	5.34	7.13	7.52	18
14.0	9.3	7.2	6.1	4.9	0.41	0.51	1.08	3.22	1.73	19
15.4	10.2	7.8	6.1	5.2	2.91	3.35	5.17	3.15	6.63	20
15.3	10.2	7.7	6.3	5.2	2.42	3.24	4.15	8.37	7.16	21
22.8	15.1	10.4	9.0	7.3	2.48	2.42	0.25	3.10	1.33	22
24.1	16.0	12.0	9.1	7.9	3.60	3.92	5.27	0.97	3.82	23
30.5	19.4	15.4	10.8	10.3	1.65	0.29	3.88	0.02	4.69	24
31.0	20.2	14.6	11.8	9.8	3.14	1.49	0.44	0.69	0.66	25
31.5	21.3	16.0	12.7	10.4	1.50	3.68	4.52	4.02	1.76	26
31.0	16.0	15.5	15.4	10.4	$5\epsilon^5$	$4\epsilon^9$	$8\epsilon^5$	2.56	ϵ^4	27
288.	192.	144.	115.	95.9	2.27	3.46	3.92	2.49	2.98	28
688.	458.	344.	275.	229.	3.10	2.04	2.85	1.15	1.33	29
Верхняя граница из (40):					3.63	5.92	9.87	14.89	23.87	

после компьютерных исследований для почти кубических решеток размерностью от 2 до 5. Это предложение было сделано, в частности, потому, что множитель можно легко запомнить (см. книгу под редакцией С. К. Зарембы [*Applications of Number Theory to Numerical Analysis*, edited by S. K. Zaremba (New York: Academic Press, 1972), 275]).

В строке 17 как множитель используется первообразный корень по модулю простого числа $2^{31} - 1$. В строке 18 приведен наилучший с точки зрения спектрального критерия первообразный корень для $2^{31} - 1$, найденный в исчерпывающем исследовании Дж. С. Фишмана (G. S. Fishman) и Л. Р. Мура III (L. R. Moore III); *SIAM J. Sci. Stat. Comput.* **7** (1986), 24–45. Похожий, но не менее выдающийся множитель $16807 = 7^5$ в строке 19 стал более часто использоваться для этого модуля, после того как его предложили Левис, Гудман и Миллер (см. работу Lewis, Goodman, and Miller в *IBM Systems J.* **8** (1969), 136–146). Генератор с этим множителем является основным с 1971 года в популярной библиотеке программ IMSL. Основная причина продолжительного использования $a = 16807$ состоит в том, что a^2 меньше модуля m, поэтому операция $ax \bmod m$ может быть выполнена с высокой эффек-

тивностью на языках высокого уровня, использующих технику из упр. 3.2.1.1–9. Однако такие малые множители имеют известные дефекты. С. К. Парк (S. K. Park) и К. В. Миллер (K. W. Miller) заметили, что эта же техника может быть применена к некоторым множителям, большим, чем $\sqrt{m}$, поэтому они попросили Дж. С. Фишмана найти наилучший "эффективно применимый" множитель из этого широкого класса. Результат поисков приводится в строке 20 [*CACM* **31** (1988), 1192–1201]. В строке 21 содержится другой хороший множитель, предложенный П. Лекуером (см. P. L'Ecuyer, *CACM* **31** (1988), 742–749, 774); этот генератор использует немного меньший простой модуль.

Если генераторы в строках 20 и 21 объединить с помощью вычитания, как показано в формуле 3.2.2–(15), то генерируемые числа $\langle Z_n \rangle$ будут удовлетворять рекуррентным соотношениям

$$X_{n+1} = 48271X_n \bmod (2^{31}-1), \qquad Y_{n+1} = 40692Y_n \bmod (2^{31}-249),$$
$$Z_n = (X_n - Y_n) \bmod (2^{31}-1). \tag{38}$$

В упр. 32 показано, что разумно проверить $\langle Z_n \rangle$ с помощью спектрального критерия для $m = (2^{31}-1)(2^{31}-249)$ и $a = 1431853894371298687$. (Это значение a удовлетворяет равенствам $a \bmod (2^{31}-1) = 48271$ и $a \bmod (2^{31}-249) = 40692$.) Результат приведен в строке 24. Не следует особо заботиться о нижней грани значения μ_5, так как $\nu_5 > 1000$. Длина периода генератора (38) равна $(2^{31}-2)(2^{31}-250)/62 \approx 7\times 10^{16}$.

В строке 25 таблицы приведена последовательность

$$X_n = (271828183X_{n-1} - 314159269X_{n-2}) \bmod (2^{31}-1), \tag{39}$$

которая имеет длину периода $(2^{31}-1)^2-1$, что легко показать. Она проверена с использованием обобщенного спектрального критерия в упр. 24.

В последних трех строках табл. 1 содержатся генераторы, основанные на методах суммирования с переносом и вычитания с заимствованием. Они моделируют линейные конгруэнтные последовательности с крайне большими модулями (см. упр. 3.2.1.1–14). В строке 27 приведен генератор

$$X_n = (X_{n-1} + 65430X_{n-2} + C_n) \bmod 2^{31},$$
$$C_{n+1} = \lfloor (X_{n-1} + 65430X_{n-2} + C_n)/2^{31} \rfloor,$$

который соответствует генератору $\mathcal{X}_{n+1} = (65430\cdot 2^{31}+1)\mathcal{X}_n \bmod (65430\cdot 2^{62}+2^{31}-1)$. Числа в таблице отвечают "супервеличинам"

$$\mathcal{X}_n = (65430\cdot 2^{31}+1)X_{n-1} + 65430X_{n-2} + C_n$$

лучше, чем величинам X_n, действительно вычисляемым и используемым как случайные числа. В строке 28 представлен более типичный генератор, основанный на методе вычитания с заимствованием

$$X_n = (X_{n-10} - X_{n-24} - C_n) \bmod 2^{24}, \quad C_{n+1} = [X_{n-10} < X_{n-24} + C_n],$$

но модифицированный так, что при генерировании 389 элементов последовательности используются только первые (или последние) 24. Этот генератор называется RANLUX и предложен Мартином Люшером (Martin Lüscher) после того, как он прошел проверку такими строгими критериями, которые забраковали предыдущие

генераторы (см. *Computer Physics Communications* **79** (1994), 100–110). Подобная последовательность

$$X_n = (X_{n-22} - X_{n-43} - C_n) \bmod (2^{32} - 5), \quad C_{n+1} = [X_{n-22} < X_{n-43} + C_n],$$

такая, что из 400 генерируемых чисел выбираются 43, приведена в строке 29. Она обсуждается в ответе к упр. 3.2.1.2–22. В обоих случаях элементы таблицы соответствуют спектральному критерию, примененному к числам $\mathcal{X}_n$, которые имеют высокую точность, вместо того, чтобы соответствовать индивидуальным "цифрам" X_n, но большие значения μ показывают, что процесс генерирования 389 или 400 случайных чисел для отбора 24 или 43 — это идеальный путь устранения смещения благодаря крайней простоте схемы генерирования.

Теоретические верхние грани для μ_t, которые не могут быть превышены для любого m, приведены сразу под табл. 1. Известно, что для каждой решетки, на единицу объема которой приходится m точек, выполняется неравенство

$$\nu_t \le \gamma_t^{1/2} m^{1/t}, \tag{40}$$

где γ_t принимает соответствующие значения

$$(4/3)^{1/2}, \quad 2^{1/3}, \quad 2^{1/2}, \quad 2^{3/5}, \quad (64/3)^{1/6}, \quad 4^{3/7}, \quad 2 \tag{41}$$

для $t = 2, \ldots, 8$. [См. упр. 9 и книги Дж. В. С. Касселя (J. W. S. Cassels, *Introduction to the Geometry of Numbers* (Berlin: Springer, 1959), 332); Дж. Х. Конвей и Н. Дж. А. Слоан (J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups* (New York: Springer, 1988), 20).] Эти грани достигаются для решеток, порожденных векторами с произвольными действительными координатами. Например, оптимальная решетка для $t = 2$ будет шестиугольной, порождается она векторами длиной $2/\sqrt{3m}$, образующими две стороны равностороннего треугольника. Для размерности 3 оптимальная решетка порождена векторами V_1, V_2, V_3, которые могут иметь вид $(v, v, -v)$, $(v, -v, v)$, $(-v, v, v)$, где $v = 1/\sqrt[3]{4m}$.

Л. К. Киллингбек провел исчерпывающие исследования множителей $a \equiv 1 \bmod 4$, когда $m = 2^{32}$. Он показал, что для множителя $a = 2650845021$ справедливы равенства $\nu_2^2 = 4938969760$, $\nu_3^2 = 2646962$, $\nu_4^2 = 68342$, $\nu_5^2 = 8778$ и $\nu_6^2 = 1506$. Таким образом, он превосходит другие множители, приведенные в таблице для этого модуля. Действительно, замечательные значения μ_k $(3.61, 4.20, 5.37, 8.85, 4.11)$ превышают все значения μ_2, μ_3, μ_4 и μ_5 для всех модулей таблицы.

***F. Связь с критерием серий.** В ряде значительных работ, опубликованных в 70-е годы, Гаральд Нидеррейтер (Harald Niederreiter) показал, как проводить исследования t-мерных векторов (1) с помощью экспоненциальных сумм. Одним из основных результатов его теории было следующее: он показал, что генератор случайных чисел проходит проверку с помощью критерия серий для нескольких измерений, если этот генератор выдерживает проверку спектральным критерием, даже когда вместо полного периода рассматривается большая его часть. Кратко рассмотрим этот интересный метод для линейной конгруэнтной последовательности (X_0, a, c, m) с длиной периода m.

Первое (в дальнейшем — необходимое) понятие — *разброс* в t измерениях, т. е. величина, которую определим как максимум разности между средним числом и

реальным числом t-мерных векторов $(x_n, x_{n+1}, \ldots, x_{n+t-1})$, попадающих в гиперпрямоугольную область, по всем таким областям. Точнее, если $\langle x_n \rangle$ — последовательность целых чисел в области $0 \le x_n < m$, определим

$$D_N^{(t)} = \max_R \left| \frac{\text{число } (x_n, \ldots, x_{n+t-1}) \text{ в } R \text{ для } 0 \le n < N}{N} - \frac{\text{объем } R}{m^t} \right|, \qquad (42)$$

где R — области вида

$$R = \{(y_1, \ldots, y_t) \mid \alpha_1 \le y_1 < \beta_1, \ \ldots, \ \alpha_t \le y_t < \beta_t\}; \qquad (43)$$

здесь α_j и β_j — целые числа из областей $0 \le \alpha_j < \beta_j \le m$ для $1 \le j \le t$. Объем R, очевидно, равен $(\beta_1 - \alpha_1) \ldots (\beta_t - \alpha_t)$. Чтобы найти разброс $D_N^{(t)}$, окинем взглядом все эти множества R и найдем такое, у которого самое большее или самое меньшее число точек вида $(x_n, \ldots, x_{n+t-1})$.

Верхняя грань разброса может быть найдена с помощью экспоненциальных сумм. Пусть $\omega = e^{2\pi i/m}$ — первообразный m-й корень из единицы. Если $(x_1, \ldots, x_t)$ и $(y_1, \ldots, y_t)$ — два вектора с компонентами из области $0 \le x_j, y_j < m$, то справедливо равенство

$$\sum_{0 \le u_1, \ldots, u_t < m} \omega^{(x_1 - y_1)u_1 + \cdots + (x_t - y_t)u_t} = \begin{cases} m^t, & \text{если } (x_1, \ldots, x_t) = (y_1, \ldots, y_t), \\ 0, & \text{если } (x_1, \ldots, x_t) \ne (y_1, \ldots, y_t). \end{cases}$$

Поэтому число векторов $(x_n, \ldots, x_{n+t-1})$ в R для $0 \le n < N$, когда область R определена в (43), может быть выражено следующим образом:

$$\frac{1}{m^t} \sum_{0 \le n < N} \sum_{0 \le u_1, \ldots, u_t < m} \omega^{x_n u_1 + \cdots + x_{n+t-1} u_t} \sum_{\alpha_1 \le y_1 < \beta_1} \cdots \sum_{\alpha_t \le y_t < \beta_t} \omega^{-(y_1 u_1 + \cdots + y_t u_t)}.$$

Когда в этой сумме $u_1 = \cdots = u_t = 0$, то она равна N/m^t, умноженному на объем R. Следовательно, $D_N^{(t)}$ можно выразить, как максимум по R выражения

$$\left| \frac{1}{Nm^t} \sum_{0 \le n < N} \sum_{\substack{0 \le u_1, \ldots, u_t < m \\ (u_1, \ldots, u_t) \ne (0, \ldots, 0)}} \omega^{x_n u_1 + \cdots + x_{n+t-1} u_t} \sum_{\alpha_1 \le y_1 < \beta_1} \cdots \sum_{\alpha_t \le y_t < \beta_t} \omega^{-(y_1 u_1 + \cdots + y_t u_t)} \right|.$$

Поскольку комплексные числа удовлетворяют неравенствам $|w + z| \le |w| + |z|$ и $|wz| = |w||z|$, справедливы соотношения

$$\begin{aligned} D_N^{(t)} &\le \max_R \frac{1}{m^t} \sum_{\substack{0 \le u_1, \ldots, u_t < m \\ (u_1, \ldots, u_t) \ne (0, \ldots, 0)}} \left| \sum_{\alpha_1 \le y_1 < \beta_1} \cdots \sum_{\alpha_t \le y_t < \beta_t} \omega^{-(y_1 u_1 + \cdots + y_t u_t)} \right| g(u_1, \ldots, u_t) \\ &\le \frac{1}{m^t} \sum_{\substack{0 \le u_1, \ldots, u_t < m \\ (u_1, \ldots, u_t) \ne (0, \ldots, 0)}} \max_R \left| \sum_{\alpha_1 \le y_1 < \beta_1} \cdots \sum_{\alpha_t \le y_t < \beta_t} \omega^{-(y_1 u_1 + \cdots + y_t u_t)} \right| g(u_1, \ldots, u_t) \\ &= \sum_{\substack{0 \le u_1, \ldots, u_t < m \\ (u_1, \ldots, u_t) \ne (0, \ldots, 0)}} f(u_1, \ldots, u_t)\, g(u_1, \ldots, u_t), \qquad (44) \end{aligned}$$

где

$$g(u_1,\dots,u_t)=\left|\frac{1}{N}\sum_{0\le n<N}\omega^{x_nu_1+\cdots+x_{n+t-1}u_t}\right|;$$

$$f(u_1,\dots,u_t)=\max_R\frac{1}{m^t}\left|\sum_{\alpha_1\le y_1<\beta_1}\dots\sum_{\alpha_t\le y_t<\beta_t}\omega^{-(y_1u_1+\cdots+y_tu_t)}\right|$$

$$=\max_R\left|\frac{1}{m}\sum_{\alpha_1\le y_1<\beta_1}\omega^{-u_1y_1}\right|\dots\left|\frac{1}{m}\sum_{\alpha_t\le y_t<\beta_t}\omega^{-u_ty_t}\right|.$$

И f, и g можно упростить в дальнейшем для того, чтобы получить хорошую верхнюю грань для $D_N^{(t)}$. Справедливо равенство

$$\left|\frac{1}{m}\sum_{\alpha\le y<\beta}\omega^{-uy}\right|=\left|\frac{1}{m}\,\frac{\omega^{-\beta u}-\omega^{-\alpha u}}{\omega^{-u}-1}\right|\le\frac{2}{m\,|\omega^u-1|}=\frac{1}{m\sin(\pi u/m)},$$

где $u\ne0$ и сумма ≤1, когда $u=0$. Следовательно,

$$f(u_1,\dots,u_t)\le r(u_1,\dots,u_t),\tag{45}$$

где

$$r(u_1,\dots,u_t)=\prod_{\substack{1\le k\le t\\ u_k\ne0}}\frac{1}{m\sin(\pi u_k/m)}.\tag{46}$$

Кроме того, когда $\langle x_n\rangle$ порождена по модулю m линейной конгруэнтной последовательностью, справедливы равенства

$$\begin{aligned}x_nu_1+\cdots+x_{n+t-1}u_t&=x_nu_1+(ax_n+c)u_2+\cdots+\big(a^{t-1}x_n+c(a^{t-2}+\cdots+1)\big)u_t\\&=(u_1+au_2+\cdots+a^{t-1}u_t)x_n+h(u_1,\dots,u_t),\end{aligned}$$

где $h(u_1,\dots,u_t)$ не зависит от n. Значит,

$$g(u_1,\dots,u_t)=\left|\frac{1}{N}\sum_{0\le n<N}\omega^{q(u_1,\dots,u_t)x_n}\right|,\tag{47}$$

где

$$q(u_1,\dots,u_t)=u_1+au_2+\cdots+a^{t-1}u_t.\tag{48}$$

Здесь устанавливается связь со спектральным критерием. Покажем, что сумма $g(u_1,\dots,u_t)$ будет маленькой, если только не выполняется $q(u_1,\dots,u_t)\equiv0$ (по модулю m); другими словами, вклад суммы (44) определяется, в основном, решением (15). Кроме того, в упр. 27 показано, что $r(u_1,\dots,u_t)$ будет малым, когда $(u_1,\dots,u_t)$ является "большим" решением (15). Следовательно, разброс $D_N^{(t)}$ будет малым, когда (15) имеет только "большие" решения, а именно — когда пройдена проверка спектральным критерием. Осталось определить количественные аналоги этих качественных утверждений, чтобы осуществлять точные вычисления.

Сначала рассмотрим величину $g(u_1,\ldots,u_t)$. Когда $N=m$, так что сумма (47) берется по всему периоду, $g(u_1,\ldots,u_t)=0$, кроме случая, когда $(u_1,\ldots,u_t)$ удовлетворяет уравнению (15). Поэтому разброс ограничен сверху суммой $r(u_1,\ldots,u_t)$, взятой по всем ненулевым решениям (15). Теперь рассмотрим, что произойдет с такой же, как (47), суммой, когда N меньше m и $q(u_1,\ldots,u_t)$ не кратно m. Справедливы равенства

$$\begin{aligned}\frac{1}{N}\sum_{0\le n<N}\omega^{x_n} &= \frac{1}{N}\sum_{0\le n<N}\frac{1}{m}\sum_{0\le k<m}\omega^{-nk}\sum_{0\le j<m}\omega^{x_j+jk}\\ &= \frac{1}{N}\sum_{0\le k<m}\left(\frac{1}{m}\sum_{0\le n<N}\omega^{-nk}\right)S_{k0},\end{aligned}\tag{49}$$

где

$$S_{kl}=\sum_{0\le j<m}\omega^{x_{j+l}+jk}.\tag{50}$$

Сейчас $S_{kl}=\omega^{-lk}S_{k0}$, поэтому $|S_{kl}|=|S_{k0}|$ для всех l, и можно вычислить это общее значение, выполнив экспоненциальное суммирование:

$$\begin{aligned}|S_{k0}|^2 &= \frac{1}{m}\sum_{0\le l<m}|S_{kl}|^2\\ &= \frac{1}{m}\sum_{0\le l<m}\sum_{0\le j<m}\omega^{x_{j+l}+jk}\sum_{0\le i<m}\omega^{-x_{i+l}-ik}\\ &= \frac{1}{m}\sum_{0\le i,j<m}\omega^{(j-i)k}\sum_{0\le l<m}\omega^{x_{j+l}-x_{i+l}}\\ &= \frac{1}{m}\sum_{0\le i<m}\sum_{i\le j<m+i}\omega^{(j-i)k}\sum_{0\le l<m}\omega^{(a^{j-i}-1)x_{i+l}+(a^{j-i}-1)c/(a-1)}.\end{aligned}$$

Пусть s — минимальное число среди чисел, для которых $a^s\equiv 1$ (по модулю m), и пусть

$$s'=(a^s-1)c/(a-1)\bmod m.$$

Тогда s — это делитель m (см. лемму 3.2.1.2P) и $x_{n+js}\equiv x_n+js'$ (по модулю m). Сумма по l равна нулю, за исключением случая, когда $j-i$ кратно s, поэтому получим, что

$$|S_{k0}|^2=m\sum_{0\le j<m/s}\omega^{jsk+js'}.$$

Справедливо равенство $s'=q's$, где q' и m — взаимно простые (см. упр. 3.2.1.2–21), поэтому оказывается, что

$$|S_{k0}|=\begin{cases}0, & \text{если } k+q'\not\equiv 0 \text{ (по модулю } m/s),\\ m/\sqrt{s}, & \text{если } k+q'\equiv 0 \text{ (по модулю } m/s).\end{cases}\tag{51}$$

Используя эти равенства в (49) и вспомнив неравенство (45), можно показать, что

$$\left|\frac{1}{N}\sum_{0\le n<N}\omega^{x_n}\right|\le\frac{m}{N\sqrt{s}}\sum_k r(k),\tag{52}$$

где сумма берется по $0 < k < m$, таким, что $k + q' \equiv 0$ (по модулю m/s). Если воспользоваться упр. 25, чтобы оценить оставшуюся сумму, то получится, что

$$\left| \frac{1}{N} \sum_{0 \le n < N} \omega^{x_n} \right| \le \frac{2}{\pi} \frac{\sqrt{s}}{N} \ln s + O\left(\frac{m}{N\sqrt{s}}\right). \tag{53}$$

Те же грани могут быть использованы, чтобы оценить $|N^{-1} \sum_{0 \le n < N} \omega^{qx_n}|$ для любого $q \not\equiv 0$ (по модулю m), так как можно заменить m делителем m. В действительности верхняя грань будет даже меньше, когда q имеет общий делитель с m, так как s и $m/\sqrt{s}$, вообще говоря, становятся меньше (см. упр. 26).

Мы доказали, что часть $g(u_1, \ldots, u_t)$ нашей верхней грани разброса (44) мала, когда N достаточно большое и когда $(u_1, \ldots, u_t)$ не удовлетворяет (15). В упр. 27 доказывается, что часть $f(u_1, \ldots, u_t)$ нашей верхней грани мала, когда сумма берется по всем не равным нулю векторам $(u_1, \ldots, u_t)$, удовлетворяющим (15), и таким, что эти векторы достаточно далеки от $(0, \ldots, 0)$. Объединив результаты, получим следующую теорему Нидеррейтера (Niederreiter).

Теорема N. *Пусть $\langle X_n \rangle$ — линейная конгруэнтная последовательность (X_0, a, c, m) с периодом длиной m и пусть s — наименьшее положительное число, такое, что $a^s \equiv 1$* (по модулю m). *Тогда t-мерный разброс $D_N^{(t)}$ относительно первых N значений $\langle X_n \rangle$, как определено в* (42), *удовлетворяет равенствам*

$$D_N^{(t)} = O\left(\frac{\sqrt{s} \log s \, (\log m)^t}{N}\right) + O\left(\frac{m(\log m)^t}{N\sqrt{s}}\right) + O\big((\log m)^t \, r_{\max}\big); \tag{54}$$

$$D_m^{(t)} = O\big((\log m)^t \, r_{\max}\big). \tag{55}$$

Здесь $r_{\max}$ — максимальное значение величины $r(u_1, \ldots, u_t)$, определенной в (46), *которая взята по всем удовлетворяющим уравнению* (15) *ненулевым целым векторам $(u_1, \ldots, u_t)$.*

Доказательство. Первые два члена O в (54) определяются векторами $(u_1, \ldots, u_t)$ из (44), не удовлетворяющими (15), так как в упр. 25 доказывается, что $f(u_1, \ldots, u_t)$, где сумма берется по *всем* $(u_1, \ldots, u_t)$, равна $O\big(((2/\pi) \ln m)^t\big)$, и упр. 26 ограничивает каждое $g(u_1, \ldots, u_t)$. (Эти члены отсутствуют в (55), поскольку в этом случае $g(u_1, \ldots, u_t) = 0$.) Оставшийся член O в (54) и (55) определяется ненулевым вектором $(u_1, \ldots, u_t)$, который удовлетворяет (15), если использовать грани, полученные в упр. 27. (Внимательно проверяя данное доказательство, каждое O в этой формуле можно заменить функцией от t.) ▮

Равенство (55) относится к критерию серий при размерности t для полного периода, тогда как равенство (54) дает полезную информацию о распределении первых N сгенерированных значений, когда N меньше m, если только N не слишком мало́. Заметим, что (54) гарантирует малый разброс только тогда, когда s достаточно большое, иначе будет доминировать член $m/\sqrt{s}$. Если $m = p_1^{e_1} \ldots p_r^{e_r}$ и $\gcd(a - 1, m) = p_1^{f_1} \ldots p_r^{f_r}$, то s равно $p_1^{e_1 - f_1} \ldots p_r^{e_r - f_r}$ по лемме 3.2.1.2P. Таким образом, наибольшие значения s соответствуют большому потенциалу. В общем

случае $m = 2^e$, $a \equiv 5$ (по модулю 8) и получаем $s = \frac{1}{4}m$. Значит, $D_N^{(t)}$ равно $O\bigl(\sqrt{m}\,(\log m)^{t+1}/N\bigr) + O\bigl((\log m)^t r_{\max}\bigr)$. Не составляет труда доказать, что

$$r_{\max} \le \frac{1}{\sqrt{8}\,\nu_t} \tag{56}$$

(см. упр. 29). Следовательно, равенство (54), в частности, указывает, что разброс в t-мерном случае будет мал, если критерий серий пройден и если N в некоторой степени больше $\sqrt{m}\,(\log m)^{t+1}$.

В смысле теоремы N это почти так же строго. Из результатов упр. 30 следует, что линейные конгруэнтные последовательности, подобные приведенным в строках 8 и 13 табл. 1, имеют разброс порядка $(\log m)^2/m$ при размерности 2. Разброс в этом случае чрезвычайно мал несмотря на то, что существуют области, имеющие вид параллелограмма площадью $\approx 1/\sqrt{m}$, в которых нет точки (U_n, U_{n+1}). Тот факт, что разброс может столь резко измениться, когда точки вращаются, предупреждает о том, что критерий серий может быть не так точен при определении случайности, как инвариантный относительно вращения спектральный критерий.

G. Историческая справка. В 1959 году, когда верхнюю грань ошибки вычисления t-мерного интеграла определяли методом Монте-Карло, Н. М. Коробов придумал метод оценки множителя линейной конгруэнтной последовательности. Его усложненная формула связана, скорее, со спектральным критерием, так как на нее сильно влияют "малые" решения (15); но это не совсем одно и то же. Критерий Коробова широко обсуждался в литературе и изучался Купером и Нидеррейтером (см. Kuipers and Niederreiter, *Uniform Distribution of Sequences* (New York: Wiley, 1974), §2.5).

Необычно сформулировали спектральный критерий Р. Р. Ковэю и Р. Д. Мак-Ферсон (R. R. Coveyou and R. D. MacPherson, *JACM* **14** (1967), 100–119), введя его интересным косвенным путем. Вместо того чтобы работать с решеточной структурой последовательных точек, они рассматривали случайные числа генераторов как источник t-мерных "волн". Числа $\sqrt{x_1^2 + \cdots + x_t^2}$, такие, что $x_1 + \cdots + a^{t-1}x_t \equiv 0$ (по модулю m), в их трактовке рассматривались как "частоты" волн или точки "спектра", определенного генератором случайных чисел, с низкочастотными волнами, которые неблагоприятны для случайности. Отсюда название *спектральный критерий*. Ковэю и Мак-Ферсон ввели аналогичную алгоритму S процедуру для выполнения их критерия, основанную на принципах леммы A. Тем не менее их оригинальная процедура (в которой используются матрицы UU^T и VV^T вместо U и V) имела дело с крайне большими числами. Идея работы непосредственно с U и V была независимо предложена Ф. Янссенсом и У. Дитером (см. F. Janssens and U. Dieter, *Math. Comp.* **29** (1975), 827–833).

Несколько других авторов указывают, что спектральный критерий можно было бы объяснять в намного более конкретных терминах, и предлагают изучить решетки и решетчатые структуры соответствующих линейных конгруэнтных последовательностей, что сделает фундаментальные ограничения случайности графически наглядными. [См. работы Марсальи, Вуда, Ковэю, Беера, Руфа и Вильямсон, Марсальи и Беера: G. Marsaglia, *Proc. Nat. Acad. Sci.* **61** (1968), 25–28; W. W. Wood, *J. Chem. Phys.* **48** (1968), 427; R. R. Coveyou, *Studies in Applied Math.* **3** (Philadelphia: SIAM, 1969), 70–111; W. A. Beyer, R. B. Roof, and D. Williamson, *Math. Comp.* **25** (1971),

345–360; G. Marsaglia and W. A. Beyer, *Applications of Number Theory to Numerical Analysis*, edited by S. K. Zaremba (New York: Academic Press, 1972), 249–285, 361–370.]

Р. Дж. Стоунхам (R. G. Stoneham), используя оценки экспоненциальных сумм, показал, что $p^{1/2+\epsilon}$ или более элементов последовательности $a^k X_0 \bmod p$ имеют асимптотически малый разброс, когда a — примитивный корень по модулю простого числа p [*Acta Arithmetica* **22** (1973), 371–389]. У Гаральда Нидеррейтера это объяснение растянулось на несколько статей (см. работы Harald Niederreiter, *Math. Comp.* **28** (1974), 1117–1132; **30** (1976), 571–597; *Advances in Math.* **26** (1977), 99–181; *Bull. Amer. Math. Soc.* **84** (1978), 957–1041. См. также его книгу H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods* (Philadelphia: SIAM, 1992)).

УПРАЖНЕНИЯ

1. [*M10*] Что произойдет со спектральным критерием, если размерность уменьшить до *единицы*? (Другими словами, что произойдет при $t = 1$?)

2. [*HM20*] Пусть $V_1, \ldots, V_t$ — линейно независимые векторы в t-мерном пространстве, пусть L_0 — решетка из точек, определенных в (10), и пусть $U_1, \ldots, U_t$ определены в (19). Докажите, что максимальное расстояние между $(t-1)$-мерными гиперплоскостями семейства параллельных гиперплоскостей, покрывающих L_0, равно $1/\min\{f(x_1, \ldots, x_t)^{1/2} \mid (x_1, \ldots, x_t) \neq (0, \ldots, 0)\}$, где f определено в (17).

3. [*M24*] Определите ν_3 и ν_4 для всех линейных конгруэнтных генераторов с потенциалом 2 и периодом длиной m.

▶ **4.** [*M23*] Пусть u_{11}, u_{12}, u_{21}, u_{22} — элементы матрицы размера 2×2, состоящей из целых чисел и такой, что $u_{11} + au_{12} \equiv u_{21} + au_{22} \equiv 0$ (по модулю m) и $u_{11}u_{22} - u_{21}u_{12} = m$.

a) Докажите, что все целые решения (y_1, y_2) уравнения $y_1 + ay_2 \equiv 0$ (по модулю m) имеют вид $(y_1, y_2) = (x_1u_{11} + x_2u_{21}, x_1u_{12} + x_2u_{22})$ для целых x_1, x_2.

b) Если вдобавок $2|u_{11}u_{21} + u_{12}u_{22}| \leq u_{11}^2 + u_{12}^2 \leq u_{21}^2 + u_{22}^2$, докажите, что $(y_1, y_2) = (u_{11}, u_{12})$ минимизирует $y_1^2 + y_2^2$ по всем соответствующим ненулевым решениям этого уравнения.

5. [*M30*] Докажите, что двумерный спектральный критерий на шагах S1–S3 алгоритма S выполняется правильно. [*Указание.* См. упр. 4; доказательство того, что $(h' + h)^2 + (p' + p)^2 \geq h^2 + p^2$, начните с шага S2.]

То, что указанное неравенство, несомненно, впервые выполняется на шаге S2, является неожиданным. Целое число q', минимизирующее $(h' - q'h)^2 + (p' - q'p)^2$, согласно (24) равно $q' =$ округление$((h'h + p'p)/(h^2 + p^2))$. Если $(h' - q'h)^2 + (p' - q'p)^2 < h^2 + p^2$, получим $q' \neq 0$, $q' \neq -1$. Следовательно, $(p' - q'p)^2 \geq p^2$; отсюда $(h' - q'h)^2 < h^2$, т. е. $|h' - q'h| < h$ либо q' равно q или $q + 1$. Получим $hu + pv \geq h(h' - q'h) + p(p' - q'p) \geq -\frac{1}{2}(h^2 + p^2)$. Если $u^2 + v^2 < s$, то на следующей итерации шага S2 будет сохранено предположение указания. Если $u^2 + v^2 \geq s > (u-h)^2 + (v-p)^2$, получим $2|h(u-h) + p(v-p)| = 2(h(h-u) + p(p-v)) = (u-h)^2 + (v-p)^2 + h^2 + p^2 - (u^2 + v^2) \leq (u-h)^2 + (v-p)^2 \leq h^2 + p^2$. Следовательно, согласно упр. 4 $(u-h)^2 + (v-p)^2$ является минимальным. Наконец, если и $u^2 + v^2$, и $(u-h)^2 + (v-p)^2$ будут $\geq s$, положим $u' = h' - q'h$, $v' = p' - q'p$; тогда согласно упр. 4 $2|hu' + pv'| \leq h^2 + p^2 \leq u'^2 + v'^2$ и $h^2 + p^2$ минимальны.

[Общие правила нахождения кратчайших 2-D-векторов относительно других метрик обсуждают Кэйб и Шнорр в работе Kaib and Schnorr, *J. Algorithms* **21** (1996), 565–578.]

6. [*M30*] Пусть $a_0, a_1, \ldots, a_{t-1}$ — частичные отношения a/m, определенные в разделе 3.3.3, и пусть $A = \max_{0 \leq j < t} a_j$. Докажите, что $\mu_2 > 2\pi/(A + 1 + 1/A)$.

7. [*HM22*] Докажите, что вопросы (a) и (b), поставленные после (23), имеют такое же решение для действительных чисел $q_1, \ldots, q_{j-1}, q_{j+1}, \ldots, q_t$ (см. (24) и (26)).

8. [*M18*] В строке 10 табл. 1 значение μ_2 очень мало́, однако μ_3 совершенно удовлетворительно. Чему равно наибольшее возможное значение μ_3, когда $\mu_2 = 10^{-6}$ и $m = 10^{10}$?

9. [*HM32*] (Ч. Эрмит (C. Hermite), 1846.) Пусть $f(x_1, \ldots, x_t)$ — положительно определенная квадратичная форма, определенная матрицей U, как в (17), и пусть θ — минимальное значение f в не равной нулю целой точке. Докажите, что $\theta \le (\frac{4}{3})^{(t-1)/2} |\det U|^{2/t}$. [*Указание.* Если W — любая целочисленная матрица с определителем, равным 1, матрица WU определена формой, эквивалентной f. Если же S — любая ортогональная матрица (т. е. если $S^{-1} = S^T$), матрица US определена формой, равной f. Покажите, что существует эквивалентная форма g, минимум которой θ достигается в $(1, 0, \ldots, 0)$. Затем докажите общий результат индукцией по t, записывая $g(x_1, \ldots, x_t) = \theta(x_1 + \beta_2 x_2 + \cdots + \beta_t x_t)^2 + h(x_2, \ldots, x_t)$, где h — положительно определенная квадратичная форма $t-1$ переменной.]

10. [*M28*] Пусть y_1 и y_2 — взаимно простые числа, такие, что $y_1 + a y_2 \equiv 0$ (по модулю m) и $y_1^2 + y_2^2 < \sqrt{4/3}\, m$. Покажите, что существуют целые числа u_1 и u_2, такие, что $u_1 + a u_2 \equiv 0$ (по модулю m), $u_1 y_2 - u_2 y_1 = m$, $2|u_1 y_1 + u_2 y_2| \le \min(u_1^2 + u_2^2, y_1^2 + y_2^2)$ и $(u_1^2 + u_2^2)(y_1^2 + y_2^2) \ge m^2$. (Следовательно, согласно упр. 4 $\nu_2^2 = \min(u_1^2 + u_2^2, y_1^2 + y_2^2)$.)

▸ **11.** [*HM30*] (Алан Г. Вотерман (Alan G. Waterman), 1974.) Придумайте эффективную процедуру вычисления множителя $a \equiv 1$ (по модулю 4), для которой существует относительно простое решение уравнения $y_1 + a y_2 \equiv 0$ (по модулю m) с $y_1^2 + y_2^2 = \sqrt{4/3}\, m - \epsilon$, где $\epsilon > 0$ настолько мало́, насколько это возможно при заданном $m = 2^e$. (Согласно упр. 10 такой выбор a гарантирует, что $\nu_2^2 \ge m^2/(y_1^2 + y_2^2) > \sqrt{3/4}\, m$, и существует возможность, что ν_2^2 будет близко к своему оптимальному значению $\sqrt{4/3}\, m$. На практике будем подсчитывать несколько таких множителей для малых ϵ, выбирая затем один из них с наилучшими спектральными значениями ν_2, ν_3, ...)

12. [*HM23*] Не прибегая к использованию графических методов, докажите, что любое решение вопроса (b), поставленного после (23), должно удовлетворять уравнениям (26).

13. [*HM22*] В лемме A используется факт, что U не вырождена, для доказательства того, что положительно определенная квадратичная форма принимает некоторое не равное нулю минимальное значение в точке с целыми не равными нулю координатами. Покажите, что это предположение необходимо, рассмотрев квадратичную форму (19), матрица коэффициентов которой не вырождена и для которой значения $f(x_1, \ldots, x_t)$ расположены произвольно в окрестности нуля (но никогда его не достигают) в не равной нулю точке с целыми координатами $(x_1, \ldots, x_t)$.

14. [*24*] Вручную выполните алгоритм S для $m = 100$, $a = 41$, $T = 3$.

▸ **15.** [*M20*] Пусть U — вектор с целыми координатами, удовлетворяющий (15). Сколько $(t-1)$-мерных гиперплоскостей, определенных U, пересекают единичный гиперкуб $\{(x_1, \ldots, x_t) \mid 0 \le x_j < 1 \text{ для } 1 \le j \le t\}$? (Это примерно равно числу гиперплоскостей в семействе, которого достаточно, чтобы покрыть L_0.)

16. [*M30*] (У. Дитер (U. Dieter).) Покажите, как можно преобразовать алгоритм S, чтобы вычислить минимальное число N_t параллельных гиперплоскостей, пересекающих единичный гиперкуб, как в упр. 15 для всех U, которые удовлетворяют (15). [*Указание.* Каким приблизительно будет аналог положительно определенной квадратичной формы и леммы A?]

17. [*20*] Преобразуйте алгоритм S таким образом, чтобы он не только вычислял величины ν_t, но и давал на выходе все векторы с целыми координатами $(u_1, \ldots, u_t)$, удовлетворяющие (15), и такие, что $u_1^2 + \cdots + u_t^2 = \nu_t^2$ при $2 \le t \le T$.

18. [*M30*] В этом упражнении рассмотрен наихудший случай использования алгоритма S.

a) С помощью "комбинаторной матрицы", элементы которой имеют вид $y + x\delta_{ij}$ (см. упр. 1.2.3–39), найдите целочисленные матрицы размера 3×3 U и V, удовлетворяющие (29) и такие, что преобразование на шаге S5 ничего не дает для любого j, но соответствующие значения z_k в (31) настолько велики, что перебор всех значений невозможен. (Матрица U не обязана удовлетворять (28); нас интересует здесь *произвольная* положительно определенная квадратичная форма с определителем m.)

b) Хотя преобразование (23) не используется для матриц, построенных в (a), найдите другое преобразование, которое дает значительное сокращение.

▸ **19.** [*HM25*] Предположим, что шаг S5 изменен так, что преобразование с $q = 1$ осуществляется, когда $2V_i \cdot V_j = V_j \cdot V_j$. (Таким образом, $q = \lfloor (V_i \cdot V_j \,/\, V_j \cdot V_j) + \frac{1}{2} \rfloor$, каково бы ни было $i \neq j$.) Возможно ли, что при этом алгоритм S зациклится?

20. [*M23*] Обсудите, как применить подходящий спектральный критерий к линейной конгруэнтной последовательности, у которой $c = 0$, X_0 — нечетное, $m = 2^e$, $a \bmod 8 = 3$ или 5 (см. упр. 3.2.1.2–9.)

21. [*M20*] (Р. В. Госпер (R. W. Gosper).) Для некоторых задач случайные числа используются группами по четыре числа, но отбрасывается второе число из каждого множества. Как можно исследовать структуру решетки $\{\frac{1}{m}(X_{4n}, X_{4n+2}, X_{4n+3})\}$, которую производит линейный конгруэнтный генератор периода $m = 2^e$?

22. [*M46*] Какова наилучшая верхняя грань для μ_3, если μ_2 очень близко к своему максимальному значению $\sqrt{4/3}\,\pi$? Какова наилучшая верхняя грань μ_2, если μ_3 очень близко к своему максимальному значению $\frac{4}{3}\pi\sqrt{2}$?

23. [*M46*] Пусть U_i, V_j — векторы, координаты которых являются действительными числами с $U_i \cdot V_j = \delta_{ij}$ для $1 \leq i, j \leq t$, и такие, что $U_i \cdot U_i = 1$, $2|U_i \cdot U_j| \leq 1$, $2|V_i \cdot V_j| \leq V_j \cdot V_j$ для $i \neq j$. Насколько большим может быть $V_1 \cdot V_1$? (Этот вопрос имеет отношение к граням на шаге S7, если и (23), и преобразование из упр. 18, (b) не производят никаких сокращений. Известно, что максимальное значение равно $(t+2)/3$. Оно достигается, когда $U_1 = I_1$, $U_j = \frac{1}{2}I_1 + \frac{1}{2}\sqrt{3}\,I_j$, $V_1 = I_1 - (I_2 + \cdots + I_t)/\sqrt{3}$, $V_j = 2I_j/\sqrt{3}$ для $2 \leq j \leq t$, где $(I_1, \ldots, I_t)$ — единичная матрица. Эту схему предложил Б. В. Алексеев.)

▸ **24.** [*M28*] Обобщите спектральный критерий для последовательностей второго порядка вида $X_n = (aX_{n-1} + bX_{n-2}) \bmod p$, имеющих длину периода $p^2 - 1$ (см. формулу 3.2.2–(8)). Как следует изменить алгоритм S?

25. [*HM24*] Пусть d — делитель m и пусть $0 \leq q < d$. Докажите, что сумма $\sum r(k)$, где суммирование производится по всем $0 \leq k < m$, таким, что $k \bmod d = q$, меньше либо равна $(2/d\pi)\ln(m/d) + O(1)$. (Здесь $r(k)$ определено формулой (46) при $t = 1$.)

26. [*M22*] Объясните, почему, если использовать метод доказательства (53), можно получить грани, подобные полученным, для

$$\left| N^{-1} \sum_{0 \leq n < N} \omega^{qx_n} \right|$$

при $0 < q < m$. Почему доказательство (53) ничего не дает, когда $m = 1$?

27. [*HM39*] (Э. Хлавка (E. Hlawka) и Г. Нидеррейтер (H. Niederreiter).) Пусть $r(u_1, \ldots, u_t)$ — функция, определенная в (46). Докажите, что сумма $\sum r(u_1, \ldots, u_t)$, где суммирование производится по всем $0 \leq u_1, \ldots, u_t < m$ так, что $(u_1, \ldots, u_t) \neq (0, \ldots, 0)$, и выполняется равенство (15), меньше или равна $2((\pi + 2\pi \lg m)^t\, r_{\max})$, где $r_{\max}$ — максимальный член $r(u_1, \ldots, u_t)$ этой суммы.

▸ **28.** [*M28*] (Г. Нидеррейтер.) Найдите аналог теоремы N для случая, когда m — простое число, $c = 0$, a — первообразный корень по модулю m, $X_0 \not\equiv 0$ (по модулю m).

[*Указание.* Ваши экспоненциальные суммы должны включать $\zeta = e^{2\pi i/(m-1)}$ так же, как ω.] Докажите, что в этом случае "средний" первообразный корень имеет разброс $D_{m-1}^{(t)} = O\left(t(\log m)^t/\varphi(m-1)\right)$, следовательно, хороший первообразный корень существует для всех m.

29. [*HM22*] Докажите, что величина $r_{\max}$ из упр. 27 никогда не будет больше $1/\sqrt{8}\,\nu_t$.

30. [*M33*] (С. К. Заремба (S. K. Zaremba).) Докажите, что $r_{\max} = O(\max(a_1, \ldots, a_s)/m)$ в двумерном случае, когда $a_1, \ldots, a_s$ — это частичные отношения, полученные в результате применения алгоритма Евклида к m и a. [*Указание.* В обозначениях из раздела 4.5.3 справедливо равенство $a/m = //a_1, \ldots, a_s//$; примените упр. 4.5.3–42.]

31. [*HM47*] (И. Борош (I. Borosh).) Докажите, что для всех достаточно больших m существует взаимно простое с m число a, такое, что все частичные отношения a/m меньше или равны 3. Кроме того, множество всех m, удовлетворяющих этому условию, но с частичными отношениями ≤ 2, имеет положительную плотность.

▸ **32.** [*M21*] Пусть $m_1 = 2^{31} - 1$ и $m_2 = 2^{31} - 249$ — модули генератора (38).

a) Покажите, что если $U_n = (X_n/m_1 - Y_n/m_2) \bmod 1$, то $U_n \approx Z_n/m_1$.

b) Пусть $W_0 = (X_0 m_2 - Y_0 m_1) \bmod m$ и $W_{n+1} = aW_n \bmod m$, где a и m имеют значения, приведенные в разделе после формулы (38). Докажите, что существует простое соотношение между W_n и U_n.

В следующем издании данной книги я планирую ввести новый раздел 3.3.5 под названием "L^3-алгоритм". Это будет отклонение от общей темы ("Случайные числа"), но в нем будет продолжено рассмотрение решетчатого базиса, описанного в разделе 3.3.4. Основным предметом изучения станет неоклассический алгоритм А. К. Ленстра (A. K. Lenstra), H. W. Lenstra, Jr., and L. Lovász, Math. Annalen **261** *(1982), 515–534, для нахождения приближенно оптимального множества базисных векторов и демонстрации того, что алгоритм можно применять к другим исследованиям. Примеры таких исследований приводятся в следующих статьях и содержащейся в них библиографии: M. Seysen, Combinatorica* **13** *(1993), 363–375; C. P. Schnorr and H. H. Hörner, Lecture Notes in Comp. Sci.* **921** *(1995), 1–12.*

3.4. ДРУГИЕ ВИДЫ СЛУЧАЙНЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ

В ПРЕДЫДУЩИХ РАЗДЕЛАХ мы обсуждали, как генерировать на компьютере последовательность чисел $U_0, U_1, U_2, \ldots$, которые ведут себя так, как если бы каждое число выбиралось независимо и случайно между 0 и 1 с равномерным распределением. Однако при использовании случайных чисел часто требуются другие виды распределений. Например, чтобы сделать случайный выбор из k альтернатив, нужны случайные *целые* числа, лежащие между 1 и k. Если необходимо моделировать случайное время ожидания между появлениями независимых событий, желательно получить случайные числа с *показательным распределением*. Иногда в случайных *числах* нет необходимости, но нужны случайные *перестановки* (случайное размещение n объектов) или случайное *сочетание* (случайный выбор k объектов из совокупности, содержащей n объектов).

В принципе, любая из этих случайных величин может быть получена из равномерно распределенных случайных величин $U_0, U_1, U_2, \ldots$. Значительное число "случайных трюков" было придумано для эффективного преобразования равномерно распределенных случайных чисел. Изучив эти методы, получим возможность правильно использовать случайные числа при любом применении метода Монте-Карло.

Вероятно, что кто-нибудь когда-нибудь придумает генератор случайных чисел, который будет вырабатывать одну из этих случайных величин *непосредственно*, а не косвенно через равномерное распределение. Но прямые методы, как доказано, не практичны, за исключением генератора "случайный двоичный разряд", описанного в разделе 3.2.2. (См. также упр. 3.4.1–31; в нем равномерное распределение используется, главным образом, для инициализации, после которой метод является почти полностью прямым.)

В следующем разделе предполагается наличие случайной последовательности равномерно распределенных между 0 и 1 независимых действительных чисел. Равномерно распределенная случайная величина U генерируется всякий раз, когда в ней возникает необходимость. Эти числа обычно представлены в компьютере словом с десятичной точкой слева.

3.4.1. Численные распределения

В этом разделе объединены наиболее известные методы получения случайных чисел для различных важных распределений. Многие из методов первоначально были предложены Джоном фон Нейманом (John von Neumann) в начале 50-х. Постепенно они усовершенствовались другими математиками, особенно Джорджем Марсалья (George Marsaglia), И. Г. Аренсом (J. H. Ahrens) и У. Дитером (U. Dieter).

A. Случайный выбор из ограниченного множества. Самые простые и наиболее общие типы распределений, используемых в приложениях, — это распределения случайных *целых* чисел. Целые числа между 0 и 7 могут быть извлечены из трех двоичных разрядов U на бинарном компьютере; поэтому эти три двоичных разряда можно извлечь из *старшей значащей* (слева) части компьютерного слова, поскольку самые младшие двоичные разряды, производимые многими генераторами случайных чисел, недостаточно случайны (см. раздел 3.2.1.1).

В общем случае случайные целые числа X, которые лежат между 0 и $k-1$, можно получить, *умножив* U на k и положив $X = \lfloor kU \rfloor$. На MIX можно записать

```
LDA  U
MUL  K
```
(1)

После выполнения этих двух операций требуемое целое число появится в регистре A. Чтобы получить случайное целое число, лежащее между 1 и k, следует добавить единицу к этому результату. (Операция "INCA 1" последует за (1).)

С помощью данного метода каждое целое число можно получить с приблизительно равной вероятностью. Существует незначительная ошибка, так как длина слова компьютера конечна (см. упр. 2), но эта ошибка совершенно незначительна, если k мало́, например $k/m < 1/10000$.

В более общем случае можно получить, если необходимо, различные веса для различных целых чисел. Предположим, что значение $X = x_1$ должно быть получено с вероятностью p_1, $X = x_2$ — с вероятностью p_2, ... и $X = x_k$ — с вероятностью p_k. Генерируем равномерное число U и положим

$$X = \begin{cases} x_1, & \text{если } 0 \le U < p_1; \\ x_2, & \text{если } p_1 \le U < p_1 + p_2; \\ \vdots & \\ x_k, & \text{если } p_1 + p_2 + \cdots + p_{k-1} \le U < 1. \end{cases} \tag{2}$$

(Заметим, что $p_1 + p_2 + \cdots + p_k = 1$.)

Существует "наилучший возможный" способ сравнения U с различными значениями $p_1 + p_2 + \cdots + p_s$, как подразумевается в (2) (см. раздел 2.3.4.5). В частных случаях можно обойтись более эффективными методами; например, для того, чтобы получить одно из одиннадцати чисел 2, 3, ..., 12 с соответствующими "игре в кости" вероятностями $\frac{1}{36}, \frac{2}{36}, \ldots, \frac{6}{36}, \ldots, \frac{2}{36}, \frac{1}{36}$, можно вычислить два независимых случайных целых числа между 1 и 6 и сложить их.

Тем не менее существует действительно более быстрый способ выбора $x_1, \ldots, x_k$ с произвольно заданной вероятностью, основанный на остроумном подходе, который был введен в употребление А. Дж. Уолкером (см. A. J. Walker, *Electronics Letters* **10**, 8 (1974), 127–128; *ACM Trans. Math. Software* **3** (1977), 253–256). Предположим, что мы образуем kU и рассматриваем целую часть $K = \lfloor kU \rfloor$ и дробную часть $V = (kU) \bmod 1$ раздельно, например после выполнения операций (1) получим K в регистре A и V — в регистр. Затем всегда можно получить желаемое распределение, выполнив операции

$$\text{если } \quad V < P_K, \quad \text{то } \quad X \leftarrow x_{K+1}, \quad \text{иначе } \quad X \leftarrow Y_K \tag{3}$$

для некоторых подходящих таблиц $(P_0, \ldots, P_{k-1})$ и $(Y_0, \ldots, Y_{k-1})$. В упр. 7 показано, как вообще можно вычислить такие таблицы. Метод Уолкера иногда называют методом псевдонимов .

На бинарном компьютере обычно полезно предполагать, что k является степенью 2, потому что умножение может быть заменено сдвигом. Это можно делать без потери общности, введя дополнительные x_j, которые появляются с вероятностью 0. Например, снова рассмотрим игру в кости. Предположим, что равенство $X = j$

должно произойти со следующими 16 вероятностями.

$$\begin{array}{rcccccccccccccccc} j = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ p_j = & 0 & 0 & \frac{1}{36} & \frac{2}{36} & \frac{3}{36} & \frac{4}{36} & \frac{5}{36} & \frac{6}{36} & \frac{5}{36} & \frac{4}{36} & \frac{3}{36} & \frac{2}{36} & \frac{1}{36} & 0 & 0 & 0 \end{array}$$

Это можно осуществить, используя (3), если $k = 16$ и $x_{j+1} = j$ при $0 \le j < 16$ и если таблицы для P и Y имеют следующий вид.

$$\begin{array}{rcccccccccccccccc} j = & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 \\ P_j = & 0 & 0 & \frac{4}{9} & \frac{8}{9} & 1 & \frac{7}{9} & 1 & 1 & 1 & \frac{7}{9} & \frac{7}{9} & \frac{8}{9} & \frac{4}{9} & 0 & 0 & 0 \\ Y_j = & 5 & 9 & 7 & 4 & * & 6 & * & * & * & 8 & 4 & 7 & 10 & 6 & 7 & 8 \end{array}$$

(Когда $P_j = 1$, Y_j не используются.) Например, значение 7 встречается с вероятностью $\frac{1}{16} \cdot \big((1 - P_2) + P_7 + (1 - P_{11}) + (1 - P_{14})\big) = \frac{6}{36}$, как и требуется. Это необычный способ бросания игральных костей, но результаты получаются такие же, как и в реальной ситуации.

Вероятности p_j, безусловно, могут быть представлены неотрицательными весами $w_1, w_2, \ldots, w_k$; если обозначить сумму весов через W, то $p_j = w_j/W$. В разных применениях отдельные веса весьма изменчивы. Матиас, Виттер и Ни (см. работу Matias, Vitter, and Ni, *SODA* **4** (1993), 361–370) показали, как изменять веса и генерировать X с постоянным средним временем.

B. Общие методы для непрерывных распределений. В общем случае распределение действительных чисел может быть выражено в терминах "функции распределения" $F(x)$, которая точно определяет вероятность того, что случайная величина X не превысит значение x:

$$F(x) = \Pr(X \le x). \tag{4}$$

Эта функция всегда монотонно возрастает от 0 до 1, т. е.

$$F(x_1) \le F(x_2), \quad \text{если } x_1 \le x_2; \qquad F(-\infty) = 0, \qquad F(+\infty) = 1. \tag{5}$$

Примеры функций распределения приведены в разделе 3.3.1 (см. рис. 3). Если $F(x)$ непрерывна и строго возрастающая (так что $F(x_1) < F(x_2)$, когда $x_1 < x_2$), то она принимает все значения между 0 и 1 и существует *обратная функция* $F^{[-1]}(y)$, такая, что для $0 < y < 1$

$$y = F(x) \qquad \text{тогда и только тогда, когда} \qquad x = F^{[-1]}(y). \tag{6}$$

В большинстве случаев, когда $F(x)$ непрерывна и строго возрастающая, можно вычислить случайную величину X с распределением $F(x)$, полагая

$$X = F^{[-1]}(U), \tag{7}$$

где U — равномерно распределенная случайная величина. Действительно, вероятность того, что $X \le x$, равна вероятности, что $F^{[-1]}(U) \le x$, а именно — вероятности того, что $U \le F(x)$, т. е. $F(x)$.

Теперь проблема сводится к решению задачи численного анализа — к нахождению хороших методов вычисления $F^{[-1]}(U)$ с требуемой точностью. Численный

анализ в этой книге о получисленных алгоритмах не рассматривается, однако существует ряд важных методов, способных улучшить общий подход (7), и здесь они будут рассмотрены.

Заметим, что если X_1 — случайная величина, имеющая функцию распределения $F_1(x)$, и если X_2 — независимая от X_1 случайная величина с функцией распределения $F_2(x)$, то

$$\begin{aligned} &\max(X_1, X_2) \quad \text{имеет распределение} \quad F_1(x)F_2(x), \\ &\min(X_1, X_2) \quad \text{имеет распределение} \quad F_1(x) + F_2(x) - F_1(x)F_2(x). \end{aligned} \tag{8}$$

(См. упр. 4.) Например, равномерно распределенная случайная величина U имеет распределение $F(x) = x$ для $0 \le x \le 1$; если U_1, U_2, ..., U_t — независимые равномерно распределенные случайные величины, то $\max(U_1, U_2, \ldots, U_t)$ имеет функцию распределения $F(x) = x^t$ при $0 < x \le 1$. Эта формула является основой критерия "максимум-t", описанного в разделе 3.3.2. Обратная функция равна $F^{[-1]}(y) = \sqrt[t]{y}$. В частном случае при $t = 2$ получаем, следовательно, что формулы

$$X = \sqrt{U} \qquad \text{и} \qquad X = \max(U_1, U_2) \tag{9}$$

дадут одинаковое распределение случайной величины X, хотя, на первый взгляд, это не очевидно. Нет необходимости извлекать квадратный корень из равномерно распределенной случайной величины.

Количество подобных хитростей бесконечно: *любой* алгоритм, использующий случайные числа на входе, дает на выходе случайные величины с *некоторым* распределением. Задача состоит в нахождении общих методов составления алгоритма, обеспечивающего заданную функцию распределения на выходе. Вместо того чтобы рассматривать подобные методы в исключительно абстрактных терминах, изучим, как они могут применяться в важных случаях.

C. Нормальное распределение. Возможно, наиболее значительным неравномерным, непрерывным распределением является *нормальное распределение с нулевым средним значением и среднеквадратичным отклонением, равным единице:*

$$F(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt. \tag{10}$$

Значительность данного распределения показана в разделе 1.2.10. В нашем случае обратную функцию $F^{[-1]}$ не так легко вычислить; но, как мы увидим, существует несколько технических приемов моделирования этого распределения.

1) *Метод полярных координат*, предложенный Дж. Э. П. Боксом, М. Э. Мюллером и Дж. Марсальей [см. G. E. P. Box, M. E. Muller, and G. Marsaglia, *Annals Math. Stat.* **29** (1958), 610–611, и Boeing Scientific Res. Lab. report D1-82-0203 (1962)].

Алгоритм P (*Метод полярных координат для нормальных случайных величин*). Этот алгоритм вычисляет две независимые нормально распределенные случайные величины: X_1 и X_2.

P1. [Получение равномерно распределенных случайных величин.] Генерируем две независимые случайные величины U_1 и U_2, равномерно распределенные между 0

и 1. Присвоить $V_1 \leftarrow 2U_1 - 1$, $V_2 \leftarrow 2U_2 - 1$. (Здесь V_1 и V_2 равномерно распределены между $-$ё1 и $+1$. На большинстве компьютеров предпочтительнее представление V_1 и V_2 в виде чисел с плавающей точкой.)

P2. [Вычисление S.] Присвоить $S \leftarrow V_1^2 + V_2^2$.

P3. [Проверить $S \geq 1$?] Если $S \geq 1$, возврат к шагу P1. (Шаги P1–P3 выполняются в среднем 1.27 раз со среднеквадратичным отклонением, равным 0.587; см. упр. 6.)

P4. [Вычисление X_1, X_2.] Присвоить X_1 и X_2 следующие значения:

$$X_1 \leftarrow V_1\sqrt{\frac{-2\ln S}{S}}, \qquad X_2 \leftarrow V_2\sqrt{\frac{-2\ln S}{S}}. \tag{11}$$

Это требуемые нормально распределенные случайные величины. ▮

Для доказательства законности данного метода используем элементарную аналитическую геометрию и вычисления: если на шаге P3 $S < 1$, точка плоскости с декартовыми координатами (V_1, V_2) *является случайной точкой, равномерно распределенной внутри единичного круга*. Перейдя к полярным координатам $V_1 = R\cos\Theta$, $V_2 = R\sin\Theta$, получим

$$S = R^2, \quad X_1 = \sqrt{-2\ln S}\cos\Theta, \quad X_2 = \sqrt{-2\ln S}\sin\Theta.$$

Используя также полярные координаты $X_1 = R'\cos\Theta'$ и $X_2 = R'\sin\Theta'$, получим $\Theta' = \Theta$ и $R' = \sqrt{-2\ln S}$. Ясно, что R' и Θ' независимы, поскольку R и Θ независимы в единичном круге. К тому же Θ' равномерно распределено между 0 и 2π, и вероятность того, что $R' \leq r$, равна вероятности, что $-2\ln S \leq r^2$, т. е. вероятности, что $S \geq e^{-r^2/2}$. Эта вероятность равна $1 - e^{-r^2/2}$, так как $S = R^2$ равномерно распределено между 0 и 1. Вероятность того, что R' лежит между r и $r+dr$, поэтому равна дифференциалу от $1 - e^{-r^2/2}$, т. е. $re^{-r^2/2}\,dr$. Аналогично вероятность того, что Θ' лежит между θ и $\theta + d\theta$, равна $(1/2\pi)\,d\theta$. Совместная вероятность того, что $X_1 \leq x_1$ и $X_2 \leq x_2$, равняется

$$\begin{aligned}\int_{\{(r,\theta)\,|\,r\cos\theta\leq x_1,\ r\sin\theta\leq x_2\}} &\frac{1}{2\pi}e^{-r^2/2}r\,dr\,d\theta \\ &= \frac{1}{2\pi}\int_{\{(x,y)\,|\,x\leq x_1,\ y\leq x_2\}} e^{-(x^2+y^2)/2}\,dx\,dy \\ &= \left(\sqrt{\frac{1}{2\pi}}\int_{-\infty}^{x_1} e^{-x^2/2}\,dx\right)\left(\sqrt{\frac{1}{2\pi}}\int_{-\infty}^{x_2} e^{-y^2/2}\,dy\right).\end{aligned}$$

Это и доказывает, что X_1 и X_2 независимы и нормально распределены.

2) *Метод прямоугольника-клина-хвоста* предложен Дж. Марсальей. Здесь используется функция

$$F(x) = \operatorname{erf}(x/\sqrt{2}\,) = \sqrt{\frac{2}{\pi}}\int_0^x e^{-t^2/2}\,dt, \qquad x \geq 0, \tag{12}$$

которая является функцией распределения *абсолютного значения* нормальной случайной величины. Затем X вычисляется в соответствии с распределением (12).

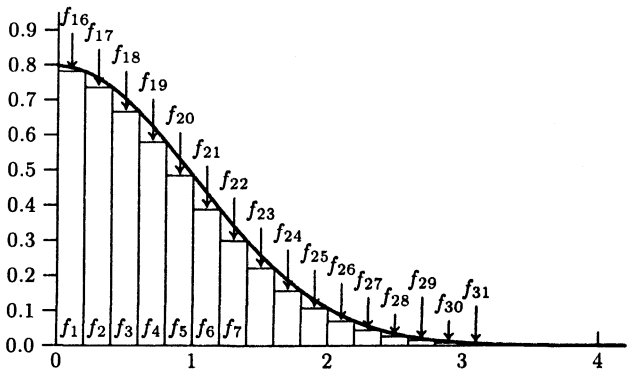


Рис. 9. Площадь под графиком плотности распределения разделена на 31 часть. Площадь каждой части равна среднему числу вычислений случайной величины с такой плотностью.

Припишем случайный знак ее значению, и это сделает ее действительно нормальной случайной величиной.

Метод прямоугольника-клина-хвоста основан на важных общих технических приемах, которые будут рассмотрены ниже по мере построения алгоритма. Первая ключевая идея — рассматривать $F(x)$ как *смесь* нескольких других функций, т. е. записать

$$F(x) = p_1F_1(x) + p_2F_2(x) + \cdots + p_nF_n(x), \tag{13}$$

где $F_1, F_2, \ldots, F_n$ — подходящие распределения и $p_1, p_2, \ldots, p_n$ — неотрицательные вероятности, сумма которых равна 1. Если генерировать случайную переменную X, выбирая распределение F_j с вероятностью p_j, то легко видеть, что X точно будет иметь F-распределение. С некоторыми распределениями $F_j(x)$, пожалуй, трудно иметь дело, даже труднее, чем с F, но мы обычно устраиваем так, что вероятности p_j в этом случае очень малы. Большинство распределений $F_j(x)$ будут довольно хорошо устроены, поскольку они будут простой модификацией равномерного распределения. Изложение завершается чрезвычайно эффективной программой, поскольку *среднее* время счета этой программы очень мало́.

Рассма́триваемый здесь метод легче понять, если работать с *производными* распределений, а не с самими распределениями. Пусть

$$f(x) = F'(x), \qquad f_j(x) = F_j{}'(x)$$

будут *плотностями распределений*. Тогда равенство (13) можно записать как

$$f(x) = p_1f_1(x) + p_2f_2(x) + \cdots + p_nf_n(x). \tag{14}$$

Каждая $f_j(x)$ есть ≥ 0, и общая площадь под графиком $f_j(x)$ равна 1. Поэтому существует подходящий графический метод отображения зависимости (14): площадь под $f(x)$ разделена на n частей и части, соответствующие $f_j(x)$, имеют площадь p_j. На рис. 9 иллюстрируется интересующий нас случай при $f(x) = F'(x) = \sqrt{2/\pi}\,e^{-x^2/2}$; площадь под этой кривой разделена на $n = 31$ часть. Существует 15 прямоугольников, представляющих $p_1f_1(x), \ldots, p_{15}f_{15}(x)$, 15 клинообразных частей, представляющих $p_{16}f_{16}(x), \ldots, p_{30}f_{30}(x)$, и оставшаяся часть $p_{31}f_{31}(x)$ — "хвост", т. е. график $f(x)$ при $x \geq 3$.

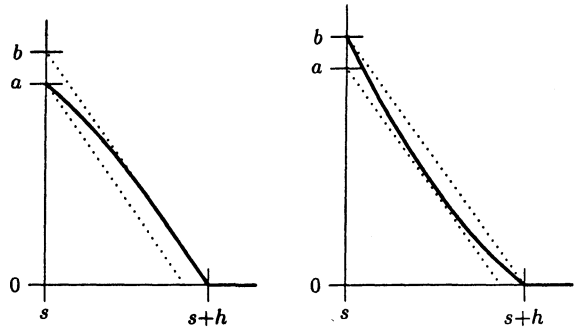


Рис. 10. Плотность распределения, для которой алгоритм L может использоваться при генерировании случайных чисел.

Прямоугольные части $f_1(x)$, ..., $f_{15}(x)$ представляют *равномерное распределение*. Например, $f_3(x)$ представляет случайную равномерно распределенную величину, лежащую между $\frac{2}{5}$ и $\frac{3}{5}$. Высота $p_j f_j(x)$ равна $f(j/5)$; следовательно, площадь j-го прямоугольника равна

$$p_j = \frac{1}{5} f(j/5) = \sqrt{\frac{2}{25\pi}}\, e^{-j^2/50} \qquad \text{для } 1 \le j \le 15. \tag{15}$$

Чтобы генерировать распределение, соответствующее таким прямоугольным частям, просто вычислим

$$X = \tfrac{1}{5}U + S, \tag{16}$$

где U равномерно и S принимает значение $(j-1)/5$ с вероятностью p_j и не зависит от U. Так как $p_1 + \cdots + p_{15} = .9183$, можно просто использовать равномерно распределенные случайные величины, подобные этим приблизительно в 92% случаев.

В оставшихся 8% случаев будем обычно генерировать одно из клинообразных распределений F_{16}, ..., F_{30}. Типичный пример, который показывает, что необходимо делать, представлен на рис. 10. Когда $x < 1$, часть кривой вогнутая, а когда $x > 1$, она выпуклая, но в каждом случае часть кривой достаточно близка к прямой линии и, как показано, может быть заключена между двумя параллельными линиями.

Чтобы перебрать эти клиновидные распределения, будем использовать другой общий технический прием: *метод отбраковки* фон Неймана получения сложной плотности из другой плотности, которая ее "заключает". Описанный выше метод полярных координат является простым примером такого подхода: шаги P1–P3 получают случайную точку внутри единичного круга, генерируя ее в большем круге, отбраковывая ее и начиная снова, если точка была вне круга.

Общий метод отбраковки является даже более сильным, чем этот. Пусть нужно генерировать случайную величину X с плотностью f и пусть g — другая плотность распределения, такая, что

$$f(t) \le cg(t) \tag{17}$$

для всех t, где c — константа. Тогда генерируем случайную величину X с плотностью g, а также независимую равномерно распределенную случайную величину U. Если $U \ge f(X)/cg(X)$, отбрасываем X и начинаем снова с другими X и U. Когда

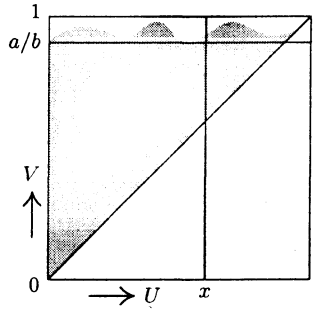


Рис. 11. Область "принятия гипотезы" алгоритма L.

условие $U < f(X)/cg(X)$ в конце концов выполняется, на выходе X будет иметь требуемую плотность f. [*Доказательство.* $X \le x$ произойдет с вероятностью $p(x) = \int_{-\infty}^{x} \big(g(t)\,dt \cdot f(t)/cg(t)\big) + qp(x)$, где величина $q = \int_{-\infty}^{\infty} \big(g(t)\,dt \cdot (1 - f(t)/cg(t))\big) = 1 - 1/c$ равна вероятности отбраковки; следовательно, $p(x) = \int_{-\infty}^{x} f(t)\,dt$.]

Техника отбраковки наиболее эффективна, когда c мало́, так как должно быть в среднем c итераций, прежде чем значение будет принято (см. упр. 6). В одних случаях $f(x)/cg(x)$ всегда равно 0 или 1 и нет необходимости генерировать U. В других случаях, если $f(x)/cg(x)$ трудно вычислить, следует постараться "втиснуть" его между двумя более простыми граничными функциями

$$r(x) \le f(x)/cg(x) \le s(x) \tag{18}$$

и точное значение $f(x)/cg(x)$ не нужно вычислять, если не выполняется неравенство $r(x) \le U < s(x)$. Следующий алгоритм разрешает проблему клина, совершенствуя метод отбраковки.

Алгоритм L (*Плотности, близкие к линейным*). Этот алгоритм можно использовать для генерирования случайной величины X с любым распределением, плотность $f(x)$ которого удовлетворяет следующим условиям (см. рис. 10):

$$\begin{aligned} f(x) &= 0 \qquad\qquad \text{для } x < s \text{ и для } x > s + h; \\ a - b(x-s)/h \le f(x) &\le b - b(x-s)/h \qquad \text{для } s \le x \le s + h. \end{aligned} \tag{19}$$

L1. [Получить $U \le V$.] Генерировать две независимые случайные величины U и V, равномерно распределенные между 0 и 1. Если $U > V$, заменить $U \leftrightarrow V$.

L2. [Простой случай?] Если $V \le a/b$, перейти к шагу L4.

L3. [Попытаемся снова?] Если $V > U + (1/b)f(s + hU)$, возвратиться к шагу L1. (Если a/b близко к 1, данный шаг алгоритма нужен не очень часто).

L4. [Вычисление X.] Присвоить $X \leftarrow s + hU$. ▮

Когда достигнут шаг L4, точка (U, V) является случайной точкой на площади, заштрихованной на рис. 11, а именно — $0 \le U \le V \le U + (1/b)f(s + hU)$. Условия (19) гарантируют, что

$$\frac{a}{b} \le U + \frac{1}{b} f(s + hU) \le 1.$$

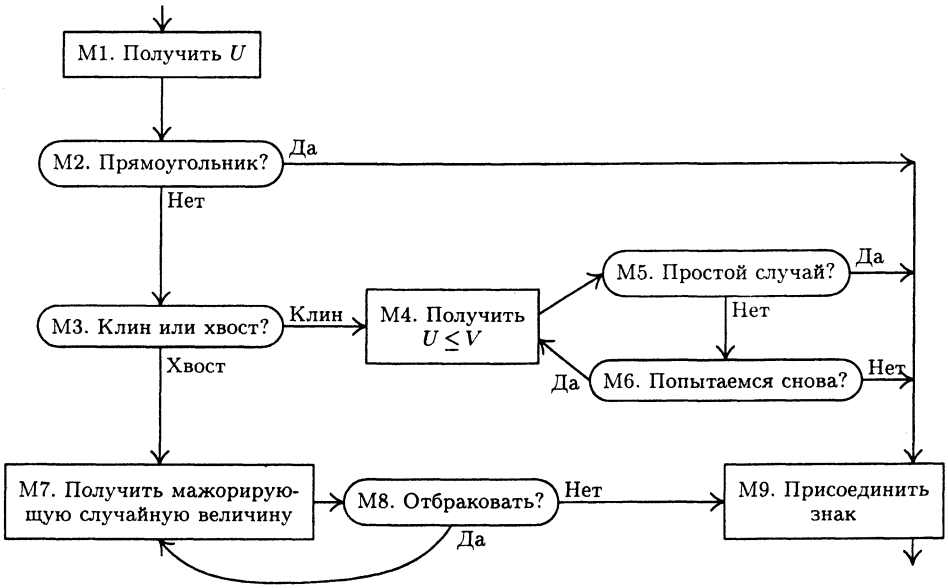


Рис. 12. Алгоритм прямоугольника-клина-хвоста для генерирования нормальных случайных величин.

Сейчас вероятность того, что $X \le s + hx$ для $0 \le x \le 1$, — это площадь, лежащая слева от вертикальной линии $U = x$ (см. рис. 11) и деленная на общую площадь, т. е.

$$\int_0^x \frac{1}{b} f(s+hu)\,du \Big/ \int_0^1 \frac{1}{b} f(s+hu)\,du = \int_s^{s+hx} f(v)\,dv.$$

Следовательно, X имеет нужное распределение.

С подходящими константами a_j, b_j и s_j алгоритм L можно применять к клинообразным плотностям f_{j+15} (см. рис. 9) для $1 \le j \le 15$. Последнее распределение F_{31} нуждается в обработке приблизительно один раз из 370; оно используется тогда, когда получаем результат $X \ge 3$. В упр. 11 показано, что стандартная схема отбраковки может использоваться для этого "хвоста". Рассмотрим процедуру в целом.

Алгоритм M (*Метод прямоугольника-клина-хвоста для нормально распределенных случайных величин*). Для этого алгоритма (рис. 12) используем вспомогательные таблицы $(P_0, \ldots, P_{31})$, $(Q_1, \ldots, Q_{15})$, $(Y_0, \ldots, Y_{31})$, $(Z_0, \ldots, Z_{31})$, $(S_1, \ldots, S_{16})$, $(D_{16}, \ldots, D_{30})$, $(E_{16}, \ldots, E_{30})$, построенные так, как в упр. 10; примеры выбираются из табл. 1. Предполагается, что используется бинарный компьютер (подобные процедуры могут быть построены для десятичных машин).

M1. [Получить U.] Генерируем равномерно распределенное случайное число $U = (.b_0 b_1 b_2 \ldots b_t)_2$. (Здесь b_j — двоичные разряды в бинарном представлении U. Для приемлемой точности t должно быть по крайней мере равно 24.) Присвоить $\psi \leftarrow b_0$. (Позже ψ будет использовано для определения знака результата.)

M2. [Прямоугольник?] Присвоить $j \leftarrow (b_1 b_2 b_3 b_4 b_5)_2$ (двоичное число определяется старшими двоичными разрядами U) и присвоить $f \leftarrow (.b_6 b_7 \ldots b_t)_2$ (дробная часть определяется оставшимися двоичными разрядами). Если $f \geq P_j$, присвоить $X \leftarrow Y_j + fZ_j$ и перейти к шагу M9. Иначе, если $j \leq 15$ (т. е. $b_1 = 0$), присвоить $X \leftarrow S_j + fQ_j$ и перейти к шагу M9. (Это использование метода псевдонимов Уолкера (Walker) (3).)

M3. [Клин или хвост?] (Сейчас $15 \leq j \leq 31$ и каждое определенное значение j появляется с вероятностью p_j.) Если $j = 31$, перейти к шагу M7.

M4. [Получить $U \leq V$.] Генерировать две новые независимые равномерно распределенные случайные величины U и V; если $U > V$, заменить $U \leftrightarrow V$. (Сейчас выполняется алгоритм L.) Присвоить $X \leftarrow S_{j-15} + \frac{1}{5}U$.

M5. [Простой случай?] Если $V \leq D_j$, перейти к шагу M9.

M6. [Попытаемся снова?] Если $V > U + E_j(e^{(S_{j-14}^2 - X^2)/2} - 1)$, вернуться к шагу M4; иначе — перейти к шагу M9. (Вероятность, что этот шаг нужно будет выполнять, мала.)

M7. [Получить мажорирующую случайную величину.] Генерировать две новые независимые равномерно распределенные случайные величины U и V и присвоить $X \leftarrow \sqrt{9 - 2\ln V}$.

M8. [Отбраковать?] Если $UX \geq 3$, возвратиться к шагу M7. (Это будет случаться приблизительно в одном из двенадцати случаев после достижения шага M8.)

M9. [Присоединить знак.] Если $\psi = 1$, присвоить $X \leftarrow -X$. ▮

Этот алгоритм является очень миленьким примером тесного переплетения математической теории с изобретательностью программирования — прекрасная иллюстрация искусства программирования! На выполнение шагов M1, M2 и M9 уходит почти все время; остальные шаги осуществляются намного быстрее. Впервые метод прямоугольника-клина-хвоста опубликовали Дж. Марсалья, Дж. Марсалья, М. Д. Мак-Ларен и Т. А. Брей (см. работы G. Marsaglia, *Annals Math. Stat.* **32** (1961), 894–899; G. Marsaglia, M. D. MacLaren, and T. A. Bray, *CACM* **7** (1964), 4–10). В дальнейшем алгоритм M усовершенствовали Дж. Марсалья, К. Ананханараяпан и Н. Дж. Поль (см. работу G. Marsaglia, K. Ananthanarayanan, and N. J. Paul, *Inf. Proc. Letters* **5** (1976), 27–30).

3) *Метод "чет-нечет"* принадлежит Дж. Э. Форсайту (G. E. Forsythe). Поразительно простая техника генерирования случайных величин с обычной экспоненциальной плотностью вида

$$f(x) = Ce^{-h(x)}\,[a \leq x < b], \tag{20}$$

где

$$0 \leq h(x) \leq 1 \qquad \text{для } a \leq x < b, \tag{21}$$

была открыта Джоном фон Нейманом и Дж. Е. Форсайтом приблизительно в 1950 году. Идея основана на методе отбраковки, описанном ранее. Предполагается, что $g(x)$ — это равномерное распределение на интервале $[a\,..\,b)$. Присвоим $X \leftarrow a + (b - a)U$, где U — равномерно распределенная случайная величина, и примем X с вероятностью $e^{-h(X)}$. Последняя операция может быть осуществлена путем сравнения $e^{-h(X)}$ с V либо $h(X)$ с $-\ln V$, когда V — другая равномерно

Таблица 1

ПРИМЕРЫ ТАБЛИЦ, ИСПОЛЬЗУЕМЫХ С АЛГОРИТМОМ M*

j	P_j	P_{j+16}	Q_j	Y_j	Y_{j+16}	Z_j	Z_{j+16}	S_{j+1}	D_{j+15}	E_{j+15}
0	.000	.067		0.00	0.59	0.20	0.21	0.0		
1	.849	.161	.236	− 0.92	0.96	1.32	0.24	0.2	.505	25.00
2	.970	.236	.206	− 5.86	−0.06	6.66	0.26	0.4	.773	12.50
3	.855	.285	.234	− 0.58	0.12	1.38	0.28	0.6	.876	8.33
4	.994	.308	.201	−33.13	1.31	34.93	0.29	0.8	.939	6.25
5	.995	.304	.201	−39.55	0.31	41.35	0.29	1.0	.986	5.00
6	.933	.280	.214	− 2.57	1.12	2.97	0.28	1.2	.995	4.06
7	.923	.241	.217	− 1.61	0.54	2.61	0.26	1.4	.987	3.37
8	.727	.197	.275	0.67	0.75	0.73	0.25	1.6	.979	2.86
9	1.000	.152	.200	0.00	0.56	0.00	0.24	1.8	.972	2.47
10	.691	.112	.289	0.35	0.17	0.65	0.23	2.0	.966	2.16
11	.454	.079	.440	− 0.17	0.38	0.37	0.22	2.2	.960	1.92
12	.287	.052	.698	0.92	−0.01	0.28	0.21	2.4	.954	1.71
13	.174	.033	1.150	0.36	0.39	0.24	0.21	2.6	.948	1.54
14	.101	.020	1.974	− 0.02	0.20	0.22	0.20	2.8	.942	1.40
15	.057	.086	3.526	0.19	0.78	0.21	0.22	3.0	.936	1.27

*Практически эти данные могут быть приведены с намного большей точностью; в таблице приведено только достаточное количество цифр, чтобы интересующиеся читатели могли более точно проверить собственные алгоритмы вычисления значений.

распределенная случайная величина, но работу можно проделать без применения каких-либо трансцендентных функций следующим интересным способом. Присвоим $V_0 \leftarrow h(X)$ и будем генерировать обычные равномерно распределенные случайные величины V_1, V_2, ..., пока не найдутся $K \geq 1$ с $V_{K-1} < V_K$. Для фиксированных X и k вероятность того, что $h(X) \geq V_1 \geq \cdots \geq V_k$, равна $1/k!$, умноженному на вероятность того, что $\max(V_1, \ldots, V_k) \leq h(X)$, т. е. $h(X)^k/k!$. Следовательно, вероятность того, что $K = k$, равна $h(X)^{k-1}/(k-1)! - h(X)^k/k!$, а вероятность того, что K — нечетное число, равна

$$\sum_{k \text{ нечетное}, k \geq 1} \left(\frac{h(X)^{k-1}}{(k-1)!} - \frac{h(X)^k}{k!} \right) = e^{-h(X)}. \tag{22}$$

Следовательно, мы отбраковываем X и начинаем снова, если K — четное число. Принимаем X как случайную величину с плотностью распределения (20), если K — нечетное число. Обычно мы не генерируем много значений V_j, чтобы определить K, поскольку среднее значение K (при заданном X) равно $\sum_{k \geq 0} \Pr(K > k) = \sum_{k \geq 0} h(X)^k/k! = e^{h(X)} \leq e$.

Несколькими годами позже Форсайт показал, что этот подход приводит к получению эффективного метода вычисления нормальных случайных величин без использования вспомогательных программ для вычисления квадратных корней или логарифмов, как в алгоритмах P и M. Его процедуру с улучшенным выбором интервалов $[a\,..\,b)$, осуществленную И. Г. Аренсом (J. H. Ahrens) и У. Дитером (U. Dieter), можно представить в виде алгоритма.

Алгоритм F (*Метод "чет-нечет" для нормальных случайных величин*). Этот алгоритм генерирует нормальные случайные величины на бинарном компьютере; предполагается, что имеется приблизительно $t+1$ двоичных разрядов точности. Он нуждается в таблице значений $d_j = a_j - a_{j-1}$ для $1 \leq j \leq t+1$, где a_j определяется

отношением

$$\sqrt{\frac{2}{\pi}}\int_{a_j}^{\infty} e^{-x^2/2}\,dx = \frac{1}{2^j}. \tag{23}$$

F1. [Получить U.] Генерировать равномерно распределенное число $U=(.b_0b_1\ldots b_t)_2$, где b_0, b_1, $\ldots$, b_t означают двоичные разряды в бинарной записи. Присвоить $\psi \leftarrow b_0$, $j \leftarrow 1$ и $a \leftarrow 0$.

F2. [Найти первый нуль b_j.] Если $b_j = 1$, присвоить $a \leftarrow a+d_j$, $j \leftarrow j+1$ и повторить этот шаг. (Если $j = t+1$, трактовать b_j как нуль.)

F3. [Генерировать X.] (Сейчас $a = a_{j-1}$, и текущее значение j появится с вероятностью $\approx 2^{-j}$. Будем генерировать X в интервале $[a_{j-1}\,..\,a_j)$, используя метод отбраковки, описанный выше, с $h(x) = x^2/2 - a^2/2 = y^2/2 + ay$, где $y = x - a$. В упр. 12 доказывается, что $h(x) \le 1$, как требуется в (21).) Присвоить $Y \leftarrow d_j$, умноженное на $(.b_{j+1}\ldots b_t)_2$ и $V \leftarrow (\frac{1}{2}Y + a)Y$. (Так как среднее значение j равно 2, обычно достаточно старших двоичных разрядов в $(.b_{j+1}\ldots b_t)_2$ для обеспечения приличной точности. Вычисления без труда выполняются арифметикой с фиксированной точкой.)

F4. [Отбраковать?] Генерируем равномерно распределенную случайную величину U. Если $V < U$, перейти к шагу F5. Иначе — присвоить V новую равномерно распределенную случайную величину. Если сейчас $U < V$ (т. е. если K четное, как обсуждалось выше), возвратиться к шагу F3, иначе — повторить шаг F4.

F5. [Выход из программы по X.] Присвоить $X \leftarrow a + Y$. Если $\psi = 1$, присвоить $X \leftarrow -X$. ▮

Значения d_j для $1 \le j \le 47$ появились в статье Аренса и Дитера (Ahrens and Dieter, *Math. Comp.* **27** (1973), 927–937). В ней обсуждается усовершенствованный алгоритм, который повышает скорость его выполнения за счет использования больших таблиц. Алгоритм F привлекателен тем, что работает почти с такой же скоростью, как алгоритм M, и его легче выполнить. Среднее число равномерно распределенных случайных величин на каждую нормальную случайную величину равно 2.53947. Р. П. Брент (R. P. Brent, *CACM* **17** (1974), 704–705) показал, как можно сократить это число до 1.37446 с помощью двух вычитаний и одного деления на каждую хранимую равномерно распределенную случайную величину.

4) *Отношение равномерно распределенных случайных величин.* Существует еще один хороший метод генерирования нормальных случайных величин, открытый в 1976 году А. Дж. Киндерманом (A. J. Kinderman) и Дж. Ф. Монаханом (J. F. Monahan). Его суть — генерирование случайной точки (U, V) в области, определенной как

$$0 < u \le 1, \qquad -2u\sqrt{\ln(1/u)} \le v \le 2u\sqrt{\ln(1/u)}, \tag{24}$$

и получение отношения $X \leftarrow V/U$. Заштрихованная площадь на рис. 13 — это магическая область неравенства (24), осуществляющая эту работу. Прежде чем изучать теорию, приведем алгоритм, эффективность и простота которого очевидны.

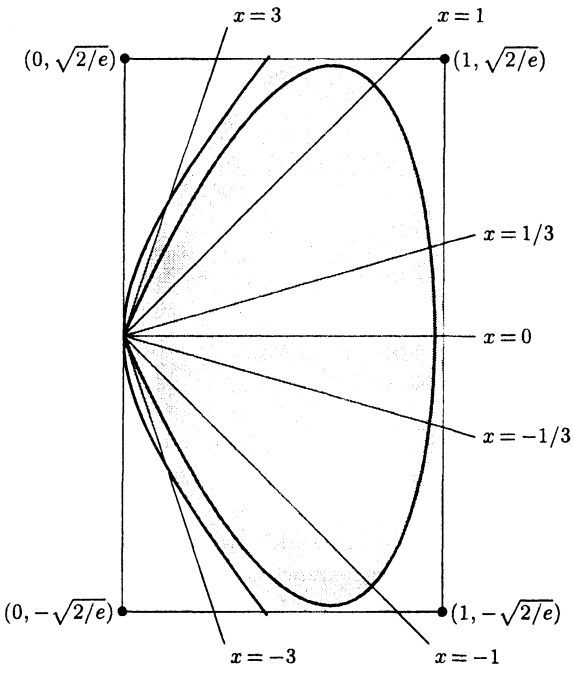


Рис. 13. Область "принятия" в методе отношения равномерных случайных величин для нормально распределенных случайных величин. Длины линий с отношением координат x распределены нормально.

Алгоритм R (*Метод отношений для нормальных случайных величин*). Этот алгоритм генерирует нормальные случайные величины X.

R1. [Получить U, V.] Генерировать две независимые равномерно распределенные случайные величины U и V, где $U \neq 0$, и присвоить $X \leftarrow \sqrt{8/e}\,(V - \frac{1}{2})\,/U$. (Сейчас X равно отношению координат $(U, \sqrt{8/e}\,(V - \frac{1}{2}))$ случайной точки в прямоугольнике, содержащем заштрихованную область на рис. 13. Принимаем X, если соответствующая точка в самом деле находится в заштрихованной области, иначе — начинаем сначала.)

R2. [Необязательная проверка верхней грани.] Если $X^2 \leq 5 - 4e^{1/4}U$, на выходе — X и завершение алгоритма. (Этот шаг можно опустить, если пожелаете; он так или иначе проверяет, находятся ли избранные точки во внешней области рис. 13, делая излишним вычисление логарифма.)

R3. [Необязательная проверка нижней грани.] Если $X^2 \geq 4e^{-1.35}/U + 1.4$, возвратиться к шагу R1. (Этот шаг также может быть опущен; он так или иначе проверяет, находятся ли выбранные точки за внешней областью рис. 13, делая ненужным вычисление логарифма).

R4. [Окончательная проверка.] Если $X^2 \leq -4 \ln U$, выход X и завершение алгоритма; иначе — возврат к шагу R1. ▮

В упр. 20 и 21 представлен временной анализ; анализируются четыре различных алгоритма, так как шаги R2 и R3 при желании могут быть включены или пропущены. В следующей таблице показано, сколько в среднем времени уходит на

выполнение каждого шага в зависимости от применения необязательной проверки.

Шаг	Ни одного	Только R2	Только R3	Оба
R1	1.369	1.369	1.369	1.369
R2	0	1.369	0	1.369
R3	0	0	1.369	0.467
R4	1.369	0.467	1.134	0.232

(25)

Таким образом, стоит отбросить необязательную проверку, если есть быстрые операции логарифмирования, но если подпрограмма вычисления логарифмов выполняется довольно медленно, то проверку стоит включить.

Но зачем делать эту работу? По одной единственной причине — чтобы вычислить вероятность того, что $X \le x$, которая оказывается точным значением (10). Но такое вычисление вряд ли будет очень простым, если не удастся найти хороший прием. Во всяком случае, в первую очередь, нужно понять, как может быть открыт такой алгоритм. Киндерман (Kinderman) и Монахам (Monahan) открыли его, разрабатывая следующую теорию, которая может использоваться с любой хорошо себя ведущей плотностью $f(x)$ [см. *ACM Trans. Math. Software* **3** (1977), 257–260].

Предположим, что точка (U, V) равномерно генерируется в области (u, v)-плоскости, определенной неравенствами

$$u > 0, \qquad u^2 \le g(v/u) \tag{26}$$

для некоторой неотрицательной интегрируемой функции g. Если присвоить $X \leftarrow V/U$, то вероятность того, что $X \le x$, можно вычислить путем интегрирования по $du\,dv$ по всей области, определенной двумя соотношениями в (26) и добавочным условием $v/u \le x$ с последующим делением на этот же интеграл, но без дополнительного условия. Допустим, $v = tu$, так что $dv = u\,dt$. Тогда интеграл имеет вид

$$\int_{-\infty}^{x} dt \int_{0}^{\sqrt{g(t)}} u\,du = \frac{1}{2}\int_{-\infty}^{x} g(t)\,dt.$$

Следовательно, вероятность, что $X \le x$, равна

$$\int_{-\infty}^{x} g(t)\,dt \Big/ \int_{-\infty}^{+\infty} g(t)\,dt. \tag{27}$$

Нормальное распределение возникает, когда $g(t) = e^{-t^2/2}$, а условие $u^2 \le g(v/u)$ упрощается в этом случае до $(v/u)^2 \le -4\ln u$. Легко видеть, что множество всех (u, v), удовлетворяющих этому соотношению, полностью содержится в прямоугольнике на рис. 13.

Грани шагов R2 и R3 определяют внутреннюю и внешнюю области простыми граничными уравнениями. Хорошо известное неравенство

$$e^x \ge 1 + x,$$

справедливое для всех действительных чисел x, можно использовать, чтобы показать, что неравенства

$$1 + \ln c - cu \;\le\; -\ln u \;\le\; 1/(cu) - 1 + \ln c \tag{28}$$

выполняются для любой константы $c > 0$. В упр. 21 доказывается, что $c = e^{1/4}$ является наилучшей возможной константой для использования на шаге R2. Более сложная ситуация складывается на шаге R3, и в этом случае, кажется, не существует простого выражения для оптимального c, но вычислительные эксперименты показывают, что наилучшее значение c для шага R3 приблизительно равно $e^{1.35}$. Аппроксимирующей кривой (28) является касательная к истинной границе, когда $u = 1/c$.

Существует возможность получения быстрого метода путем разделения области на подобласти, с большинством из которых иметь дело намного быстрее. Конечно, подразумевается, что нужны дополнительные таблицы, как в алгоритмах M и F. Интересная альтернатива, требующая меньшего числа вспомогательных таблиц, предложена Аренсом и Дитером в *CACM* **31** (1988), 1330–1337.

5) *Нормальная случайная величина из нормальной случайной величины.* В упр. 31 обсуждается интересный подход, который позволяет экономить время работы, так как сразу использует нормальные случайные величины вместо равномерно распределенных случайных величин. Этот метод, введенный в употребление в 1996 году К. С. Уэллесом (C. S. Wallace), в настоящее время не нашел достаточного теоретического обоснования, но удовлетворяет многим эмпирическим критериям.

6) *Преобразования нормального распределения.* До сих пор рассматривалось нормальное распределение со средним, равным нулю, и среднеквадратичным отклонением, равным единице. Если X имеет такое распределение, то

$$Y = \mu + \sigma X \tag{29}$$

имеет нормальное распределение со средним, равным μ, и среднеквадратичным отклонением, равным σ. Кроме того, если X_1 и X_2 — независимые нормальные случайные величины со средним, равным нулю, среднеквадратичным отклонением, равным единице, и

$$Y_1 = \mu_1 + \sigma_1 X_1, \qquad Y_2 = \mu_2 + \sigma_2\big(\rho X_1 + \sqrt{1-\rho^2}\, X_2\big), \tag{30}$$

то Y_1 и Y_2 — *зависимые* нормально распределенные случайные величины со средними, равными μ_1, μ_2, среднеквадратичными отклонениями, равными σ_1, σ_2, и коэффициентом корреляции ρ. (Для генерирования n переменных обратитесь к упр. 13.)

D. Показательное распределение. После равномерного и нормального распределений следующим наиболее важным распределением случайной величины является *показательное распределение.* Такие распределения появляются в ситуациях "время поступления". Например, если радиоактивное вещество излучает альфа-частицы так, что одна частица в среднем испускается каждые μ секунд, то время между двумя последовательными испусканиями имеет показательное распределение со средним, равным μ. Это распределение задается формулой

$$F(x) = 1 - e^{-x/\mu}, \qquad x \geq 0. \tag{31}$$

1) *Метод логарифма.* Очевидно, если $y = F(x) = 1 - e^{-x/\mu}$, то $x = F^{[-1]}(y) = -\mu \ln(1-y)$. Следовательно, $-\mu \ln(1-U)$ имеет экспоненциальное распределение

согласно (7). Поскольку $1 - U$ равномерно распределено, когда U имеет такое же распределение, можно сделать вывод, что

$$X = -\mu \ln U \tag{32}$$

имеет экспоненциальное распределение со средним, равным μ. (Случай, когда $U = 0$, должен трактоваться особо; его можно заменять любым подходящим значением ϵ, так как вероятность возникновения этого случая крайне мала.)

2) *Метод случайной минимизации.* Как было показано в алгоритме F, существует простой и быстрый способ альтернативного вычисления логарифма равномерной случайной величины. Следующий особенно эффективный подход разработали Дж. Марсалья, М. Сибая (M. Sibuya) и И. Г. Аренс (J. H. Ahrens) [см. *CACM* **15** (1972), 876–877].

Алгоритм S (*Экспоненциальное распределение со средним, равным* μ). Этот алгоритм вырабатывает экспоненциальные случайные величины на бинарном компьютере, используя равномерно распределенные случайные величины с точностью до $(t + 1)$ двоичных разрядов. Постоянные

$$Q[k] = \frac{\ln 2}{1!} + \frac{(\ln 2)^2}{2!} + \cdots + \frac{(\ln 2)^k}{k!}, \qquad k \geq 1, \tag{33}$$

могут вычисляться заранее до тех пор, пока они не превысят следующее значение: $Q[k] > 1 - 2^{-t}$.

S1. [Получить U и сдвинуть.] Генерировать $(t+1)$ двоичных разрядов равномерной случайной двоичной дроби $U = (.b_0 b_1 b_2 \ldots b_t)_2$; определить местоположение первого нулевого двоичного разряда b_j и избавиться от старших $j + 1$ двоичных разрядов с помощью присвоения $U \leftarrow (.b_{j+1} \ldots b_t)_2$. (Как и в алгоритме F, среднее число отбрасываемых двоичных разрядов равно 2.)

S2. [Немедленное принятие?] Если $U < \ln 2$, присвоить $X \leftarrow \mu(j \ln 2 + U)$ и завершить алгоритм. (Отметим, что $Q[1] = \ln 2$.)

S3. [Минимизация.] Найти наименьшее $k \geq 2$, такое, что $U < Q[k]$. Генерировать k новых равномерно распределенных случайных величин $U_1, \ldots, U_k$ и присвоить $V \leftarrow \min(U_1, \ldots, U_k)$.

S4. [Получение ответа.] Присвоить $X \leftarrow \mu(j + V) \ln 2$. ▮

Можно также использовать альтернативный метод генерирования случайных величин с показательным распределением (например, метод отношений равномерных случайных величин, как в алгоритме R).

E. Другие непрерывные распределения. Кратко рассмотрим, как обращаться с другими распределениями, которые достаточно часто возникают на практике.

1) *Гамма-распределение* порядка $a > 0$ определяется как

$$F(x) = \frac{1}{\Gamma(a)} \int_0^x t^{a-1} e^{-t}\, dt, \qquad x \geq 0. \tag{34}$$

При $a = 1$ оно является показательным распределением со средним, равным 1; при $a = \frac{1}{2}$ — это распределение случайной величины $\frac{1}{2}Z^2$, где Z — нормально

распределенная случайная величина (среднее равно 0, дисперсия — 1). Если X и Y — независимые случайные величины, имеющие гамма-распределение порядка a и b соответственно, то $X+Y$ имеет гамма-распределение порядка $a+b$. Так, например, сумма k независимых случайных величин, имеющих показательное распределение со средним, равным 1, имеет гамма-распределение порядка k. Если метод логарифма (32) использовался для генерирования таких случайных величин, имеющих показательное распределение, то здесь необходимо вычислить только один логарифм: $X \leftarrow -\ln(U_1 \ldots U_k)$, где U_1, ..., U_k — равномерно распределенные случайные величины, не равные нулю. Таким методом можно генерировать все случайные величины с гамма-распределением порядка a, где a — целое. Для завершения картины соответствующий метод для $0 < a < 1$ приведен в упр. 16.

Простой метод логарифма также очень медленный, когда a большое, поскольку он требует $\lfloor a \rfloor$ равномерно распределенных случайных величин. Более того, существует реальный риск, что произведение $U_1 \ldots U_{\lfloor a \rfloor}$ приведет к переполнению (потере плавающей точки). Для большого a следующий алгоритм, предложенный И. Г. Аренсом, является достаточно эффективным и легко записывается в терминах стандартных программ. [См. *Ann. Inst. Stat. Math.* **13** (1962), 231–237.]

Алгоритм A (*Гамма-распределение порядка* $a > 1$).

A1. [Генерирование кандидата.] Присвоить $Y \leftarrow \tan(\pi U)$, где U — равномерно распределенная случайная величина, и присвоить $X \leftarrow \sqrt{2a-1}\,Y + a - 1$. (Вместо $\tan(\pi U)$ можно использовать метод полярных координат, вычисляя отношение V_2/V_1, как на шаге P4 алгоритма P.)

A2. [Принимать?] Если $X \le 0$, возврат к шагу A1. Иначе — генерировать равномерно распределенную случайную величину V и возвратиться к шагу A1, если $V > (1+Y^2)\exp\bigl((a-1)\ln(X/(a-1)) - \sqrt{2a-1}\,Y\bigr)$. Иначе — принять X. ▮

Среднее время выполнения шага A1 < 1.902, когда $a \ge 3$.

Существует также заманчивый подход для больших a, основанный на таком замечательном факте, что гамма-распределение случайных величин приблизительно равно aX^3, когда X — нормально распределенная величина со средним значением $1 - 1/(9a)$ и среднеквадратичным отклонением $1/\sqrt{9a}$ [см. работы Э. Б. Уилсон (E. B. Wilson) и М. М. Гилферти (M. M. Hilferty), *Proc. Nat. Acad. Sci.* **17** (1931), 684–688, а также Дж. Марсалья (G. Marsaglia, *Computers and Math.* **3** (1977), 321–325)]*.

В некоторой степени усложненный, но значительно более быстрый алгоритм, который генерирует случайные величины, имеющие гамма-распределение, за приблизительно вдвое большее время, чем случайные величины, имеющие нормальное распределение, приведен в статье И. Г. Аренса и У. Дитера, *CACM* **25** (1982), 47–54, в которой содержится полезное описание принципов, используемых при построении алгоритма.

2) *Бета-распределение* с положительными параметрами a и b определяется как

$$F(x) = \frac{\Gamma(a+b)}{\Gamma(a)\,\Gamma(b)} \int_0^x t^{a-1}(1-t)^{b-1}\,dt, \qquad 0 \le x \le 1. \tag{35}$$

---

* Заменить "$+(3a-1)$" на "$-(3a-1)$" на шаге 3 алгоритма на с. 323.

Пусть X_1 и X_2 — независимые случайные величины, имеющие гамма-распределение соответственно порядка a и b. Присвоить $X \leftarrow X_1/(X_1+X_2)$. Другим полезным для малых a и b методом является использование присвоений

$$Y_1 \leftarrow U_1^{1/a} \qquad \text{и} \qquad Y_2 \leftarrow U_2^{1/b},$$

повторяемых до тех пор, пока не получится $Y_1+Y_2 \le 1$; тогда $X \leftarrow Y_1/(Y_1+Y_2)$. [См. работу М. Д. Йонка (M. D. Jöhnk, *Metrika* **8** (1964), 5–15).] Еще одним подходом, если a и b — целые числа, которые не настолько велики, является присвоение X b-й самой большой по величине из $a+b-1$ независимых равномерно распределенных случайных величин (см. упр. 9 в начале гл. 5). [См. также более прямой метод, описанный Р. Ч. С. Ченгом (R. C. H. Cheng, *CACM* **21** (1978), 317–322).]

3) *χ^2-распределение* с ν степенями свободы (3.3.1–(22)) можно получить, если положить, что $X \leftarrow 2Y$, где Y — случайная величина, имеющая гамма-распределение порядка $\nu/2$.

4) *F-распределение* (распределение отношения дисперсий) со степенями свободы ν_1 и ν_2 определено следующим образом:

$$F(x) = \frac{\nu_1^{\nu_1/2}\,\nu_2^{\nu_2/2}\,\Gamma\big((\nu_1+\nu_2)/2\big)}{\Gamma(\nu_1/2)\,\Gamma(\nu_2/2)} \int_0^x t^{\nu_1/2-1}(\nu_2+\nu_1 t)^{-\nu_1/2-\nu_2/2}\,dt, \tag{36}$$

где $x \ge 0$. Пусть Y_1 и Y_2 — независимые случайные величины, имеющие χ^2-распределение со степенями свободы ν_1 и ν_2 соответственно. Положим $X \leftarrow Y_1\nu_2/Y_2\nu_1$ или $X \leftarrow \nu_2 Y/\nu_1(1-Y)$, где Y_k, $k=1,2$, — случайные величины, имеющие бета-распределение с параметрами $\nu_1/2$ и $\nu_2/2$.

5) *t-распределение* с ν степенями свободы определено следующим образом:

$$F(x) = \frac{\Gamma\big((\nu+1)/2\big)}{\sqrt{\pi\nu}\,\Gamma(\nu/2)} \int_{-\infty}^x (1+t^2/\nu)^{-(\nu+1)/2}\,dt. \tag{37}$$

Пусть Y_1 — нормально распределенная случайная величина (среднее — 0, дисперсия — 1) и Y_2 — случайная независимая от Y_1 величина, имеющая χ^2-распределение с ν степенями свободы. Положим $X \leftarrow Y_1/\sqrt{Y_2/\nu}$. С другой стороны, когда $\nu > 2$, можно поступить следующим образом. Пусть Y_1 — нормально распределенная случайная величина (среднее — 0, дисперсия — 1) и Y_2 — независимая от нее случайная величина, имеющая показательное распределение со средним $2/(\nu-2)$. Положим $Z \leftarrow Y_1^2/(\nu-2)$ и отбросим (Y_1, Y_2), если $e^{-Y_2-Z} \ge 1-Z$. Иначе положим

$$X \leftarrow Y_1/\sqrt{(1-2/\nu)(1-Z)}.$$

Последний метод предложен Джорджем Марсальей, *Math. Comp.* **34** (1980), 235–236. [См. также работу А. Дж. Киндермана, Дж. Ф. Монахама и Дж. Дж. Рамаджа (A. J. Kinderman, J. F. Monahan, and J. G. Ramage, *Math. Comp.* **31** (1977), 1009–1018).]

6) *Случайные точки на n-мерной сфере единичного радиуса.* Пусть X_1, X_2, ..., X_n — независимые нормально распределенные случайные величины (среднее — 0,

дисперсия — 1); требуемая точка выбирается на единичной сфере следующим образом:

$$(X_1/r, X_2/r, \ldots, X_n/r), \qquad \text{где } r = \sqrt{X_1^2 + X_2^2 + \cdots + X_n^2}. \tag{38}$$

Если мы вычисляем X_k, используя метод полярных координат (алгоритм P), необходимо каждый раз вычислять две независимые случайные величины X_k, $k = 1, 2$, и в обозначениях алгоритма выполняется равенство $X_1^2 + X_2^2 = -2 \ln S$. Так можно сэкономить немного времени, требуемого для вычисления r. Справедливость (38) вытекает из того факта, что функция распределения точки $(X_1, \ldots, X_n)$ имеет плотность, зависящую только от расстояния от начала координат, поэтому при проецировании на единичную сферу она имеет равномерное распределение. Данный метод впервые был предложен Дж. В. Брауном (G. W. Brown, *Modern Mathematics for the Engineer*, First series, edited by E. F. Beckenbach (Бекенбах Э. Ф.), New York: McGraw-Hill, 1956, 302). Чтобы получить случайную точку *внутри* n-мерной сферы, Р. П. Брент (R. P. Brent) предложил взять точку на поверхности этой сферы и умножить ее координаты на $U^{1/n}$.

Для размерности 3 можно использовать значительно более простой метод, так как каждая координата этой точки равномерно распределена между -1 и 1. Если найти V_1, V_2 и S, используя шаги P1–P3 алгоритма P, искомая точка на поверхности шара будет иметь вид $(\alpha V_1, \alpha V_2, 2S - 1)$, где $\alpha = 2\sqrt{1 - S}$. [Robert E. Knop, *CACM* **13** (1970), 326.]

F. Важные целочисленные распределения. Вероятностные распределения на множестве целых чисел (случайные величины, принимающие только целые значения. — *Прим. ред.*) могут быть получены, в сущности, с помощью технических приемов, описанных в начале раздела, но некоторые из этих распределений настолько важны, что заслуживают специального упоминания.

1) *Геометрическое распределение.* Если некоторое событие происходит с вероятностью p, то число N независимых испытаний, проведенных до появления события (или до момента, когда событие происходит впервые), имеет геометрическое распределение, т. е. $N = 1$ с вероятностью p, $N = 2$ с вероятностью $(1 - p)p$, ..., $N = n$ с вероятностью $(1-p)^{n-1}p$. Это, по существу, ситуация, которая уже рассматривалась в разделе 3.3.2. Данное распределение непосредственно связано с числом циклов алгоритмов из настоящего раздела, как и циклы на шагах P1–P3 метода полярных координат.

Удобный метод генерирования случайной величины с таким распределением состоит в следующем:

$$N \leftarrow \lceil \ln U / \ln(1 - p) \rceil. \tag{39}$$

Чтобы проверить эту формулу, заметим, что $\lceil \ln U / \ln(1 - p) \rceil = n$ тогда и только тогда, когда $n - 1 < \ln U / \ln(1 - p) \leq n$, т. е. $(1 - p)^{n-1} > U \geq (1 - p)^n$ и это происходит с требуемой вероятностью $(1 - p)^{n-1}p$. Величину $\ln U$ можно также заменить величиной $-Y$, где Y имеет показательное распределение со средним 1.

Частный случай, когда $p = \frac{1}{2}$, совсем просто реализуется на бинарном компьютере, так как формула (39) сводится к присвоению $N \leftarrow \lceil -\lg U \rceil$, т. е. N становится на единицу больше числа старших нулевых разрядов в двоичном представлении числа U.

2) *Биномиальное распределение* (t,p). Если некоторое событие происходит с вероятностью p и проводится t независимых испытаний, то общее число N появлений этого события равно n с вероятностью $\binom{t}{n} p^n(1-p)^{t-n}$ (см. раздел 1.2.10). Другими словами, если генерируется U_1, ..., U_t, достаточно подсчитать, сколько из них не превосходят p. Для малых t с помощью этого метода легко можно получить точное значение N.

Для больших t можно генерировать случайную величину X, имеющую бета-распределение с целыми параметрами a и b, где $a+b-1=t$. При этом эффективно получаем b-й наибольший из t элементов, не беспокоясь о генерировании остальных элементов. Сейчас, если $X \geq p$, положим $N \leftarrow N_1$, где N_1 имеет биномиальное распределение $(a-1, p/X)$, так как оно показывает, сколько из $a-1$ случайных величин в области $[0..X)$ меньше, чем p. Если $X < p$, положим $N \leftarrow a+N_1$, где N_1 имеет биномиальное распределение $\big(b-1, (p-X)/(1-X)\big)$, так как N_1 показывает нам, сколько из $b-1$ случайных чисел в области $[X..1)$ меньше p. Выбирая $a = 1 + \lfloor t/2 \rfloor$, параметр t можно свести к разумной величине после примерно $\lg t$ преобразований такого рода. (Этот подход предложен И. Г. Аренсом, который также предложил альтернативный метод для значений параметра t средней величины; см. упр. 27.)

3) *Пуассоновское распределение* со средним μ. Пуассоновское распределение соотносится с показательным распределением, как биномиальное распределение с геометрическим: если некоторое событие может произойти в любой момент времени, то пуассоновское распределение — это распределение случайной величины, которая равна числу событий, происшедших в единицу времени (при некоторых других ограничениях. — *Прим. ред.*). Например, число альфа-частиц, которые испускаются радиоактивным веществом за одну секунду, имеет распределение Пуассона.

В соответствии с этим принципом можно получить пуассоновскую случайную величину N, генерируя независимые показательные случайные величины $X_1, X_2, \ldots$ со средним $1/\mu$ и останавливаясь, как только будет получено $X_1 + \cdots + X_m \geq 1$, где $N \leftarrow m-1$. Вероятность того, что $X_1 + \cdots + X_m \geq 1$, равна вероятности, что случайная величина, имеющая гамма-распределение порядка m, будет $\geq \mu$, а это равно $\int_\mu^\infty t^{m-1}e^{-t}\,dt/(m-1)!$. Следовательно, вероятность, что $N=n$, равна

$$\frac{1}{n!}\int_\mu^\infty t^n e^{-t}\,dt - \frac{1}{(n-1)!}\int_\mu^\infty t^{n-1}e^{-t}\,dt = e^{-\mu}\frac{\mu^n}{n!}, \qquad n \geq 0. \tag{40}$$

При генерировании показательной случайной величины методом логарифма, описанным выше, нужно остановиться, когда $-(\ln U_1 + \cdots + \ln U_m)/\mu \geq 1$. Упрощая это выражение, получим, что требуемую пуассоновскую величину можно получить, вычислив $e^{-\mu}$ до определенной точности и генерируя одну или более равномерно распределенных случайных величин U_1, U_2, ... до тех пор, пока их произведение не будет удовлетворять неравенству $U_1 \ldots U_m \leq e^{-\mu}$. Окончательно полагаем, что $N \leftarrow m-1$. В среднем нужно генерировать $\mu+1$ равномерно распределенную величину, поэтому рассматриваемый подход очень полезен, когда μ не слишком велико.

Когда μ большое, можно получить метод порядка $\log\mu$, используя сведения о том, как соотносятся гамма- и биномиальное распределения высокого порядка.

Сначала генерируем случайную величину X, имеющую гамма-распределение порядка $m = \lfloor \alpha\mu \rfloor$, где α — подходящая постоянная. (Так как X эквивалентно $-\ln(U_1 \dots U_m)$, по существу, производим m шагов предыдущего метода.) Если $X < \mu$, полагаем $N \leftarrow m + N_1$, где N_1 — пуассоновская случайная величина со средним $\mu - X$, а если $X \geq \mu$, то полагаем $N \leftarrow N_1$, где N_1 имеет биномиальное распределение $(m-1, \mu/X)$. Этот метод предложен И. Г. Аренсом и У. Дитером, которые предположили, что $\frac{7}{8}$ — хорошее значение для α.

Справедливость сформулированного утверждения, когда $X \geq \mu$, является следствием такого важного принципа: "Пусть $X_1, \dots, X_m$ — независимые показательные случайные величины с одинаковым средним и пусть $S_j = X_1 + \cdots + X_j$ и $V_j = S_j/S_m$ для $1 \leq j \leq m$. Тогда распределение $V_1, V_2, \dots, V_{m-1}$ такое же, как распределение $m-1$ независимых случайных величин, расположенных в порядке возрастания". Чтобы формально доказать этот принцип, подсчитаем вероятность того, что $V_1 \leq v_1, \dots, V_{m-1} \leq v_{m-1}$, если $S_m = s$ для произвольных значений $0 \leq v_1 \leq \cdots \leq v_{m-1} \leq 1$. Пусть $f(v_1, v_2, \dots, v_{m-1})$ — $(m-1)$-кратный интеграл

$$\int_0^{v_1 s} \mu e^{-t_1/\mu}\, dt_1 \int_0^{v_2 s - t_1} \mu e^{-t_2/\mu}\, dt_2 \dots \times \int_0^{v_{m-1}s - t_1 - \cdots - t_{m-2}} \mu e^{-t_{m-1}/\mu}\, dt_{m-1} \cdot \mu e^{-(s - t_1 - \cdots - t_{m-1})/\mu};$$

тогда

$$\frac{f(v_1, v_2, \dots, v_{m-1})}{f(1, 1, \dots, 1)} = \frac{\int_0^{v_1} du_1 \int_{u_1}^{v_2} du_2 \dots \int_{u_{m-2}}^{v_{m-1}} du_{m-1}}{\int_0^1 du_1 \int_{u_1}^1 du_2 \dots \int_{u_{m-2}}^1 du_{m-1}},$$

если произвести замену переменных $t_1 = su_1$, $t_1 + t_2 = su_2, \dots, t_1 + \cdots + t_{m-1} = su_{m-1}$. Последнее отношение соответствует вероятности того, что равномерно распределенные случайные величины $U_1, \dots, U_{m-1}$ удовлетворяют неравенствам $U_1 \leq v_1, \dots, U_{m-1} \leq v_{m-1}$, если задано, что они также удовлетворяют неравенствам $U_1 \leq \cdots \leq U_{m-1}$.

В некоторой степени более эффективным, но более сложным будет метод для биномиальной и пуассоновской случайных величин, предложенный в упр. 22.

G. Для дальнейшего чтения. Факсимиле письма фон Неймана (von Neumann), датированного 21 мая 1947 года, в котором метод отбраковки впервые увидел свет, появилось в *Stanisław Ulam 1909–1984*, в специальном выпуске *Los Alamos Science* (Los Alamos National Lab., 1987), 135–136. В книге *Non-Uniform Random Variate Generation* Л. Девроя (L. Devroye) (Springer, 1986) обсуждается намного больше алгоритмов генерирования случайных величин с неравномерным распределением; там же подробно рассматривается эффективность каждого метода для типичных компьютеров.

В. Херманн и Г. Дерфлингер [W. Hörmann and G. Derflinger, *ACM Trans. Math. Software* **19** (1993), 489–495] указали, что может быть опасно использовать метод отбраковки для линейных конгруэнтных генераторов с малыми множителями $a \approx \sqrt{m}$.

С теоретической точки зрения интересно рассмотреть *оптимальный* метод генерирования случайных величин с заданным распределением. Оптимальность здесь означает, что этот метод генерирует требуемую случайную величину, используя минимальное возможное число случайных разрядов. Начала этой теории можно

найти в книге Д. Э. Кнута и Э. Ч. Яо (D. E. Knuth and A. C. Yao, *Algorithms and Complexity*, edited by J. F. Traub (New York: Academic Press, 1976), 357–428).

Упр. 16 рекомендуется как обзор различных методов, приведенных в этом разделе.

УПРАЖНЕНИЯ

1. [*10*] Пусть α и β — действительные числа, $\alpha < \beta$. Как можно генерировать случайные действительные числа, равномерно распределенные между α и β?

2. [*M16*] Пусть mU — случайное целое число, принимающее значения между 0 и $m-1$ с одинаковыми вероятностями. Чему *точно* равна вероятность того, что $\lfloor kU \rfloor = r$, если $0 \le r < k$? Сравните с требуемой вероятностью $1/k$.

▶ **3.** [*14*] Как можно рассмотреть U в качестве целого числа и *разделить* его на k, чтобы получить случайное целое число между 0 и $k-1$, вместо того чтобы выполнять умножение, как предложено в разделе. Тогда (1) было бы заменено на

```
ENTA 0;  LDX U;  DIV K
```

с результатом в регистре X. Хороший ли это метод?

4. [*M20*] Докажите два соотношения в (8).

5. [*21*] Предложите эффективный метод генерирования случайной величины с функцией распределения $F(x) = px + qx^2 + rx^3$ ($0 \le x \le 1$. — *Прим. ред.*), где $p \ge 0$, $q \ge 0$, $r \ge 0$ и $p + q + r = 1$.

6. [*HM21*] Величина X получена следующим образом.

Шаг 1. Генерировать две независимые равномерно распределенные случайные величины U и V.

Шаг 2. Если $U^2 + V^2 \ge 1$, возврат к шагу 1; иначе — присвоить $X \leftarrow U$.

Какова функция распределения X? Сколько времени выполняется шаг 1? (Найдите среднее и среднеквадратичное отклонения).

7. [*20*] (А. Дж. Уолкер (A. J. Walker).) Предположим, что имеется набор кубиков k различных цветов, скажем n_j кубиков цвета C_j при $1 \le j \le k$, и k коробок $\{B_1, \ldots, B_k\}$, в каждую из которых можно поместить n кубиков. Кроме того, $n_1 + \cdots + n_k = kn$, так что кубики будут полностью заполнять коробки. Докажите (конструктивно), что всегда можно разместить кубики в коробках так, чтобы в каждой из них содержались кубики самое большее двух различных цветов. Действительно, можно устроить так, чтобы всякий раз в коробке B_j содержалось два цвета, один из которых — цвет C_j. Покажите, как использовать это утверждение, чтобы подсчитать P и Y, требуемые в (3), при заданном вероятностном распределении $(p_1, \ldots, p_k)$.

8. [*M15*] Покажите, что операцию (3) можно заменить операцией

$$\text{если} \quad U < P_K, \quad \text{то} \quad X \leftarrow x_{K+1}; \quad \text{иначе} \quad X \leftarrow Y_K.$$

(Таким образом, вместо V используется истинное значение U.) Будет ли это более удобным или подходящим для изменения P_0, P_1, ..., P_{k-1}.

9. [*HM10*] Почему кривая $f(x)$ на рис. 9 выпукла для Выпуклая кривая Вогнутая кривая $x < 1$ и вогнута для $x > 1$?

▶ **10.** [*HM24*] Объясните, как вычислить вспомогательные постоянные P_j, Q_j, Y_j, Z_j, S_j, D_j, E_j, чтобы алгоритм M давал ответ с правильным распределением.

▸ **11.** [*HM27*] Докажите, что шаги M7 и M8 алгоритма M порождают случайную величину с хвостом, близким к нормальному распределению, т. е. вероятность того, что $X \le x$ должна быть точно равна

$$\int_3^x e^{-t^2/2}\,dt \Big/ \int_3^\infty e^{-t^2/2}\,dt, \qquad x \ge 3.$$

[*Указание.* Покажите, что это частный случай метода отбраковки с $g(t) = Cte^{-t^2/2}$ для некоторого C.]

12. [*HM23*] (Р. П. Брент (R. P. Brent).) Докажите, что числа a_j, определенные в (23), удовлетворяют соотношению

$$a_j^2 - a_{j-1}^2 < 2\ln 2 \qquad \text{для всех } j \ge 1.$$

[*Указание.* Если $f(x) = e^{x^2/2}\int_x^\infty e^{-t^2/2}\,dt$, покажите, что $f(x) < f(y)$ для $0 \le x < y$.]

13. [*HM25*] Дано множество n независимых нормальных случайных величин X_1, X_2, ..., X_n со средним 0 и дисперсией 1. Покажите, как найти константы b_j и a_{ij}, $1 \le j \le i \le n$, такие, что если

$$Y_1 = b_1 + a_{11}X_1, \quad Y_2 = b_2 + a_{21}X_1 + a_{22}X_2, \quad \ldots, \quad Y_n = b_n + a_{n1}X_1 + \cdots + a_{nn}X_n,$$

то $Y_1, Y_2, \ldots, Y_n$ — зависимые нормально распределенные случайные величины, Y_j имеют среднее μ_j и заданную ковариационную матрицу (c_{ij}). (Ковариации c_{ij} между Y_i и Y_j определяются как среднее значение случайной величины $(Y_i - \mu_i)(Y_j - \mu_j)$. В частности, c_{jj} — это дисперсия Y_j, квадрат их среднеквадратичных отклонений. Не все матрицы (c_{ij}) могут быть ковариационными, и ваше построение, конечно, будет работать всякий раз, когда данное условие будет выполняться.)

14. [*M21*] Найдите распределение cX, если X — случайная величина с непрерывной функцией распределения $F(x)$ и если c — постоянная (возможно, отрицательная).

15. [*HM21*] Если X_1 и X_2 — независимые случайные величины с соответствующими функциями распределения $F_1(x)$ и $F_2(x)$ и плотностями $f_1(x) = F_1'(x)$, $f_2(x) = F_2'(x)$, какова функция распределения и плотность случайной величины $X_1 + X_2$?

▸ **16.** [*HM22*] (И. Г. Аренс.) Предложите алгоритм для генерирования случайной величины, имеющей гамма-распределение порядка a, когда $0 < a \le 1$, с помощью метода отбраковки с $cg(t) = t^{a-1}/\Gamma(a)$ для $0 < t < 1$ и с $cg(t) = e^{-t}/\Gamma(a)$ для $t \ge 1$.

▸ **17.** [*M24*] Чему равна *функция распределения* $F(x)$ случайной величины, имеющей геометрическое распределение с параметром p? Чему равна *производящая функция* $G(z)$? Чему равны среднее и дисперсия этого распределения?

18. [*M24*] Предложите метод вычисления случайной целочисленной величины N, такой, что N принимает значение n с вероятностью $np^2(1-p)^{n-1}$, $n \ge 0$. (Когда p сравнительно мало́, этот случай представляет особый интерес.)

19. [*22*] Случайная величина N с *отрицательным биномиальным распределением* с параметрами (t, p) принимает значения $N = n$ с вероятностью $\binom{t-1+n}{n}p^t(1-p)^n$. (В отличие от обычного биномиального распределения t не обязано быть целым, так как эта величина неотрицательна для всех n, каково бы ни было $t > 0$.) Обобщая упр. 18, объясните как генерировать целые случайные числа N с этим распределением, когда t — малое положительное целое число. Какой вы предложите метод для $t = p = \frac{1}{2}$?

20. [*M20*] Пусть A — площадь заштрихованной области на рис. 13, R — площадь содержащего ее прямоугольника, I — площадь внутренней области, определенной на шаге R2, и E — площадь между внутренней областью, отбрасываемой на шаге R3, и другой частью

прямоугольника. Определите длительность каждого шага алгоритма R для каждого из четырех вариантов в (25) в терминах A, R, I и E.

21. [*HM29*] Найдите формулы для величин A, R, I и E, определенных в упр. 20. (Для I и особенно для E можно использовать интерактивную алгебраическую систему.) Покажите, что $c = e^{1/4}$ — наилучшая возможная постоянная на шаге R2 для проверки "$X^2 \leq 4(1 + \ln c) - 4cU$".

22. [*HM40*] Можно ли получить точное пуассоновское распределение для больших μ, генерируя приближенно нормально распределенную величину, превращая ее в целочисленную некоторым подходящим способом и применяя (возможно, сложную) коррекцию в небольшом числе случаев?

23. [*HM23*] (Дж. фон Нейман.) Будут ли следующие два способа генерирования случайной величины X эквивалентны (т. е. будут ли случайные величины X при этом одинаково распределены)?

Метод 1. Присвоить $X \leftarrow \sin((\pi/2)U)$, где U — равномерно распределенная случайная величина.

Метод 2. Генерировать две равномерно распределенные случайные величины U и V; если $U^2 + V^2 \geq 1$, то повторять генерирование до тех пор, пока $U^2 + V^2 < 1$. Затем присвоить $X \leftarrow |U^2 - V^2|/(U^2 + V^2)$.

24. [*HM40*] (С. Улам (S. Ulam) и Дж. фон Нейман.) Пусть V_0 — случайно выбранное действительное число между 0 и 1. Определим последовательность $\langle V_n \rangle$ с помощью правила $V_{n+1} = 4V_n(1 - V_n)$. Если произвести вычисление с хорошей точностью, то в результате получится такая же последовательность, как случайная последовательность $\sin^2 \pi U_n$, где U_n — равномерно распределенные случайные величины, т. е. с функцией распределения члена последовательности, равной $F(x) = \int_0^x dx/\sqrt{2\pi x(1-x)}$. Если записать $V_n = \sin^2 \pi U_n$, то получим, что $U_{n+1} = (2U_n) \bmod 1$. Так как почти все действительные числа имеют случайное двоичное представление (см. раздел 3.5), полученная последовательность U_n будет равнораспределенной. Однако если вычисление V_n выполняется только с ограниченной точностью, то наши аргументы становятся несостоятельными, поскольку будет мешать ошибка округления. [См. работу фон Неймана *Collected Works* **5**, 768–770.]

Проанализируйте последовательность $\langle V_n \rangle$, определенную в разделе 3.3.4, когда вычисления производятся только с ограниченной точностью. Выполните эмпирический и теоретический анализ (для различных V_0). Будет ли последовательность иметь распределение, подобное ожидаемому?

25. [*M25*] Пусть X_1, X_2, ..., X_5 — двоичные слова, каждый двоичный разряд которых является независимой случайной величиной, принимающей значение 0 или 1 с вероятностью $\frac{1}{2}$. Чему равна вероятность того, что расположение данных двоичных разрядов $X_1 \vee (X_2 \wedge (X_3 \vee (X_4 \wedge X_5)))$ содержит 1? Обобщите результат.

26. [*M18*] Пусть N_1 и N_2 — независимые пуассоновские случайные величины со средними μ_1 и μ_2, где $\mu_1 > \mu_2 \geq 0$. Докажите или опровергните следующие утверждения: (a) $N_1 + N_2$ имеет распределение Пуассона со средним $\mu_1 + \mu_2$; (b) $N_1 - N_2$ имеет распределение Пуассона со средним $\mu_1 - \mu_2$.

27. [*22*] (И. Г. Аренс.) В большинстве бинарных компьютеров существует эффективный способ подсчета единиц в бинарном слове (см. раздел 7.1). Следовательно, существует хороший путь получения биномиального распределения (t, p), когда $p = \frac{1}{2}$, путем генерирования t случайных двоичных разрядов и подсчета числа единиц.

Предложите алгоритм, который генерирует случайные числа с биномиальным распределением (t, p) для *произвольного* p, используя в качестве источника случайных чисел только программу для частного случая, когда $p = \frac{1}{2}$. [*Указание.* Моделируйте сначала

процесс, который выглядит, как выбор самого старшего двоичного разряда равномерно распределенной случайной величины t, затем — как выбор второго двоичного разряда этой случайной величины, если старшего двоичного разряда недостаточно, чтобы определить, будет ли случайная величина $< p$, и т. д.]

28. [*HM35*] (Р. П. Брент.) Разработайте метод генерирования случайной точки на поверхности эллипсоида, определенного следующим образом: $\sum a_k x_k^2 = 1$, где $a_1 \geq \cdots \geq a_n > 0$.

29. [*M20*] (Дж. Л. Бентли (J. L. Bentley) и Дж. Б. Сакс (J. B. Saxe).) Найдите простой метод генерирования n равномерно распределенных между 0 и 1 чисел $X_1, \ldots, X_n$, требуя, чтобы они были упорядочены: $X_1 \leq \cdots \leq X_n$. Алгоритм должен состоять только из $O(n)$ шагов.

30. [*M30*] Объясните, как генерировать множество случайных точек (X_j, Y_j), таких, что если R — некоторый прямоугольник площадью α, содержащийся в единичной сфере, числа (X_j, Y_j), лежащие в R, имеют пуассоновское распределение со средним $\alpha\mu$.

31. [*HM39*] (*Непосредственное генерирование нормальных случайных величин.*)

a) Докажите, что если $a_1^2 + \cdots + a_k^2 = 1$ и $X_1, \ldots, X_k$ — независимые нормально распределенные случайные величины со средним 0 и дисперсией 1, то $a_1X_1 + \cdots + a_kX_k$ — нормальные случайные числа со средним 0 и дисперсией 1.

b) В доказательстве (a) предполагается, что можно генерировать новые нормальные случайные величины, используя старые точно так, как получают новые равномерно распределенные случайные величины из старых. Например, можно использовать идею 3.2.2–(7), но с рекуррентным соотношением, подобным

$$X_n = (X_{n-24} + X_{n-55})/\sqrt{2} \qquad \text{или} \qquad X_n = \tfrac{3}{5}X_{n-24} + \tfrac{4}{5}X_{n-55},$$

после получения множества нормальных случайных величин $X_0, \ldots, X_{54}$. Объясните, почему это *нехорошая* идея.

c) Покажите, тем не менее, что *существует* более быстрый по сравнению с другими метод генерирования нормальных случайных величин, использующий усовершенствованные идеи (a) и (b). [*Указание.* Если X и Y — независимые нормальные случайные величины, то такими же будут $X' = X\cos\theta + Y\sin\theta$ и $Y' = -X\sin\theta + Y\cos\theta$ для любых углов θ.]

32. [*HM30*] (К. С. Уоллес (C. S. Wallace).) Пусть X и Y — независимые случайные величины с показательным распределением со средним 1. Докажите, что X' и Y' также являются независимыми случайными величинами, имеющими показательное распределение со средним 1, если получить их из X и Y любым из следующих способов.

a) Задана $0 < \lambda < 1$,

$$X' = (1-\lambda)X - \lambda Y + (X+Y)[(1-\lambda)X < \lambda Y], \qquad Y' = X + Y - X'.$$

b) $(X', Y') = \begin{cases} (2X, Y-X), & \text{если } X \leq Y; \\ (2Y, X-Y), & \text{если } X > Y. \end{cases}$

c) Если X и Y имеют двоичное представление $X = (\ldots x_2x_1x_0.x_{-1}x_{-2}x_{-3}\ldots)_2$ и $Y = (\ldots y_2y_1y_0.y_{-1}y_{-2}y_{-3}\ldots)_2$, то X' и Y' имеют "перемешанные" значения

$$X' = (\ldots x_2y_1x_0.y_{-1}x_{-2}y_{-3}\ldots)_2, \qquad Y' = (\ldots y_2x_1y_0.x_{-1}y_{-2}x_{-3}\ldots)_2.$$

33. [*20*] Алгоритмы P, M, F и R генерируют нормальные случайные величины, поглощая неизвестное число равномерно распределенных , случайных величин U_1, U_2, ... Как их можно преобразовать, чтобы на выходе получить функцию только от одной случайной величины U?

3.4.2. Случайные выборки и перемешивания

В многочисленных случаях при обработке данных для беспристрастного выбора n случайных записей приходится обращаться к файлу, содержащему N записей. Такая проблема возникает, например, при контроле качества или других статистических вычислениях, где используются выборки. Обычно N очень велико, поэтому невозможно хранить сразу все данные в памяти, да и отдельные записи часто настолько большие, что в памяти нельзя содержать даже n записей. Поэтому необходима эффективная процедура для выбора n записей, которая определяла бы одно из двух — принимать или отбрасывать каждую запись при ее появлении, записывая принятую запись в выходной файл.

Для решения данной проблемы разработано несколько методов. Наиболее очевидным подходом является выбор каждой записи с вероятностью n/N. Возможно, иногда такой подход оправдан, но он дает в *среднем* только n записей в выборке. Среднеквадратичное отклонение равно $\sqrt{n(1-n/N)}$, и выборка может оказаться либо слишком большой либо слишком малой, чтобы дать необходимые результаты.

К счастью, простое преобразование "очевидной" процедуры позволяет получить то, что нужно: $(t+1)$-я запись будет выбрана с вероятностью $(n-m)/(N-t)$, если m записей уже выбраны. Эта вероятность обоснована, поскольку среди всех возможных способов такого выбора n записей из N, чтобы точно m записей было выбрано из первых t, доля способов, когда при этом выбирается также и $(t+1)$-я запись, равна

$$\binom{N-t-1}{n-m-1} \Big/ \binom{N-t}{n-m} = \frac{n-m}{N-t}. \tag{1}$$

(Иначе говоря, это вероятность выбора $(t+1)$-й записи при условии, что n записей из N выбирались так, что среди первых t было выбрано ровно m записей. — *Прим. ред.*)

Идея, развитая в предыдущем разделе, немедленно приводит к следующему алгоритму.

Алгоритм S (*Техника подбора выборки*). Выбрать n записей случайным образом из множества N, где $0 < n \le N$.

S1. [Инициализация.] Присвоить $t \leftarrow 0$, $m \leftarrow 0$. (В этом алгоритме m является числом записей, выбранных ранее, а t — общим числом введенных записей, с которыми мы работаем.)

S2. [Генерировать U.] Генерировать случайное число U, равномерно распределенное между 0 и 1.

S3. [Проверка.] Если $(N-t)U \ge n-m$, перейти к шагу S5.

S4. [Выбор.] Включить следующую запись в выборку и увеличить m и t на 1. Если $m < n$, перейти к шагу S2. Иначе выборка является полной и алгоритм завершен.

S5. [Пропустить.] Пропустить следующую запись (не включать ее в выборку), увеличить t на 1 и вернуться к шагу S2. ▮

На первый взгляд, алгоритм может показаться ненадежным и на самом деле неправильным, но тщательный анализ (см. упражнение, приведенное ниже) показывает, что он вполне надежен. Нетрудно проверить следующие факты.

a) Будет введено максимум N записей (мы никогда не достигнем конца файла, пока не выберем n элементов).

b) Выборка полностью беспристрастна. В частности, вероятность того, что любой заданный элемент выбран, как, например, последний элемент файла, равна n/N.

Утверждение (b) справедливо несмотря на то, что выбран $(t+1)$-й элемент *не* с вероятностью n/N, а с вероятностью, заданной в (1)! Это послужило причиной некоторой неразберихи в опубликованной литературе. Может ли читатель объяснить это кажущееся противоречие?

(*Замечание.* Используя алгоритм S, следует позаботиться о наличии различных источников случайных чисел U при каждом выполнении программы во избежание зависимости между выборками, полученными в разные дни. Это можно сделать, например, выбирая каждый раз различные значения X_0 в линейном конгруэнтном методе. Начальному значению X_0 может быть присвоено число месяца в день выполнения программы или последнее случайное число X, которое генерировалось при предыдущем прогоне программы.)

Обычно мы не проходим все N записей. В самом деле, так как указанное выше утверждение (b) гласит, что последняя запись выбирается с вероятностью n/N, вероятность того, что работа алгоритма завершится *до* выбора последней записи, точно равна $(1-n/N)$. Среднее число рассмотренных записей, когда $n=2$, приблизительно равно $\frac{2}{3}N$; общие формулы даны в упр. 5 и 6.

Алгоритм S и другие подобные методы обсуждаются в работе Ч. Т. Фана, Мервина Э. Мюллера и Ивана Резуха (C. T. Fan, Mervin E. Muller, and Ivan Rezucha, *J. Amer. Stat. Assoc.* **57** (1962), 387–402). Независимо этот метод был открыт Т. Г. Джонсом (T. G. Jones, *CACM* **5** (1962), 343).

Если значение N заранее неизвестно, то возникают трудности, поскольку знание точного значения N является решающим для выполнения алгоритма S. Допустим, необходимо случайно выбрать n записей из файла, если точно неизвестно, сколько записей в настоящее время содержится в этом файле. Можно сначала пересчитать все записи, а затем перейти к выбору n записей, но лучше так выбрать $m \geq n$ начальных записей на первом проходе, чтобы m было намного меньше N. Тогда на втором проходе нужно будет рассмотреть только m записей. Искусство состоит, конечно, в том, чтобы при подобном выборе в результате получить истинную случайную выборку из исходного файла.

Так как заранее неизвестно, когда завершится ввод, "модель" случайной выборки из записей следует хранить на входе при первом просмотре, чтобы всегда быть готовым к его окончанию. При чтении файла мы строим "резервуар", содержащий только записи, которые образуют предварительные выборки. Первые n записей всегда будут входить в резервуар. Когда $(t+1)$-я запись введена для $t \geq n$, в памяти формируется таблица n индексов записей, которые были выбраны среди первых t записей. Проблемой является сохранение этого состояния с t, увеличенным на единицу, т. е. поиск новой случайной выборки среди $t+1$ имеющихся известных сейчас записей. Нетрудно видеть, что новую запись следует включить в выборку с вероятностью $n/(t+1)$; при этом она заменит случайный элемент предыдущей выборки.

Итак, описанную работу выполняет следующий алгоритм.

Алгоритм R (*Резервуар выбора*). Выбрать случайно n записей из файла неизвестного размера $\geq n$, $n > 0$ задано. Дополнительный файл назовем резервуаром, содержащим все записи, которые являются кандидатами для окончательной выборки. Алгоритм использует таблицу различных индексов $I[j]$ для $1 \leq j \leq n$, каждый из которых указывает на одну из записей в резервуаре.

R1. [Инициализация.] Введем первые n записей и скопируем их в резервуар. Присвоим $I[j] \leftarrow j$ для $1 \leq j \leq n$ и присвоим $t \leftarrow m \leftarrow n$. (Если файл будет выборкой, имеющей меньше n записей, то, конечно, необходимо будет прервать выполнение алгоритма и сообщить о неблагоприятном исходе. В алгоритме индексы $I[1], \ldots, I[n]$ указывают записи в текущей выборке; m является размерностью резервуара, а t — номером входных записей, рассмотренных до сих пор.)

R2. [Конец файла?] Если записей на ввод больше нет, то перейти к шагу R6.

R3. [Генерирование и проверка.] Увеличить t на 1, затем генерировать случайное целое число M между 1 и t (включительно). Если $M > n$, перейти к шагу R5.

R4. [Увеличить резервуар.] Копировать следующую запись входного файла в резервуар, увеличить m на 1 и присвоить $I[M] \leftarrow m$. (Запись, предварительно указанная как $I[M]$, заменяется в выборке новой записью.) Возврат к шагу R2.

R5. [Пропустить.] Пропустить следующую запись входного файла (не включать ее в резервуар) и возвратиться к шагу R2.

R6. [Следующий просмотр.] Упорядочить индексы I таблицы входов так, чтобы $I[1] < \cdots < I[n]$, затем тщательно разобрать резервуар, копируя записи с этими индексами в выходной файл, который фиксирует окончательный выбор. ▮

Алгоритм R предложен Аланом Дж. Вотерманом (Alan G. Waterman). Читатель при желании может выполнить упр. 9, используя эти операции.

Если записи достаточно короткие, конечно, излишне помещать их в резервуар. n записей текущего файла можно постоянно хранить в памяти, и алгоритм станет более простым (см. упр. 10).

Относительно алгоритма R возникает естественный вопрос: "Какова предполагаемая размерность резервуара?". В упр. 11 показано, что среднее значение m точно равно $n(1 + H_N - H_n)$, т. е. приблизительно $n\big(1 + \ln(N/n)\big)$. Так, если $N/n = 1000$, в резервуаре будет содержаться лишь около $1/125$ целых записей исходного файла.

Заметим, что алгоритмы S и R можно применять, чтобы получать выборки для различных независимых классов одновременно. Например, если есть большой файл фамилий и адресов постоянных жителей США, можно случайным образом сделать выборку точно 10 жителей из каждого из 50 штатов, не просматривая 50 раз файл и не выполняя первую сортировку файла по штатам.

Возможно значительное улучшение алгоритмов S и R, когда n/N мало́. Скажите, сколько записей можно пропустить, вместо того чтобы решать, пропускать ли каждую запись, если генерируется единственная случайная величина? (См. упр. 8.)

Проблема выборки может быть рассмотрена как вычисление случайного *сочетания* в соответствии с обычным определением сочетания N по n (см. раздел 1.2.6). Рассмотрим задачу вычисления случайной *перестановки* t объектов. Назовем ее

задачей *перемешивания* (*тасования*), так как, например, тасование колоды карт является ничем иным, как ее случайной перестановкой.

Достаточно легко убедить любого игрока в карты, что традиционная процедура тасования карт ужасно несовершенна. Нет надежды получить таким методом каждую из $t!$ перестановок с примерно равными вероятностями. Рассказывают, что знаток игры в бридж использует это обстоятельство, когда принимает решение, стоит ли ему блефовать. По крайней мере семь "перемешиваний" колоды из 52 карт происходит с вероятностью, примерно равной 10% от вероятности получить эти перемешивания при равномерном распределении, в то время как 14 случайных перемешиваний имеют вероятность появления, примерно равную вероятности их получения при равномерном распределении [см. Aldous and Diaconis, *AMM* **93** (1986), 333–348].

Если t мало́, можно получить случайную перестановку очень быстро, генерируя случайное целое число между 1 и $t!$. Например, когда $t = 4$, случайного числа между 1 и 24 достаточно, чтобы выбрать случайную перестановку из таблицы всех возможностей. Но для больших t необходимо быть более осторожным, если требуется, чтобы каждая перестановка была равновероятной, так как $t!$ намного больше точности отдельного случайного числа.

Подходящая процедура перемешивания может быть получена с помощью уже упоминаемого алгоритма 3.3.2P, который дает простое соответствие между каждой из $t!$ возможных перестановок и последовательностью чисел $(c_1, c_2, \ldots, c_{t-1})$ с $0 \le c_j \le j$. Можно легко получить такое множество случайных чисел, и это соответствие можно использовать для получения случайных перестановок.

Алгоритм P (*Перемешивание*). Пусть X_1, X_2, ..., X_t — множество t чисел для перемешивания.

P1. [Инициализация.] Присвоить $j \leftarrow t$.

P2. [Генерирование U.] Генерировать случайное число U, равномерно распределенное между 0 и 1.

P3. [Замена.] Присвоить $k \leftarrow \lfloor jU \rfloor + 1$. (Сейчас k — случайное целое число между 1 и j. В упр. 3.4.1–3 объясняется, что деление на j *не* следует использовать для вычисления k.) Заменим $X_k \leftrightarrow X_j$.

P4. [Уменьшение j.] Уменьшить j на 1. Если $j > 1$, возвратиться к шагу P2. ▮

Впервые этот алгоритм опубликовали Р. А. Фишер и Ф. Ятс (R. A. Fisher and F. Yates, *Statistical Tables* (London, 1938), Example 12) на обычном языке, а Р. Дурстенфелд — на языке компьютерном (R. Durstenfeld, *CACM* **7** (1964), 420). Чтобы просто генерировать случайную перестановку $\{1, \ldots, t\}$ вместо перемешивания заданной последовательности $(X_1, \ldots, X_t)$, можно избежать операции замены $X_k \leftrightarrow X_j$, выполнив увеличение j от 1 до t и присвоив $X_j \leftarrow X_k$, $X_k \leftarrow j$ [см. D. E. Knuth, *The Stanford GraphBase* (New York: ACM Press, 1994), 104].

Р. Салфи (R. Salfi, *COMPSTAT 1974* (Vienna, 1974), 28–35) указал, что алгоритм P не имеет возможности генерировать более m различных перестановок, когда мы получаем равномерно распределенные U_j из линейной конгруэнтной последовательности по модулю m или используем рекуррентное соотношение $U_{n+1} = f(U_n)$,

для которого U_n может дать только m различных значений, потому что окончательная перестановка в таком случае полностью определяется первыми генерированными значениями U. Так, например, если $m = 2^{32}$, определенные перестановки из 13 элементов никогда не появятся, поскольку $13! \approx 1.45 \times 2^{32}$. В большинстве случаев на самом деле нет *необходимости* получать все 13! перестановок, но огорчает, что исключение данных перестановок определяется фактически простым математическим правилом, таким как решетчатая структура (см. раздел 3.3.4).

Эта проблема не возникает, когда используется генератор Фибоначчи с запаздыванием, подобный 3.2.2–(7), с достаточно длинным периодом. Но даже таким методом невозможно получать все перестановки постоянно, если нельзя точно задать по крайней мере $t!$ различных начальных значений для инициализации генератора. Другими словами, не существует возможности получить $\lg t!$ истинно случайных двоичных разрядов на выходе, если не получить $\lg t!$ истинно случайных двоичных разрядов на входе. Как говорится в разделе 3.5, не стоит из-за этого огорчаться.

Алгоритм S можно легко преобразовать для получения случайной перестановки случайных сочетаний (см. упр. 15). Случайные объекты комбинаторики других видов (например, разбиения) рассматриваются в разделе 7.2 и в книге *Combinatorial Algorithms* by Nijenhuis and Wilf (New York: Academic Press, 1975).

УПРАЖНЕНИЯ

1. [*M12*] Объясните (1).

2. [*20*] Докажите , что алгоритм S никогда не пытается прочесть более N записей своего входного файла.

▶ **3.** [*22*] $(t+1)$-я запись в алгоритме S выбирается с вероятностью $(n-m)/(N-t)$, а не n/N, однако алгоритм требует, чтобы выборка была беспристрастной, поэтому каждая запись должна выбираться с *одинаковой* вероятностью. Почему оба эти утверждения правильны?

4. [*M23*] Пусть $p(m,t)$ — вероятность того, что выбрано точно m записей среди первых t в методе отбора выборок. Покажите непосредственно из алгоритма S, что

$$p(m,t) = \binom{t}{m}\binom{N-t}{n-m}\Big/\binom{N}{n} \qquad \text{для } 0 \le t \le N.$$

5. [*M24*] Чему равно среднее значение t по завершении алгоритма S? (Другими словами, сколько из N записей будет просмотрено в среднем, прежде чем выборка станет полной?)

6. [*M24*] Чему равно среднеквадратичное отклонение значения, вычисленного в упр. 5?

7. [*M25*] Докажите, что любой *заданный* выбор n записей из множества N записей может быть получен алгоритмом S с вероятностью $1/\binom{N}{n}$. Следовательно, выборка является совершенно беспристрастной.

▶ **8.** [*M39*] (Д. С. Виттер (J. S. Vitter).) Алгоритм S производит одну равномерно распределенную случайную величину для каждой входной записи. Цель этого упражнения — рассмотреть более эффективный подход, при котором можно быстрее вычислить точное число X входных записей, которые следует пропустить, прежде чем сделать первый выбор.

a) Чему равна вероятность того, что $X \ge k$, k задано?

b) Покажите, что результат (a) позволяет выполнить вычисление X, генерируя только одну равномерно распределенную случайную величину U, и производит $O(X)$ других вычислений.

c) Покажите, что можно также присвоить $X \leftarrow \min(Y_N, Y_{N-1}, \ldots, Y_{N-n+1})$, где Y_t независимы и каждое Y_t является случайным целым числом в области $0 \le Y_t < t$.

d) Найдите максимальную скорость, т. е. покажите, что X также может быть вычислено в среднем за $O(1)$ шагов, если использовать "метод втискивания", подобный 3.4.1–(18).

9. [*12*] Пусть $n = 3$. Если алгоритм R применяется к файлу, содержащему 20 записей, которые пронумерованы от 1 до 20, и если случайные числа, генерированные на шаге R3, равны соответственно

$$4, 1, 6, 7, 5, 3, 5, 11, 11, 3, 7, 9, 3, 11, 4, 5, 4,$$

какая запись попадет в резервуар? Какая из них окажется в окончательной выборке?

10. [*15*] Преобразуйте алгоритм R так, чтобы обойтись без резервуара, предполагая, что n записей текущей выборки можно хранить в памяти.

▶ **11.** [*M25*] Пусть p_m — вероятность того, что точно m элементов занесены в резервуар в течение первого прохода алгоритма R. Определите производящую функцию $G(z) = \sum_m p_m z^m$ и найдите среднее значение и среднеквадратичное отклонение. (Используйте идеи из раздела 1.2.10.)

12. [*M26*] Суть алгоритма P состоит в том, что любую перестановку π можно однозначно записать как результат транспонирования в виде $\pi = (a_t t)\ldots(a_3 3)(a_2 2)$, где $1 \le a_j \le j$ для $t \ge j > 1$. Докажите, что существует также единственное представление вида $\pi = (b_2 2)(b_3 3)\ldots(b_t t)$, где $1 \le b_j \le j$ для $1 < j \le t$, и составьте алгоритм для вычисления b_k через a_k за $O(t)$ шагов.

13. [*M23*] (С. В. Голомб (S. W. Golomb).) Один из наиболее простых способов тасования игральных карт — разделение колоды карт на две максимально равные части и их тасование вместе. (В правилах Хойла карточных игр при обсуждении профессиональной этики игроков в карты сказано: "Тасование такого вида можно выполнить приблизительно трехкратным тщательным перемешиванием игральных карт".) Рассмотрим колоду из $2n-1$ игральных карт $X_1, X_2, \ldots, X_{2n-1}$. "Идеальное перемешивание" — это разделение s раз этой колоды на части $X_1, X_2, \ldots, X_n$ и $X_{n+1}, \ldots, X_{2n-1}$. Чередуя их, получим X_1, X_{n+1}, X_2, X_{n+2}, $\ldots$, X_{2n-1}, X_n. Операция "разрезания" c^j меняет $X_1, X_2, \ldots, X_{2n-1}$ на $X_{j+1}, \ldots, X_{2n-1}, X_1, \ldots, X_j$. Покажите, что, комбинируя идеальное перемешивание и разрезание, можно получить не более $(2n-1)(2n-2)$ различных упорядочений компоновок игральных карт, если $n > 1$.

14. [*22*] Разрезав и перетасовав перестановку $a_0 a_1 \ldots a_{n-1}$, можно заменить ее последовательностью, содержащей подпоследовательности

$$a_x\, a_{(x+1) \bmod n} \cdots a_{(y-1) \bmod n} \qquad \text{и} \qquad a_y\, a_{(y+1) \bmod n} \cdots a_{(x-1) \bmod n},$$

которые перемешаны определенным способом для некоторых x и y. Таким образом, последовательность 3890145267 является разрезанием и перетасовкой 0123456789 с $x = 3$ и $y = 8$.

a) Начав с расположенных обычным образом 52 игральных карт

2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A 2 3 4 5 6 7 8 9 10 J Q K A,
♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♣ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♢ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♡ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠ ♠

мистер Д. Г. Квик (J. H. Quick) ("Студент") сделал случайное разрезание и перетасовку, затем убрал крайнюю слева карту и вставил ее в случайное место. Получилась последовательность

9 10 K J Q A K A 2 Q 3 2 3 4 5 6 7 4 8 9 5 10 6 J 7 Q 8 K 9 10 J Q A K 2 3 A 4 2 3 4 5 6 5 6 7 8 7 9 10 J 8.
♣ ♣ ♡ ♣ ♡ ♣ ♣ ♢ ♡ ♢ ♠ ♠ ♠ ♠ ♠ ♢ ♠ ♠ ♢ ♠ ♢ ♠ ♢ ♠ ♢ ♠ ♢ ♢ ♢ ♢ ♠ ♢ ♣ ♣ ♢ ♣ ♡ ♡ ♡ ♣ ♡ ♡ ♡ ♡ ♣ ♡ ♡ ♡ ♣

Какую карту он убрал из крайней слева позиции?

b) Начав снова с колоды в ее обычном порядке, Квик сделал *три* разрезания и перетасовки, прежде чем сдвинул крайнюю слева карту на новое место:

Какую карту он сдвинул на этот раз?

▸ **15.** [*30*] (Оле-Йохан Дахл (Ole-Johan Dahl).) Если $X_k = k$ для $1 \le k \le t$ в начале алгоритма P и если остановить алгоритм, когда j достигнет значения $t-n$, последовательность $X_{t-n+1}, \ldots, X_t$ станет случайной перестановкой случайных комбинаций из n элементов. Покажите, как эффективно смоделировать эту процедуру, используя только $O(n)$ ячеек памяти.

▸ **16.** [*M25*] Придумайте метод вычисления случайной выборки по n записей из N при заданных N и n, основываясь на идее рандомизации (раздел 6.4). Можете использовать в нем $O(n)$ ячеек для хранения и в среднем $O(n)$ единиц времени. Метод должен представлять выборку как упорядоченное множество целых чисел $1 \le X_1 < X_2 < \cdots < X_n \le N$.

17. [*M22*] (P. B. Флойд (R. W. Floyd).) Докажите, что следующий алгоритм генерирует случайную выборку S из n целых чисел из $\{1, \ldots, N\}$. Присвойте сначала $S \leftarrow \emptyset$, а затем для $j \leftarrow N-n+1, N-n+2, \ldots, N$ (в таком порядке) присвойте $k \leftarrow \lfloor jU \rfloor + 1$ и

$$S \leftarrow \begin{cases} S \cup \{k\}, & \text{если } k \notin S; \\ S \cup \{j\}, & \text{если } k \in S. \end{cases}$$

*3.5. ЧТО ТАКОЕ СЛУЧАЙНАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ

А. Вводные замечания. Выше в данной главе рассказывалось, как генерировать последовательности

$$\langle U_n \rangle = U_0,\ U_1,\ U_2,\ \ldots \tag{1}$$

действительных чисел в области $0 \le U_n < 1$. Эти последовательности назывались случайными, даже несмотря на то, что они совершенно детерминированные по характеру. Чтобы оправдать использование этой терминологии, мы утверждали, что числа "ведут себя так, как будто они действительно случайны". Такое утверждение может быть удовлетворительным для практических целей (в настоящее время), но это шаг в сторону очень важного методического и теоретического вопроса "Что именно подразумевается под случайным поведением?". Необходимо количественное определение. Нежелательно говорить о понятиях, которых мы на самом деле не понимаем, главным образом потому, что, по-видимому, можно сделать много парадоксальных утверждений, касающихся случайных чисел.

Теория вероятностей и математическая статистика тщательно избегают обсуждения спорных вопросов. Они воздерживаются от безусловных утверждений и вместо этого выражают все в терминах *вероятностей*, связанных с последовательностью случайных событий. Аксиомы теории вероятностей установлены так, что теоретические вероятности можно легко вычислить, но о том, что на самом деле означает "вероятность" или как это понятие можно разумно применить к действительности, ничего не говорится. В книге *Probability, Statistics, and Truth* (New York: Macmillan, 1957), Р. фон Мизес (R. von Mises) подробно обсуждает эту ситуацию и предлагает следующую точку зрения: определение вероятностей зависит от того, какое используется определение случайных последовательностей.

Процитируем здесь несколько утверждений двух авторов, комментировавших эту тему.

> *Д. Г. Лехмер* (D. H. Lehmer) (1951): "Случайная последовательность является смутным понятием, олицетворяющим идею последовательности, в которой каждый член является непредсказуемым для непосвященных и значения которой проходят определенное количество проверок, традиционных у статистиков и отчасти зависящих от пользователей, которым предложена последовательность".
>
> *Д. Н. Франклин* (*J. N. Franklin*) (1962): "Последовательность (1) случайна, если она обладает любыми свойствами, присущими всем бесчисленным последовательностям независимых выборок случайных равномерно распределенных величин".

Утверждение Франклина, по существу, обобщает высказывание Лехмера о том, что последовательность должна удовлетворять *всем* статистическим критериям. Его определение не вполне точное, и позже мы убедимся, что разумное объяснение его утверждения приводит к заключению о том, что не существует такого объекта, как случайная последовательность! Так давайте начнем с менее ограничительного утверждения Лехмера и попытаемся сделать *его* точным. Что нам действительно необходимо — так это сравнительно короткий перечень математических свойств, каждое из которых удовлетворяет нашим интуитивным представлениям о случайной

последовательности. К тому же этого перечня будет вполне достаточно, чтобы согласиться с тем, что *любая* последовательность, удовлетворяющая этим свойствам, является "случайной". В данном разделе рассматривается то, что кажется адекватным точному определению случайности согласно этим критериям, хотя много интересных вопросов остается без ответов.

Пусть u и v — действительные числа, $0 \le u < v \le 1$. Если U — случайная величина, равномерно распределенная между 0 и 1, вероятность того, что $u \le U < v$, равна $v - u$. Например, вероятность, что $\frac{1}{5} \le U < \frac{3}{5}$, равна $\frac{2}{5}$. Как можно выразить это свойство единственного числа U свойством бесконечной последовательности U_0, U_1, U_2, ...? Очевидный ответ — подсчитать, сколько раз U_n находится между u и v, и сказать, что среднее число попаданий в этот интервал равно $v - u$. Наше интуитивное понятие вероятности основано, таким образом, на частоте появления события.

Точнее, пусть $\nu(n)$ — число значений j, $0 \le j < n$, таких, что $u \le U_j < v$, и отношением $\nu(n)/n$ необходимо приблизить значение $v - u$, когда n стремится к бесконечности:

$$\lim_{n\to\infty} \frac{\nu(n)}{n} = v - u. \tag{2}$$

Если это условие выполняется для всех вариантов u и v, говорят, что последовательность *равнораспределена*.

Пусть утверждение $S(n)$ относится и к целому числу n, и к последовательности U_0, U_1, ..., например $S(n)$ может быть приведенным выше утверждением "$u \le U_n < v$". Можно обобщить понятие, используемое в предыдущем разделе, чтобы определить вероятность того, что $S(n)$ справедливо по отношению к некоторой бесконечной последовательности.

Определение А. *Пусть $\nu(n)$ — число значений j, $0 \le j < n$, таких, что $S(j)$ верно. Мы говорим, что $S(n)$ выполняется с вероятностью λ, если предел $\nu(n)/n$, когда n стремится к бесконечности, равен λ.* А именно: $\Pr\big(S(n)\big) = \lambda$, если $\lim_{n\to\infty} \nu(n)/n = \lambda$.

В терминах этой записи последовательность $U_0, U_1, \ldots$ равнораспределена тогда и только тогда, когда $\Pr(u \le U_n < v) = v - u$ для всех действительных чисел u, v при $0 \le u < v \le 1$.

Последовательность может быть равнораспределена, даже если она не случайна. Например, если $U_0, U_1, \ldots$ и $V_0, V_1, \ldots$ — равнораспределенные последовательности, то нетрудно показать, что последовательность

$$W_0, W_1, W_2, W_3, \ldots = \tfrac{1}{2}U_0,\ \tfrac{1}{2}+\tfrac{1}{2}V_0,\ \tfrac{1}{2}U_1,\ \tfrac{1}{2}+\tfrac{1}{2}V_1,\ \ldots \tag{3}$$

также равнораспределена, поскольку подпоследовательность $\frac{1}{2}U_0, \frac{1}{2}U_1, \ldots$ равнораспределена между 0 и $\frac{1}{2}$, в то время как промежуточные члены $\frac{1}{2}+\frac{1}{2}V_0, \frac{1}{2}+\frac{1}{2}V_1, \ldots$ равнораспределены между $\frac{1}{2}$ и 1. Но в последовательности W_j, $j = 0, 1, 2, \ldots$, значения, меньшие $\frac{1}{2}$, всегда следуют за значениями, бóльшими или равными $\frac{1}{2}$ соответственно. Значит, последовательность не случайна согласно любому разумному определению. Необходимы свойства, которые сильнее равнораспределенности.

Естественная возможность обобщить свойство равнораспределенности, которое позволяет развеять сомнения предыдущего раздела, — рассмотреть смежные пары

членов нашей последовательности. Можно потребовать, чтобы последовательность удовлетворяла условиям

$$\Pr(u_1 \le U_n < v_1 \text{ и } u_2 \le U_{n+1} < v_2) = (v_1 - u_1)(v_2 - u_2) \tag{4}$$

для любых членов u_1, v_1, u_2, v_2 при $0 \le u_1 < v_1 \le 1$, $0 \le u_2 < v_2 \le 1$. И вообще, для любого положительного целого k можно потребовать, чтобы наша последовательность была *k-распределенной* в смысле определения B.

Определение B. *Говорят, что последовательность* (1) *будет k-распределенной, если*

$$\Pr(u_1 \le U_n < v_1, \ \ldots, \ u_k \le U_{n+k-1} < v_k) = (v_1 - u_1) \ldots (v_k - u_k) \tag{5}$$

для всех вариантов действительных чисел u_j, v_j при $0 \le u_j < v_j \le 1$ для $1 \le j \le k$.

Равнораспределенная последовательность является 1-распределенной. Заметим, что, если $k > 1$, k-распределенная последовательность всегда является $(k-1)$-распределенной, так как в (5) можно положить $u_k = 0$ и $v_k = 1$. Таким образом, в частности, любая последовательность, о которой известно, что она 4-распределена, должна быть также 3- и 2-распределенной. Можно определить наибольшее k, для которого данная последовательность является k-распределенной, что приведет нас к формулировке более сильного свойства.

Определение C. *Говорят, что последовательность ∞-распределена, если она k-распределена для всех положительных целых k.*

До сих пор мы рассматривали $[0\,..\,1)$-последовательности, т. е. последовательности действительных чисел, лежащих между 0 и 1. Такие же понятия применяются к целочисленным последовательностям. Говорят, что последовательность $\langle X_n \rangle = X_0, X_1, X_2, \ldots$ является *b-ичной последовательностью*, если каждый член последовательности X_n является одним из целых чисел $0, 1, \ldots, b-1$. Таким образом, 2-ичная (бинарная) последовательность — это последовательность нулей и единиц.

Определим также *b-ичное число*, состоящее из k цифр, как строку k целых чисел $x_1 x_2 \ldots x_k$, где $0 \le x_j < b$ для $1 \le j \le k$.

Определение D. *Говорят, что b-ичная последовательность является k-распределенной, если*

$$\Pr(X_n X_{n+1} \ldots X_{n+k-1} = x_1 x_2 \ldots x_k) = 1/b^k \tag{6}$$

для всех b-ичных чисел $x_1 x_2 \ldots x_k$.

Из этого определения ясно, что если $U_0, U_1, \ldots$ является k-распределенной последовательностью $[0\,..\,1)$, то последовательность $\lfloor bU_0 \rfloor, \lfloor bU_1 \rfloor, \ldots$ является k-распределенной b-ичной последовательностью. (Если положить $u_j = x_j/b$, $v_j = (x_j + 1)/b$, $X_n = \lfloor bU_n \rfloor$, то равенство (5) превратится в равенство (6).) Более того, каждая k-распределенная b-ичная последовательность является также $(k-1)$-распределенной, если $k > 1$: мы складываем вероятности для b-ичных чисел $x_1 \ldots x_{k-1}\,0, x_1 \ldots x_{k-1}\,1, \ldots, x_1 \ldots x_{k-1}\,(b-1)$, чтобы получить

$$\Pr(X_n \ldots X_{n+k-2} = x_1 \ldots x_{k-1}) = 1/b^{k-1}$$

(Вероятности для несовместных событий аддитивны; см. упр. 5.) Следовательно, естественно ввести понятие ∞-распределенных b-ичных последовательностей, как в определении C.

Представление положительных действительных чисел в системе с основанием b можно рассматривать как b-ичную последовательность, например π соответствует 10-ичной последовательности 3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5, 8, 9, ... Предполагается, что эта последовательность ∞-распределенная, но никто, однако, не в состоянии даже доказать, что она является точно 1-распределенной.

Проанализируем это понятие более подробно для случая, когда k равно миллиону. Бинарная последовательность, являющаяся 1 000 000-распределенной, может содержать серию из миллиона нулей! Аналогично $[0\,..\,1)$-последовательность, являющаяся 1 000 000-распределенной, может содержать миллион последовательных значений, каждое из которых меньше $\frac{1}{2}$. Правда, в среднем такое случается только в одной из $2^{1000000}$ ситуаций, но это действительно *происходит*. Действительно, данный феномен встречается в любой поистине случайной последовательности. Мы используем пока наше интуитивное понятие "истинная случайность". Каждый может легко себе представить, что такая ситуация будет иметь значительные последствия, если такое множество из миллиона "истинно случайных" чисел использовать в эксперименте компьютерного моделирования. Это будет хорошим поводом для того, чтобы вызвать недовольство генератором случайных чисел. Однако при наличии последовательности чисел, которые никогда не пробегают миллион последовательных U_j, меньших $\frac{1}{2}$, последовательность будет не случайной и она не будет подходящим источником чисел для других предполагаемых применений, использующих на входе крайне длинные блоки U_j. В итоге *истинно случайная последовательность будет проявлять локальную "неслучайность"* . Локальная "неслучайность" необходима в одних применениях, но она гибельна в других. Можно сделать вывод, что нет последовательности "случайных" чисел, которую можно было бы использовать в любом случае.

В подобном духе каждый может утверждать, что невозможно решить, будет ли *конечная* последовательность случайной; любая конкретная последовательность ничем не отличается от любой другой. Эти факты действительно представляют собой камни преткновения всякий раз, когда необходимо дать полезное определение случайности, но они не являются истинной причиной смятения. Все еще можно сформулировать такое определение случайности бесконечной последовательности действительных чисел, чтобы соответствующая теория (рассмотренная должным образом) много дала для понимания обычных конечных последовательностей рациональных чисел, которые на самом деле генерируются на компьютере. Более того, ниже в этом разделе будет показано, что существует несколько внушающих доверие определений случайности конечных последовательностей.

B. ∞-распределенные последовательности. Кратко рассмотрим теорию ∞-распределенных последовательностей. Чтобы описать ее адекватно, понадобится немного высшей математики, поэтому в остальной части этого раздела предполагается, что читатель знаком с понятиями, необходимыми для понимания дальнейшего материала.

Во-первых, нужно обобщить определение A, так как фигурирующий в нем предел не существует для всех последовательностей. Определим

$$\overline{\Pr}\big(S(n)\big) = \limsup_{n\to\infty} \frac{\nu(n)}{n}, \qquad \underline{\Pr}\big(S(n)\big) = \liminf_{n\to\infty} \frac{\nu(n)}{n}. \tag{7}$$

Тогда вероятность $\Pr\big(S(n)\big)$ является общим значением $\underline{\Pr}\big(S(n)\big)$ и $\overline{\Pr}\big(S(n)\big)$ (если она существует).

Мы видели, что k-распределенная $[0\,..\,1)$-последовательность приводит к k-распределенной b-ичной последовательности, если U заменить $\lfloor bU \rfloor$. Наша теорема показывает, что обратное утверждение также справедливо.

Теорема A. *Пусть* $\langle U_n \rangle = U_0,\ U_1,\ U_2,\ \ldots$ — $[0\,..\,1)$*-последовательность. Если последовательность*

$$\langle \lfloor b_j U_n \rfloor \rangle = \lfloor b_j U_0 \rfloor,\ \lfloor b_j U_1 \rfloor,\ \lfloor b_j U_2 \rfloor,\ \ldots$$

является k*-распределенной* b_j*-ичной последовательностью для любого* b_j *из бесконечной последовательности целых чисел* $1 < b_1 < b_2 < b_3 < \cdots$, *то исходная последовательность* $\langle U_n \rangle$ k*-распределенная.*

В качестве примера предположим, что $b_j = 2^j$. Последовательность $\lfloor 2^j U_0 \rfloor$, $\lfloor 2^j U_1 \rfloor, \ldots$ является, по существу, последовательностью первых j двоичных разрядов бинарного представления $U_0, U_1, \ldots$. Если все эти последовательности целых чисел k-распределены в смысле определения D, то последовательность действительных чисел $U_0, U_1, \ldots$ также должна быть k-распределенной в смысле определения B.

Доказательство. Если последовательность $\lfloor bU_0 \rfloor, \lfloor bU_1 \rfloor, \ldots$ k-распределена, то с помощью сложения вероятностей получим, что (5) выполняется всякий раз, когда каждое u_j и v_j — рациональные числа со знаменателем b. Предположим, что u_j, v_j — любые действительные числа, и пусть u'_j, v'_j — рациональные числа со знаменателем b, такие, что

$$u'_j \le u_j < u'_j + 1/b, \qquad v'_j \le v_j < v'_j + 1/b.$$

Пусть $S(n)$ — утверждение, что $u_1 \le U_n < v_1, \ldots, u_k \le U_{n+k-1} < v_k$. Получим

$$\begin{aligned}\overline{\Pr}\big(S(n)\big) &\le \Pr\Big(u'_1 \le U_n < v'_1 + \frac{1}{b},\ \ldots,\ u'_k \le U_{n+k-1} < v'_k + \frac{1}{b}\Big) \\ &= \Big(v'_1 - u'_1 + \frac{1}{b}\Big) \ldots \Big(v'_k - u'_k + \frac{1}{b}\Big); \\ \underline{\Pr}\big(S(n)\big) &\ge \Pr\Big(u'_1 + \frac{1}{b} \le U_n < v'_1,\ \ldots,\ u'_k + \frac{1}{b} \le U_{n+k-1} < v'_k\Big) \\ &= \Big(v'_1 - u'_1 - \frac{1}{b}\Big) \ldots \Big(v'_k - u'_k - \frac{1}{b}\Big).\end{aligned}$$

Сейчас $\big|(v'_j - u'_j \pm 1/b) - (v_j - u_j)\big| \le 2/b$. Так как наше неравенство выполняется для всех $b = b_j$ и так как $b_j \to \infty$ при $j \to \infty$, получим

$$(v_1 - u_1) \ldots (v_k - u_k) \le \underline{\Pr}\big(S(n)\big) \le \overline{\Pr}\big(S(n)\big) \le (v_1 - u_1) \ldots (v_k - u_k). \quad \blacksquare$$

Следующая теорема является основным орудием для доказательства различных утверждений о k-распределенных последовательностях.

Теорема В. *Предположим, что* $\langle U_n \rangle$ — *k-распределенная* $[0\,..\,1)$*-последовательность, и пусть* $f(x_1, x_2, \ldots, x_k)$ — *интегрируемая по Риману функция k переменных. Тогда*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{0\le j<n} f(U_j, U_{j+1}, \ldots, U_{j+k-1}) = \int_0^1 \cdots \int_0^1 f(x_1, x_2, \ldots, x_k)\, dx_1 \ldots dx_k. \tag{8}$$

Доказательство. Из определения k-распределенной последовательности следует, что этот результат справедлив в частном случае, когда

$$f(x_1, \ldots, x_k) = [u_1 \le x_1 < v_1,\ \ldots,\ u_k \le x_k < v_k], \tag{9}$$

для некоторых постоянных u_1, v_1, $\ldots$, u_k, v_k. Более того, (8) выполняется каждый раз, когда $f = a_1 f_1 + a_2 f_2 + \cdots + a_m f_m$ и каждая функция f_j является функцией вида (9). Другими словами, (8) выполняется каждый раз, когда f является "ступенчатой функцией", полученной путем разбиения единичного k-мерного куба на подкубы, грани которых параллельны оси координат, и f принимает постоянные значения на каждом подкубе.

Пусть f — любая интегрируемая по Риману функция. Если ϵ — любое положительное число, то (по определению интегрируемости по Риману) получим, что существуют ступенчатые функции $\underline{f}$ и $\overline{f}$, такие, что $\underline{f}(x_1, \ldots, x_k) \le f(x_1, \ldots, x_k) \le \overline{f}(x_1, \ldots, x_k)$, и такие, что разность интегралов $\underline{f}$ и $\overline{f}$ меньше ϵ. Поэтому (8) выполняется для $\underline{f}$ и $\overline{f}$. И поскольку

$$\begin{aligned} \frac{1}{n} \sum_{0\le j<n} \underline{f}(U_j, \ldots, U_{j+k-1}) &\le \frac{1}{n} \sum_{0\le j<n} f(U_j, \ldots, U_{j+k-1}) \\ &\le \frac{1}{n} \sum_{0\le j<n} \overline{f}(U_j, \ldots, U_{j+k-1}), \end{aligned}$$

можно заключить, что (8) верно также для f. ∎

Теорема В может применяться, например, в *критерии перестановок* из раздела 3.3.2. Пусть $(p_1, p_2, \ldots, p_k)$ — любая перестановка чисел $\{1, 2, \ldots, k\}$. Покажем, что

$$\Pr(U_{n+p_1-1} < U_{n+p_2-1} < \cdots < U_{n+p_k-1}) = 1/k!. \tag{10}$$

Для доказательства предположим, что последовательность $\langle U_n \rangle$ k-распределена и пусть

$$f(x_1, \ldots, x_k) = [x_{p_1} < x_{p_2} < \cdots < x_{p_k}].$$

Имеем

$$\begin{aligned} &\Pr(U_{n+p_1-1} < U_{n+p_2-1} < \cdots < U_{n+p_k-1}) \\ &\quad = \int_0^1 \cdots \int_0^1 f(x_1, \ldots, x_k)\, dx_1 \ldots dx_k \\ &\quad = \int_0^1 dx_{p_k} \int_0^{x_{p_k}} \cdots \int_0^{x_{p_3}} dx_{p_2} \int_0^{x_{p_2}} dx_{p_1} = \frac{1}{k!}. \end{aligned}$$

Следствие Р. *Если* $[0\,..\,1)$*-последовательность k-распределена, то она удовлетворяет критерию перестановок порядка k в смысле равенства* (10). ∎

Также можно показать, что последовательность удовлетворяет *критерию сериальной корреляции.*

Следствие S. *Если* $[0..1)$*-последовательность* $(k+1)$*-распределена, то коэффициент сериальной корреляции между* U_n *и* U_{n+k} *стремится к нулю:*

$$\lim_{n\to\infty} \frac{\frac{1}{n}\sum U_jU_{j+k} - \left(\frac{1}{n}\sum U_j\right)\left(\frac{1}{n}\sum U_{j+k}\right)}{\sqrt{\left(\frac{1}{n}\sum U_j^2 - \left(\frac{1}{n}\sum U_j\right)^2\right)\left(\frac{1}{n}\sum U_{j+k}^2 - \left(\frac{1}{n}\sum U_{j+k}\right)^2\right)}} = 0.$$

(Все суммирования здесь выполняются по $0 \le j < n$.)

Доказательство. По теореме B значения

$$\frac{1}{n}\sum U_jU_{j+k}, \qquad \frac{1}{n}\sum U_j^2, \qquad \frac{1}{n}\sum U_{j+k}^2, \qquad \frac{1}{n}\sum U_j, \qquad \frac{1}{n}\sum U_{j+k}$$

стремятся соответственно к пределу $\frac{1}{4}, \frac{1}{3}, \frac{1}{3}, \frac{1}{2}, \frac{1}{2}$ при $n \to \infty$. ▮

Рассмотрим несколько более общие свойства распределений последовательностей. Мы определяли понятие k-распределения, рассматривая все смежные строки размерности k, например последовательность является 2-распределенной тогда и только тогда, когда точки

$$(U_0,U_1),\ (U_1,U_2),\ (U_2,U_3),\ (U_3,U_4),\ (U_4,U_5),\ \ldots$$

равнораспределены в единичном квадрате. Это вполне возможно несмотря на то, что пары точек (U_1,U_2), (U_3,U_4), (U_5,U_6), ... могут быть *не* равнораспределенными. Если в некоторой области не хватает точек (U_{2n-1},U_{2n}), их можно компенсировать другими точками: (U_{2n},U_{2n+1}). Например, периодическая бинарная последовательность

$$\langle X_n\rangle = 0,0,0,1,\ 0,0,0,1,\ 1,1,0,1,\ 1,1,0,1,\ 0,0,0,1,\ \ldots \tag{11}$$

с периодом длины 16 будет 3-распределенной; однако последовательность четных элементов $\langle X_{2n}\rangle$ = 0, 0, 0, 0, 1, 0, 1, 0, ... имеет в три раза больше нулей, чем единиц, тогда как подпоследовательность нечетных элементов $\langle X_{2n+1}\rangle$ = 0, 1, 0, 1, 1, 1, 1, 1, ... имеет в три раза больше единиц, чем нулей.

Предположим, что последовательность $\langle U_n\rangle$ является ∞-распределенной. Пример (11) показывает, что подпоследовательность чередующихся членов $\langle U_{2n}\rangle = U_0$, U_2, U_4, U_6, ... не обязана быть ∞-распределенной или даже 1-распределенной. Но $\langle U_{2n}\rangle$ на самом деле является ∞-распределенной и многое еще будет верным.

Определение E. $[0..1)$*-последовательность* $\langle U_n\rangle$ *называют* (m,k)*-распределенной, если*

$$\Pr(u_1 \le U_{mn+j} < v_1,\ u_2 \le U_{mn+j+1} < v_2,\ \ldots,\ u_k \le U_{mn+j+k-1} < v_k) = (v_1-u_1)\ldots(v_k-u_k)$$

для любого выбора действительных чисел u_r, v_r *при* $0 \le u_r < v_r \le 1$ *для* $1 \le r \le k$ *и для всех целых* j *при* $0 \le j < m$.

Поэтому k-распределенная последовательность является частным случаем определения E при $m = 1$; случай, когда $m = 2$, означает, что строки размерности k,

начинающиеся с четного номера, должны иметь такую же плотность, как и строки размерности k, начинающиеся с нечетного номера, и т. д.

Следующие свойства определения E очевидны:

$$(m,k)\text{-распределенная последовательность является } (m,\kappa)\text{-распределенной для } 1 \le \kappa \le k; \tag{12}$$

$$(m,k)\text{-распределенная последовательность является } (d,k)\text{-распределенной для всех делителей } d \text{ числа } m. \tag{13}$$

(См. упр. 8.) Также можно определить понятие (m,k)-распределенности b-ичной последовательности, как в определении D, и доказать теорему A, остающуюся верной для (m,k)-распределенных последовательностей.

Следующая теорема, которая во многих отношениях является, скорее, сюрпризом, показывает, что свойства, присущие ∞-распределению, действительно очень строгие, более строгие, чем мы представляли себе, когда впервые рассматривали определение этого понятия.

Теорема C. (Иван Нивен (Ivan Niven) и Г. С. Цукерман (H. S. Zuckerman).) *∞-распределенная последовательность является (m,k)-распределенной для всех положительных m и k.*

Доказательство. Достаточно доказать теорему для b-ичной последовательности, используя обобщение только что упомянутой теоремы A. Более того, можно предположить, что $m = k$, так как (12) и (13) утверждают, что последовательность (m,k)-распределена, если она (mk,mk)-распределена.

Таким образом, докажем, что *любая ∞-распределенная b-ичная последовательность $X_0, X_1, \ldots$ является (m,m)-распределенной для всех положительных целых m.* Наше доказательство — это упрощенная версия первоначального доказательства, приведенного в статье Niven and Zuckerman, *Pacific J. Math.* **1** (1951), 103–109.

Ключевая используемая идея является важным методом, применяемым во многих ситуациях в математике: "Если и сумма m величин, и сумма их квадратов согласуются с гипотезой, что m величин равны, то гипотеза верна". В строгом виде этот принцип можно сформулировать так.

Лемма E. *Заданы m последовательностей чисел $\langle y_{jn}\rangle = y_{j0}, y_{j1}, \ldots$ для $1 \le j \le m$. Предположим, что*

$$\begin{aligned} \lim_{n\to\infty} (y_{1n} + y_{2n} + \cdots + y_{mn}) &= m\alpha, \\ \limsup_{n\to\infty} (y_{1n}^2 + y_{2n}^2 + \cdots + y_{mn}^2) &\le m\alpha^2. \end{aligned} \tag{14}$$

Тогда для каждого j существует $\lim_{n\to\infty} y_{jn}$ и он равен α.

Невероятно простое доказательство этой леммы приведено в упр. 9. ▮

Продолжим доказательство теоремы C. Пусть $x = x_1x_2\ldots x_m$ — b-ичное число; скажем, что x *встречается* на позиции p, если $X_{p-m+1}X_{p-m+2}\ldots X_p = x$. Пусть $\nu_j(n)$ — число x, находящихся на позиции p, когда $p < n$ и $p \bmod m = j$. Пусть

$y_{jn} = \nu_j(n)/n$ и нужно доказать, что

$$\lim_{n\to\infty} y_{jn} = \frac{1}{mb^m}. \tag{15}$$

Во-первых, известно, что

$$\lim_{n\to\infty} (y_{0n} + y_{1n} + \cdots + y_{(m-1)n}) = \frac{1}{b^m}, \tag{16}$$

так как последовательность m-распределена. Согласно лемме E и равенству (16) теорема доказана, если показать, что

$$\limsup_{n\to\infty} (y_{0n}^2 + y_{1n}^2 + \cdots + y_{(m-1)n}^2) \le \frac{1}{mb^{2m}}. \tag{17}$$

Однако это неравенство не очевидно; необходимо несколько деликатных маневров, прежде чем можно будет его доказать. Пусть q кратно m. Рассмотрим

$$C(n) = \sum_{0\le j<m} \binom{\nu_j(n) - \nu_j(n-q)}{2}. \tag{18}$$

Это число появлений пар x-в на позициях p_1 и p_2, для которых $n - q \le p_1 < p_2 < n$ и $p_2 - p_1$ кратно m. Рассмотрим сумму

$$S_N = \sum_{n=1}^{N+q} C(n). \tag{19}$$

Каждое появление пары x-в, встречающейся на позициях p_1 и p_2 с $p_1 < p_2 < p_1 + q$, где $p_2 - p_1$ кратно m и $p_1 \le N$, учитывается точно $p_1 + q - p_2$ раз в общей сумме S_N (т. е. когда $p_2 < n \le p_1 + q$), а такие пары, которые появляются на позициях p_1 и p_2 с $N < p_1 < p_2 < N + q$, учитываются точно $N + q - p_2$ раз.

Пусть $d_t(n)$ — число пар x, встречающихся на позициях p_1 и p_2 с $p_1 + t = p_2 < n$. Приведенный выше анализ показывает, что

$$\sum_{0<t<q/m} (q - mt) d_{mt}(N + q) \ge S_N \ge \sum_{0<t<q/m} (q - mt) d_{mt}(N). \tag{20}$$

Так как начальная последовательность является q-распределенной, то

$$\lim_{N\to\infty} \frac{1}{N} d_{mt}(N) = \frac{1}{b^{2m}} \tag{21}$$

для всех t, $0 < t < q/m$, и, следовательно, из (20) получаем

$$\lim_{N\to\infty} \frac{S_N}{N} = \sum_{0<t<q/m} \frac{q - mt}{b^{2m}} = \frac{q(q-m)}{2mb^{2m}}. \tag{22}$$

Из этих равенств после нескольких преобразований получим утверждение теоремы.

По определению

$$2S_N = \sum_{n=1}^{N+q} \sum_{0\le j<m} \big((\nu_j(n) - \nu_j(n-q))^2 - (\nu_j(n) - \nu_j(n-q))\big),$$

и можно удалить не возведенные в квадрат члены и, применяя (16), получить

$$\lim_{N\to\infty}\frac{T_N}{N}=\frac{q(q-m)}{mb^{2m}}+\frac{q}{b^m},\tag{23}$$

где

$$T_N=\sum_{n=1}^{N+q}\sum_{0\le j<m}\big(\nu_j(n)-\nu_j(n-q)\big)^2.$$

Используя неравенство

$$\frac{1}{r}\Big(\sum_{j=1}^{r}a_j\Big)^2\le\sum_{j=1}^{r}a_j^2$$

(см. упр. 1.2.3–30), находим, что

$$\limsup_{N\to\infty}\sum_{0\le j<m}\frac{1}{N(N+q)}\Big(\sum_{n=1}^{N+q}\big(\nu_j(n)-\nu_j(n-q)\big)\Big)^2\le\frac{q(q-m)}{mb^{2m}}+\frac{q}{b^m}.\tag{24}$$

Также получим

$$q\,\nu_j(N)\le\sum_{N<n\le N+q}\nu_j(n)=\sum_{n=1}^{N+q}\big(\nu_j(n)-\nu_j(n-q)\big)\le q\nu_j(N+q).$$

Подставив это неравенство в (24), получим

$$\limsup_{N\to\infty}\sum_{0\le j<m}\Big(\frac{\nu_j(N)}{N}\Big)^2\le\frac{q-m}{qmb^{2m}}+\frac{1}{qb^m}.\tag{25}$$

Данная формула справедлива всякий раз, когда q кратно m, и если мы устремим $q\to\infty$, то получим (17), что и завершает доказательство.

Более простое доказательство можно найти у Дж. В. С. Касселя (J. W. S. Cassels, *Pacific J. Math.* **2** (1952), 555–557). ▮

В упр. 29 и 30 иллюстрируется нетривиальность этой теоремы и тот факт, что q-распределенная последовательность имеет вероятности, отклоняющиеся от истинных вероятностей (m,m)-распределения, по существу, не более чем на $1/\sqrt{q}$ (см. (25)). Для доказательства теоремы необходима гипотеза о ∞-распределении последовательности.

Используя теорему С, можно доказать, что ∞-распределенная последовательность проходит критерий серий, критерий "максимум-t", критерий конфликтов, критерий промежутков между днями рождений и критерий подпоследовательностей, о которых упоминалось в разделе 3.3.2. Нетрудно показать, что она также удовлетворяет критерию интервалов, покер-критерию и критерию монотонности (см. упр. 12–14). Критерий собирания купонов является значительно более трудным, но и его последовательность проходит (см. упр. 15 и 16).

Существование ∞-распределенной последовательности достаточно простого вида гарантирует следующая теорема.

Теорема F. (Дж. Н. Франклин (J. N. Franklin).) $[0\,..\,1)$-*последовательность* U_0, U_1, U_2, ... *с*

$$U_n = \theta^n \bmod 1 \tag{26}$$

является ∞-*распределенной для почти всех действительных чисел* $\theta > 1$. *Другими словами, множество*

$$\{\theta \mid \theta > 1 \text{ и (26) не } \infty\text{-распределено}\}$$

имеет меру нуль.

Доказательство этой теоремы и некоторые обобщения приведены в *Math. Comp.* **17** (1963), 28–59. ▮

Франклин показал, что θ должно быть трансцендентным числом для того, чтобы (26) была ∞-распределенной. Раньше, в 60-е годы, степени $\langle \pi^n \bmod 1 \rangle$ получали в результате трудоемких вычислений, использующих многократную точность для $n \leq 10000$, и 35 самых старших двоичных разрядов этих чисел, оставленных в файле на диске, успешно использовались как источник равномерно распределенных случайных чисел. Согласно теореме F вероятность того, что степени $\langle \pi^n \bmod 1 \rangle$ ∞-распределены, равна 1, однако существует несчетное множество действительных чисел, поэтому теорема не дает информации о том, действительно ли последовательность для π имеет ∞-распределение. Можно совершенно спокойно держать пари, что никто при нашей жизни не *докажет*, что именно данная последовательность *не* является ∞-распределенной, хотя это и возможно. Руководствуясь такими соображениями, он может законно удивиться, если окажется, что существует *какая-либо* ∞-распределенная последовательность: *существует ли алгоритм вычисления действительных чисел* U_n *для всех* $n \geq 0$, *такой, что последовательность* $\langle U_n \rangle$ ∞-*распределена?* Ответ — "Да", как показано, например, в работе D. E. Knuth, *BIT* **5** (1965), 246–250. Построенная последовательность полностью состоит из рациональных чисел. На самом деле каждое число U_n имеет ограниченное представление в двоичной системе счисления. Другое построение определенной ∞-распределенной последовательности, отчасти более сложное, чем построение предыдущей последовательности, вытекает из теоремы W, приведенной ниже. (См. также Н. М. Коробов, *Изв. Акад. наук СССР* **20** (1956), 649–660.)

C. Эквивалентно ли понятие ∞-распределенности понятию случайности? Принимая во внимание все теоретические результаты относительно ∞-распределенных последовательностей, можно быть уверенным в одном: понятие "∞-распределенная последовательность" является важным в математике. Кроме того, кажется очевидным, что интуитивное понятие случайности можно формализовать следующим образом.

Определение R1. $[0\,..\,1)$-*последовательность называется случайной, если она является* ∞-*распределенной.*

Мы видели, что последовательности, удовлетворяющие этому определению, удовлетворяют всем статистическим критериям раздела 3.3.2 и многим другим.

Попытаемся объективно критиковать это определение. Прежде всего, всякая ли "поистине случайная" последовательность ∞-распределена? Существует бесконечное число последовательностей U_0, U_1, ... действительных чисел между нулем и единицей. Если генератор истинно случайных чисел выдает значения U_0, U_1, ..., то любую из возможных последовательностей можно рассматривать как в равной степени подходящую и некоторые из последовательностей (на самом деле бессчетное количество) даже не равнораспределены. С другой стороны, используя любое разумное определение вероятности на этом пространстве всех возможных последовательностей, можно прийти к заключению, что случайная последовательность является ∞-распределенной *с вероятностью 1*. Итак, получена следующая формализация определения случайности Франклина, приведенного в начале этого раздела.

Определение R2. $[0\,..\,1)$-*последовательность* $\langle U_n \rangle$ *называется случайной, если всякий раз, когда* P *является таким свойством, что* $P(\langle V_n \rangle)$ *выполняется с вероятностью 1 для последовательности* $\langle V_n \rangle$ *независимых случайных равномерно распределенных величин,* $P(\langle U_n \rangle)$ *также выполняется.*

Можно ли предположить, что определение R1 эквивалентно определению R2? Давайте выдвинем возможные возражения против определения R1 и проанализируем их.

Во-первых, определение R1 распространяется только на предельные свойства последовательностей при $n \to \infty$. Существуют ∞-распределенные последовательности, в которых первый миллион элементов — нули. Можно ли такие последовательности рассматривать как случайные?

Это возражение не очень существенно. Если ϵ — любое положительное число, то нет причины каждому элементу последовательности из первого миллиона не быть меньше ϵ. С вероятностью 1 истинно случайная последовательность содержит бесконечно много рядов в миллион последовательных элементов, меньших ϵ, так почему это не может произойти в начале последовательности?

Во-вторых, рассмотрим определение R2, и пусть P — такое свойство, что все элементы последовательности различны; P справедливо с вероятностью 1, поэтому любая последовательность с миллионом нулей не является случайной по *этому* критерию.

Пусть P — такое свойство, что *нет* элемента последовательности, равного нулю. P снова справедливо с вероятностью 1, поэтому по определению R2 любая последовательность с нулевым элементом не случайна. Вообще говоря, пусть x_0 — любое фиксированное число между нулем и единицей и пусть P — такое свойство: нет элемента последовательности, равного x_0. Из определения R2 сейчас следует, что нет случайной последовательности, которая может содержать элемент x_0! Можно доказать, что *не существует последовательности, удовлетворяющей условиям определения R2.* (Если $U_0, U_1, \ldots$ — такая последовательность, то выберем $x_0 = U_0$.)

Следовательно, если R1 — слишком слабое определение, то R2 является, несомненно, слишком строгим. "Правильное" определение должно быть менее строгим, чем R2. В действительности мы не показали, что определение R1 слишком слабое, поэтому продолжим исследование. Как упоминалось выше, ∞-распределенная последовательность *рациональных* чисел построена. (В самом деле, это не так уди-

вительно; см. упр. 18.) Почти все действительные числа иррациональны. Возможно, следовало бы потребовать, чтобы для случайной последовательности выполнялось

$$\Pr(U_n \text{ рациональное}) = 0.$$

Из определения равнораспределенности (см. равенство (2)) следует, что $\Pr(u \le U_n < v) = v-u$. Существует очевидный способ обобщения этого определения, используя теорию меры: "если $S \subseteq [0\,..\,1)$ — множество меры μ, то

$$\Pr(U_n \in S) = \mu \tag{27}$$

для всех случайных последовательностей $\langle U_n \rangle$". В частности, если S — множество рациональных чисел, то оно имеет меру нуль; значит, нет последовательности рациональных чисел, равнораспределенных в этом обобщенном смысле. Разумно ожидать, что теорема B может распространяться на интегрирование по Лебегу вместо интегрирования по Риману, если оговорено свойство (27). Однако мы снова найдем, что определение (27) слишком строгое, так как *нет* последовательностей, удовлетворяющих этому свойству. Если U_0, U_1, ... — любая последовательность, множество $S = \{U_0, U_1, \ldots\}$ есть множество меры нуль. Кроме того, $\Pr(U_n \in S) = 1$. Поэтому в силу тех же аргументов, из-за которых рациональные числа были исключены из случайных последовательностей, можно исключить все случайные последовательности.

До сих пор определение R1 можно было считать приемлемым. Однако существует несколько совершенно обоснованных возражений по этому поводу. Например, если имеется случайная в интуитивном смысле последовательность, то бесконечная подпоследовательность

$$U_0,\ U_1,\ U_4,\ U_9,\ \ldots,\ U_{n^2},\ \ldots \tag{28}$$

также должна быть случайной. Это не всегда верно для ∞-распределенных последовательностей. В самом деле, если взять любую ∞-распределенную последовательность и присвоить $U_{n^2} \leftarrow 0$ для всех n, количество $\nu_k(n)$, появляющихся в критерии k-распределенности, изменится самое большее на $\sqrt{n}$. Значит, отношение $\nu_k(n)/n$ не изменится. Таким образом, определение R1 не удовлетворяет этому критерию случайности.

Можно было бы усилить R1 следующим образом.

Определение R3. $[0\,..\,1)$-*последовательность называется случайной, если каждая ее бесконечная подпоследовательность является* ∞-*распределенной.*

Однако еще раз определение вышло очень строгим; любая равнораспределенная последовательность $\langle U_n \rangle$ имеет монотонную подпоследовательность с $U_{s_0} < U_{s_1} < U_{s_2} < \cdots$.

Секрет состоит в том, чтобы ограничиться подпоследовательностями, при построении которых можно было бы заранее сказать, принадлежит ли заданное U_n этой подпоследовательности. Предлагаем следующее определение.

Определение R4. $[0\,..\,1)$-*последовательность* $\langle U_n \rangle$ *называется случайной, если для любого эффективного алгоритма, точно определяющего бесконечную последовательность различных неотрицательных целых чисел* s_n *для* $n \ge 0$, *подпоследова-*

тельность $U_{s_0}, U_{s_1}, U_{s_2}, \ldots$, соответствующая этому алгоритму, является ∞-распределенной.

Алгоритм, относящийся к определению R4, — это эффективная процедура вычисления s_n при заданном n (см. обсуждение в разделе 1.1). Так, например, последовательность $\langle \pi^n \bmod 1 \rangle$ *не* удовлетворяет R4, поскольку она или не равнораспределена, или существует эффективный алгоритм, определяющий бесконечную подпоследовательность s_n с $(\pi^{s_0} \bmod 1) < (\pi^{s_1} \bmod 1) < (\pi^{s_2} \bmod 1) < \cdots$. Точно так же никакая *явно определенная последовательность не может удовлетворять определению R4*. Это справедливо, если согласиться с тем, что никакая явно определенная последовательность не является случайной. Судя по ее виду, явная последовательность $\langle \theta^n \bmod 1 \rangle$ на самом деле, однако, удовлетворяет определению R4 для почти всех действительных чисел $\theta > 1$; это не противоречие, так как почти все θ не могут быть вычислены алгоритмом. Ж. Ф. Коксма (J. F. Koksma) доказал, что $\langle \theta^{s_n} \bmod 1 \rangle$ является 1-распределенной для почти всех $\theta > 1$, если $\langle s_n \rangle$ — любая последовательность различных положительных целых чисел [*Compositio Math.* **2** (1935), 250–258]. Г. Нидеррейтер (H. Niederreiter) и Р. Ф. Тичи (R. F. Tichy) усилили теорему Коксма, заменив "1-распределенность" "∞-распределенностью" [*Mathematika* **32** (1985), 26–32]. Только счетное множество последовательностей $\langle s_n \rangle$ эффективно определимо; значит, $\langle \theta^n \bmod 1 \rangle$ почти всегда удовлетворяет R4.

Определение R4 более строгое, чем определение R1, но все еще можно утверждать, что определение R4 слишком слабое. Пусть, например, $\langle U_n \rangle$ — истинно случайная последовательность. Определим подпоследовательность $\langle U_{s_n} \rangle$ следующим образом: $s_0 = 0$ и, если $n > 0$, s_n — наименьшее целое число $\geq n$, для которого все $U_{s_n-1}, U_{s_n-2}, \ldots, U_{s_n-n}$ меньше $\frac{1}{2}$. Таким образом мы определили подпоследовательность значений, следующих за первой серией n значений, меньших $\frac{1}{2}$. Предположим, что $U_n < \frac{1}{2}$ соответствует выпадению "герба" при бросании монеты. Игроки склонны считать, что длинный ряд "гербов" предполагает, что появление "решки" становится более вероятным, если монета не поддельная. В этом случае подпоследовательность $\langle U_{s_n} \rangle$ определяет систему азартной игры для человека, который делает n-ю ставку на первую решку, следующую после серии из n "гербов." Игрок, возможно, думает, что $\Pr(U_{s_n} \geq \frac{1}{2})$ больше $\frac{1}{2}$, но, конечно, в истинной случайной последовательности $\langle U_{s_n} \rangle$ будет совершенно случайным. Нет системы азартных игр, которая всегда приводит к победе! Определение R4 ничего не говорит о подпоследовательности, формируемой в соответствии с такой системой азартных игр; так что, по-видимому, необходимо нечто большее.

Пусть определено "правило подпоследовательности" $\mathcal{R}$ как бесконечной последовательности функций $\langle f_n(x_1, \ldots, x_n) \rangle$, где для $n \geq 0$, f_n — функция n переменных и значение $f_n(x_1, \ldots, x_n)$ равно либо 0, либо 1. Здесь $x_1, \ldots, x_n$ — элементы некоторого множества S. (Таким образом, в данном случае f_0 — это постоянная функция, равная либо 0, либо 1.) Правило подпоследовательности $\mathcal{R}$ определяет подпоследовательность бесконечной последовательности $\langle X_n \rangle$ элементов S следующим образом: *n-й член X_n принадлежит подпоследовательности $\langle X_n \rangle \mathcal{R}$ тогда и только тогда, когда* $f_n(X_0, X_1, \ldots, X_{n-1}) = 1$. Заметим, что подпоследовательность

$\langle X_n\rangle\mathcal{R}$, определенная таким образом, необязательно бесконечна; фактически в ней вообще может не быть элементов.

Например, подпоследовательность азартного игрока, описанная выше, соответствует такому правилу подпоследовательности: $f_0 = 1$ и для $n > 0$, $f_n(x_1, \ldots, x_n) = 1$ тогда и только тогда, когда найдется некоторое k, $0 < k \le n$, такое, что k последовательных чисел x_m, x_{m-1}, $\ldots$, x_{m-k+1} все будут $< \frac{1}{2}$, когда $m = n$, но не когда $k \le m < n$.

Правило подпоследовательностей $\mathcal{R}$ называют *исчислимым*, если существует эффективный алгоритм, определяющий значение $f_n(x_1, \ldots, x_n)$, когда n и $x_1, \ldots, x_n$ заданы на входе. При попытке определить случайность лучше ограничиться исчислимыми правилами подпоследовательностей, когда пытаемся определить случайность, чтобы не получить чрезмерно ограничительное определение, подобное R3. Но эффективный алгоритм нельзя рассматривать с произвольными входными действительными числами. Например, если действительное число x точно определено бесконечным разложением в десятичной системе счисления, то не существует алгоритма для того, чтобы определить, будет ли $x < \frac{1}{3}$, так как для этого нужно исследовать все цифры числа $0.333\ldots$. Поэтому исчислимое правило подпоследовательностей неприменимо ко всем $[0\,..\,1)$-последовательностям и служит основой для следующего определения b-ичных последовательностей.

Определение R5. *b-ичная последовательность называется случайной, если каждая бесконечная подпоследовательность, определенная исчислимым правилом подпоследовательностей, является 1-распределенной.*

$[0\,..\,1)$-последовательность $\langle U_n\rangle$ называется случайной, если b-ичная последовательность $\langle\lfloor bU_n\rfloor\rangle$ является случайной для всех целых чисел $b \ge 2$.

Заметим, что определение R5 говорит только об 1-распределении, а не о ∞-распределении. Интересно проверить, что это может быть сделано без потери общности. Очевидно, можно определить исчислимое правило подпоследовательности $\mathcal{R}(a_1 \ldots a_k)$ следующим образом при любом заданном b-ичном числе $a_1 \ldots a_k$: пусть $f_n(x_1, \ldots, x_n) = 1$ тогда и только тогда, когда $n \ge k - 1$ и $x_{n-k+1} = a_1$, $\ldots$, $x_{n-1} = a_{k-1}$, $x_n = a_k$. Если $\langle X_n\rangle$ является k-распределенной b-ичной последовательностью, то правило $\mathcal{R}(a_1 \ldots a_k)$, с помощью которого выбирается подпоследовательность, содержащая точно следующие за появлением $a_1 \ldots a_k$ члены, определяет бесконечную подпоследовательность, и, если эта подпоследовательность 1-распределена, каждая из строк $a_1 \ldots a_k a_{k+1}$ размерности $(k+1)$ для $0 \le a_{k+1} < b$ появляется с вероятность $1/b^{k+1}$ в $\langle X_n\rangle$. Таким образом, индукцией по k можно доказать, что последовательность, удовлетворяющая определению R5, является k-распределенной для всех k. Подобным образом, рассматривая "композицию" правил подпоследовательностей, получаем: если $\mathcal{R}_1$ определяет бесконечную подпоследовательность $\langle X_n\rangle\mathcal{R}_1$, то можно определить правило подпоследовательности $\mathcal{R}_1\mathcal{R}_2$, для которого $\langle X_n\rangle\mathcal{R}_1\mathcal{R}_2 = (\langle X_n\rangle\mathcal{R}_1)\mathcal{R}_2$, и найти, что все подпоследовательности, рассматриваемые в определении R5, ∞-распределенные (см. упр. 32).

Факт, что ∞-распределение следует из определения R5 как очень частный случай, ободряет, и это хороший признак того, что можно, наконец, найти требуемое определение случайности. Но, увы, это все еще проблема. Не ясно, почему последовательность, удовлетворяющая определению R5, должна удовлетворять определе-

нию R4. Введенные нами исчислимые правила подпоследовательностей всегда дают подпоследовательности $\langle X_{s_n}\rangle$, для которых $s_0 < s_1 < \cdots$, но $\langle s_n\rangle$ не должны быть монотонны в R4; они только должны удовлетворять условию $s_n \neq s_m$ для $n \neq m$.

Итак, определения R4 и R5 можно скомбинировать следующим образом.

Определение R6. *b-ичная последовательность $\langle X_n\rangle$ называется случайной, если для каждого эффективного алгоритма, точно определяющего бесконечную последовательность различных неотрицательных целых чисел $\langle s_n\rangle$ как функцию n и значений $X_{s_0}, \ldots, X_{s_{n-1}}$, подпоследовательность $\langle X_{s_n}\rangle$, которая соответствует этому алгоритму, является случайной в смысле определения R5.*

$[0\,..\,1)$-последовательность $\langle U_n\rangle$ называется случайной, если b-ичная последовательность $\langle \lfloor bU_n\rfloor\rangle$ является случайной для всех целых чисел $b \geq 2$.

Автор утверждает*, что это определение точно удовлетворяет всем разумным философским требованиям случайности, а значит, отвечает на основной вопрос, поставленный в этом разделе.

D. Существование случайных последовательностей. Как было показано, определение R3 слишком строгое в том смысле, что не существует удовлетворяющей ему последовательности, и в приведенных выше определениях R4–R6 предпринимались попытки сохранить существенные элементы определения R3. Для того чтобы показать, что определение R6 не чрезмерно ограничительно, все еще необходимо доказать, что последовательность, удовлетворяющая всем этим условиям, существует. Интуитивно мы совершенно правильно ощущаем, что это не проблема, так как мы верим, что истинно случайная последовательность существует и удовлетворяет определению R6, но доказательство действительно необходимо, чтобы показать, что определение состоятельно.

Интересный метод построения последовательностей, удовлетворяющих определению R5, найден А. Вальдом (A. Wald). Он начинается с построения простой 1-распределенной последовательности.

Лемма T. *Пусть последовательность действительных чисел $\langle V_n\rangle$ определена следующим образом в терминах двоичной системы:*

$$V_0 = 0, \quad V_1 = .1, \quad V_2 = .01, \quad V_3 = .11, \quad V_4 = .001, \quad \ldots$$
$$V_n = .c_r \ldots c_1 1, \qquad \text{если } n = 2^r + c_1 2^{r-1} + \cdots + c_r. \tag{29}$$

Обозначим через $I_{b_1\ldots b_r}$ множество всех действительных чисел в $[0\,..\,1)$, которые имеют двоичное представление, начиная с $0.b_1\ldots b_r$. Таким образом,

$$I_{b_1\ldots b_r} = \big[(0.b_1\ldots b_r)_2\ ..\ (0.b_1\ldots b_r)_2 + 2^{-r}\big). \tag{30}$$

Тогда, если $\nu(n)$ — число V_k в $I_{b_1\ldots b_r}$ при $0 \leq k < n$, получим

$$\big|\nu(n)/n - 2^{-r}\big| \leq 1/n. \tag{31}$$

Доказательство. Так как $\nu(n)$ — число k, для которых $k \bmod 2^r = (b_r\ldots b_1)_2$, получим $\nu(n) = t$ или $t+1$, когда $\lfloor n/2^r\rfloor = t$. Следовательно, $\big|\nu(n) - n/2^r\big| \leq 1$. ∎

* По крайней мере, он сделал такое заявление, когда готовил этот материал в 1966 году.

Из (31) следует, что последовательность $\langle \lfloor 2^r V_n \rfloor \rangle$ — это равнораспределенная 2^r-ичная последовательность. Отсюда согласно теореме A получаем, что $\langle V_n \rangle$ — равнораспределенная $[0\,..\,1)$-последовательность. Действительно, достаточно ясно, что $\langle V_n \rangle$ настолько равнораспределена на $[0\,..\,1)$, насколько это вообще возможно. (Чтобы получить дополнительную информацию о данной и родственных последовательностях, обратитесь к работам J. G. van der Corput, *Proc. Koninklijke Nederl. Akad. Wetenschappen* **38** (1935), 813–821, 1058–1066; J. H. Halton, *Numerische Math.* **2** (1960), 84–90, 196; S. Haber, *J. Research National Bur. Standards* **B70** (1966), 127–136; R. Béjian and H. Faure, *Comptes Rendus Acad. Sci. Paris* **A285** (1977), 313–316; H. Faure, *J. Number Theory* **22** (1986), 4–20; S. Tezuka, *ACM Trans. Modeling and Comp. Simul.* **3** (1993), 99–107.

Л. Г. Рамшоу (L. H. Ramshaw) показал, что последовательность $\langle \phi n \bmod 1 \rangle$ немного более равнораспределена, чем $\langle V_n \rangle$ (см. *J. Number Theory* **13** (1981), 138–175).

Пусть сейчас $\mathcal{R}_1$, $\mathcal{R}_2$, ... — бесконечное множество правил подпоследовательностей, и необходимо найти последовательность $\langle U_n \rangle$, для которой все бесконечные подпоследовательности $\langle U_n \rangle \mathcal{R}_j$ равнораспределены.

Алгоритм W (*Последовательность Вальда*). Задана бесконечная последовательность правил подпоследовательностей $\mathcal{R}_1$, $\mathcal{R}_2$, ..., которая определяет подпоследовательности $[0\,..\,1)$-последовательностей рациональных чисел. Эта процедура определяет $[0\,..\,1)$-последовательность $\langle U_n \rangle$. Для вычисления требуется бесконечно много вспомогательных переменных $C[a_1, \ldots, a_r]$, где $r \geq 1$ и $a_j = 0$ или 1 для $1 \leq j \leq r$. Вначале все эти переменные равны нулю.

W1. [Инициализация n.] Присвоить $n \leftarrow 0$.

W2. [Инициализация r.] Присвоить $r \leftarrow 1$.

W3. [Проверка $\mathcal{R}_r$.] Если элемент U_n должен принадлежать подпоследовательности, определенной $\mathcal{R}_r$, которая основана на значениях U_k для $0 \leq k < n$, присвоить $a_r \leftarrow 1$; иначе — присвоить $a_r \leftarrow 0$.

W4. [Случай $[a_1, \ldots, a_r]$ не окончен?] Если $C[a_1, \ldots, a_r] < 3 \cdot 4^{r-1}$, перейти к шагу W6.

W5. [Увеличить r.] Присвоить $r \leftarrow r + 1$ и возвратиться к шагу W3.

W6. [Присвоить U_n.] Увеличить значение $C[a_1, \ldots, a_r]$ на 1, и пусть k будет его новым значением. Присвоить $U_n \leftarrow V_k$, где V_k определено в приведенной выше лемме T.

W7. [Увеличение n.] Увеличить n на 1 и возвратиться к шагу W2. ▮

Строго говоря, это не алгоритм, так как он не заканчивается, но мы, конечно, слегка преобразуем процедуру, чтобы он останавливал работу, когда n достигает заданного значения. Чтобы понять идею построения, выполните алгоритм вручную, заменяя число $3 \cdot 4^{r-1}$ шага W4 значением 2^r.

Алгоритм W не предназначен для получения случайных чисел на практике. Имеется в виду, что он имеет только теоретическое значение.

Теорема W. *Пусть $\langle U_n \rangle$ — последовательность рациональных чисел, определенная алгоритмом W, и пусть k — положительное целое число. Если подпоследовательность $\langle U_n \rangle \mathcal{R}_k$ бесконечна, то она 1-распределена.*

Доказательство. Пусть $A[a_1, \ldots, a_r]$ означает подпоследовательность $\langle U_n \rangle$ (возможно пустую), включающую те элементы U_n, которые для всех $j \le r$ принадлежат подпоследовательности $\langle U_n \rangle \mathcal{R}_j$, если $a_j = 1$, и не принадлежат подпоследовательности $\langle U_n \rangle \mathcal{R}_j$, если $a_j = 0$.

Достаточно доказать для всех $r \ge 1$ и всех пар двоичных чисел $a_1 \ldots a_r$ и $b_1 \ldots b_r$, что $\Pr(U_n \in I_{b_1 \ldots b_r}) = 2^{-r}$ по отношению к подпоследовательности $A[a_1, \ldots, a_r]$ всякий раз, когда последняя бесконечна (см. равенство (30)). Если $r \ge k$, для бесконечной последовательности $\langle U_n \rangle \mathcal{R}_k$ существует конечное объединение непересекающихся подпоследовательностей $A[a_1, \ldots, a_r]$ для $a_k = 1$ и $a_j = 0$ или 1 для $1 \le j \le r$, $j \ne k$. Из этого следует, что $\Pr(U_n \in I_{b_1 \ldots b_r}) = 2^{-r}$ по отношению к $\langle U_n \rangle \mathcal{R}_k$ (см. упр. 33). Этого достаточно, чтобы показать, что согласно теореме A последовательность 1-распределена.

Пусть $B[a_1, \ldots, a_r]$ означает подпоследовательность $\langle U_n \rangle$ для тех n, в которых $C[a_1, \ldots, a_r]$ увеличивается на единицу на шаге W6 алгоритма. Согласно алгоритму $B[a_1, \ldots, a_r]$ — конечная последовательность с самое большее $3 \cdot 4^{r-1}$ элементами. За исключением конечного числа, все члены $A[a_1, \ldots, a_r]$ являются членами подпоследовательностей $B[a_1, \ldots, a_r, \ldots, a_t]$, где $a_j = 0$ или 1 для $r < j \le t$.

Предположим, что $A[a_1, \ldots, a_r]$ бесконечна, и пусть $A[a_1, \ldots, a_r] = \langle U_{s_n} \rangle$, где $s_0 < s_1 < s_2 < \cdots$. Если N — большое целое число с $4^r \le 4^q < N \le 4^{q+1}$, логически вытекает, что число значений $k < N$, для которых U_{s_k} принадлежит $I_{b_1 \ldots b_r}$, равно (за исключением конечного множества элементов начальных подпоследовательностей)

$$\nu(N) = \nu(N_1) + \cdots + \nu(N_m).$$

Здесь m — число перечисленных выше подпоследовательностей $B[a_1, \ldots, a_t]$, в которых U_{s_k} появляется для некоторых $k < N$, N_j — число значений k с U_{s_k} в соответствующей подпоследовательности и $\nu(N_j)$ — число таких значений, которые также принадлежат $I_{b_1 \ldots b_r}$. Значит, согласно лемме T

$$\begin{aligned} \left|\nu(N) - 2^{-r}N\right| &= \left|\nu(N_1) - 2^{-r}N_1 + \cdots + \nu(N_m) - 2^{-r}N_m\right| \\ &\le \left|\nu(N_1) - 2^{-r}N_1\right| + \cdots + \left|\nu(N_m) - 2^{-r}N_m\right| \\ &\le m \le 1 + 2 + 4 + \cdots + 2^{q-r+1} < 2^{q+1}. \end{aligned}$$

Неравенство для m следует здесь из того, что согласно сделанному выбору N элемент U_{s_N} принадлежит $B[a_1, \ldots, a_t]$ для некоторых $t \le q + 1$.

Мы доказали, что $|\nu(N)/N - 2^{-r}| \le 2^{q+1}/N < 2/\sqrt{N}$. ∎

Чтобы показать окончательно, что существуют последовательности, удовлетворяющие определению R5, сначала заметим, что, если $\langle U_n \rangle$ — $[0\,..\,1)$-последовательность рациональных чисел и если $\mathcal{R}$ — исчисляемое правило подпоследовательностей для b-ичной последовательности, $\mathcal{R}$ можно преобразовать в исчисляемое правило подпоследовательности $\mathcal{R}'$ для $\langle U_n \rangle$, полагая $f'_n(x_1, \ldots, x_n)$ в $\mathcal{R}'$ равным $f_n(\lfloor bx_1 \rfloor, \ldots, \lfloor bx_n \rfloor)$ в $\mathcal{R}$. Если $[0\,..\,1)$-последовательность $\langle U_n \rangle \mathcal{R}'$ равнораспределена, значит, существует b-ичная последовательность $\langle \lfloor bU_n \rfloor \rangle \mathcal{R}$. Сейчас множество всех исчисляемых правил подпоследовательностей для b-ичных последовательностей для всех значений b является счетным (так как возможно только счетное множество эффективных алгоритмов). Значит, они могут образовать

некоторую последовательность $\mathcal{R}_1$, $\mathcal{R}_2$, ...; таким образом, алгоритм W задает $[0..1)$-последовательность, случайную в смысле определения R5.

Это приводит к возникновению в некоторой степени парадоксальной ситуации. Как уже упоминалось, не существует эффективного алгоритма для задания последовательности, удовлетворяющей определению R4. По тем же соображениям неэффективен алгоритм, который задает последовательность, удовлетворяющую определению R5. Доказательство существования такой случайной последовательности неизбежно неконструктивно. Как тогда алгоритм W может построить такую последовательность?

Здесь нет противоречия, есть только заминка, связанная с тем фактом, что множество всех эффективных алгоритмов не может быть пронумеровано эффективным алгоритмом. Другими словами, не существует эффективного алгоритма выбора j-го исчислимого правила подпоследовательности $\mathcal{R}_j$, поскольку нет эффективного алгоритма определения того, заканчивается ли данный вычислительный метод. Но важные большие классы алгоритмов *можно* систематически перенумеровывать. Так, например, в алгоритме W показано, что с помощью эффективного алгоритма можно построить последовательность, удовлетворяющую определению R5, если ограничиться рассмотрением "примитивных рекуррентных" правил подпоследовательностей.

Преобразовав шаг W6 алгоритма W (присвоив $U_n \leftarrow V_{k+t}$ вместо V_k, где t — любое неотрицательное целое число, зависящее от a_1, ..., a_r), можно показать, что существует *несчетное* множество $[0..1)$-последовательностей, удовлетворяющих определению R5.

В приведенной ниже теореме дан другой метод доказательства существования несчетного множества случайных последовательностей, который использует менее прямые доводы, основанные на теории меры, даже если применять строгое определение случайной последовательности R6.

Теорема М. *Пусть действительное число* x, $0 \le x < 1$, *соответствует двоичной последовательности* $\langle X_n \rangle$, *если* $(0.X_0X_1\ldots)_2$ *является двоичным представлением* x. *При этом почти все* x *соответствуют случайным в смысле определения R6 последовательностям.* (Другим словами, множество всех действительных чисел x, соответствующих неслучайным по определению R6 двоичным последовательностям, имеет меру нуль.)

Доказательство. Пусть $\mathcal{S}$ — эффективный алгоритм, определяющий бесконечную последовательность различных неотрицательных чисел $\langle s_n \rangle$, где выбор s_n зависит только от n и X_{s_k} для $0 \le k < n$, и пусть $\mathcal{R}$ — исчислимое правило подпоследовательности. Тогда любая двоичная последовательность $\langle X_n \rangle$ приводит к подпоследовательности $\langle X_{s_n} \rangle \mathcal{R}$ и по определению R6 эта подпоследовательность должна быть либо конечной, либо 1-распределенной. Достаточно доказать, что *для фиксированных* $\mathcal{R}$ *и* $\mathcal{S}$ *множество* $N(\mathcal{R}, \mathcal{S})$ *всех действительных* x, *соответствующих* $\langle X_n \rangle$, *такое, что* $\langle X_{s_n} \rangle \mathcal{R}$ *бесконечна и не* 1*-распределена, имеет меру нуль.* Для x существует неслучайное двоичное представление тогда и только тогда, когда x принадлежит $\bigcup N(\mathcal{R}, \mathcal{S})$, взятому по счетному множеству выборов $\mathcal{R}$ и $\mathcal{S}$.

Следовательно, пусть $\mathcal{R}, \mathcal{S}$ фиксированы. Рассмотрим множество $T(a_1a_2\ldots a_r)$, определенное для всех двоичных чисел $a_1a_2\ldots a_r$ как множество всех x, соответству-

ющих $\langle X_n \rangle$, такое, что $\langle X_{s_n} \rangle \mathcal{R}$ имеет $\geq r$ элементов, из которых первые r элементов равны $a_1, a_2, \ldots, a_r$ соответственно. Первым результатом будет неравенство

$$T(a_1 a_2 \ldots a_r) \text{ имеет меру } \leq 2^{-r}. \tag{32}$$

Доказательство начнем с замечания, что $T(a_1 a_2 \ldots a_r)$ является измеримым множеством: каждый элемент $T(a_1 a_2 \ldots a_r)$ — действительное число $x = (0.X_0 X_1 \ldots)_2$, для которого существует такое целое число m, что алгоритм $\mathcal{S}$ определяет различные значения $s_0, s_1, \ldots, s_m$, и правило $\mathcal{R}$ определяет подпоследовательность $X_{s_0}, X_{s_1}, \ldots, X_{s_m}$, такую, что X_{s_m} является r-м элементом этой подпоследовательности. Множество всех действительных $y = (0.Y_0 Y_1 \ldots)_2$, таких, что $Y_{s_k} = X_{s_k}$ для $0 \leq k \leq m$, также принадлежит $T(a_1 a_2 \ldots a_r)$ и является измеримым множеством, состоящим из конечного объединения двоичных подынтервалов $I_{b_1 \ldots b_t}$. Поскольку существует только счетное множество таких двоичных интервалов, то $T(a_1 a_2 \ldots a_r)$ — счетное объединение двоичных интервалов, и оно, следовательно, измеримо. Более того, данный довод может быть распространен, чтобы показать, что мера $T(a_1 \ldots a_{r-1} 0)$ равна мере $T(a_1 \ldots a_{r-1} 1)$, так как последнее множество является объединением двоичных интервалов, полученных из предшествующей рекуррентной формулы $Y_{s_k} = X_{s_k}$ для $0 \leq k < m$ и $Y_{s_m} \neq X_{s_m}$. Поскольку

$$T(a_1 \ldots a_{r-1} 0) \cup T(a_1 \ldots a_{r-1} 1) \subseteq T(a_1 \ldots a_{r-1}),$$

мера $T(a_1 a_2 \ldots a_r)$ равна самое большее половине меры $T(a_1 \ldots a_{r-1})$. Неравенство (32) теперь легко получить индукцией по r.

Сейчас, когда (32) установлено, осталось, по существу, показать, что двоичное представление почти всех действительных чисел равнораспределено. Пусть для $0 < \epsilon < 1$ $B(r, \epsilon)$ — это $\bigcup T(a_1 \ldots a_r)$, где объединение берется по всем двоичным строкам $a_1 \ldots a_r$, для которых число единиц $\nu(r)$ среди $a_1 \ldots a_r$ удовлетворяет неравенству

$$\left| \nu(r) - \tfrac{1}{2} r \right| \geq \epsilon r.$$

Число таких двоичных строк равно $C(r, \epsilon) = \sum \binom{r}{k}$, и суммирование выполняется по всем значениям k с $|k - \frac{1}{2} r| \geq \epsilon r$. В упр. 1.2.10–21 доказано, что $C(r, \epsilon) \leq 2^{r+1} e^{-\epsilon^2 r}$, отсюда согласно (32)

$$B(r, \epsilon) \text{ имеет меру } \leq 2^{-r} C(r, \epsilon) \leq 2 e^{-\epsilon^2 r}. \tag{33}$$

На следующем шаге определим

$$B^*(r, \epsilon) = B(r, \epsilon) \cup B(r+1, \epsilon) \cup B(r+2, \epsilon) \cup \cdots.$$

Мера $B^*(r, \epsilon)$ равна самое большее $\sum_{k \geq r} 2 e^{-\epsilon^2 k}$ и является остатком сходящегося ряда, так что

$$\lim_{r \to \infty} \big(\text{мера } B^*(r, \epsilon) \big) = 0. \tag{34}$$

Теперь, если x — действительное число, двоичное разложение $(0.X_0 X_1 \ldots)_2$ которого приводит к бесконечной не 1-распределенной последовательности $\langle X_{s_n} \rangle \mathcal{R}$, и если $\nu(r)$ обозначает число 1 в первых r элементах последней последовательности, то

$$\left| \nu(r)/r - \tfrac{1}{2} \right| \geq \epsilon$$

для некоторого $\epsilon > 0$ и бесконечного множества r. Это означает, что x принадлежит $B^*(r,\epsilon)$ для всех r. И наконец, находим, что

$$N(\mathcal{R},\mathcal{S}) = \bigcup_{t\ge 2} \bigcap_{r\ge 1} B^*(r,1/t).$$

Согласно (34) $\bigcap_{r\ge 1} B^*(r,1/t)$ имеет меру нуль для всех t. Следовательно, $N(\mathcal{R},\mathcal{S})$ имеет меру нуль. ▮

Основываясь на факте существования *двоичных* последовательностей, удовлетворяющих определению R6, можно показать существование $[0\,..\,1)$ случайных в этом смысле последовательностей (подробности — в упр. 36). Состоятельность определения R6 в связи с этим установлена.

E. Случайные конечные последовательности. Доводы, приведенные выше, показывают, что невозможно определить понятие случайности конечной последовательности: заданная конечная последовательность так же вероятна, как и любая другая. До сих пор почти каждый согласен, что последовательность 011101001 "более случайна", чем 101010101, и последняя последовательность "более случайна", чем 000000000. Хотя верно, что истинно случайные последовательности проявляют локально неслучайное поведение, мы предполагаем такое поведение только у длинной конечной последовательности, а не у короткой.

Предлагались различные способы определения случайности конечной последовательности, но здесь будут сделаны только наброски нескольких идей. Для простоты ограничимся рассмотрением b-ичных последовательностей.

Пусть задана b-ичная последовательность X_0, X_1, ..., X_{N-1}. Можно сказать, что

$$\Pr\bigl(S(n)\bigr) \approx p, \qquad \text{если } \bigl|\nu(N)/N - p\bigr| \le 1/\sqrt{N}, \tag{35}$$

где $\nu(n)$ — величина, появившаяся в определении A в начале раздела. Приведенную выше последовательность можно назвать k-распределенной, если

$$\Pr(X_n X_{n+1} \ldots X_{n+k-1} = x_1 x_2 \ldots x_k) \approx 1/b^k \tag{36}$$

для всех b-ичных чисел $x_1 x_2 \ldots x_k$. (Ср. с определением D. К сожалению, последовательность может быть k-распределенной согласно этому новому определению, когда она не $(k-1)$-распределена.)

Сейчас можно дать следующее аналогичное определению R1 определение случайности.

Определение Q1. *b-ичная последовательность длины N случайна, если она k-распределена (в вышеприведенном смысле) для всех положительных целых чисел $k \le \log_b N$.*

Например, согласно этому определению существует 178 неслучайных двоичных последовательностей длины 11,

```
00000001111  10000000111  11000000011  11100000001  11110000000
00000001110  10000000110  11000000010  11100000000  11010000000
00000001101  10000000101  11000000001  10100000001  10110000000 ,
00000001011  10000000011  01000000011  01100000001  01110000000
00000000111
```

плюс 01010101010 и все последовательности с девятью или более нулями, плюс все последовательности, полученные из предшествующих последовательностей, если единицы и нули поменять местами.

Подобным образом можно сформулировать определение, аналогичное определению R6, для конечной последовательности. Пусть **A** — множество алгоритмов, каждый из которых является процедурой выделения и выбора, дающей подпоследовательность $\langle X_{s_n}\rangle\mathcal{R}$, как в доказательстве теоремы M.

Определение Q2. *b-ичная последовательность X_0, X_1, ..., X_{N-1} является (n,ϵ)-случайной относительно множества алгоритмов* **A**, *если для каждой подпоследовательности X_{t_1}, X_{t_2}, ..., X_{t_m}, определенной алгоритмом* **A**, *мы имеем либо $m < n$, либо*

$$\left|\frac{1}{m}\nu_a(X_{t_1},\ldots,X_{t_m}) - \frac{1}{b}\right| \le \epsilon \qquad \text{для } 0 \le a < b.$$

Здесь $\nu_a(x_1,\ldots,x_m)$ — количество чисел a в последовательности x_1, ..., x_m.

(Другими словами, каждая достаточно длинная подпоследовательность, определенная алгоритмом из множества **A**, должна быть приближенно равнораспределенной.) Основной идеей в этом случае является предположение, что **A** — множество "простых" алгоритмов и количество (и сложность) алгоритмов в **A** может увеличиваться при росте N.

В качестве примера определения Q2 рассмотрим двоичную последовательность. Пусть **A** состоит из следующих четырех алгоритмов.

a) Взять всю последовательность.
b) Взять нечетные члены последовательности, начиная с первого.
c) Взять члены последовательности, следующие за нулем.
d) Взять члены последовательности, следующие за единицей.

Сейчас последовательность X_0, X_1, ..., X_7 является $(4,\frac{1}{8})$-случайной относительно **A**, если:

по (a) $\left|\frac{1}{8}(X_0+X_1+\cdots+X_7)-\frac{1}{2}\right| \le \frac{1}{8}$, т. е. если последовательность содержит 3, 4 или 5 единиц;

по (b) $\left|\frac{1}{4}(X_0+X_2+X_4+X_6)-\frac{1}{2}\right| \le \frac{1}{8}$, т. е. если последовательность содержит точно 2 единицы на четной позиции;

по (c) существуют три возможности в зависимости от того, как много нулей занимают позиции X_0, ..., X_6: если здесь 2 или 3 нуля, то нет необходимости в проверке (так как $n = 4$); если 4 нуля, то за ними должны следовать 2 нуля и 2 единицы; если 5 нулей, то за ними должны следовать 2 единицы и 3 нуля;

по (d) мы получим условия, подобные условиям в (c).

Оказывается, что только следующие двоичные последовательности длины 8 являются $(4, \frac{1}{8})$-случайными в соответствии с этими правилами,

00001011	00101001	01001110	01101000
00011010	00101100	01011011	01101100
00011011	00110010	01011110	01101101
00100011	00110011	01100010	01110010 '
00100110	00110110	01100011	01110110
00100111	00111001	01100110	

плюс те, которые получены из этих, если единицы и нули поменять местами.

Ясно, что множество алгоритмов можно сделать таким большим, что не будет последовательностей, не удовлетворяющих определению, когда n и ϵ малы. А. Н. Колмогоров доказал, что (n, ϵ)-случайная двоичная последовательность всегда *будет* существовать для любого заданного N, если число алгоритмов в **A** не превышает

$$\tfrac{1}{2} e^{2n\epsilon^2(1-\epsilon)}. \tag{37}$$

Этот результат не достаточно строгий, чтобы показать, что последовательности, удовлетворяющие определению Q1, существуют, но последние могут быть эффективно построены с использованием процедуры Риса (Rees) из упр. 3.2.2–21. Обобщенный спектральный критерий, основанный на дискретном преобразовании Фурье, можно использовать для проверки, насколько последовательность соответствует определению Q1 [см. A. Compagner, *Physical Rev.* **E52** (1995), 5634–5645].

Другие интересные подходы к определению случайности приведены Пером Мартин-Лёфом (Per Martin-Löf, *Information and Control* **9** (1966), 602–619). Пусть задана конечная b-ичная последовательность $X_1, \ldots, X_N$, $l(X_1, \ldots, X_N)$ — длина самой короткой программы Тьюринга, которая генерирует эту последовательность. (Вместо этого можно использовать другие классы эффективных алгоритмов, например такие, которые обсуждались в разделе 1.1.) Тогда $l(X_1, \ldots, X_N)$ — мера "хаотичности" последовательности и это понятие можно отождествить со случайностью. Последовательности длины N, максимизирующие $l(X_1, \ldots, X_N)$, можно называть случайными. (С практической точки зрения генерирование случайного числа компьютером — это, конечно, наихудшее определение "случайности", какое можно себе представить!)

По существу, почти в то же время такое определение случайности независимо предложил Г. Чайтин (G. Chaitin, *JACM* **16** (1969), 145–159.) Интересно отметить, что хотя в данных определениях не упоминается о свойствах равнораспределенности, как в наших определениях, Мартин-Лёф и Чайтин доказали, что случайные последовательности этого вида также имеют ожидаемые свойства равнораспределенности. На самом деле Мартин-Лёф продемонстрировал, что такие последовательности удовлетворяют *всем* вычислимым статистическим критериям случайности в соответствующем смысле.

Дополнительную информацию об определении случайной конечной последовательности можно найти в следующих работах: Звонкин А. К. и Левин Л. А. *Успехи мат. наук* **25**, 6 (Ноябрь, 1970), 85–127; Левин Л. А. *Докл. Акад. наук СССР* **212** (1973), 548–550; Левин Л. А. *Информация и контроль* **61** (1984), 15–37.

F. Псевдослучайные числа. С теоретической точки зрения утешительно знать, что существуют случайные конечные последовательности с разными особенностями, но такие теоремы не дают ответов на вопросы, с которыми сталкиваются в действительности программисты. Недавние исследования привели к более практичной теории, основанной на изучении *множеств* конечных последовательностей. Точнее, рассмотрим *мультимножества*, в которых последовательности могут появляться более одного раза.

Пусть S — мультимножество, содержащее двоичные строки длины N; назовем S *N-источником.* Пусть $\$_N$ означает определенный N-источник, содержащий все 2^N возможных N-двоичных строк. Каждый элемент S представляет последовательность, которую можно использовать в качестве источника псевдослучайных двоичных разрядов, выбор различных "начальных" значений приводит к различным элементам S. Например, возможно такое множество S

$$\{B_1B_2\ldots B_N \mid B_j \text{ старший значащий двоичный разряд } X_j\} \tag{38}$$

в линейной конгруэнтной последовательности, определенной равенством $X_{j+1} = (aX_j + c) \bmod 2^e$, в котором существует одна строка $B_1B_2\ldots B_N$ для каждого из 2^e начальных значений X_0.

Основной идеей псевдослучайных последовательностей, как будет показано в этой главе, является получение N двоичных разрядов, появляющихся случайно, несмотря на то что мы используем лишь несколько "истинно случайных" двоичных разрядов, когда выбираем начальное значение. В только что рассмотренном примере понадобилось e истинно случайных двоичных разрядов для выбора X_0. Вообще, для использования отбирается $\lg|S|$ из S истинно случайных двоичных разрядов, после чего процедура становится детерминированной. Если $N = 10^6$ и $|S| = 2^{32}$, получаем более 30 000 "кажущихся случайными" двоичных разрядов из выбранного истинно случайного двоичного разряда. При $\$_N$ вместо S мы не получим такого большого числа "случайных разрядов", поскольку $\lg|\$_N| = N$.

Что означает "кажущихся случайными"? Э. Ч. Яо (A. C. Yao) в 1982 году предложил хорошее определение: рассмотрим любой алгоритм A, который при применении к строке двоичных разрядов $B = B_1\ldots B_N$ выдает значение $A(B) = 0$ или 1. Можно рассматривать A как критерий случайности, например алгоритм A может вычислить распределение серий последовательных нулей и единиц, выдавая на выходе единицу, если длины серий значительно отличаются для ожидаемого распределения. Что бы ни делал A, вероятность $P(A, S)$ можно рассматривать как вероятность того, что $A(B) = 1$, когда B — случайно выбранный элемент из S, и можно сравнивать с вероятностью $P(A, \$_N)$ того, что $A(B) = 1$, когда B — истинно случайная строка двоичных разрядов длины N. Если $P(A, S)$ будет очень близким к $P(A, \$_N)$ для всех статистических критериев A, то мы ничего не сможем сказать о различии между последовательностями S и истинно случайными двоичными последовательностями.

Определение Р. *Мы говорим, что N-источник S проходит статистический критерий A с допустимым отклонением ϵ, если $\big|P(A, S) - P(A, \$_N)\big| < \epsilon$. Критерий "проваливается", если $\big|P(A, S) - P(A, \$_N)\big| \geq \epsilon$.*

Нет необходимости в том, чтобы алгоритм A задавали статистики. *Любой* алгоритм можно рассматривать как статистический критерий случайности согласно определению P. Мы позволяем A бросать монеты (т. е. использовать истинно случайные двоичные разряды), а также выполнять вычисления. Единственное требование — A должен выдавать на выходе 0 или 1.

Конечно, в действительности существуют другие требования: мы утверждаем, что алгоритм A должен давать результат на выходе за разумное время, по крайней мере в среднем. Нам не интересен алгоритм, который работает много лет, потому что мы никогда не заметим какого-нибудь несоответствия между S и $\$_N$, если наш компьютер не сможет обнаружить их в течение нашей жизни. Последовательности S содержат только $\lg|S|$ двоичных разрядов информации, так что, несомненно, существуют алгоритмы, которые в конечном счете обнаружат избыточность. Но ведь нас интересует, как долго S будет проходить все реально имеющие значение критерии.

Эти качественные идеи можно выразить, как мы сейчас увидим, в количественной форме. Такая теория весьма тонкая, но она достаточно красивая и важная, так что читатель, нашедший время внимательно изучить детали, будет вознагражден.

В последующем обсуждении *время выполнения* $T(A)$ алгоритмом A на N двоичных строк определяется как максимальное ожидаемое число шагов, необходимых для выхода $A(B)$, максимум берется по всем $B \in \$_N$, ожидаемое число является средним по распределению, соответствующему подбрасыванию монеты.

Первый шаг количественного анализа — показать, что можно ограничиться критерием очень специального вида. Пусть A_k — алгоритм, зависящий только от первых k двоичных разрядов во входной строке $B = B_1 \ldots B_N$, где $0 \le k < N$, и пусть $A_k^P(B) = \big(A_k(B) + B_{k+1} + 1\big) \bmod 2$. Тогда A_k^P выводит 1 тогда и только тогда, когда A_k успешно предсказало B_{k+1}; назовем A_k^P критерием *прогноза*.

Лемма P1. *Пусть S — N-источник. Если S не проходит критерий A с допустимым отклонением ϵ, то существует целое число $k \in \{0, 1, \ldots, N-1\}$ и критерий прогноза A_k^P с $T(A_k^P) \le T(A) + O(N)$, такой, что S не проходит A_k^P с допустимым отклонением ϵ/N.*

Доказательство. Дополнительно при необходимости можно предположить, что $P(A,S) - P(A,\$_N) \ge \epsilon$. Рассмотрим алгоритмы F_k, которые начинают подбрасывать $N-k$ монет, и заменим $B_{k+1} \ldots B_N$ случайными двоичными разрядами $B'_{k+1} \ldots B'_N$ до выполнения A. Алгоритм F_N такой же, как и A, в то время как F_0 действует на S, как A действует на $\$_N$. Пусть $p_k = P(F_k, S)$. Поскольку $\sum_{k=0}^{N-1} (p_{k+1} - p_k) = p_N - p_0 = P(A,S) - P(A,\$_N) \ge \epsilon$, существуют k, такие, что $p_{k+1} - p_k \ge \epsilon/N$.

Пусть A_k^P — алгоритм, выполняющий вычисления F_k и предсказывающий значение $(F_k(B) + B'_{k+1} + 1) \bmod 2$. Другими словами, он выводит

$$A_k^P(B) = \big(F_k(B) + B_{k+1} + B'_{k+1}\big) \bmod 2. \tag{39}$$

Внимательный анализ вероятностей показывает, что $P(A_k^P, S) - P(A_k^P, \$_N) = p_{k+1} - p_k$ (см. упр. 40). ∎

Большая часть N-источников S, имеющих практическое преимущество, являются источниками с *симметричным сдвигом* в том смысле, что каждая подстрока

$B_1 \ldots B_k$, $B_2 \ldots B_{k+1}$, $\ldots$, $B_{N-k+1} \ldots B_N$ длины k имеет одно и то же вероятностное распределение. Это выполняется, например, когда S соответствует линейной конгруэнтной последовательности, как в (38). В таких случаях лемму P1 можно улучшить, взяв $k = N - 1$.

Лемма P2. *Если S является N-источником с симметричным сдвигом, который не проходит критерий A с допустимым отклонением ϵ, то существует алгоритм A' с $T(A') \le T(A) + O(N)$, который предсказывает B_N из $B_1 \ldots B_{N-1}$ с вероятностью, равной по крайней мере $\frac{1}{2} + \epsilon/N$.*

Доказательство. Если $P(A, S) > P(A, \$_N)$, пусть A' есть A_k^P в доказательстве леммы P1, только примененное к $B_{N-k} \ldots B_{N-1} 0 \ldots 0$ вместо $B_1 \ldots B_N$. Тогда A' в среднем ведет себя так же из-за симметричного сдвига. Если $P(A, S) < P(A, \$_N)$, пусть A' есть $1 - A_k^P$ в том же виде. Ясно, что $P(A', \$_N) = \frac{1}{2}$. ▮

Введем более жесткие ограничения на S. Предположим, что каждая из последовательностей $B_1 B_2 \ldots B_N$ имеет вид $f\big(g(X_0)\big) f\big(g(g(X_0))\big) \ldots f\big(g^{[N]}(X_0)\big)$, где X_0 — это упорядочение некоторого множества X, g является перестановкой X и $f(x)$ равно 0 или 1 для всех $x \in X$. Наш линейный конгруэнтный пример удовлетворяет этим ограничениям с $X = \{0, 1, \ldots, 2^e - 1\}$, $g(x) = (ax + c) \bmod 2^e$ и $f(x) =$ старший значащий двоичный разряд x. Такие N-источники назовем *итеративными*.

Лемма P3. *Если S — итеративный N-источник, который не удовлетворяет критерию A с допустимым отклонением ϵ, то существует алгоритм A' с $T(A') \le T(A) + O(N)$, предсказывающий B_1 по $B_2 \ldots B_N$ по крайней мере с вероятностью $\frac{1}{2} + \epsilon/N$.*

Доказательство. Итеративный N-источник является источником с симметричным сдвигом. Значит, он является своим отражением $S^R = \{B_N \ldots B_1 \mid B_1 \ldots B_N \in S\}$. Следовательно, лемма P2 применима к S^R. ▮

Перестановка $g(x) = (ax + c) \bmod 2^e$ легко обратима в том смысле, что x можно определить через $g(x)$ всякий раз, когда a нечетное. Однако множества легко вычисляемых функций перестановок являются "односторонними" (трудно обратимыми), и мы увидим, что это делает их вероятно хорошими источниками псевдослучайных чисел.

Лемма P4. *Пусть S — итеративный N-источник, соответствующий f, g и X. Если S не удовлетворяет критерию A с допустимым отклонением ϵ, то существует алгоритм G, который правильно оценивает $f(x)$ по заданной $g(x)$ с вероятностью $\ge \frac{1}{2} + \epsilon/N$, когда x — случайный элемент из X. Время вычисления $T(G)$ будет не более чем $T(A) + O(N)\big(T(f) + T(g)\big)$.*

Доказательство. Зададим $y = g(x)$. Требуемый алгоритм G вычисляет $B_2 = f\big(g(x)\big)$, $B_3 = f\big(g(g(x))\big)$, $\ldots$, $B_N = f\big(g^{[N-1]}(x)\big)$ и применяет алгоритм A' леммы P3. Он приближенно оценивает $f(x) = B_1$ с вероятностью $\ge \frac{1}{2} + \epsilon/N$, так как g является перестановкой X, и $B_1 \ldots B_N$ — элемент S, соответствующий начальному значению X_0, для которого выполняется $g(X_0) = x$. ▮

Для того чтобы использовать лемму P4, нужно усилить способность приближенно оценивать один двоичный разряд $f(x)$ для того, чтобы уметь оценивать x только по значению $g(x)$. Для этого существует только тонкий общий способ —

использовать свойства булевых функций, если расширить S так, что появится потребность приближенно оценивать множество различных функций $f(x)$. (Однако метод является несколько техническим. Так, при первом чтении следует, возможно, перейти к теореме G, прежде чем внимательно рассматривать следующие ниже детали.)

Предположим, $G(z_1 \dots z_R)$ — бинарнозначная функция (принимающая значения 0 или 1) на строках из R двоичных разрядов, которая хорошо приближает функцию вида $f(z_1 \dots z_R) = (x_1 z_1 + \cdots + x_R z_R) \bmod 2$ для некоторого фиксированного $x = x_1 \dots x_R$. Удобно измерять, насколько это приближение успешно, вычисляя среднее значение

$$s = \mathrm{E}\big((-1)^{G(z_1 \dots z_R) + x_1 z_1 + \cdots + x_R z_R}\big), \tag{40}$$

где усреднение берется по всем возможным значениям $z_1 \dots z_R$. Это сумма правильных оценок минус неправильные оценки деленная на 2^R; так, если p — вероятность того, что G правильно, то получим $s = p - (1-p)$ или $p = \frac{1}{2} + \frac{1}{2}s$.

Например, предположим, что $R = 4$ и $G(z_1 z_2 z_3 z_4) = [z_1 \neq z_2][z_3 + z_4 < 2]$. Функция дает хорошую оценку $s = \frac{3}{4}$ (и $p = \frac{7}{8}$), если $x = 1100$, так как она равна $x \cdot z \bmod 2 = (z_1 + z_2) \bmod 2$ для всех строк z, содержащих 4 двоичных разряда, исключая 0111 или 1011. Она также дает хорошую оценку $\frac{1}{4}$, когда $x = 0000$, 0011, 1101 или 1110, так что существует пять вероятных возможностей для x. Остальные одиннадцать x_k дают $s \leq 0$.

Следующий алгоритм магически находит x в большинстве случаев, когда G успешно оценивает в только что описанном смысле. Точнее, алгоритм строит короткий перечень элементов, имеющих хорошую возможность содержать x.

Алгоритм L (*Усиление линейных оценок*). Пусть задана бинарнозначная функция $G(z_1 \dots z_R)$ и положительное целое число k. Этот алгоритм дает таблицу 2^k двоичных последовательностей $x = x_1 \dots x_R$ с таким свойством, что x, вероятно, находится в этой таблице, когда $G(z_1 \dots z_R)$ является хорошим приближением функции $(x_1 z_1 + \cdots + x_R z_R) \bmod 2$.

L1. [Построение случайной матрицы.] Генерировать случайные двоичные разряды B_{ij} для $1 \leq i \leq k$ и $1 \leq j \leq R$.

L2. [Подсчет знаков.] Для $1 \leq i \leq R$ и для всех двоичных разрядов строк $b = b_1 \dots b_k$ вычислить

$$h_i(b) = \sum_{c \neq 0} (-1)^{b \cdot c + G(cB + e_i)}, \tag{41}$$

где e_i — строка, содержащая R двоичных разрядов, $0 \dots 010 \dots 0$ с 1 на позиции i и где cB — строка $d_1 \dots d_R$ с $d_j = (B_{1j} c_1 + \cdots + B_{kj} c_k) \bmod 2$. (Другими словами, двоичный вектор $c_1 \dots c_k$ является кратным двоичной матрице B размера $k \times R$.) Сумма взята по всем $2^k - 1$ строкам двоичных разрядов, $c_1 \dots c_k \neq 0 \dots 0$. Для каждого i можно оценить $h_i(b)$ $k \cdot 2^k$ сложениями и вычитаниями с помощью метода Ятеса (Yates) для преобразования Уолша (см. замечание к равенству 4.6.4–(38)).

L3. [Вывод оценок.] Для всех 2^k выборов $b = b_1 \dots b_k$ вывод строки $x(b) = [h_1(b) < 0] \dots [h_R(b) < 0]$. ∎

Для доказательства того, что алгоритм L хорошо работает, необходимо показать, что заданная строка x, вероятно, выводится всякий раз, когда она этого заслуживает. Сначала заметим, что, если заменить G на G', где $G'(z) = (G(z) + z_j) \bmod 2$, начальное $G(z)$ будет хорошим приближением $x \cdot z \bmod 2$ тогда и только тогда, когда новое $G'(z)$ будет хорошим приближением $(x + e_j) \cdot z \bmod 2$, где e_j — единичный вектор-строка, определенная на шаге L2. Кроме того, если применить алгоритм G' вместо G, можно получить

$$h'_i(b) = \sum_{c\neq 0}(-1)^{b\cdot c+G(cB+e_i)+(cB+e_i)\cdot e_j} = (-1)^{\delta_{ij}} h_i\big((b + B_j) \bmod 2\big),$$

где B_j — j-й столбец B. Следовательно, на шаге L3 выводится вектор $x'(b) = x\big((b+B_j) \bmod 2\big) + e_j$ по модулю 2. Поскольку b пробегает все строки, состоящие из k двоичных разрядов, $(b + B_j) \bmod 2$ также пробегает эти строки, следствием чего является дополнение j-м двоичным разрядом каждого x на выходе.

Следовательно, достаточно доказать, что вектор $x = 0 \ldots 0$ можно вывести, как только $G(z)$ хорошо аппроксимирует постоянную функцию 0. В действительности мы покажем, что $x(0 \ldots 0)$ равняется $0 \ldots 0$ на шаге L3 с большой вероятностью всякий раз, когда $G(z)$ с большей вероятностью принимает значение 0, чем 1, и k является достаточно большим. Точнее говоря, условие

$$\sum_{c\neq 0}(-1)^{G(cB+e_i)} > 0$$

выполняется для $1 \le i \le R$ с вероятностью $> \frac{1}{2}$, если $s = \mathrm{E}\big((-1)^{G(z)}\big)$ положительно, где среднее берется по всем 2^R возможным z и если k достаточно велико.

Ключом исследования является утверждение, что для каждого фиксированного $c = c_1 \ldots c_k \neq 0 \ldots 0$ строка $d = cB$ равномерно распределена: каждое значение d появляется с вероятностью $1/2^R$, так как двоичные разряды B случайны. Более того, когда $c \neq c' = c'_1 \ldots c'_k$, строки $d = cB$ и $d' = c'B$ *независимы*: каждое значение пары (d, d') происходит с вероятностью $1/2^{2R}$. Следовательно, можно рассуждать, как при доказательстве неравенства Чебышева: для любого фиксированного i сумма $\sum_{c\neq 0}(-1)^{G(cB+e_i)}$ отрицательна с вероятностью, не большей, чем $1/((2^k - 1)s^2)$. (Подробности содержатся в упр. 42.) Поэтому $R/((2^k - 1)s^2)$ — верхняя грань вероятности того, что $x(0)$ не является нулем на шаге L3.

Теорема G. *Если* $s = \mathrm{E}\big((-1)^{G(z)+x\cdot z}\big) > 0$ *и* $2^k > \lceil 2R/s^2 \rceil$, *то алгоритм L выводит* x *с вероятностью* $\ge \frac{1}{2}$. *Время счета равно* $O(k2^kR)$ *плюс время получения* 2^kR *оценок* G. ▮

Сейчас мы готовы доказать, что последовательность смешанно-квадратичного метода, заданная в 3.2.2–(17), является хорошим источником (псевдо)случайных чисел. Предположим, что $2^{R-1} < M = PQ < 2^R$, где P и Q — простые числа вида $4k+3$, удовлетворяющие неравенствам $2^{(R-2)/2} < P < 2^{(R-1)/2}$, $2^{R/2} < Q < 2^{(R+1)/2}$. Назовем M, состоящее из R двоичных разрядов, *целым числом Блюма*, поскольку на важность таких чисел для криптографии было впервые указано Мануэлем Блюмом (Manuel Blum, *COMPCON* **24** (Spring, 1982), 133–137). Блюм первоначально предложил выбрать P и Q так, чтобы они оба имели $R/2$ двоичных разрядов, но

алгоритм 4.5.4D показал, что лучше выбрать P и Q, как сформулировано здесь: чтобы $Q - P > .29 \times 2^{R/2}$.

Выбрать X_0 наугад среди чисел $0 < X_0 < M$ с $X_0 \perp M$. Пусть также Z — случайная, состоящая из R двоичных разрядов, маска. Можно построить итеративный N-источник S, полагая X множеством всех (x, z, m), которые возможны для (X_0, Z, M) с дополнительным ограничением $x \equiv a^2$ (по модулю m) для некоторых a. Легко показать, что функция $g(x, z, m) = (x^2 \bmod m, z, m)$ — это перестановка X (см., например, упр. 4.5.4–35). Функция $f(x, z, m)$, извлекающая двоичные разряды в этом итеративном источнике, равна $x \cdot z \bmod 2$. Наше начальное значение (X_0, Z, M) не является необходимым в X, но $g(X_0, Z, M)$ имеет равномерное распределение в X, так как точно четыре значения X_0 имеют данный квадрат $X_0^2 \bmod M$.

Теорема Р. *Пусть S — N-источник, который определен смешанно-квадратичным методом согласно модулю, содержащему R двоичных разрядов, и предположим, что S не удовлетворяет некоторому статистическому критерию A с допустимым отклонением $\epsilon \geq 1/2^N$. Тогда можно построить алгоритм F, который найдет множители состоящего из R двоичных разрядов случайного целого числа Блюма $M = PQ$, имеющего вид, описанный выше, с вероятностью по крайней мере $\epsilon/(4N)$ и временем счета $T(F) = O(N^2R^2\epsilon^{-2}T(A) + N^3R^4\epsilon^{-2})$.*

Доказательство. Умножение по модулю M можно осуществить за $O(R^2)$ шагов; следовательно, $T(f) + T(g) = O(R^2)$. Поэтому лемма Р4 утверждает существование оценочного алгоритма G с успешной оценкой ϵ/N и $T(G) \leq T(A) + O(NR^2)$. Построить G по A можно, используя метод из упр. 41. Этот алгоритм G имеет такое свойство, что $s = \mathrm{E}\big((-1)^{G(y,z,m)+z\cdot x}\big) \geq (\frac{1}{2} + \epsilon/N) - (\frac{1}{2} - \epsilon/N) = 2\epsilon/N$, где среднее значение взято по всем $(x, z, m) \in X$ и где $(y, z, m) = g(x, z, m)$.

Требуемый алгоритм F получается следующим образом. Задано случайное число $M = PQ$ с неизвестными P и Q. Алгоритм вычисляет случайное число X_0 между 0 и M и немедленно останавливается с известным разложением, если $\gcd(X_0, M) \neq 1$. В других случаях применяется алгоритм L с $G(z) = G(X_0^2 \bmod M, z, M)$ и $k = \lceil \lg(1 + 2N^2R/\epsilon^2) \rceil$. Если одно из 2^k значений x на его выходе удовлетворяет $x^2 \equiv X_0^2$ (по модулю M), существует 50:50 шансов, что $x \not\equiv \pm X_0$. Тогда $\gcd(X_0 - x, M)$ и $\gcd(X_0 + x, M)$ являются простыми множителями M (см. "SQRT-ящик" Рабина (Rabin) в разделе 4.5.4).

Ясно, что время счета этого алгоритма равно $O(N^2R^2\epsilon^{-2}T(A) + N^3R^4\epsilon^{-2})$, так как $\epsilon \geq 2^{-N}$. Вероятность, что алгоритм достигнет цели в разложении M, можно оценить следующим образом. Пусть $n = |X|/2^R$ — число выборов (x, m) и пусть $s_{xm} = 2^{-R}\sum(-1)^{G(y,z,m)+z\cdot x}$ — суммирование по всем содержащим R двоичных разрядов числам z. Тогда $s = \sum_{x,m} s_{xm}/n \geq 2\epsilon/N$. Пусть t — число таких (x, m), что $s_{xm} \geq \epsilon/N$. Вероятность, что наш алгоритм оперирует подобными парами (x, m), равна

$$\frac{t}{n} \geq \sum_{x,m} [s_{xm} \geq \epsilon/N] \frac{s_{xm}}{n} = \sum_{x,m} \big(1 - [s_{xm} < \epsilon/N]\big) \frac{s_{xm}}{n}$$
$$\geq \frac{2\epsilon}{N} - \sum_{x,m} [s_{xm} < \epsilon/N] \frac{s_{xm}}{n} \geq \frac{\epsilon}{N}.$$

И в таком случае алгоритм по теореме G найдет x с вероятностью $\geq \frac{1}{2}$, поскольку мы имеем $2^k > \lceil 2R/s_{xm}^2 \rceil$. Значит, он находит множитель с вероятностью $\geq \frac{1}{4}$. ▮

Что дает теорема P с практической точки зрения? Наше доказательство показывает, что константы, включенные в O, малы. Предположим, что время счета для разложения на множители равно самое большее $10(N^2R^2\epsilon^{-2}T(A) + N^3R^4\epsilon^{-2})$. Многие известнейшие математики мира работали над проблемой разложения на множители больших чисел, в особенности после того, как в конце 70-х годов было показано, что разложение на множители в высшей степени связано с криптографией. Так как они не могли найти хорошее решение, мы имеем основание считать, что разложение на множители является трудным делом. Следовательно, теорема P показывает, что $T(A)$ должно быть большим для всех алгоритмов, которые обнаруживают неслучайность двоичных разрядов, полученных смешанно-квадратичным методом.

Длительные вычисления удобно измерять в MIP-годах (это число операций, выполняемых за год машиной, которая совершает миллион операций в секунду, т. е. $31{,}556{,}952{,}000{,}000 \approx 3.16 \times 10^{13}$). В 1995 году время разложения на множители числа из 120 десятичных цифр (400 двоичных разрядов) при использовании в высшей степени совершенных алгоритмов было больше 250 MIP-лет. Наиболее оптимистически настроенные исследователи, работающие над разложением на множители, могут удивиться, если алгоритм обнаружит, что требуется всего $\exp\bigl(R^{1/4}(\ln R)^{3/4}\bigr)$ команд, когда $R \to \infty$. Только допустим, что это количество может быть достигнуто для по крайней мере не слишком малых частей целых чисел Блюма M, состоящих из R двоичных разрядов. Тогда можно будет умножить много чисел, состоящих из приблизительно 50 000 двоичных разрядов (15 000 цифр), за 2×10^{25} MIP-лет. Если генерировать $N = 1000$ случайных двоичных разрядов смешанно-квадратичным методом с $R = 50000$ и если предположить, что все алгоритмы достаточно хороши, то умножение по крайней мере $\frac{1}{400000}$ на состоящие из 50 000 двоичных разрядов числа Блюма должно выполняться минимум 2×10^{25} MIP-лет. Из теоремы P следует, что каждое такое множество из 1 000 двоичных разрядов проходит все статистические критерии на случайность, время счета $T(A)$ которых меньше 70 000 MIP-лет: не существует алгоритма A, который мог бы отличить такие двоичные разряды от истинно случайной последовательности с вероятностью $\geq \epsilon = \frac{1}{100}$.

Впечатляет? Нет. Такой результат вряд ли является сюрпризом, так как необходимо точно определить около 150 000 истинно случайных двоичных разрядов, точно начинающихся в смешанно-квадратичном методе с X_0, Z и M, когда $R = 50000$. *Конечно*, можно получить 1 000 случайных двоичных разрядов из такого вклада!

Но вообще, формула

$$T(A) \geq \frac{1}{100000} N^{-2}R^{-2}\exp\bigl(R^{1/4}(\ln R)^{3/4}\bigr) - NR^2$$

справедлива при наших умеренных предположениях, когда $\epsilon = \frac{1}{100}$, NR^2 членов являются незначительными и когда R велико. Положим, $R = 200000$ и $N = 10^{10}$. Тогда смешанно-квадратичным методом получим десять миллиардов псевдослучайных двоичных разрядов из $3R \approx 600000$ истинно случайных двоичных разрядов, проходящих все статистические критерии, которые требуют меньше 7.486×10^{10} MIP-лет, что равно 74.86 гигаMIP-годам. При $N = 10^{13}$ и $R = 333333$ время

вычисления, необходимое для определения статистического смещения, возрастает до 53.5 тераMIP-лет.

Простой псевдослучайный генератор 3.2.2–(16), который избегает случайной маски Z, что также можно показать, проходит все полиномиальные критерии случайности, если разложение на множители трудно осуществить (см. упр. 4.5.4–43). Но известные преобразования гарантируют для методов, которые отчасти слабее смешанно-квадратичного метода, порядок роста $O\big(N^4R\epsilon^{-4}\log(NR\epsilon^{-1})\big)$ по сравнению с $O(N^2R^2\epsilon^{-2})$ теоремы P.

Каждый думает, что не существует алгоритма разложения на множители для чисел, состоящих из R двоичных разрядов, время счета которых равно полиному в степени R. Если это предположение верно в строгом виде, то нельзя будет получить даже $1/R^k$ для целого числа Блюма, состоящего из R двоичных разрядов, за полиномиальное время для любого фиксированного k. Теорема P доказывает, что смешанно-квадратичный метод генерирует псевдослучайные числа, проходящие все полиномиальные критерии случайности.

Сформулируем это другим способом: если генерировать случайные двоичные разряды смешанно-квадратичным методом для подходящим образом выбранных N и R, можно также получить числа, проходящие все разумные статистические критерии, или открыть новый алгоритм разложения на множители.

G. Выводы, история и библиография. Выше были определены различные степени случайности, которыми может обладать последовательность.

Конечная ∞-распределенная последовательность удовлетворяет великому множеству полезных свойств, которыми могут обладать случайные последовательности, и существует огромная теория, посвященная ∞-распределенным последовательностям. (В упражнениях, которые приводятся ниже, развиваются некоторые важные, не упомянутые в разделе, свойства таких последовательностей.) Определение R1 поэтому является подходящей основой для теоретического изучения случайности.

Понятие "∞-распределение b-ичной последовательности" было введено в 1909 году Эмилем Борелем (Emile Borel). Он, по существу, ввел понятие (m,k)-распределенной последовательности и показал, что b-ичное представление почти всех действительных чисел является (m,k)-распределенным для всех m и k. Борель назвал такие числа *нормальными* по отношению к основанию b. Превосходное обсуждение этой темы появилось в его хорошо известной книге *Leçons sur la Théorie des Fonctions*, 2nd edition (1914), 182–216.

Понятие ∞-распределенной последовательности *действительных* чисел, также носящее название *полностью равнораспределенной последовательности*, впервые появилось в заметке Н. М. Коробова (*Доклады Акад. Наук СССР* **62** (1948), 21–22). Коробов и несколько его коллег достаточно широко развили теорию таких последовательностей в ряде статей в течение 50-х годов. Полностью равнораспределенные последовательности независимо изучались Джоэлем Н. Франклиным (Joel N. Franklin, *Math. Comp.* **17** (1963), 28–59) в статье, которая особенно заслуживает внимания, так как она была вдохновлена проблемой генерирования случайных чисел. Книга L. Kuipers and H. Niederreiter, *Uniform Distribution of Sequences* (New York: Wiley, 1974) является чрезвычайно полным источником информации об огромной математической литературе, содержащей k-распределенные последовательности всех видов.

Тем не менее мы увидели, что ∞-распределенные последовательности не обладают достаточным количеством свойств, чтобы их можно было считать совершенно "случайными". Определения R4–R6, приведенные выше, предусматривают дополнительные условия; в частности, определение R6, видимо, было подходящим способом определения понятия бесконечной случайной последовательности. Это точное количественное утверждение, хорошо совпадающее с интуитивным понятием истинной случайности.

Исторически развитие этих определений было стимулировано, главным образом, поисками Р. фон Мизесом (R. von Mises) хорошего определения вероятности. В *Math. Zeitschrift* **5** (1919), 52–99, фон Мизес предложил определение, по духу сходное с определением R5, хотя формулировка была слишком строга (подобно нашему определению R3), так что последовательностей, удовлетворяющих этим условиям, не существует. Многие исследователи отмечали это противоречие, и А. Г. Коуплэнд (A. H. Copeland, *Amer. J. Math.* **50** (1928), 535–552) предлагал ослабить определение фон Мизеса заменой, которую он назвал допустимыми числами (или последовательностями Бернулли). Существует эквивалент ∞-распределенных $[0\,..\,1)$-последовательностей, в которых все входные U_n заменяются 1, если $U_n < p$ или 0 и если $U_n \geq p$ для заданной вероятности p. Так, Коуплэнд, по существу, предложил вернуться к определению R1. Затем Абрахам Вальд (Abraham Wald) показал, что нет необходимости так решительно ослаблять определение фон Мизеса, и предложил заменить счетное множество правилами подпоследовательностей. В важной статье *Ergebnisse eines math. Kolloquiums* **8** (Vienna, 1937), 38–72, Вальд, по существу, доказал теорему W, хотя он сделал ошибочный вывод, что последовательность, построенная алгоритмом W, также удовлетворяет сильным условиям — $\Pr(U_n \in A) =$ мера A для всех измеримых по Лебегу $A \subseteq [0\,..\,1)$. Заметим, что не существует последовательности, которая может удовлетворять этому условию.

Понятие "вычисляемость" играло большую роль на ранней стадии, когда Вальд написал статью и А. Черч (A. Church, *Bull. Amer. Math. Soc.* **46** (1940), 130–135) показал, как точное понятие "эффективный алгоритм" можно присоединить к теории Вальда, делая его определение совершенно строгим. Практически тогда же дополнение к определению R6 было предложено А. Н. Колмогоровым [*Sankhyā* **A25** (1963), 369–376], как и определение Q2 для конечных последовательностей. Другое определение случайности для конечных последовательностей, находящееся где-то между определениями Q1 и Q2, намного раньше сформулировал А. С. Безикович (A. S. Besicovitch, *Math. Zeitschrift* **39** (1934), 146–156).

В публикациях Черча и Колмогорова рассматривались только двоичные последовательности, для которых $\Pr(X_n = 1) = p$ с заданной вероятностью p. В этом разделе анализировалась более общая ситуация, поскольку $[0\,..\,1)$-последовательность, по существу, представляет все p сразу. Определение фон Мизеса-Вальда-Черча еще одним интересным способом усовершенствовал Дж. В. Говард (J. V. Howard, *Zeitschr. für math. Logik und Grundlagen der Math.* **21** (1975), 215–224).

Следующий важный вклад был сделан Дональдом В. Лавлендом (Donald W. Loveland, *Zeitschr. für math. Logik und Grundlagen der Math.* **12** (1966), 279–294), который обсудил определения R4–R6 и несколько других понятий. Лавленд доказал, что существуют R5-случайные последовательности, не удовлетворяющие определению R4. В связи с этим он установил, что необходимо более строгое определение,

такое как R6. На самом деле Лавленд определил простую перестановку $\langle f(n)\rangle$ неотрицательных целых чисел и алгоритм W′, сходный с алгоритмом W, такой, что

$$\overline{\Pr}(U_{f(n)} \ge \tfrac{1}{2}) - \underline{\Pr}(U_{f(n)} \ge \tfrac{1}{2}) \ge \tfrac{1}{2}$$

для каждой R5-случайной последовательности $\langle U_n\rangle$, выдаваемой алгоритмом W′, когда он задан бесконечным множеством правил подпоследовательностей $\mathcal{R}_k$

Хотя определение R6 интуитивно строже определения R4, очевидно, строго доказать это совсем не просто. В течение нескольких лет данный вопрос оставался открытым, поскольку R4 так или иначе включает в себя R6. В конце концов, Томас Герцог (Thomas Herzog) и Джеймс К. Оуингс (мл.) (James C. Owings, Jr.) сумели построить большое семейство последовательностей, удовлетворяющих R4, но не удовлетворяющих R6. [См. *Zeitschr. für math. Logik und Grundlagen der Math.* **22** (1976), 385–389.]

Другую важную статью написал Колмогоров [*Проблемы передачи информации* **1** (1965), 3–11]. В ней он рассмотрел проблему определения "информационного содержимого" последовательности, и эта работа привела к интересному определению Чайтина (Chaitin) и Мартин-Лёфа (Martin-Löf) конечных случайных последовательностей через "хаотичность". [См. *IEEE Trans.* **IT-14** (1968), 662–664.] Их идея может быть также прослежена в работах Р. Дж. Соломонова (R. J. Solomonoff, *Information and Control* **7** (1964), 1–22, 224–254; *IEEE Trans.* **IT-24** (1978), 422–432; *J. Comp. System Sci.* **55** (1997), 73–88).

Обсуждение случайных последовательностей с философской точки зрения можно найти у К. Р. Поппера (K. R. Popper, *The Logic of Scientific Discovery* (London, 1959)); особенно интересно построение на с. 162–163, впервые опубликованное в 1934 году.

Дальнейшие связи между случайными последовательностями и теорией рекурсивных функций исследовались в работе D. W. Loveland, *Trans. Amer. Math. Soc.* **125** (1966), 497–510. См. также работу К.-П. Шнорр (C.-P. Schnorr, *Zeitschr. Wahr. verw. Geb.* **14** (1969), 27–35), нашедшего сильные связи между случайными последовательностями и "категориями меры", которые были определены Л. Э. Я. Брауэром (L. E. J. Brouwer) в 1919 году. В следующей книге Шнорра, *Zufälligkeit und Wahrscheinlichkeit* [*Lecture Notes in Math.* **218** (Berlin: Springer, 1971)], дан подробный обзор всей темы случайности и превосходное вступление к новым публикациям по этой теме. Обзор важнейших усовершенствований в течение следующих двух десятилетий можно найти в книге *An Introduction to Kolmogorov Complexity and Its Applications* (Springer, 1993), Ming Li and Paul M. B. Vitányi.

Основы теории псевдослучайных последовательностей и эффективной информации заложены Мануэлем Блюмом (Manuel Blum), Сильвио Микали (Silvio Micali) и Эндре Яо (Andrew Yao) в работах [*FOCS* **23** (1982), 80–91, 112–117; *SICOMP* **13** (1984), 850–864], в которых построены первые явные последовательности, удовлетворяющие всем возможным статистическим критериям. Блюм и Микали ввели понятие жесткого ядра двоичного разряда, булевой функции f, такой, что $f(x)$ и $g(x)$ легко вычисляются, хотя функция $f(g^{[-1]}(x))$ не вычисляется. В их статье берет начало лемма P4. Леонид Левин развил теорию в работе *Combinatorica* **7** (1987), 357–363. Затем он и Одед Голдрейч (Oded Goldreich) [*STOC* **21** (1989), 25–32] проанализировали такие алгоритмы, как смешанно-квадратичный метод, и

показали, что, используя маску подобным образом, можно получить жесткое ядро во многих случаях. Наконец, Левин в работе *J. Symbolic Logic* **58** (1993), 1102–1103, усовершенствовал методы предыдущей работы, введя алгоритм L и проанализировав его.

Свой вклад в теорию внесли многие авторы — особенно Импаглиаззо (Impagliazzo), Левин, Лаби (Luby) и Хастад (Håstad) [*STOC* **21** (1989), 12–24; **22** (1990), 395–404], которые показали, что псевдослучайные последовательности можно построить из любой однозначной функции. Однако такие результаты здесь не рассматриваются, так как они применяются, главным образом, в сложной абстрактной теории, а не в практическом генерировании случайных чисел. Практическое применение теоретических работ к псевдослучайности впервые эмпирически исследовано в работе P. L'Ecuyer and R. Proulx, *Proc. Winter Simulation Conf.* **22** (1989), 467–476.

Если числа не случайны, то
они по крайней мере в полном беспорядке.

— ДЖОРДЖ МАРСАЛЬЯ (GEORGE MARSAGLIA) (1984)

УПРАЖНЕНИЯ

1. [*10*] Может ли периодическая последовательность быть равнораспределенной?

2. [*10*] Рассмотрите периодическую двоичную последовательность $0, 0, 1, 1, 0, 0, 1, 1, \ldots$. Она 1-, 2- или 3-распределенная?

3. [*M22*] Постройте троичную периодическую 3-распределенную последовательность.

4. [*HM14*] Докажите, что $\Pr(S(n)\text{ и }T(n)) + \Pr(S(n)\text{ или }T(n)) = \Pr(S(n)) + \Pr(T(n))$ для любых двух утверждений $S(n)$ и $T(n)$, предполагая, что по крайней мере три из этих пределов существуют. Например, если последовательность 2-распределена, то можно найти, что

$$\Pr(u_1 \le U_n < v_1 \text{ или } u_2 \le U_{n+1} < v_2) = v_1 - u_1 + v_2 - u_2 - (v_1 - u_1)(v_2 - u_2).$$

▶ **5.** [*HM22*] Пусть $U_n = (2^{\lfloor \lg(n+1) \rfloor}/3) \bmod 1$. Чему равна $\Pr(U_n < \frac{1}{2})$?

6. [*HM23*] Пусть $S_1(n)$, $S_2(n)$, ... — бесконечная последовательность утверждений о совместных непересекающихся событиях, т. е. $S_i(n)$ и $S_j(n)$ не могут выполняться одновременно, если $i \ne j$. Предположим, что $\Pr(S_j(n))$ существует для каждого $j \ge 1$. Покажите, что $\underline{\Pr}(S_j(n)$ выполняется для некоторого $j \ge 1) \ge \sum_{j\ge1} \Pr(S_j(n))$, и приведите пример, показывающий, что равенство может не выполняться.

7. [*HM27*] Пусть $\{S_{ij}(n)\}$ — семейство утверждений, таких, что $\Pr(S_{ij}(n))$ существует для всех $i, j \ge 1$. Предположим, что для всех $n > 0$ $S_{ij}(n)$ выполняется для точно одной пары целых чисел i, j. Если $\sum_{i,j\ge1} \Pr(S_{ij}(n)) = 1$, то следует ли из этого, что "$\Pr(S_{ij}(n)$ выполняется для некоторого $j \ge 1)$" существует для всех $i \ge 1$ и равна $\sum_{j\ge1} \Pr(S_{ij}(n))$?

8. [*M15*] Докажите (13).

9. [*HM20*] Докажите лемму E. [*Указание.* Рассмотрите $\sum_{j=1}^{m}(y_{jn} - \alpha)^2$.]

▶ **10.** [*HM22*] Где в доказательстве теоремы C используется тот факт, что m делит q?

11. [*M10*] Применяя теорему C, докажите, что если последовательность $\langle U_n \rangle$ ∞-распределена, то она является подпоследовательностью $\langle U_{2n} \rangle$.

12. [*HM20*] Покажите, что k-распределенная последовательность удовлетворяет критерию "максимум-k" в следующем смысле: $\Pr(u \le \max(U_n, U_{n+1}, \ldots, U_{n+k-1}) < v) = v^k - u^k$.

▶ **13.** [*HM27*] Покажите, что ∞-распределенная $[0\,..\,1)$-последовательность проходит критерий интервалов в следующем смысле: если $0 \le \alpha < \beta \le 1$ и $p = \beta - \alpha$, пусть $f(0) = 0$ и для $n \ge 1$ пусть $f(n)$ — наименьшее целое число $m > f(n-1)$, такое, что $\alpha \le U_m < \beta$, тогда

$$\Pr(f(n) - f(n-1) = k) \;=\; p(1-p)^{k-1}.$$

14. [*HM25*] Покажите, что ∞-распределенная последовательность проходит критерий монотонности в следующем смысле: если $f(0) = 0$ и если для $n \ge 1$ $f(n)$ — наименьшее целое число $m > f(n-1)$, такое, что $U_{m-1} > U_m$, тогда

$$\Pr(f(n) - f(n-1) = k) \;=\; 2k/(k+1)! - 2(k+1)/(k+2)!.$$

▶ **15.** [*HM30*] Покажите, что ∞-распределенная последовательность проходит критерий собирания купонов, в котором существует только два вида купонов, в следующем смысле: пусть $X_1, X_2, \ldots$ — ∞-распределенная двоичная последовательность. Пусть $f(0) = 0$ и для $n \ge 1$ пусть $f(n)$ — наименьшее целое число $m > f(n-1)$, такое, что $\{X_{f(n-1)+1}, \ldots, X_m\}$ является множеством $\{0, 1\}$. Докажите, что $\Pr(f(n) - f(n-1) = k) = 2^{1-k}$ для $k \ge 2$ (см. упр. 7).

16. [*HM38*] Выполняется ли критерий собирания купонов для ∞-распределенных последовательностей, когда существует больше двух видов купонов? (См. предыдущее упражнение.)

17. [*HM50*] Для любого заданного рационального числа r Франклин (Franklin) доказал, что последовательность $\langle r^n \bmod 1\rangle$ не является 2-распределенной. Но существует ли рациональное число r, для которого эта последовательность равнораспределена? В частности, равнораспределена ли последовательность при $r = \frac{3}{2}$? [См. K. Mahler, *Mathematika* **4** (1957), 122–124.]

▶ **18.** [*HM22*] Докажите, что если $U_0, U_1, \ldots$ k-распределена, то такой же будет последовательность $V_0, V_1, \ldots$, где $V_n = \lfloor nU_n \rfloor / n$.

19. [*HM35*] Рассмотрите модификацию определения R4, требуя, чтобы подпоследовательность была только 1-распределенной, а не ∞-распределенной. Существует ли последовательность, удовлетворяющая этому более слабому определению, которая не будет ∞-распределенной? (Действительно ли это определение более слабое?)

▶ **20.** [*HM36*] (Н. Г. де Брейн (N. G. de Bruijn) и П. Эрдеш (P. Erdős).) Первые n точек любой $[0\,..\,1)$-последовательности $\langle U_n\rangle$ с $U_0 = 0$ делят интервал $[0\,..\,1)$ на n подынтервалов. Пусть эти подынтервалы имеют длины $l_n^{(1)} \ge l_n^{(2)} \ge \cdots \ge l_n^{(n)}$. Очевидно, что $l_n^{(1)} \ge \frac{1}{n} \ge l_n^{(n)}$, поскольку $l_n^{(1)} + \cdots + l_n^{(n)} = 1$. Одним из способов измерения равномерности распределения $\langle U_n\rangle$ является рассмотрение пределов

$$\overline{L} = \limsup_{n\to\infty} n l_n^{(1)} \qquad \text{и} \qquad \underline{L} = \liminf_{n\to\infty} n l_n^{(n)}.$$

a) Чем являются $\overline{L}$ и $\underline{L}$ для последовательности Ван дер Корпута (van der Corput) (29)?

b) Покажите, что $l_{n+k-1}^{(1)} \ge l_n^{(k)}$ для $1 \le k \le n$. Используйте этот результат для доказательства того, что $\overline{L} \ge 1/\ln 2$.

c) Докажите, что $\underline{L} \le 1/\ln 4$. [*Указание.* Для каждого n существуют такие числа a_1, $\ldots$, a_{2n}, что $l_{2n}^{(k)} \ge l_{n+a_k}^{(n+a_k)}$ для $1 \le k \le 2n$. Кроме того, каждое целое число $2, \ldots, n$ появляется самое большее дважды в $\{a_1, \ldots, a_{2n}\}$.]

d) Покажите, что последовательность $\langle W_n\rangle$, определенная равенством $W_n = \lg(2n+1) \bmod 1$, удовлетворяет $1/\ln 2 > nl_n^{(1)} \ge nl_n^{(n)} > 1/\ln 4$ для всех n. Следовательно, она достигает оптимальных значений $\overline{L}$ и $\underline{L}$.

21. [*HM40*] (Л. Г. Рамшоу (L. H. Ramshaw).)

a) Продолжаем предыдущее упражнение. Будет ли последовательность $\langle W_n \rangle$ равнораспределена?

b) Покажите, что $\langle W_n \rangle$ является только $[0\,..\,1)$-последовательностью, для которой выполняется $\sum_{j=1}^{k} l_n^{(j)} \le \lg(1 + k/n)$ всякий раз, как только $1 \le k \le n$.

c) Пусть $\langle f_n(l_1, \ldots, l_n) \rangle$ — любая последовательность непрерывных функций на множествах строк размерности n $\{(l_1, \ldots, l_n) \mid l_1 \ge \cdots \ge l_n$ и $l_1 + \cdots + l_n = 1\}$, удовлетворяющая следующим двум свойствам:

$$f_{mn}(\tfrac{1}{m}l_1, \ldots, \tfrac{1}{m}l_1, \ \ldots\ , \tfrac{1}{m}l_n, \ldots, \tfrac{1}{m}l_n) = f_n(l_1, \ldots, l_n);$$

$$\text{если} \quad \sum_{j=1}^{k} l_j \ge \sum_{j=1}^{k} l'_j \text{ для } 1 \le k \le n, \quad \text{то} \quad f_n(l_1, \ldots, l_n) \ge f_n(l'_1, \ldots, l'_n).$$

[Примеры: $nl_n^{(1)}$; $-nl_n^{(n)}$; $l_n^{(1)}/l_n^{(n)}$; $n(l_n^{(1)2} + \cdots + l_n^{(n)2})$.] Пусть

$$\overline{F} = \limsup_{n\to\infty} f_n(l_n^{(1)}, \ldots, l_n^{(n)})$$

для последовательности $\langle W_n \rangle$. Покажите, что $f_n(l_n^{(1)}, \ldots, l_n^{(n)}) \le \overline{F}$ для всех n относительно $\langle W_n \rangle$, а также $\limsup_{n\to\infty} f_n(l_n^{(1)}, \cdots, l_n^{(n)}) \ge \overline{F}$ относительно каждой другой $[0\,..\,1)$-последовательности.

▸ **22.** [*HM30*] (Герман Вейль (Hermann Weyl).) Покажите, что $[0\,..\,1)$-последовательность $\langle U_n \rangle$ k-распределена тогда и только тогда, когда

$$\lim_{N\to\infty} \frac{1}{N} \sum_{0\le n<N} \exp(2\pi i(c_1U_n + \cdots + c_kU_{n+k-1})) = 0$$

для каждого множества не всех равных нулю целых чисел c_1, c_2, $\ldots$, c_k.

23. [*M32*] (a) Покажите, что $[0\,..\,1)$-последовательность $\langle U_n \rangle$ k-распределена тогда и только тогда, когда все последовательности $\langle (c_1U_n + c_2U_{n+1} + \cdots + c_kU_{n+k-1}) \bmod 1 \rangle$ 1-распределены всякий раз, когда c_1, c_2, $\ldots$, c_k — целые числа, не все равные нулю. (b) Покажите, что b-ичная последовательность $\langle X_n \rangle$ k-распределена тогда и только тогда, когда все последовательности $\langle (c_1X_n + c_2X_{n+1} + \cdots + c_kX_{n+k-1}) \bmod b \rangle$ 1-распределены всякий раз , когда c_1, c_2, $\ldots$, c_k — целые числа с $\gcd(c_1, \ldots, c_k) = 1$.

▸ **24.** [*M35*] (Й. Г. Ван дер Корпут (J. G. van der Corput).) (a) Докажите, что $[0\,..\,1)$-последовательность $\langle U_n \rangle$ равнораспределена всегда, когда последовательности $\langle (U_{n+k} - U_n) \bmod 1 \rangle$ равнораспределены для всех $k > 0$. (b) Следовательно, $\langle (\alpha_d n^d + \cdots + \alpha_1 n + \alpha_0) \bmod 1 \rangle$ равнораспределена, когда $d > 0$ и α_d — иррациональные числа.

25. [*HM20*] Последовательность называется белой, если все сериальные коэффициенты корреляции равны нулю, т. е. если соотношение в следствии S выполняется для *всех* $k \ge 1$. (Согласно следствию S ∞-распределенная последовательность является белой.) Покажите, что если $[0\,..\,1)$-последовательность равнораспределена, то она белая тогда и только тогда, когда

$$\lim_{n\to\infty} \frac{1}{n} \sum_{0\le j<n} (U_j - \tfrac{1}{2})(U_{j+k} - \tfrac{1}{2}) = 0 \qquad \text{для всех } k \ge 1.$$

26. [*HM34*] (Дж. Франклин (J. Franklin).) Белая последовательность, определенная в предыдущем упражнении, может не быть случайной. Пусть U_0, U_1, $\ldots$ — ∞-распределенная последовательность. Определим последовательность V_0, V_1, $\ldots$ следующим образом:

$$(V_{2n-1}, V_{2n}) = (U_{2n-1}, U_{2n}), \qquad \text{если } (U_{2n-1}, U_{2n}) \in G,$$
$$(V_{2n-1}, V_{2n}) = (U_{2n}, U_{2n-1}), \qquad \text{если } (U_{2n-1}, U_{2n}) \notin G,$$

где G — множество

$$\{(x,y) \mid x - \tfrac{1}{2} \le y \le x \text{ или } x + \tfrac{1}{2} \le y\}.$$

Покажите, что (a) последовательность V_0, V_1, ... равнораспределенная и белая, (b) $\Pr(V_n > V_{n+1}) = \frac{5}{8}$. (Это указывает на слабость критерия сериальной корреляции.)

27. [*HM48*] Чему равно наибольшее возможное значение для $\Pr(V_n > V_{n+1})$ в равнораспределенной белой последовательности? (Д. Копперсмит (D. Coppersmith) построил такую последовательность, для которой это значение достигает $\frac{7}{8}$.)

▶ **28.** [*HM21*] Воспользуйтесь последовательностью (11), чтобы построить 3-распределенную $[0\,..\,1)$-последовательность, для которой $\Pr(U_{2n} \ge \frac{1}{2}) = \frac{3}{4}$.

29. [*HM34*] Пусть X_0, X_1, ... — $(2k)$-распределенная двоичная последовательность. Покажите, что

$$\overline{\Pr}(X_{2n} = 0) \le \frac{1}{2} + \binom{2k-1}{k} \Big/ 2^{2k}.$$

▶ **30.** [*M39*] Постройте $(2k)$-распределенную двоичную последовательность, для которой

$$\Pr(X_{2n} = 0) = \frac{1}{2} + \binom{2k-1}{k} \Big/ 2^{2k}.$$

(Таким образом, неравенство в предыдущем упражнении является наилучшим.)

31. [*M30*] Покажите, что существуют $[0\,..\,1)$-последовательности, удовлетворяющие определению R5, однако $\nu_n/n \ge \frac{1}{2}$ для всех $n > 0$, где ν_n — число $j < n$, для которых $U_n < \frac{1}{2}$. (Это можно рассматривать как неслучайное свойство последовательности.)

32. [*M24*] Задана $\langle X_n \rangle$ "случайная" b-ичная последовательность, удовлетворяющая определению R5, и $\mathcal{R}$ — исчисляемое правило подпоследовательности, точно задающее бесконечную подпоследовательность $\langle X_n \rangle \mathcal{R}$. Покажите, что последняя подпоследовательность не только 1-распределена, но и "случайна" согласно определению R5.

33. [*HM22*] Пусть $\langle U_{r_n} \rangle$ и $\langle U_{s_n} \rangle$ — бесконечные непересекающиеся подпоследовательности последовательности $\langle U_n \rangle$. (Иначе говоря, $r_0 < r_1 < r_2 < \cdots$ и $s_0 < s_1 < s_2 < \cdots$ — возрастающие последовательности целых чисел и $r_m \ne s_n$ для любых m, n.) Предположим, что $\langle U_{t_n} \rangle$ — комбинированная подпоследовательность, такая, что $t_0 < t_1 < t_2 < \cdots$, и положим $\{t_n\} = \{r_n\} \cup \{s_n\}$. Покажите, что если $\Pr(U_{r_n} \in A) = \Pr(U_{s_n} \in A) = p$, то $\Pr(U_{t_n} \in A) = p$.

▶ **34.** [*M25*] Определите правила подпоследовательностей $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$, ..., такие, что с этими правилами можно использовать алгоритм W, чтобы задать эффективный алгоритм построения $[0\,..\,1)$-последовательности, удовлетворяющей определению R1.

▶ **35.** [*HM35*] (Д. В. Лавленд (D. W. Loveland).) Покажите, что если двоичная последовательность $\langle X_n \rangle$ R5-случайна и если $\langle s_n \rangle$ — любая исчисляемая последовательность соответственно с определением R4, то $\overline{\Pr}(X_{s_n} = 1) \ge \frac{1}{2}$ и $\underline{\Pr}(X_{s_n} = 1) \le \frac{1}{2}$.

36. [*HM30*] Пусть $\langle X_n \rangle$ — двоичная последовательность, "случайная" согласно определению R6. Покажите, что $[0\,..\,1)$-последовательность $\langle U_n \rangle$, определенная в двоичной системе счисления по схеме

$$U_0 = (0.X_0)_2, \quad U_1 = (0.X_1X_2)_2, \quad U_2 = (0.X_3X_4X_5)_2, \quad U_3 = (0.X_6X_7X_8X_9)_2, \quad \ldots,$$

случайна в смысле определения R6.

37. [*M37*] (Д. Копперсмит (D. Coppersmith).) Постройте последовательность, удовлетворяющую определению R4, но не определению R5. [*Указание.* Рассмотрите преобразование U_0, U_1, U_4, U_9, ... истинно случайной последовательности.]

38. [*M49*] (A. N. Колмогоров.) Пусть заданы N, n и ϵ. Чему равно наименьшее число алгоритмов в множестве **A**, таких, чтобы не существовали (n,ϵ)-случайные двоичные последовательности длины N по отношению к **A**? (Если нельзя задать точные формулы, можно ли найти асимптотические формулы? Суть этой проблемы — обнаружить, как точная грань (37) становится "наилучшей возможной".)

39. [*HM45*] (B. M. Шмидт (W. M. Schmidt).) Пусть U_n — $[0\,..\,1)$-последовательность и пусть $\nu_n(u)$ — число таких неотрицательных целых чисел $j \le n$, что $0 \le U_j < u$. Докажите, что существует такая положительная постоянная c, что для любого N и любой $[0\,..\,1)$-последовательности $\langle U_n \rangle$ мы получим

$$|\nu_n(u) - un| > c \ln N$$

для некоторых n и u при $0 \le n < N$, $0 \le u < 1$. (Другими словами, никакая $[0\,..\,1)$-последовательность не может быть *слишком* равнораспределена.)

40. [*M28*] Завершите доказательство леммы P1.

41. [*M21*] В лемме P2 показано существование критерия прогноза, но при доказательстве предполагается существование подходящего k без объяснения, как конструктивно находить k из A. Покажите, что любой алгоритм A можно обратить в алгоритм A' с $T(A') \le T(A) + O(N)$, который предсказывает B_N по $B_1 \ldots B_{N-1}$ с вероятностью, по крайней мере равной $\frac{1}{2} + (P(A,S) - P(A,\$_N))/N$ на любом симметричном сдвиге N-источника S.

▸ **42.** [*M28*] (*Попарная независимость.*)

a) Пусть $X_1, \ldots, X_n$ — случайные величины со средним, равным $\mu = \mathrm{E}\,X_j$, и дисперсией $\sigma^2 = \mathrm{E}\,X_j^2 - (\mathrm{E}\,X_j)^2$ при $1 \le j \le n$. Докажите неравенство Чебышева

$$\Pr((X_1 + \cdots + X_n - n\mu)^2 \ge tn\sigma^2) \le 1/t$$

при дополнительных предположениях, что $\mathrm{E}(X_i X_j) = (\mathrm{E}\,X_i)(\mathrm{E}\,X_j)$ всякий раз, когда $i \ne j$.

b) Пусть B — случайная двоичная матрица размера $k \times R$. Докажите, что если c и c' — фиксированные не равные нулю двоичные векторы размерности k, то cB и $c'B$ — независимые случайные двоичные векторы размерности R (по модулю 2).

c) Примените (a) и (b) к анализу алгоритма L.

43. [*20*] Кажется, точно так же тяжело найти множители любого *фиксированного* целого числа Блюма M, состоящего из R двоичных разрядов. Как найти множители *случайного* целого числа, состоящего из R двоичных разрядов? Почему тогда теорема P сформулирована для случайного, а не фиксированного M?

▸ **44.** [*16*] (И. Дж. Гуд (I. J. Good).) Может ли правильная таблица случайных чисел содержать точно одну ошибку?

3.6. ВЫВОДЫ

В ЭТОЙ ГЛАВЕ было рассмотрено довольно много тем: генерирование случайных чисел, их проверка, их видоизменение при использовании и методы получения теоретических фактов. Возможно, главным для многих читателей был вопрос "Что получено в результате всей этой теории и что такое простой добротный генератор, который можно использовать в программах в качестве надежного источника случайных чисел?".

Подробные исследования в этой главе наводят на мысль, что следующая процедура позволяет получить простейший генератор случайных чисел для машинного языка большинства компьютеров. В начале программы присвойте целой переменной X некоторое значение X_0. Эта величина X используется только для генерирования случайного числа. Как только в программе потребуется новое случайное число, положите

$$X \leftarrow (aX + c) \bmod m \tag{1}$$

и используйте новое значение X в качестве случайной величины. Необходимо тщательно выбрать X_0, a, c и m и разумно использовать случайные числа согласно следующим принципам.

i) "Начальное" число X_0 может быть выбрано произвольно. Если программа используется несколько раз и каждый раз требуются различные источники случайных чисел, то нужно присвоить X_0 последнее полученное значение X на предыдущем прогоне или, если это более удобно, присвоить X_0 текущие дату и время. Чтобы снова запустить программу с *такими же* случайными числами (например, при отладке программы), нужно напечатать X_0, если иначе его невозможно получить.

ii) Число m должно быть большим, скажем, по крайней мере 2^{30}. Возможно, удобно взять его равным размеру компьютерного слова, так как это делает вычисление $(aX + c) \bmod m$ вполне эффективным. В разделе 3.2.1.1 выбор m обсуждается более детально. Вычисление $(aX + c) \bmod m$ должно быть *точным* без округления ошибки.

iii) Если m — степень 2 (т. е. если используется двоичный компьютер), выбираем a таким, чтобы $a \bmod 8 = 5$. Если m — степень 10 (т. е. используется десятичный компьютер), выбираем a таким, чтобы $a \bmod 200 = 21$. Одновременный выбор a и c даст гарантию, что генератор случайных чисел будет вырабатывать все m различных возможных значений X прежде, чем они начнут повторяться (см. раздел 3.2.1.2), и гарантирует высокий "потенциал" (см. раздел 3.2.1.3).

iv) Множитель a предпочтительно выбирать между $.01m$ и $.99m$, и его двоичные или десятичные цифры *не* должны иметь простую регулярную структуру. Выбирая несколько случайных констант, подобных $a = 3141592621$ (которые удовлетворяют обоим условиям в (iii)), почти всегда получаем достаточно хороший множитель. Дополнительная проверка, конечно, нужна, если генератор случайных чисел используется регулярно. Например, частичные отношения не должны быть большими, когда для нахождения $\gcd$ a и m используется алгоритм Евклида (см. раздел 3.3.3). Множитель должен пройти спектральный критерий (раздел 3.3.4) и несколько критериев, описанных в разделе 3.3.2, прежде чем он получит сертификат качества.

v) Значение c не существенно, когда a — хороший множитель, за исключением того, что c не должно иметь общего множителя с m, когда m — размер компьютерного слова. Таким образом, можно выбрать $c = 1$ или $c = a$. Многие используют $c = 0$ вместе с $m = 2^e$, но они жертвуют двумя двоичными разрядами точности и половиной начальных значений, чтобы сэкономить всего несколько наносекунд счета (см. упр. 3.2.1.2–9).

vi) Младшие значащие цифры (справа) X не очень случайны, так что решения, основанные на числе X, всегда должны опираться, главным образом, на старшие значащие цифры. Обычно лучше считать X случайной дробью X/m между 0 и 1, т. е. представлять себе X с десятичной точкой слева, а не относиться к X как к случайному целому числу между 0 и $m - 1$. Чтобы подсчитать случайное целое число между 0 и $k - 1$, нужно умножить его на k (но не делить на k; см. упр. 3.4.1–3) и округлить результат.

vii) Важное ограничение случайности последовательности (1) обсуждалось в разделе 3.3.4, в котором показано, что "точность" при размерности t будет только порядка $\sqrt[t]{m}$. Применяя метод Монте-Карло, необходимо использовать случайные последовательности высокой надежности. Их можно получить с помощью технических приемов, описанных в разделе 3.2.2.

viii) Можно генерировать не больше $m/1000$ чисел, иначе последующие будут вести себя подобно предыдущим. Если $m = 2^{32}$, значит, новая схема (например, новый множитель a) должна использоваться после генерирования нескольких миллионов случайных чисел.

Комментарии, приведенные выше, относятся, главным образом, к машинному языку программирования. Некоторые понятия прекрасно работают также на языках высокого уровня, например (1) превращается в `X=a*X+c` на языке C, если `X` имеет тип `беззнаковое длинное` и если m равно модулю `беззнаково длинной` арифметики (обычно 2^{32} или 2^{64}). Но C не дает возможности рассматривать `X` в качестве дроби, как того требует (vi), если не обращаться к числам с плавающей точкой двойной точности.

Поэтому для языка программирования, подобного C, часто используют другой вариант (1): выбирается простое число m, близкое к наибольшему легко вычисляемому целому числу, a полагается равным первообразному корню m и приращение c в этом случае равняется нулю. Тогда (1) может полностью выполняться простой арифметикой над числами, остающимися между $-m$ и $+m$, так, как в упр. 3.2.1.1–9. Например, когда $a = 48271$ и $m = 2^{31} - 1$ (см. строку 20 табл. 3.3.4–1), можно вычислить $X \leftarrow aX \bmod m$ на языке C.

```
#define MM 2147483647              /* простое число Мерсена */
#define AA 48271  /* здесь хорош спектральный критерий */
#define QQ 44488  /* (long)(MM/AA) */
#define RR 3399   /* MM % AA; важно, что RR<QQ */
X=AA*(X%QQ)-RR*(long)(X/QQ);
if (X<0) X+=MM;
```

Здесь `X` имеет тип `long` и `X` можно инициализировать, как ненулевое начальное значение, меньшее `MM`. Поскольку `MM` — простое число, наименьший значащий дво-

ичный разряд X так же случаен, как и самый старший, поэтому в предостережении (vi) нет необходимости.

Чтобы получить миллионы и миллионы случайных чисел, можно скомбинировать эту программу с другой, как в 3.3.4–(38), дописав дополнительно несколько операций.

```
#define MMM 2147483399        /* простое число, но не Мерсена */
#define AAA 40692       /* другой удачный спектральный критерий */
#define QQQ 52774       /* (long)(MMM/AAA) */
#define RRR 3791        /* MMM % AAA; снова меньше, чем QQQ */
Y=AAA*(Y%QQQ)-RRR*(long)(Y/QQQ);
if (Y<0) Y+=MMM;
Z=X-Y; if (Z<=0) Z+=MM;
```

Подобно X, случайная величина Y должна быть установлена не равной нулю. Эти операции несколько отличаются от операций, приведенных в 3.3.4–(38): выходное Z никогда не равно нулю и Z всегда лежит точно между 0 и 2^{31}. Длина периода последовательности Z приблизительно равна 74 квадриллионам, и ее числа сейчас имеют точность, приблизительно вдвое бóльшую, чем числа X.

Этот метод мобильный и совершенно простой, но не очень устойчивый. Альтернативная схема, основанная на последовательности Фибоначчи с запаздыванием и вычитанием (упр. 3.2.2–23), даже более привлекательна, так как не только легко преобразуется для использования на разных компьютерах, но значительно быстрее работает и поставляет случайные числа лучшего качества, поскольку t-мерная точность, возможно, хороша для $t \le 100$. Ниже приведена программа на языке C *ran_array*(**long** *aa*[], **int** *n*), которая генерирует n новых случайных чисел и засылает их в заданный массив *aa*, используя рекуррентное соотношение

$$X_j = (X_{j-100} - X_{j-37}) \bmod 2^{30}. \tag{2}$$

Это соотношение, в частности, хорошо подходит для современных компьютеров. Значение n должно быть равно по крайней мере 100; рекомендуются большие значения, например 1 000.

```
#define KK 100                          /* длинное запаздывание */
#define LL  37                          /* короткое запаздывание */
#define MM (1L<<30)                        /* модуль */
#define mod_diff(x,y) (((x)-(y))&(MM-1))    /* (x-y) mod MM */
long ran_x[KK];                        /* состояние генератора */
void ran_array(long aa[],int n) {
  register int i,j;
  for (j=0;j<KK;j++) aa[j]=ran_x[j];
  for (;j<n;j++) aa[j]=mod_diff(aa[j-KK],aa[j-LL]);
  for (i=0;i<LL;i++,j++) ran_x[i]=mod_diff(aa[j-KK],aa[j-LL]);
  for (;i<KK;i++,j++) ran_x[i]=mod_diff(aa[j-KK],ran_x[i-LL]);
}
```

Вся информация о числах, которые генерируются путем заблаговременного обращения к *ran_array*, появляется в массиве *ran_x*. Так, этот массив можно ско-

пировать во время вычислений, чтобы позднее повторить запуск программы с такими же значениями, не проходя весь путь назад, к началу последовательности. Особенностью использования рекуррентных соотношений, подобных (2), является, конечно, необходимость получения сразу всех начальных значений путем присвоения подходящих значений $X_0, \ldots, X_{99}$. Следующая программа *ran_start*(**long** *seed*) хорошо запускает генератор, когда задано любое начальное число, находящееся между 0 и $2^{30} - 3 = 1{,}073{,}741{,}821$ включительно.

```
#define TT  70     /* гарантирует разделение потоков */
#define is_odd(x)  ((x)&1)            /* блок двоичных разрядов x */
#define evenize(x) ((x)&(MM-2))              /* делаем x четным */

void ran_start(long seed) { /* используется для размещения ran_array */
  register int t,j;
  long x[KK+KK-1];                 /* подготовка буфера */
  register long ss=evenize(seed+2);
  for (j=0;j<KK;j++) {
    x[j]=ss;                          /* самозагрузка буфера */
    ss<<=1; if (ss>=MM) ss-=MM-2;    /* циклический сдвиг 29 двоичных
                                        разрядов */
  }
  for (;j<KK+KK-1;j++) x[j]=0;
  x[1]++;                    /* делаем x[1] (и только x[1]) нечетным */
  ss=seed&(MM-1); t=TT-1; while (t) {
    for (j=KK-1;j>0;j--) x[j+j]=x[j];           /* "квадрат" */
    for (j=KK+KK-2;j>KK-LL;j-=2) x[KK+KK-1-j]=evenize(x[j]);
    for (j=KK+KK-2;j>=KK;j--) if(is_odd(x[j])) {
      x[j-(KK-LL)]=mod_diff(x[j-(KK-LL)],x[j]);
      x[j-KK]=mod_diff(x[j-KK],x[j]);
    }
    if (is_odd(ss)) {                    /* "умножаем на z" */
      for (j=KK;j>0;j--)  x[j]=x[j-1];
      x[0]=x[KK];            /* сдвигаем буфер циклично */
      if (is_odd(x[KK])) x[LL]=mod_diff(x[LL],x[KK]);
    }
    if (ss) ss>>=1; else t--;
  }
  for (j=0;j<LL;j++) ran_x[j+KK-LL]=x[j];
  for (;j<KK;j++) ran_x[j-LL]=x[j];
}
```

Довольно любопытное маневрирование *ran_start* приведено в упр. 9, в котором доказывается, что последовательности чисел, генерируемых с различными начальными значениями, независимы: *каждый блок 100 последовательных значений* X_n, X_{n+1}, $\ldots$, X_{n+99} *в последующем выходном массиве ran_array будет отличаться от блоков, появляющихся с другим начальным значением.* (Строго говоря, известно, что это справедливо только тогда, когда $n < 2^{70}$, но меньше 2^{55} нс в год.) Некоторые процессы можно, следовательно, запускать параллельно с различными начальными

значениями и быть уверенным, что они дадут независимые результаты. Разные группы ученых, работающих над задачей в различных компьютерных центрах, могут быть уверены, что они не дублируют работу других ученых, если присваивают различные начальные значения. Таким образом, более одного миллиарда практически непересекающихся групп случайных чисел предусмотрено единственными программами *ran_array* и *ran_start*. А если этого недостаточно, можете заменить параметры программы 100 и 37 другими значениями из табл. 3.2.2–1.

В программах на языке C для эффективности используется операция "поразрядное и", `&`, так что их нельзя точно перенести на другие компьютеры, кроме тех, в которых применяется представление с удвоенной точностью целых чисел. Почти все современные компьютеры основаны на арифметике с удвоенной точностью, но для этого алгоритма в действительности операция `&` не нужна. В упр. 10 показано, как получить такую же последовательность чисел на языке FORTRAN, не используя подобные трюки. Несмотря на то что приведенные здесь программы предназначены для генерирования целых чисел с 30 двоичными разрядами, их легко преобразовать в программы генерирования случайных дробей между 0 и 1 с 52 двоичными разрядами на компьютерах, имеющих арифметику с плавающей точкой (см. упр. 11).

Вы, возможно, захотите включить программу *ran_array* в библиотеку программ или найти еще кого-нибудь, кто уже так сделал. Один из способов проверки, реализуются ли программы *ran_array* и *ran_start* в соответствии с операциями, приведенными выше, состоит в запуске следующего элементарного теста программ.

```
main() { register int m; long a[2009];
ran_start(310952);
for (m=0;m<2009;m++) ran_array(a,1009);
printf("%ld\n", ran_x[0]);
ran_start(310952);
for (m=0;m<1009;m++) ran_array(a,2009);
printf("%ld\n", ran_x[0]);
}
```

Должно быть напечатано 461390032 (дважды).

Предостережение: числа, генерируемые программой *ran_array*, не удовлетворяют критерию промежутков между днями рождений, приведенному в разделе 3.3.2J, и обладают другими недостатками, которые иногда возникают в высокоточном моделировании (см. упр. 3.3.2–31 и 3.3.2–35). Один из способов избежать проблем, связанных с критерием промежутков между днями рождения, — просто использовать только половину чисел (пропуская нечетные элементы), но это не панацея от других проблем. Лучшая четная процедура, предложенная Мартином Люшером (Martin Lüscher), обсуждалась в разделе 3.2.2: используйте программу *ran_array* для генерирования, допустим, 1 009 чисел, но применяйте из них только первые 100 (см. упр. 15). *Этот метод теоретически довольно обоснован и не имеет известных недостатков.* Большинство пользователей не нуждаются в таком предостережении, но так, определенно, меньше риска и оно позволяет делать выбор между случайностью и скоростью.

Много известно о линейных конгруэнтных последовательностях, подобных (1), но сравнительно мало исследованы свойства случайных последовательностей

Фибоначчи с запаздыванием, подобных (2). Обе, кажется, похожи на практически надежные, если их использовать с учетом приведенных предостережений.

Когда эта глава была написана в конце 60-х годов, поистине ужасный генератор случайных чисел RANDU использовался в большинстве компьютеров мира (см. раздел 3.3.4). Авторы, внесшие большой вклад в науку о генерировании случайных чисел, часто не знали, что обычных методов их доказательств недостаточно. Особенно заслуживает внимания неприятный пример Алана М. Ферренберга (Alan M. Ferrenberg) и его коллег, опубликованный в *Physical Review Letters* **69** (1992), 3382–3384. Они тестировали свои алгоритмы для трехмерной задачи, рассматривая сначала двумерную задачу с известным ответом, и обнаружили, что предположительно суперкачественные современные генераторы случайных чисел дают неправильные результаты в пятом знаке. А старомодный, как крутящаяся мельница, линейный конгруэнтный генератор $X \leftarrow 16807X \bmod (2^{31} - 1)$ работал прекрасно. Возможно, дополнительные научные исследования покажут, что даже генераторы случайных чисел, рекомендованные здесь, будут неудовлетворительны. Мы надеемся, что этого не случится, но история предупреждает, что нужно быть осмотрительными. Отсюда вытекает наиболее разумная линия поведения — работая с каждой программой метода Монте-Карло, необходимо по крайней мере дважды использовать совершенно различные источники случайных чисел, прежде чем получить решения. Это будет указывать на стабильность результатов, а также оградит от опасного доверия к генераторам со скрытыми недостатками. (Каждый генератор случайных чисел будет "проваливаться" по крайней мере для одной какой-либо задачи.)

Превосходная библиография до 1972 года по генерированию случайного числа составлена Ричардом Нансом и Клодом Оверстритом (мл.) (Richard E. Nance and Claude Overstreet, Jr., *Computing Reviews* **13** (1972), 495–508) и Э. Р. Совьи (E. R. Sowey, *International Stat. Review* **40** (1972), 355–371). Период 1972–1984 годов закрыт Совьи в работах *International Stat. Review* **46** (1978), 89–102; *J. Royal Stat. Soc.* **A149** (1986), 83–107. Последующее развитие обсуждается в книге Шу Тезука (Shu Tezuka, *Uniform Random Numbers* (Boston: Kluwer, 1995)).

Для подробного изучения использования случайных чисел в численном анализе обратитесь к книге J. M. Hammersley and D. C. Handscomb, *Monte Carlo Methods* (London: Methuen, 1964). В ней показано, как некоторые численные методы улучшаются с помощью "квазислучайных" чисел, специально предназначенных для определенных целей (необязательно удовлетворяющие статистическим критериям, которые мы обсуждали). Происхождение метода Монте-Карло для компьютеров обсуждается в работе N. Metropolis and R. Eckhardt in *Stanisław Ulam 1909–1984*, специальный выпуск *Los Alamos Science* **15** (1987), 125–136.

Каждому читателю будет интересно рассмотреть упр. 6, приведенное ниже.

УПРАЖНЕНИЯ

1. [*21*] Используя метод (1), напишите программу со следующими характеристиками для машины MIX.

Вызов последовательности:	`JMP RANDI`
Входные условия:	rA = k — положительное целое число < 5000
Выходные условия:	rA $\leftarrow$ случайное целое число Y, $1 \le Y \le k$, все значения равновероятны; rX =?; переполнение выключено

▸ **2.** [*15*] Кое-кто напуган тем, что компьютеры в будущем будут править миром, но правительство их успокаивает заявлениями, что машина не может сделать ничего нового, так как она только повинуется командам ее создателя, программиста. Леди Лавлейс (Lovelace) в 1844 году писала: "Аналитическая машина не претендует *на создание* чего-либо. Она может сделать все, *если известно, как ей приказать* это выполнить". Ее утверждение развивалось многими философами. Обсудите это высказывание в связи с генераторами случайных чисел.

3. [*32*] (*Игра в кости.*) Напишите программу моделирования бросания двух игральных костей, на каждой из которых выпадают значения 1, 2, ..., 6 с равной вероятностью. Если сумма равна 7 или 11 при первом бросании, то игрок выиграл; если сумма равна 2, 3 или 12, то проиграл. При любой другой сумме называем сумму "очко" и продолжаем бросать игральные кости до тех пор, пока выпадет либо 7 (проигрыш), либо очко (выигрыш).

Играется десять игр. Результат каждого бросания игральных костей можно напечатать в виде $m\,n$, где m и n — цифры, выпавшие на каждой из двух игральных костей, и сопроводить их соответствующими комментариями (например, "змеиные глаза" либо "маленький солдат", либо "трудная дорога" и т. д.).

4. [*40*] (*Солитер, или пасьянс.*) Уйма драгоценного времени уходит на раскладывание пасьянса, и, возможно, автоматика существенно вторгнется в эту область. Напишите программу, которая (i) тасует модель колоды карт; (ii) раскладывает несколько обычных пасьянсов, основанных на порядке карт в колоде, и (iii) выводит на дисплей результат игры, показывая, насколько близко программа подходит к выигрышу. Сохраните собранные статистические данные. Программа по запросу может начать жульничать.

5. [*46*] (*Литературное творчество компьютера.*) Телевизионная программа "Думающая машина" была показана по телесети CBS 26 октября 1960 года. Среди всего прочего она отвела важнейшее место двум небольшим пьесам в стиле "вестерн", написанных компьютером. Здесь помещены два сценария в том виде, в каком они были напечатаны компьютером.

Сценарий #1. (Револьвер в правой руке, деньги в левой руке, стакан на столе, бутылка на столе, кобура на грабителе, револьвер шерифа в правой руке шерифа, кобура шерифа на шерифе.)

ГРАБИТЕЛЬ: (Грабитель за окном.) Идет к двери; открывает дверь; входит; закрывает дверь; идет в угол; кладет в углу деньги; идет к столу; кладет револьвер на стол; сидит и ерзает; сидит за столом; берет стакан правой рукой (пустой); ставит стакан на стол; берет бутылку правой рукой; наливает; ставит бутылку на стол; берет стакан правой рукой; пьет из стакана; ставит стакан на стол; берет бутылку правой рукой; сидит за столом; сидит за столом; идет в угол; идет к окну; идет к столу; ставит бутылку на стол; сидит и ерзает; сидит за столом; сидит и ерзает; идет к окну; идет к столу; подымает стакан правой рукой.

ШЕРИФ: (Шериф за окном.) Видит грабителя (грабитель видит шерифа); идет к двери.

ГРАБИТЕЛЬ: Ставит стакан на стол; берет револьвер правой рукой; проверяет револьвер.

ШЕРИФ: Ждет; открывает двери; видит грабителя (грабитель видит шерифа); входит.

ГРАБИТЕЛЬ: Идет к окну; целится; стреляет; ШЕРИФ ЛЕГКО РАНЕН.

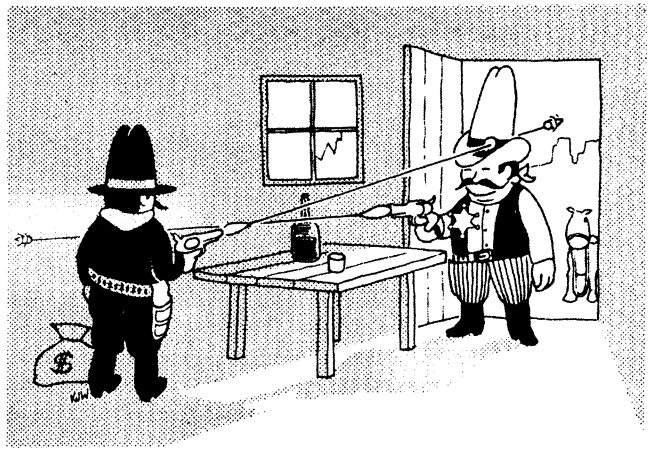
ШЕРИФ: Идет к окну; целится; стреляет; ПРОМАХНУЛСЯ; идет к двери; идет к окну.

ГРАБИТЕЛЬ: Идет к двери; целится; целится.

ШЕРИФ: Целится; стреляет; ПРОМАХНУЛСЯ.

ГРАБИТЕЛЬ: Стреляет; ШЕРИФ ЛЕГКО РАНЕН.

ШЕРИФ: Идет к двери; целится; стреляет; ПРОМАХНУЛСЯ; выходит; целится.



ГРАБИТЕЛЬ: Целится; стреляет; ПРОМАХНУЛСЯ; целится; стреляет; ПРОМАХНУЛСЯ.

ШЕРИФ: Стреляет; ПРОМАХНУЛСЯ; идет к окну; целится; стреляет; ПРОМАХНУЛСЯ.

ГРАБИТЕЛЬ: Целится; стреляет; ПРОМАХНУЛСЯ; целится; стреляет; ПРОМАХНУЛСЯ; целится; стреляет; ШЕРИФ ЛЕГКО РАНЕН.

ШЕРИФ: Целится; стреляет; ПОПАДАЕТ В ГРАБИТЕЛЯ.

ГРАБИТЕЛЬ: Роняет РЕВОЛЬВЕР; грабитель умирает.

ШЕРИФ: Кладет револьвер в кобуру; идет к столу; берет стакан правой рукой (пустой); берет стакан из правой руки в левую руку; берет бутылку правой рукой; наливает; ставит бутылку на стол; берет стакан из левой руки в правую руку; пьет из стакана; берет стакан из правой руки в левую руку; берет бутылку правой рукой; наливает; ставит бутылку на стол; берет стакан из левой руки в правую руку; пьет из стакана; ставит стакан на стол; идет в угол; берет деньги правой рукой; идет к двери; выходит; закрывает дверь. ЗАНАВЕС.

Сценарий #2. (Револьвер в правой руке; деньги в левой руке; стакан на столе; бутылка на столе; кобура на грабителе; револьвер шерифа в правой руке шерифа; кобура шерифа на шерифе.)

ГРАБИТЕЛЬ: (Грабитель за окном.) Идет к двери; открывает дверь; входит; закрывает дверь; идет в угол; кладет деньги в углу; идет к окну; кладет револьвер у окна; прислоняется к окну и смотрит; идет в угол; считает деньги; идет к столу; берет стакан правой рукой (пустой); берет стакан из правой руки в левую руку; берет бутылку правой рукой; наливает; ставит бутылку на стол; берет стакан из левой руки в правую руку; пьет из стакана; ставит стакан на стол; берет бутылку в правую руку; наливает; идет в угол; ставит бутылку в угол; идет к окну; берет револьвер в правую руку; проверяет револьвер; кладет револьвер в кобуру; идет к столу; берет стакан в правую руку; пьет из стакана; идет к окну; ставит стакан у окна.

ШЕРИФ: (Шериф за окном.) Видит грабителя (грабитель видит шерифа); идет к двери.

ГРАБИТЕЛЬ: Берет револьвер из кобуры правой рукой; проверяет револьвер; идет к двери; проверяет револьвер; кладет револьвер у двери.

ШЕРИФ: Открывает двери; видит грабителя (грабитель видит шерифа); входит; идет к окну.

ГРАБИТЕЛЬ: Берет револьвер правой рукой.
ШЕРИФ: Идет к столу.
ГРАБИТЕЛЬ: Целится; стреляет; ПРОМАХНУЛСЯ; прицеливается; стреляет; ПОПАЛ В ШЕРИФА; задул ствол; кладет револьвер в кобуру.
ШЕРИФ: Роняет револьвер; шериф умер.
ГРАБИТЕЛЬ: Идет в угол; берет деньги в правую руку; идет к двери; выходит; закрывает дверь. ЗАНАВЕС.

Внимательный читатель этих сценариев обнаружит представленную здесь в высшей степени напряженную драму. Программа старательно следит за перемещением каждого актера, за тем, что он держит в руках, и т. д. Действия актеров случайные, регулировались определенными вероятностями; вероятности глупых действий возрастали в зависимости от того, как много актер пил и как часто промахивался стрелок. Читатель в состоянии сделать вывод о дополнительных свойствах программы, изучая образцы сценариев. Конечно, даже лучшие сценарии переписывают прежде, чем их поставят, и в особенности когда неопытный писатель готовит начальный набросок. Здесь сценарии точно такие, какие действительно использовались для показа.

Сценарий #1. Музыка.
СП Грабитель всматривается через окно хижины.
КП Лицо грабителя.
СП Грабитель входит в лачугу.
КП Грабитель видит бутылку виски на столе.
КП Шериф за хижиной.
СП Грабитель видит шерифа.
ДП Шериф в дверях за плечом грабителя, оба хватаются за револьверы.
СП Шериф вытаскивает револьвер.
ДП Стреляет. Попадает в грабителя.
СП Шериф поднимает мешки с деньгами.
СП Грабитель шатается.
СП Грабитель умирает. Падает поперек стола после попытки сделать последний выстрел в шерифа.
СП Шериф идет к двери с деньгами.
СП Тело грабителя все еще лежит поперек стола. Камера катится назад. (Смех)

Сценарий #2. Музыка.
КП окно. Появляется грабитель.
СП Грабитель входит в хижину с двумя мешками денег.
СП Грабитель кладет мешки денег на бочку.
КП Грабитель видит виски на столе.
СП Грабитель наливает виски за столом. Идет считать деньги. Смеется.
СП Шериф за хижиной.
СП Вид из окна.
СП Грабитель видит шерифа через окно.
ДП Шериф входит в хижину. Вытаскивает. Стреляет.
КП Шериф. Корчится от выстрела.
СП/2 Шериф идет, шатаясь к столу, чтобы выпить . . . падает мертвый.
СП Грабитель покидает хижину с мешками денег.*

[*Замечание.* КП — "крупный план", СП — "средний план" и т. д. Приведенные сценарии любезно предоставлены Томасом Г. Вулфом (Thomas H. Wolf) — продюсером телепередачи, впервые предложившим идею написания компьютером небольшой пьесы, а также

 Дополнительную информацию можно найти в книге J. E. Pfeiffer, *The Thinking Machine* (New York: J. B. Lippincott, 1962).

Дугласу Т. Россу (Douglas T. Ross) и Гаррисону Р. Морсу (Harrison R. Morse), создавшими программу.]

Летом 1952 года Кристофер Стрейч (Christopher Strachey) использовал техническое обеспечение генератора случайного числа Ferranti Mark I для составления следующего письма.

> Милая голубушка
>
> Моя полная сочувствия привязанность прелестно привлекает ваш показной энтузиазм. Вы — мое нежное обожание, мое затаенное обожание. Мои приятельские чувства, затаив дыхание, надеются на ваш дорогой пыл. Мое томящееся от любви поклонение лелеет ваше жадное рвение.
>
> Ваш грустящий
> М. У. К.

[*Encounter* **3** (1954), 4, 25–31; другой пример появился в статье на Electronic Computers в 64 издании *Pears Cyclopedia* (London, 1955), 190–191.]

У читателя, несомненно, есть множество идей, как научить компьютер литературному творчеству. Это и является целью данного упражнения.

▶ **6.** [*40*] Просмотрите библиотеку программ каждого установленного в вашей организации компьютера и замените плохие генераторы случайных чисел хорошими. Попробуйте избежать сильных потрясений от того, что вы обнаружите.

▶ **7.** [*M40*] Решительный программист формирует свои файлы, используя линейную конгруэнтную последовательность $\langle X_n \rangle$ с периодом 2^{32}, которая генерируется (1) для $m = 2^{32}$. Он берет наибольшие значащие двоичные разряды $\lfloor X_n/2^{16} \rfloor$ и добавляет их в свои данные с исключающим "или", но держит в секрете параметры a, c и X_0.

Покажите, что это не очень надежная схема, и придумайте метод нахождения множителя a и первой разности $X_1 - X_0$ за разумное приемлемое время, задав только значения $\lfloor X_n/2^{16} \rfloor$ для $0 \le n < 150$.

8. [*M15*] Предложите хороший метод выполнения проверки, хорошо ли работают линейные конгруэнтные генераторы.

9. [*HM32*] Пусть $X_0, X_1, \ldots$ — числа, полученные программой *ran_array* после того, как *ran_start* инициализирует процесс генерирования с начальным значением s. Рассмотрим полиномы

$$P_n(z) = X_{n+62}z^{99} + X_{n+61}z^{98} + \cdots + X_n z^{37} + X_{n+99}z^{36} + \cdots + X_{n+64}z + X_{n+63}.$$

a) Докажите, что $P_n(z) \equiv z^{h(s)-n}$ (по модулю 2 и $z^{100} + z^{37} + 1$) для некоторого показателя $h(s)$.

b) Выразите $h(s)$ в терминах двоичного представления s.

c) Докажите, что если $X'_0, X'_1, \ldots$ — последовательность чисел, полученных той же программой при начальном значении $s' \neq s$, то $X_{n+k} \equiv X'_{n'+k}$ (по модулю 2) для $0 \le k < 100$ только тогда, когда $|n - n'| \ge 2^{70} - 1$.

10. [*22*] Преобразуйте программы *ran_array* и *ran_start* на языке C в программы на языке FORTRAN 77 генерирования той же последовательности чисел.

▶ **11.** [*M25*] Предположим, что арифметика с плавающей точкой на числах, имеющих тип `double`, правильно выполняет округление в смысле раздела 4.2.2 (т. е. точно тогда, когда значения должным образом ограничены). Преобразуйте программы *ran_array* и *ran_start* на языке C в сходные программы, которые выдают случайные дроби с двойной точностью в $[0\,..\,1)$ вместо целых чисел с 30-ю двоичными разрядами.

▶ **12.** [*M21*] Какой генератор случайных чисел будет подходящим для мини-компьютера, который имеет арифметику только для целых чисел в области $[-32768\,..\,32767]$?

13. [*M25*] Сравните генератор вычитания с заимствованием из упр. 3.2.1.1–12 с генератором Фибоначчи с запаздыванием, реализованным в программах этого раздела.

▸ **14.** [*M35*] (*Будущее против прошлого.*) Пусть $X_n = (X_{n-37} + X_{n-100}) \bmod 2$. Рассмотрим последовательность

$$\langle Y_0, Y_1, \ldots \rangle = \langle X_0, X_1, \ldots, X_{99}, X_{200}, X_{201}, \ldots, X_{299}, X_{400}, X_{401}, \ldots, X_{499}, X_{600}, \ldots \rangle.$$

(Она соответствует неоднократно вызываемой программе *ran_array*(a,200), когда рассматриваются только младшие значащие двоичные разряды после отбрасывания половины элементов.) Следующий эксперимент был повторен один миллион раз с использованием последовательности $\langle Y_n \rangle$: "Генерируем 100 случайных двоичных разрядов, затем, если 60 или более из них равны нулю, генерируем еще один и печатаем его". В результате было напечатано 14 527 нулей и 13 955 единиц, но вероятность того, что 28 482 случайных двоичных разряда содержат самое большее 13 955 единиц, равна приблизительно .000358.

Дайте математическое объяснение, почему так много нулей будет на выходе.

▸ **15.** [*25*] Напишите на языке C программу генерирования для случайных целых чисел, которые получаются с помощью программы *ran_array*, отбрасывая все, кроме первых 100, элементы из каждых 1 009 элементов, как рекомендуется в разделе.

ГЛАВА 4

АРИФМЕТИКА

В математической практике
нет ничего более хлопотного (правы любезные студенты-математики),
что более всего досаждало бы и мешало вычислителям, чем перемножение,
деление или извлечение квадратных и кубических корней больших чисел.
Выполнение этих операций не только приводит к значительным потерям времени,
но и сопряжено с такой массой скрытых ошибок, что я начал размышлять
о поисках надежного и удобного средства устранения подобных помех.
— ДЖОН НЕПЕР (J. NAPIER [NEPAIR]) (1616)

Терпеть не могу складывать!
Самая большая ошибка — считать арифметику точной *наукой.*
Существуют... тайные законы Чисел, которые может постигнуть
только ум, подобный моему. К примеру, при сложении чисел в столбик
сначала снизу вверх, а затем наоборот вы всегда получите разные суммы.
— М. П. ЛА ТУШ (M. P. LA TOUCHE) (1878)

Не могу представить, чтобы кому-нибудь понадобилось
выполнять умножение со скоростью 40 000 или даже 4 000 операций в час;
такое радикальное средство, как переход к восьмеричной системе счисления,
не следует навязывать всему человечеству ради нескольких личностей.
— Ф. Х. УЭЙЛС (F. X. WALES) (1936)

Большинство специалистов в теории чисел не проявляют интереса к арифметике.
— Б. ПАРЛЕТТ (B. PARLETT) (1979)

Основное назначение этой главы — тщательный анализ четырех основных действий арифметики: сложения, вычитания, умножения и деления. Арифметику многие считают тривиальной дисциплиной, которой обучают детей, а арифметические действия — уделом компьютеров; но мы увидим далее, что арифметика — это увлекательный предмет с множеством интересных аспектов. Она лежит в основе многих важных компьютерных приложений, поэтому необходимо самым тщательным образом изучить эффективные методы вычислительных операций над числами.

На самом деле, арифметика — это живая и все еще успешно развивающаяся отрасль науки, сыгравшая важную роль в мировой истории. В этой главе будут проанализированы алгоритмы выполнения операций над различными типами величин: числами с "плавающей точкой", очень большими числами, дробями (рациональными числами), полиномами и степенными рядами. Кроме того, здесь будут рассмотрены связанные с ними вопросы, такие как преобразование из одной системы счисления в другую, разложение чисел на множители и операции над полиномами.

4.1. ПОЗИЦИОННЫЕ СИСТЕМЫ СЧИСЛЕНИЯ

Способ выполнения арифметических операций тесно связан со способом представления чисел, над которыми выполняются операции. Поэтому резонно начать изучение предмета с обсуждения принципиальных подходов к представлению чисел.

Позиционное представление *с основанием* b (*или по основанию* b) определяется правилом

$$(\ldots a_3a_2a_1a_0.a_{-1}a_{-2}\ldots)_b = \cdots + a_3b^3 + a_2b^2 + a_1b^1 + a_0 + a_{-1}b^{-1} + a_{-2}b^{-2} + \cdots; \qquad (1)$$

например, $(520.3)_6 = 5\cdot 6^2 + 2\cdot 6^1 + 0 + 3\cdot 6^{-1} = 192\frac{1}{2}$. Традиционная десятичная система — это, разумеется, частный случай, когда b равно десяти, а значения a_k выбираются из "десятичных цифр" 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; в этом случае индекс b в (1) может быть опущен.

Простейшие обобщения десятичной системы получаются, когда в качестве b берется целое число, большее 1, а в качестве a_k — целые числа из интервала $0 \le a_k < b$. Таким образом приходим к стандартной двоичной ($b = 2$), троичной ($b = 3$), четверичной ($b = 4$), пятеричной ($b = 5$), ... системам счисления. В общем случае в качестве b можно взять любое ненулевое число, а числа a_k выбирать из произвольного заранее заданного ряда чисел. Как мы увидим далее, это приводит к некоторым интересным ситуациям. Точка между a_0 и a_{-1} в (1) называется *позиционной* или *разделяющей*. (Если $b = 10$, точка также называется десятичной; в случае, когда $b = 2$, она иногда называется двоичной точкой и т. д.). В странах Европейского *континента* (Великобритания, как известно, себя к таковым не относит) вместо разделяющей точки часто используется запятая.

Числа a_k в (1) называются *цифрами* представления. Цифру a_k с бóльшим k называют более значимой, чем a_k с меньшим k; крайнюю слева, или "ведущую", цифру называют *наиболее значимой*, а крайнюю справа, или "хвостовую", — *наименее значимой*. В стандартной двоичной системе двоичные цифры зачастую называют *битами*; в стандартной шестнадцатеричной системе (с основанием шестнадцать) шестнадцатеричные цифры от нуля до пятнадцати обычно обозначаются так:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

История развития способов представления чисел — увлекательная повесть, так как их развитие происходит параллельно развитию самой цивилизации. Однако при подробном рассмотрении этой истории мы ушли бы далеко в сторону от главной темы; тем не менее полезно конспективно изложить основные ее моменты.

Наиболее ранние формы представления чисел, обнаруженные в древних цивилизациях, обычно основываются на использовании групп пальцев, кучек камней и т. п. с дополнительными соглашениями о замене некоторой группы, скажем, из пяти или десяти объектов одним объектом специального вида или объектом, расположенным в специальном месте. Подобные системы естественно приводят к наиболее ранним из известных способам представления чисел в письменном виде, таким как вавилонские, египетские, греческие, китайские и римские числа, но такого рода обозначения чрезвычайно неудобны для выполнения арифметических операций, кроме разве что простейших случаев.

Глубокий анализ древних клинописных табличек, обнаруженных археологами на Среднем Востоке, который выполнен историками математики в 20 столетии, показал, что вавилоняне применяли фактически две различные системы представления чисел. Числа, которые использовались при ведении повседневных деловых записей, записывались при помощи унаследованных от более ранних цивилизаций Месопотамии обозначений, основанных на группировании по десяткам, сотням и т. д. При этом необходимость в операциях с большими числами возникала редко. При решении более сложных математических задач вавилонские математики широко использовали шестидесятеричную (по основанию шестьдесят) позиционную систему, достаточно хорошо разработанную к 1750 г. до н. э. Эта числовая система была уникальной в том смысле, что она фактически была формой представления с *плавающей точкой* с опущенным показателем степени; соответствующий масштабный множитель, или степень шестидесяти, определялся из контекста, так что, например, числа 2, 120, 7200, $\frac{1}{30}$ и т. д. записывались одинаково. Особенно удобно было пользоваться этой системой для умножения и деления при помощи вспомогательных таблиц, поскольку выравнивание порядков никак не влияло на ответ. Примером такой вавилонской системы записи может служить выписка из древних таблиц "Квадрат 30 есть 15" (что можно прочесть, как "Квадрат $\frac{1}{2}$ есть $\frac{1}{4}$"; "Число, обратное $81 = (1\ 21)_{60}$, равно $(44\ 26\ 40)_{60}$"; "Квадрат этого числа равен $(32\ 55\ 18\ 31\ 6\ 40)_{60}$"). У вавилонян был символ для обозначения нуля, но из-за их идеологии обращения с плавающей точкой он использовался между цифрами, но никогда — в крайней справа позиции для обозначения масштаба. Об интересной истории ранней вавилонской математики можно прочесть в книгах O. Neugebauer, *The Exact Sciences in Antiquity* (Princeton, N. J.: Princeton University Press, 1952), B. L. van der Waerden, *Science Awakening*, переведенной на английский A. Дрезденом (A. Dresden) (Groningen: P. Noordhoff, 1954)*, а также D. E. Knuth, *CACM* **15** (1972), 671–677; **19** (1976), 108.

Позиционная система с фиксированной точкой, очевидно, впервые появилась в Центральной Америке у индейцев Майя около 2 000 лет тому назад. Их система счисления по основанию двадцать была достаточно хорошо продуманной, особенно если учесть потребности в записи астрономических наблюдений и календарных дат.

Индейцы Центральной Америки ввели в употребление письменный знак для нуля около 200 г. н. э. Однако испанские завоеватели уничтожили почти все книги Майя по истории и науке, поэтому нам трудно судить об уровне абстракции, достигнутом аборигенами Америки в арифметике. Были найдены таблицы умножения специального назначения, но не обнаружено никаких примеров по делению. (См. J. Eric S. Thompson, *Contributions to Amer. Anthropology and History* **7** (Carnegie Inst. of Washington, 1941), 37–67; J. Justeson, "Ancient Mesoamerican computing practices", *History of Science* **3** (Rome: Istituto della Enciclopedia Italiana), в печати.)

За несколько столетий до новой эры греки применяли для своих вычислений раннюю разновидность счетной доски (абаки), используя песок и/или гальку на доске с начерченными строками и столбцами, которые естественным образом соответствуют нашей десятичной системе. Нам, привыкшим выполнять расчеты при помощи карандаша и бумаги, скорее всего, покажется странным, что тот же позиционный принцип никогда не применялся ими для записи чисел, ведь мы так к нему

* Имеется русский перевод книги: Б. Л. Ван дер Варден, Пробуждающаяся наука, Физматгиз, 1959. — *Прим. перев.*

привыкли. Но бóльшая простота вычислений на абаке (писать тогда умели далеко не все; кроме того, вычисления на абаке делали ненужным запоминание таблиц сложения и умножения), вероятно, привела греков к убеждению, что нелепо даже предполагать, что вычисления удобнее выполнять, "царапая на бумаге". В то же время греческие астрономы для записи дробей использовали шестидесятеричную систему счисления, чему они научились у вавилонян.

Привычная нам десятичная система, отличающаяся от более ранних форм прежде всего наличием фиксированной разделяющей точки, а также использованием символа нуля для обозначения пустой позиции, впервые появилась в Индии. Точная дата возникновения этой системы неизвестна, но есть основания полагать, что это произошло около 6 в. н. э. Индусская наука того времени достигла довольно высокого уровня развития, в частности это относится к астрономии. В наиболее ранних известных индийских манускриптах, в которых применяется десятичная система, числа записываются в обратном порядке (с наиболее значимой цифрой справа), но позднее стало правилом расположение наиболее значимой цифры слева.

Около 750 г. н. э. на арабский язык было переведено несколько важных работ индусских математиков, и принципы десятичной арифметики таким образом попали в Персию. Живописное описание этого периода можно найти в древнееврейской рукописи Абрахама Ибн Эзра (Abraham Ibn Ezra), перевод которой на английский язык опубликован в журнале *AMM* **25** (1918), 99–108. Вскоре после этого аль-Хорезми написал на арабском языке свое руководство. (Как отмечалось в главе 1, слово "алгоритм" произошло от его имени.) Книга аль-Хорезми была переведена на латынь и оказала значительное влияние на Леонардо Пизано (Фибоначчи) (Leonardo Pisano (Fibonacci)), чья книга по арифметике (1202 г.), в свою очередь, сыграла решающую роль в распространении индо-арабских методов работы с числами в Европе. Интересно отметить, что в результате такого двойного перевода порядок записи чисел (слева направо) не изменился, хотя арабы пишут справа налево, а индусы и европейцы — слева направо. Подробно процесс распространения десятичной нумерации и арифметики по всей Европе с 1200 по 1600 год описан в книге David Eugene Smith *History of Mathematics* **1** (Boston: Ginn and Co., 1923), гл. 6 и 8.

Вначале десятичная система счисления применялась только к целым числам (для операций с дробями она не использовалась). Арабские астрономы, которым для составления карт звездного неба и других астрономических таблиц приходилось применять дроби, продолжали пользоваться системой знаменитого греческого астронома Птолемея, основанной на шестидесятеричных дробях. Эта система единиц — рудимент шестидесятеричной системы вавилонян — дожила до наших дней и используется для измерения угловых градусов, минут и секунд, а также для некоторых единиц измерения времени. Использовались первыми европейскими математиками и шестидесятеричные дроби, когда приходилось иметь дело с нецелыми числами. К примеру, Фибоначчи приводил значение

$$1^\circ\ 22'\ 7''\ 42'''\ 33^{IV}\ 4^{V}\ 40^{VI}$$

в качестве приближенного корня уравнения $x^3+2x^2+10x=20$. (Правильный ответ: $1^\circ\ 22'\ 7''\ 42'''\ 33^{IV}\ 4^{V}\ 38^{VI}\ 30^{VII}\ 50^{VIII}\ 15^{IX}\ 43^{X}\ \ldots$.)

Кажется, не так уж много нужно изменить, чтобы использовать десятичные обозначения для десятых, сотых и других угловых и временны́х параметров, но,

конечно, ломать традиции всегда трудно, тем более что шестидесятеричные дроби имеют преимущество перед десятичными дробями в том, что такие числа, как $\frac{1}{3}$, могут быть записаны точно и просто.

Китайские математики — кстати, никогда не пользовавшиеся шестидесятеричной системой, — вероятно, были первыми, кто стал работать с величинами, эквивалентными десятичным дробям, хотя их числовая система (без нуля) и не была в строгом смысле позиционной. Китайские единицы мер и весов были десятичными, так что Цзу Чун-Чи (умер в 501 г.) смог аппроксимировать число π в следующем виде:

3 чана, 1 чжи, 4 цуня, 1 фэн, 5 ли, 9 хао, 2 мяо, 7 ху.

Здесь чан, ..., ху — единицы длины; 1 ху (диаметр шелковой нити) равен 1/10 мяо и т. д. Использование дробей, столь похожих на десятичные, получило в Китае довольно широкое распространение после примерно 1250 года. Начальная форма истинно десятичных дробей появилась в 10 веке в трактате по арифметике, написанном в Дамаске неизвестным математиком, который подписался именем "аль-Уклидиси" (последователь Евклида). Он ввел обозначение места размещения десятичной точки, в частности в связи с проблемой умножения 135 на $(1.1)^n$ для $1 \leq n \leq 5$. (См. A. S. Saidan, *The Arithmetic of al-Uqlīdisī* (Dordrecht: D. Reidel, 1975), 110, 114, 343, 355, 481–485.) Но аль-Уклидиси не развил идею до конца, и его трактат вскоре был забыт. Из датированного 1172 годом трактата аль-Самайяля, который жил и работал в Багдаде и Баку, следует, что ему был известен способ вычисления $\sqrt{10} = 3.162277\ldots$, но он не нашел подходящего способа записи результата. Через несколько столетий десятичные дроби были заново открыты арабским (среднеазиатским) математиком аль-Каши, умершим в 1429 году*. Аль-Каши был весьма искусен в выполнении всяческих вычислений и нашел следующее значение для 2π, содержащее 16 правильных десятичных знаков.

Целые		Дроби														
0	6	2	8	3	1	8	5	3	0	7	1	7	9	5	8	6	5

Это было наилучшее приближение к π до тех пор, пока Лудольф ван Цейлен (Ludolph van Ceulen), выполнив в течение 1596–1610 годов огромный объем работ, не вычислил 35 десятичных знаков.

Самый ранний пример операций с десятичными дробями в Европе обнаружен в одном тексте, написанном в 15 веке, где, например, 153.5 умножается на 16.25 и в ответе получается 2494.375. Используемый при этом метод был назван "турецким". В 1525 году Кристоф Рудольф (Christof Rudolff) из Германии самостоятельно открыл десятичные дроби, но, как и работа аль-Уклидиси, его работа осталась практически незамеченной. Франсуа Виет (François Viète) эту идею высказал снова в 1579 году. Наконец, в 1585 году приобрел популярность трактат по арифметике, написанный Симоном Стевином (Simon Stevin) из Бельгии, который самостоятельно пришел к идее десятичных дробей. Работа Стевина и последовавшее вскоре открытие логарифмов привели к тому, что в течение 17 века десятичные дроби стали общепринятыми повсюду в Европе. [Подробности и ссылки на литературу можно

* В БСЭ приведен другой год смерти — около 1436–1437. — *Прим. перев.*

найти в книгах D. E. Smith, *History of Mathematics* **2** (Boston: Ginn and Co., 1925), 228–247, и V. J. Katz, *A History of Mathematics* (New York: HarperCollins, 1993).]

Двоичная система счисления имеет собственную интересную историю. Известно, что многие племена, ведущие в наше время первобытный образ жизни, используют двоичную, или парную, систему счета (группирование по два, а не по пять или десять), но было бы ошибкой утверждать, что они действительно считают в двоичной системе, так как при этом степени двойки никоим образом не выделяются. Интересные подробности о примитивных числовых системах можно найти в статье Abraham Seidenberg *The Diffusion of Counting Practices Univ. of Calif. Publ. in Math.* **3** (1960), 215–300. Другой "примитивный" пример двоичной системы — музыкальная нотация (для обозначения ритма и длительности нот). В 17 веке предметом обсуждения в Европе были недесятичные системы. В течение многих лет астрономы от случая к случаю использовали шестидесятеричную арифметику как для целых, так и для дробных частей чисел, главным образом, при выполнении умножения (см. John Wallis, *Treatise of Algebra* (Oxford, 1685), 18–22, 30). Тот факт, что *любое* целое, большее 1, может служить основанием для системы счисления, впервые опубликован около 1658 года Блезом Паскалем (Blaise Pascal) в *De Numeris Multiplicibus.* [См. полное собрание сочинений Паскаля *Œuvres Complètes* (Paris: Éditions du Seuil, 1963), 84–89.] Паскаль писал: "Denaria enim ex instituto hominum, non ex necessitate naturæ ut vulgus arbitratur, et sane satis inepte, posita est", т. е. "Десятичная система построена довольно неразумно, в соответствии с человеческими обычаями, а вовсе не с требованиями естественной необходимости, как склонно думать большинство людей". Он утверждал, что было бы желательно перейти к двенадцатеричной (по основанию двенадцать) системе, и предложил правило проверки делимости двенадцатеричного числа на 9. Эрхард Вайгель (Erhard Weigel) в ряде публикаций начиная с 1673 года пытался пробудить интерес к четверичной (по основанию четыре) системе счисления. Подробное обсуждение арифметики по основанию двенадцать было проведено Джошуа Джордэйном (Joshua Jordaine) в работе *Duodecimal Arithmetick* (London, 1687).

Хотя в арифметике на протяжении всей этой эпохи применялась почти исключительно десятичная система счисления, системы мер и весов редко основывались (если вообще основывались) на кратности десяти, и для ведения многих торговых операций требовалась изрядная доля умения складывать величины наподобие фунтов, шиллингов и пенсов. Поэтому на протяжении столетий купцы учились складывать и вычитать величины, выражаемые специфическими денежными единицами, единицами мер и весов, а это фактически была арифметика в недесятичной системе счисления. Особого внимания заслуживают, в частности, основные единицы измерения объема жидкости в Англии, установившиеся еще в 13 веке или даже раньше.

$$\begin{aligned}
2 \text{ джила} &= 1 \text{ полуштоф} \\
2 \text{ полуштофа} &= 1 \text{ пинта} \\
2 \text{ пинты} &= 1 \text{ кварта} \\
2 \text{ кварты} &= 1 \text{ потл} \\
2 \text{ потла} &= 1 \text{ галлон} \\
2 \text{ галлона} &= 1 \text{ пек}
\end{aligned}$$

2 пека = 1 полубушель
2 полубушеля = 1 бушель или фиркин
2 фиркина = 1 килдеркин
2 килдеркина = 1 баррель
2 барреля = 1 хогсхед
2 хогсхеда = 1 пайп
2 пайпа = 1 тан

Объемы жидкости, выраженные в галлонах, потлах, квартах, пинтах и т. д., по существу, записывались в двоичной системе. Быть может, подлинными изобретателями двоичной арифметики были английские виноторговцы!

Насколько сейчас известно, впервые чисто двоичная система счисления появилась в 1605 году в нескольких неопубликованных работах Томаса Хэрриота (Thomas Harriot) (1560–1621). Хэрриот был личностью творческой и приобрел известность по прибытии в Америку в качестве представителя сэра Уолтера Рэлея (Walter Raleigh). Он изобрел (среди всего прочего) символы для обозначения отношений "меньше" и "больше", используемые и ныне, но по некоторым соображениям предпочел не публиковать большинство своих открытий. Выдержки из его заметок по двоичной арифметике воспроизведены Джоном У. Ширли (John W. Shirley) в журнале *Amer. J. Physics* **19** (1951), 452–454. Впервые заметки Хэрриота относительно двоичной системы были процитированы Морли (Morley) и появились в журнале *The Scientific Monthly* **14** (1922), 60–66.

Первый опубликованный анализ двоичной системы появился в работе испанского священника Хуана де Карамюэля Лобковица (Juan de Caramuel Lobkowitz) *Mathesis Biceps* **1** (Campaniæ, 1670), 45–48. Карамюэль рассмотрел представление чисел в системах по основаниям 2, 3, 4, 5, 6, 7, 8, 9, 10, 12 и 60, но не привел никаких примеров арифметических операций в недесятичных системах, кроме шестидесятеричной системы.

Наконец, статья Г. В. Лейбница (G. W. Leibniz) *Mémoires de l'Académie Royale des Sciences* (Paris, 1703), 110–116, в которой были описаны сложение, вычитание, умножение и деление в двоичной системе, действительно привлекла к этой системе всеобщее внимание, и именно на эту статью ссылаются, говоря о рождении арифметики по основанию 2. Лейбниц и в дальнейшем очень часто обращался к двоичной системе счисления. Он не рекомендовал ее для практических вычислений, однако подчеркивал важность этой системы в теории чисел, так как закономерности поведения числовых последовательностей часто гораздо легче усмотреть в двоичной записи, чем в десятичной. Он также вкладывал некий мистический смысл в тот факт, что все в мире можно выразить с помощью нулей и единиц. Неопубликованные работы Лейбница показывают, что он интересовался двоичной системой счисления еще в 1679 году, когда ссылался на нее как на систему "bimal" (аналогично "десятичной").

Подробное исследование ранних работ Лейбница по двоичным числам выполнено Гансом Й. Захером (Hans J. Zacher) в работе *Die Hauptschriften zur Dyadik von G. W. Leibniz* (Frankfurt am Main: Klostermann, 1973). Захер отмечал, что, предложив способ вычислений при помощи камней, ориентированных на использование абаки по основанию 2, Лейбниц стал близок к так называемой "локаль-

ной арифметике" Джона Непера. Непер в 1617 году опубликовал идею локальной арифметики в приложении к своей маленькой книге *Rhabdologia*. Эта идея может трактоваться как первая в мире "двоичная машина", тем более что она была самой дешевой машиной в мире, хотя Непер и чувствовал, что это не машина, а, скорее, игрушка. (См. обзор Мартина Гарднера (Martin Gardner) *Knotted Doughnuts and Other Mathematical Entertainments* (New York: Freeman, 1986), гл. 8.)

Интересно отметить, что в то время важная концепция отрицательных степеней числа справа от разделяющей точки еще не была по-настоящему осознана. Лейбниц попросил Якова Бернулли (James Bernoulli) вычислить π в двоичной системе счисления, и Бернулли решил задачу. Он вычислил 35-значное приближение к π, умножил его на 10^{35}, а затем, выразив полученное целое число в двоичной системе счисления, получил искомый ответ! Для меньшего масштабного множителя это рассуждение выглядело бы как $\pi \approx 3.14$, а $(314)_{10} = (100111010)_2$; следовательно, π в двоичной системе счисления равно 100111010! (См. Leibniz, *Math. Schriften*, edited by K. Gehrhardt, **3** (Halle, 1855), 97; из-за ошибок в вычислениях два из 118 бит в ответе неверны.) Бернулли, скорее всего, выполнил эти вычисления, чтобы в таком представлении числа π выявить какие-либо простые закономерности.

Шведский король Карл XII, математический талант которого, возможно, превосходил таланты всех остальных королей в мировой истории, около 1717 года увлекся восьмеричной арифметикой. Скорее всего, это было его собственное изобретение, хотя он и встречался с Лейбницем в 1707 году. Карлу казалось, что основание 8 или 64 было бы более удобным для вычислений, чем 10, и он собирался ввести восьмеричную систему в Швеции, но погиб в битве, так и не успев провести эту реформу. (См. *Сочинения Вольтера (Voltaire)* **21** (Paris, E. R. DuMont, 1901), 49; E. Swedenborg, *Gentleman's Magazine* **24** (1754), 423–424.)

Восьмеричная система счисления была также предложена в Американских колониях Хью Джонсом (Hugh Jones), профессором колледжа "Уильям и Мэри" около 1750 года (см. *Gentleman's Magazine* **15** (1745), 377–379; H. R. Phalen, *AMM* **56** (1949), 461–465).

Спустя столетие выдающийся американский инженер, швед по национальности, Джон Нистром (John W. Nystrom) решил сделать еще один шаг в развитии идей Карла XII и предложил полную систему нумерации, мер и весов, основанную на шестнадцатеричной арифметике. Он писал: "Я не боюсь и не колеблюсь выступить в защиту двоичной системы в арифметике и метрологии. Я знаю, на моей стороне природа; если мне не удастся убедить вас в ее полезности и чрезвычайной важности для человечества, это не сделает чести ни нашему поколению, ни нашим ученым и философам". Нистром разработал специальные правила произношения шестнадцатеричных чисел; например, $(\texttt{C0160})_{16}$ следовало читать как "вибонг, бисантон" (vybong, bysanton). Полная система была им названа тональной и описана в *J. Franklin Inst.* **46** (1863), 263–275, 337–348, 402–407. Аналогичная система, но использующая основание 8, была предложена Альфредом Б. Тэйлором (Alfred B. Taylor) и описана в *Proc. Amer. Pharmaceutical Assoc.* **8** (1859), 115–216; *Proc. Amer. Philosophical Soc.* **24** (1887), 296–366.

Со времен Лейбница двоичная система счисления становится хорошо известной диковинкой, и Р. К. Арчибальд (R. C. Archibald) собрал более 20 посвященных ей ранних работ [*AMM* **25** (1918), 139–142]. Она применялась, главным образом, для

вычисления степеней, как будет объяснено в разделе 4.6.3, а также при анализе некоторых игр и головоломок. Джузеппе Пеано (Giuseppe Peano) использовал двоичную систему как базис "логического" алфавита из 256 символов [*Atti della R. Accademia delle Scienze di Torino* **34** (1898), 47–55]. Джозеф Боуден (Joseph Bowden) предложил систему обозначений для шестнадцатеричных чисел [*Special Topics in Theoretical Arithmetic* (Garden City, 1936), 49].

В книге Антона Глэйзера (Anton Glaser) *History of Binary and Other Nondecimal Numeration* (Los Angeles: Tomash, 1981) приведена подробная информация и практически исчерпывающий анализ развития двоичной системы, включая перевод на английский язык многих работ, процитированных выше (см. *Historia Math.* **10** (1983), 236–243).

Большинство современных числовых систем связано с развитием вычислительных машин. Из заметок Чарльза Бэббиджа (Charles Babbage) 1838 года понятно, что он задумывался над использованием в своей аналитической машине недесятичных чисел (см. M. V. Wilkes, *Historia Math.* **4** (1977), 421). Повышенный интерес к механическим устройствам для выполнения арифметических операций, особенно умножения, побудил в 30-х годах ряд исследователей обратить внимание на двоичную систему. Прекрасный отчет об этих исследованиях вместе с записью дискуссии, состоявшейся после прочитанной им на данную тему лекции приведен в статье Э. Уильяма Филлипса (E. William Phillips) "Двоичные вычисления" [*Journal of the Institute of Actuaries* **67** (1936), 187–221]. Филлипс начал статью словами "Конечная цель (этой статьи) состоит в том, чтобы убедить весь цивилизованный мир отказаться от десятичной нумерации и заменить ее восьмеричной".

Современные читатели статьи Филлипса, возможно, будут удивлены, обнаружив, что система счисления по основанию 8 называлась в то время в соответствии со всеми словарями английского языка "octonary" или "octonal", точно так, как система по основанию 10 называлась "denary" или "decimal". Слово "octal" появилось в английском языке только после 1961 года, причем первоначально, скорее всего, как термин для описания конструкции цоколя определенного класса вакуумных электронных ламп. Слово "hexadecimal", которое вкралось в английский язык еще позже, представляет собой смесь греческого и латинского корней. Более корректными терминами должны были быть либо "senidenary", либо "sedecimal", либо даже "sexadecimal", но последний, пожалуй, для программистов звучал бы слишком рискованно.

Высказывание Ф. Х. Уэйлса, приведенное в качестве одного из эпиграфов к этой главе, извлечено из записи дискуссии, опубликованной вместе со статьей Филлипса. Другой слушатель лекции указал на неудобства восьмеричной системы для деловых целей: 5% "превращается в 3.146$\dot{3}$ per 64, что звучит довольно ужасно"*.

Филлипса вдохновила возможность реализации его идей в устройствах, работающих на электронных лампах и способных выполнять вычисления в двоичном коде [C. E. Wynn-Williams, *Proc. Roy. Soc. London* **A136** (1932), 312–324].

Во второй половине 30-х годов Джон В. Атанасофф (John V. Atanasoff) и Джордж Р. Штибитц (George R. Stibitz) в США, Л. Куффигнал (L. Couffignal) и Р. Валта (R. Valtat) во Франции, а также Гельмут Шрейер (Helmut Schreyer) и

* Поскольку слово "процент" на английском языке пишется как "per cent" (от лат. "per centum"), а "cent" (100) заменяется числом 64. — *Прим. перев.*

Конрад Цузе (Konrad Zuse) в Германии разработали первые электромеханические и электронные машины для выполнения арифметических операций. Все они использовали двоичную систему счисления, хотя позже Штибитц разработал и двоичный код "с избытком 3" для десятичных цифр. Обобщенный обзор этих ранних достижений, включая переводы и копии наиболее значимых современных документов, содержится в книге Brian Randell *The Origins of Digital Computers* (Berlin: Springer, 1973).

В первых быстродействующих вычислительных машинах, созданных в США в начале 40-х годов, использовалась десятичная арифметика. Но в 1946 году в сыгравшем важную роль отчете А. В. Беркса (A. W. Burks), Г. Г. Голдстайна (H. H. Goldstine) и Дж. фон Неймана (J. von Neumann) о проекте первой вычислительной машины с хранимой в памяти программой были подробно изложены причины, которые побудили их порвать с традицией и перейти к системе счисления по основанию 2 (см. John von Neumann, *Collected Works* **5**, 41–65). С тех пор двоичные вычислительные устройства получили всеобщее распространение. После первой дюжины лет работы с двоичными машинами в статье В. Буххольца (W. Buchholz) "Fingers or Fists?" [*CACM* **2** (December, 1959), 3–11] был выполнен анализ сравнительных достоинств и недостатков двоичной системы счисления.

Структура компьютера `MIX`, используемого в этой книге, такова, что машина может быть как двоичной, так и десятичной. Интересно отметить, что почти все программы для этого компьютера можно написать, не зная, какая именно система используется (двоичная или десятичная), даже при выполнении вычислений с многократной точностью. Итак, мы видим, что на методику программирования для компьютера выбор основания системы счисления не оказывает значительного влияния. (Заслуживает упоминания исключение из этого правила — операции "булевой" алгебры, рассматриваемые в разделе 7.1; см. также алгоритм 4.5.2.)

Отрицательные числа могут быть представлены в компьютере несколькими способами. Выбор того или иного способа зачастую оказывает влияние на метод реализации арифметических операций. Поясним сказанное. Сначала будем считать машину `MIX` десятичной; тогда каждое слово состоит из десяти цифр и знака, например

$$-12345\ 67890. \tag{2}$$

Этот способ представления называется *абсолютным значением со знаком**. Такое представление соответствует общепринятым обозначениям, и поэтому его предпочитают многие программисты. Возможное неудобство заключается в том, что допускается существование как "минус нуль", так и "плюс нуль", в то время как эти разные коды должны обозначать одно и то же число. На практике такая возможность требует принятия определенных мер предосторожности.

В большинстве механических счетных машин, выполняющих действия десятичной арифметики, используется другая система записи — *дополнение до десяти***. Если вычесть 1 из 00000 00000, в этой системе записи получим 99999 99999; другими

---

* В общепринятой русскоязычной терминологии этому понятию соответствует термин *прямой код*. — *Прим. перев.*

** В общепринятой русскоязычной терминологии этому понятию соответствует термин *дополнительный код*. — *Прим. перев.*

словами, числу явно не приписывается знак, а вычисления выполняются по модулю 10^{10}. Число $-12345\ 67890$ в форме дополнения до десяти будет выглядеть так:

$$87654\ 32110. \tag{3}$$

В этой системе обозначений принято считать отрицательным любое число, головная цифра которого — 5, 6, 7, 8 или 9, хотя с точки зрения сложения и умножения не будет большим грехом рассматривать (3), если это удобно, как число $+87654\ 32110$. Попутно отметим, что в такой системе не возникает проблема "минус нуль".

На практике основное различие между двумя описанными формами представления заключается в том, что сдвиг вправо дополнения до десяти не эквивалентен делению на 10; к примеру, число $-11 = \ldots 99989$ после сдвига вправо на одну позицию превращается в $\ldots 99998 = -2$ (в предположении, что сдвиг вправо отрицательного числа порождает в головном разряде "9"). В общем случае результатом сдвига числа x, записанного в формате дополнения до десяти, на одну позицию вправо будет число $\lfloor x/10 \rfloor$, независимо от того, положительно или отрицательно число x.

Одним из возможных неудобств записи в формате дополнения до десяти является несимметричность относительно нуля. Наибольшее отрицательное число, представимое p цифрами, есть $500\ldots0$, и оно не является результатом обращения знака никакого p — разрядного положительного числа. Таким образом, возможно, изменение знака, т. е. замена x на $-x$, приведет к переполнению. (См. упр. 7 и 31, в которых обсуждается формат дополнения до основания системы счисления, который имеет *бесконечную* точность.)

Еще одна система обозначений, принятая с самых первых дней эры быстродействующих вычислительных машин, — это представление в виде *дополнения до всех девяток**.

В этом случае число $-12345\ 67890$ записывается в виде

$$87654\ 32109. \tag{4}$$

Каждая цифра отрицательного числа $(-x)$ равна разности между 9 и соответствующей цифрой числа x. Нетрудно видеть, что для отрицательного числа дополнение до девяти всегда на единицу меньше соответствующего дополнения до десяти. Сложение и вычитание производятся по модулю $10^{10} - 1$, а это означает, что перенос из крайней слева позиции добавляется к крайней справа (см. описание арифметики по модулю $w - 1$ в разделе 3.2.1.1). Опять возникает проблема с "минус нулем", так как записи 99999 99999 и 00000 00000 обозначают одно и то же значение.

Только что изложенные идеи для арифметики по основанию 10 в полной мере применимы и к арифметике по основанию 2; здесь мы имеем *абсолютную величину со знаком, дополнение до двух и дополнение до одного***.

Арифметика в дополнительном коде — это арифметика по модулю 2^n, а арифметика в обратном коде — по модулю $2^n - 1$. Машина `MIX` имеет дело только с прямым кодом, что и используется в примерах этой главы. Тем не менее в сопроводитель-

---

* В общепринятой русскоязычной терминологии этому понятию соответствует термин *обратный код*. — *Прим. перев.*

** В общепринятой русскоязычной терминологии этим понятиям соответствуют термины *прямой*, *обратный* и *дополнительный* код, которыми мы будем пользоваться в дальнейшем. — *Прим. перев.*

ном тексте рассматриваются, если в этом есть необходимость, и альтернативные варианты процедур для дополнительного и обратного кодов.

Скрупулезные читатели и редакторы английского текста, вероятно, отметили положение апострофа в терминах "two's complement" (дополнение до *двойки*) и "ones' complement" (дополнение до *единиц*). В первом случае каждая цифра дополняется до первой степени двойки, а во втором весь код есть дополнение до кода, представленного единицами во всех разрядах. По аналогии с "ones' complement" может существовать и формат "twos' complement" (дополнение до *двоек*), который используется в системе счисления по основанию 3 и является дополнением до $(2\ldots22)_3$.

В руководствах по машинному языку программирования часто указывается, что схемотехника компьютера позволяет настраивать конкретное положение разделяющей точки в каждом машинном слове. На это сообщение не стоит обращать внимание. Целесообразнее изучить правила размещения разделяющей точки в результате выполнения каждой конкретной команды, если предположить, что до ее выполнения точки в операндах были расположены в каком-то определенном месте. Например, в случае машины MIX можно было бы рассматривать наши операнды либо как целые числа с разделяющей точкой в крайнем справа положении, либо как правильные дроби с разделяющей точкой в крайнем слева положении, либо как некоторые промежуточные варианты. Правила установки разделяющей точки после осуществления операций сложения, вычитания, перемножения и деления определяются очевидным образом и следуют из алгоритмов выполнения этих операций.

Легко видеть, что между записью чисел в системах по основанию b и b^k существует простая связь:

$$(\ldots a_3a_2a_1a_0.a_{-1}a_{-2}\ldots)_b = (\ldots A_3A_2A_1A_0.A_{-1}A_{-2}\ldots)_{b^k}, \tag{5}$$

где

$$A_j = (a_{kj+k-1}\ldots a_{kj+1}a_{kj})_b$$

(см. упр. 8). Таким образом, получается простой способ перехода "чисто визуального" от, скажем, двоичной системы к шестнадцатеричной.

Помимо стандартных систем по основанию b, обсуждавшихся выше, существует множество других интересных вариантов позиционных систем счисления. Например, можно было бы рассматривать числа по основанию (-10), так что

$$\begin{aligned}(\ldots a_3a_2a_1a_0.a_{-1}a_{-2}\ldots)_{-10}&\\ &= \cdots + a_3(-10)^3 + a_2(-10)^2 + a_1(-10)^1 + a_0 + \cdots\\ &= \cdots - 1000a_3 + 100a_2 - 10a_1 + a_0 - \tfrac{1}{10}a_{-1} + \tfrac{1}{100}a_{-2} - \cdots.\end{aligned}$$

Здесь, как и в традиционной десятичной системе, цифры удовлетворяют неравенствам $0 \le a_k \le 9$. Число 12345 67890 запишется в такой "негадесятичной" системе в виде

$$(1\ 93755\ 73910)_{-10}, \tag{6}$$

так как оно равно как раз 10305070900 − 9070503010. Интересно отметить, что обращение этого числа, −12345 67890, запишется в виде

$$(28466\ 48290)_{-10}. \tag{7}$$

И действительно, любое вещественное число, положительное или отрицательное, может быть представлено без знака в системе по основанию −10.

Системы по отрицательному основанию впервые описаны Витторио Грюнвальдом (Vittorio Grunwald) в *Giornale di Matematiche di Battaglini* **23** (1885), 203–221, 367. В этой работе изложены правила выполнения в таких системах четырех арифметических действий, а также рассмотрены правила извлечения корня, проверка делимости и перевод из одной системы счисления в другую. Однако, похоже, работа Грюнвальда осталась незамеченной, так как она была опубликована в довольно заштатном журнале и вскоре забыта. Следующее исследование по системам счисления по отрицательному основанию опубликовал О. Дж. Кемпнер (A. J. Kempner) в *AMM* **43** (1936), 610–617. В этой работе он рассмотрел свойства систем счисления с нецелыми основаниями и отметил в примечаниях, что системы счисления с отрицательными основаниями также будут иметь право на существование. Двадцать лет спустя эта идея снова была предложена; на этот раз — З. Павляком (Z. Pawlak) и А. Вакуличем (A. Wakulicz) [*Bulletin de l'Académie Polonaise des Sciences*, Classe III, **5** (1957), 233–236; Série des sciences techniques **7** (1959), 713–721], а также Л. Уэйделом (L. Wadel) [*IRE Transactions* **EC-6** (1957), 123]. Экспериментальные вычислительные машины SKRZAT 1 и BINEG, в которых −2 использовалось в качестве основания системы, были сделаны в Польше в конце 50-х годов (см. N. M. Blachman, *CACM* **4** (1961), 257; R. W. Marczyński, *Ann. Hist. Computing* **2** (1980), 37–48). Дополнительные ссылки на литературу приводятся в журналах *IEEE Transactions* **EC-12** (1963), 274–276; *Computer Design* **6** (May, 1967), 52–63. Можно полагать, что идея отрицательного основания возникла независимо сразу у целого ряда авторов. Например, Д. Э. Кнут в небольшом машинописном тексте, предназначенном для конкурса "Поиск научных талантов" среди учеников старших классов, в 1955 году обсуждал системы счисления с отрицательными основаниями. Там же обсуждалось и дальнейшее распространение этой идеи на основания, являющиеся комплексными числами.

Выбор основания $2i$ приводит к интересной системе счисления, которую естественно назвать "мнимочетверичной" (по аналогии с "четверичной"). Такая система обладает необычным свойством, заключающимся в том, что в ней любое комплексное число может быть представлено без знака при помощи цифр 0, 1, 2 и 3. (См. D. E. Knuth, *CACM* **3** (1960), 245–247.) Например,

$$(11210.31)_{2i} = 1\cdot 16 + 1\cdot(-8i) + 2\cdot(-4) + 1\cdot(2i) + 3\cdot(-\tfrac{1}{2}i) + 1(-\tfrac{1}{4}) = 7\tfrac{3}{4} - 7\tfrac{1}{2}i.$$

Здесь число $(a_{2n}\ldots a_1a_0.a_{-1}\ldots a_{-2k})_{2i}$ равно

$$(a_{2n}\ldots a_2a_0.a_{-2}\ldots a_{-2k})_{-4} + 2i(a_{2n-1}\ldots a_3a_1.a_{-1}\ldots a_{-2k+1})_{-4},$$

так что перевод числа в мнимочетверичную форму и обратно сводится к переводу в "негачетверичную" форму и обратно действительной и мнимой частей числа. Интересное свойство этой системы заключается в том, что она позволяет единообразно

выполнять умножение и деление комплексных чисел без разделения действительной и мнимой частей. Например, в этой системе можно перемножить два числа так же, как при любом другом основании, но при этом нужно использовать несколько иное "правило переноса": в случае, если цифра становится больше 3, вычесть 4 — и -1 "перенесется" на два разряда влево; когда же цифра отрицательна, к ней прибавляется 4 и $+1$ "переносится" на два разряда влево. Проиллюстрируем это своеобразное правило переноса следующим примером.

$$\begin{array}{rl}
 & 1\,2\,2\,3\,1 \quad [9-10i] \\
 & 1\,2\,2\,3\,1 \quad [9-10i] \\
\hline
 & 1\,2\,2\,3\,1 \\
1\,0\,3\,2\,0\,2\,1\,3 & \\
 \quad 1\,3\,0\,2\,2 & \\
 1\,3\,0\,2\,2 & \\
1\,2\,2\,3\,1 & \\
\hline
0\,2\,1\,3\,3\,3\,1\,2\,1 & \quad [-19-180i]
\end{array}$$

Аналогичную систему, в которой используются лишь цифры 0 и 1, можно построить и по основанию $\sqrt{2}\,i$. Однако в ней для представления мнимой единицы i требуется бесконечное непериодическое разложение. Витторио Грюнвальд (Vittorio Grunwald) предложил разрешить эту проблему, используя цифры 0 и $1/\sqrt{2}$ в нечетных позициях, однако это фактически испортило всю систему. (См. *Commentari dell'Ateneo di Brescia* (1886), 43–54.)

Используя основание $i-1$, можно также получить "бинарную" комплексную систему счисления, предложенную У. Пенни (W. Penney) [*JACM* **12** (1965), 247–248]:

$$(\ldots a_4a_3a_2a_1a_0.a_{-1}\ldots)_{i-1} = \cdots - 4a_4 + (2+2i)a_3 - 2ia_2 + (i-1)a_1 + a_0 - \tfrac{1}{2}(i+1)a_{-1} + \cdots.$$

В ней задействованы только цифры 0 и 1. Продемонстрировать, что любое комплексное число допускает такое представление, можно, рассмотрев интересное множество S, приведенное на рис. 1. Это множество по определению состоит из всех точек, которые могут быть записаны в виде $\sum_{k\ge1}a_k(i-1)^{-k}$ для бесконечной последовательности a_1, a_2, a_3, ... нулей и единиц. Она известна также как "двуглавый дракон" (см. M. F. Barnsley, *Fractals Everywhere*, second edition (Academic Press, 1993), 306, 310). На рис. 1 показано, что множество S можно разбить на 256 частей, конгруэнтных $\frac{1}{16}S$. Заметим, что если множество S повернуть по часовой стрелке на 135°, то оно распадется на два примыкающих одно к другому множества, конгруэнтных $(1/\sqrt{2})\,S$, поскольку $(i-1)S = S \cup (S+1)$. Детально доказательство того, что множество S содержит все комплексные числа, достаточно малые по модулю, рассмотрено в упр. 18.

Быть может, самой изящной из всех систем счисления является *уравновешенная троичная* система счисления (по основанию 3), в которой вместо цифр 0, 1 и 2 используются "триты" (троичные цифры) -1, 0 и $+1$. Заменив -1 символом $\bar{1}$, получим следующие примеры уравновешенных троичных чисел.

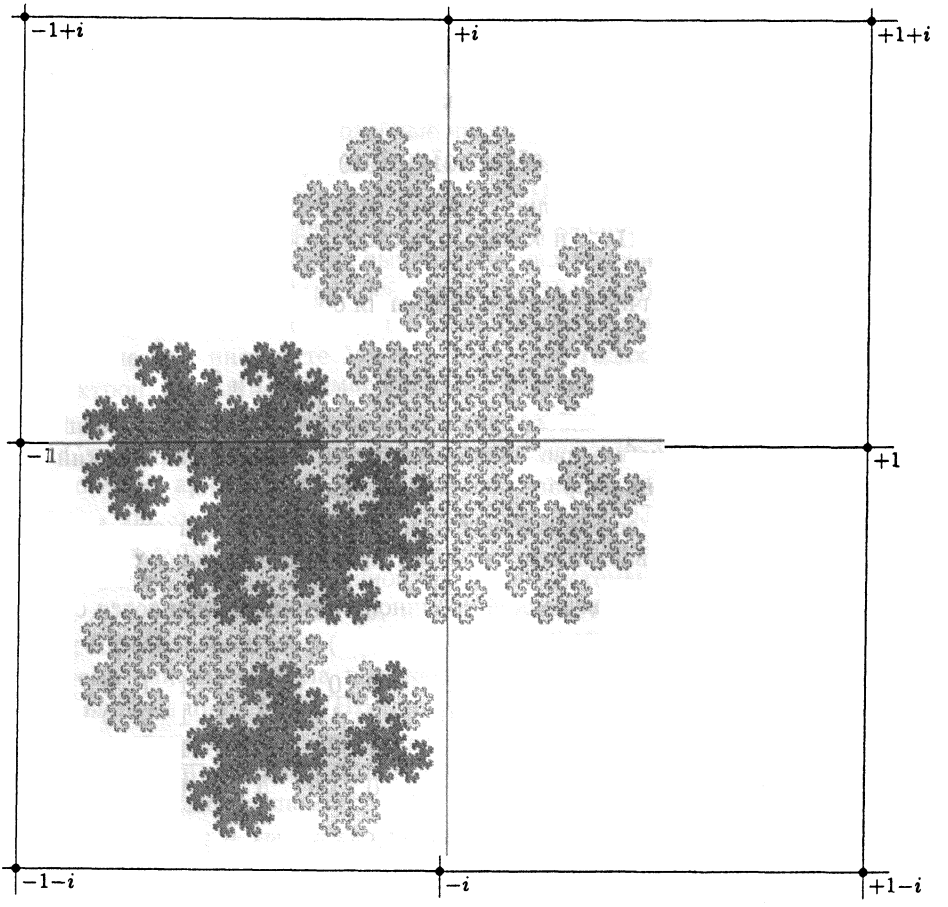


Рис. 1. Фрактальное множество S, называемое "двуглавый дракон".

Уравновешенные троичные числа	Десятичные числа
$10\bar{1}$	8
$11\bar{1}0.\bar{1}\bar{1}$	$32\frac{5}{9}$
$\bar{1}\bar{1}10.11$	$-32\frac{5}{9}$
$\bar{1}\bar{1}10$	-33
$0.11111\ldots$	$\frac{1}{2}$

Один из способов поиска числа в уравновешенной троичной системе состоит в следующем. Сначала запишем число в троичной системе счисления, к примеру

$$208.3 = (21201.022002200220\ldots)_3.$$

(Очень простой способ перевода в троичную систему, пригодный для вычисления вручную с карандашом и бумагой, описан в упр. 4.4.) Затем добавим к нему в троичной системе бесконечное число $\ldots11111.11111\ldots$, после чего получим для

вышеприведенного примера бесконечное число

$$(\ldots 11111210012.210121012101\ldots)_3.$$

Наконец, вычтем $\ldots 11111.11111\ldots$ поразрядно, уменьшая на единицу каждую цифру, и получим

$$208.3 = (10\bar{1}\bar{1}01.10\bar{1}010\bar{1}010\bar{1}0\ldots)_3. \tag{8}$$

Этот процесс можно сделать вполне строгим, если заменить искусственное бесконечное число $\ldots 11111.11111\ldots$ некоторым числом с соответствующим количеством единиц.

Уравновешенная троичная система счисления обладает многими привлекательными свойствами.

a) Отрицание числа осуществляется взаимной заменой 1 и $\bar{1}$.

b) Знак числа задается его наиболее значимым ненулевым тритом; в общем случае можно сравнивать любые два числа, используя лексикографический порядок при чтении слова слева направо, как в десятичной системе.

c) Операция округления до ближайшего целого идентична усечению; другими словами, просто отбрасывается все, что стоит правее разделяющей точки.

Операция сложения в уравновешенной троичной системе выполняется совсем просто, если воспользоваться таблицей сложения.

$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	0	0	0	1	1	1	$\bar{1}$	$\bar{1}$	$\bar{1}$	0	0	0	1	1	1	$\bar{1}$	$\bar{1}$	$\bar{1}$	0	0	0	1	1	1
$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1	$\bar{1}$	0	1
$\bar{1}0$	$\bar{1}1$	$\bar{1}$	$\bar{1}1$	$\bar{1}$	0	$\bar{1}$	0	1	$\bar{1}1$	$\bar{1}$	0	$\bar{1}$	0	1	0	1	$1\bar{1}$	$\bar{1}$	0	1	0	1	$1\bar{1}$	1	$1\bar{1}$	10

(Три входных трита — это триты двух наших слагаемых и трит переноса.) Вычитание состоит в формировании числа, противоположного по знаку вычитаемого, и последующем выполнении сложения. Умножение также сводится к операциям перемены знака и сложения, как в следующем примере.

$$\begin{array}{rl}
 & 1\ \bar{1}\ 0\ \bar{1} \quad [17] \\
 & 1\ \bar{1}\ 0\ \bar{1} \quad [17] \\
\hline
 & \bar{1}\ 1\ 0\ 1 \\
 & \bar{1}\ 1\ 0\ 1\ 0 \\
1\ \bar{1}\ 0\ \bar{1} & \\
\hline
0\ 1\ 1\ \bar{1}\ \bar{1}\ 0\ 1 & \quad [289]
\end{array}$$

Представление чисел в уравновешенной троичной системе неявно присутствует в одной знаменитой математической головоломке, обычно называемой "задача Баше (Bachet) о весах", хотя она была сформулирована еще Фибоначчи за четыре столетия до того, как Баше написал свою книгу, а перс Табари сделал это еще раньше — более чем за 100 лет до Фибоначчи. [См. W. Ahrens, *Mathematische Unterhaltungen und Spiele* **1** (Leipzig: Teubner, 1910), Section 3.4; H. Hermelink, *Janus* **65** (1978), 105–117.]

Позиционные системы счисления с отрицательными цифрами были изобретены Дж. Колсоном (J. Colson) [*Philos. Trans.* **34** (1726), 161–173], затем забыты и вновь

открыты примерно через 100 лет Джоном Лесли (Sir John Leslie) [*The Philosophy of Arithmetic* (Edinburgh, 1817); см. с. 33–34, 54, 64–65, 117, 150] и А. Коши (A. Cauchy) [*Comptes Rendus Acad. Sci. Paris* **11** (1840), 789–798]. Коши отмечал, что отрицательные цифры позволяют избежать необходимости помнить таблицу умножения после 5×5. Утверждение, что подобные числовые системы были давно известны в Индии [Я. Бхарати (J. Bharati), *Vedic Mathematics* (Delhi: Motilal Banarsidass, 1965)], было опровергнуто К. Ш. Шуклой (K. S. Shukla) [*Mathematical Education* **5**, 3 (1989), 129–133]. В "чистом" виде уравновешенная троичная система счисления появилась в статье изобретателя механических вычислительных устройств Леона Лаланна (Léon Lalanne) [*Comptes Rendus Acad. Sci. Paris* **11** (1840), 903–905]. Система оставалась незамеченной до тех пор, пока спустя 100 лет после публикации Лаланна в Электротехническом институте Мура в 1945–1946 годах не стали разрабатывать первые электронные вычислительные машины. В то время она наряду с двоичной системой серьезно рассматривалась в качестве возможной альтернативы десятичной системе. Сложность электронных схем арифметических устройств для уравновешенной троичной арифметики не намного выше, чем для двоичной системы, а чтобы задать число, в ней требуется лишь $\ln 2 / \ln 3 \approx 63\%$ цифровых позиций от того количества, которое необходимо для представления чисел в двоичной системе. Дискуссии по поводу уравновешенной троичной системы счисления опубликованы в журнале *AMM* **57** (1950), 90–93, и в сборнике *High-speed Computing Devices*, Engineering Research Associates (McGraw-Hill, 1950), 287–289. Уравновешенная троичная система счисления была положена в основу экспериментальной советской вычислительной машины СЕТУНЬ (см. *CACM* **3** (1960), 149–150)*.

Возможно, симметричные свойства и простая арифметика этой системы счисления окажутся в один прекрасный день весьма существенными (когда "флип-флоп" заменится "флип-флэп-флопом")**.

Еще одно важное обобщение позиционного способа представления чисел — это позиционная система со *смешанным основанием*. Если дана последовательность чисел $\langle b_n \rangle$, где n могут быть отрицательными, то по определению полагается

$$\begin{bmatrix} \ldots, a_3, a_2, a_1, a_0; a_{-1}, a_{-2}, \ldots \\ \ldots, b_3, b_2, b_1, b_0; b_{-1}, b_{-2}, \ldots \end{bmatrix} = \cdots + a_3 b_2 b_1 b_0 + a_2 b_1 b_0 + a_1 b_0 + a_0 + a_{-1}/b_{-1} + a_{-2}/b_{-1} b_{-2} + \cdots . \tag{9}$$

В простейших системах со смешанным основанием используются только целые числа; b_0, b_1, b_2, ... полагаются целыми числами, бóльшими единицы, и рассматриваются только такие числа, которые не содержат разделяющей точки, причем a_n должно принадлежать интервалу $0 \le a_n < b_n$.

Одна из наиболее важных систем со смешанным основанием — это *факториальная система счисления*, где $b_n = n+2$. С ее помощью можно единственным образом

---

* См. также Бруснецов Н. П. и др. *Малая цифровая вычислительная машина "Сетунь"*. — М., 1965. — *Прим. перев.*

** Здесь в оригинале — игра слов. Словосочетание "flip-flop" (дословно "щелчок-шлепок") означает в английской технической литературе элемент с двумя устойчивыми состояниями. По аналогии "flip-flap-flop" (дословно "щелчок-хлопок-шлепок") должно означать элемент с тремя устойчивыми состояниями. — *Прим. перев.*

представить любое неотрицательное целое число в виде

$$c_n\, n! + c_{n-1}\, (n-1)! + \cdots + c_2\, 2! + c_1, \tag{10}$$

где $0 \le c_k \le k$ для $1 \le k \le n$ и $c_n \neq 0$ (см. алгоритм 3.3.2).

Системы со смешанным основанием часто встречаются в нашей повседневной жизни; речь идет о единицах измерения. Например, величина "3 недели, 2 дня, 9 часов, 22 минуты, 57 секунд и 492 миллисекунды" может быть представлена в виде

$$\begin{bmatrix} 3, & 2, & 9, & 22, & 57; & 492 \\ & 7, & 24, & 60, & 60; & 1000 \end{bmatrix} \text{ секунд.}$$

Величина "10 фунтов, 6 шиллингов и три с половиной пенса" до перехода Великобритании к десятичной системе в денежных расчетах есть не что иное, как $\begin{bmatrix} 10, & 6, & 3; & 1 \\ & 20, & 12; & 2 \end{bmatrix}$ британских пенсов.

Числа со смешанным основанием можно складывать и вычитать, применяя очевидное обобщение обычных алгоритмов сложения и вычитания при условии, что для обоих операндов используется одна и та же система (см. упр. 4.3.1). Подобным образом можно легко умножать или делить числа со смешанным основанием на малые целые константы, используя простое расширение общеизвестных приемов счета карандашом на бумаге. В общем виде системы со смешанным основанием впервые были проанализированы Георгом Кантором (Georg Cantor) [*Zeitschrift für Math. und Physik* **14** (1869), 121–128]. Дополнительная информация о таких системах содержится в упр. 26 и 29.

У. Перри (W. Parry) исследовал некоторые вопросы, относящиеся к *иррациональным* основаниям (см. *Acta Math. Acad. Sci. Hung.* **11** (1960), 401–416).

Помимо описанных здесь систем счисления, существует несколько других способов представления чисел, которые упоминаются в этой серии книг: биномиальная система счисления (упр. 1.2.6–56), система Фибоначчи (упр. 1.2.8–34, 5.4.2–10), ϕ-система (упр. 1.2.8–35), модульное представление (раздел 4.3.2), код Грея (раздел 7.2.1) и система с римскими цифрами (раздел 9.1).

УПРАЖНЕНИЯ

1. [*15*] Выразите числа $-10, -9, \ldots, 9, 10$ в системе счисления по основанию -2.

▶ **2.** [*24*] Рассмотрите следующие четыре системы счисления: (a) двоичную (прямой код), (b) негадвоичную (основание -2), (c) уравновешенную троичную и (d) систему по основанию $b = \frac{1}{10}$. Используйте эти четыре системы для представления каждого из трех чисел: (i) -49; (ii) $-3\frac{1}{7}$ (укажите период); (iii) π (несколько значащих цифр).

3. [*20*] Выразите $-49 + i$ в мнимочетверичной системе.

4. [*15*] Предположим, имеется MIX-программа, в ячейке памяти A которой находится число, разделяющая точка которого расположена между 3- и 4-м байтами, а в ячейке памяти B — число, разделяющая точка которого расположена между 2- и 3-м байтами. (Крайний слева байт имеет номер 1.) Где будет располагаться разделяющая точка в регистрах A и X после выполнения команд

(a) `LDA A; MUL B` (b) `LDA A; SRAX 5; DIV B` ?

5. [*00*] Объясните, почему представление отрицательного целого числа в обратном коде всегда на единицу меньше представления в дополнительном коде, если рассматривать эти представления как положительные числа.

6. [*16*] Каковы наибольшее и наименьшее p — битовые целые числа, которые могут быть представлены в двоичной системе посредством (a) прямого кода, (b) обратного кода, (c) дополнительного кода?

7. [*M20*] В тексте раздела десятичное представление с дополнением до десяти определено только для целых чисел, записанных в одном машинном слове. Можно ли аналогично определить представление в этом же формате *для всех действительных чисел*, имеющих "бесконечную точность"? Существует ли подобный способ определения десятичного представления в обратном коде для всех действительных чисел?

8. [*M10*] Докажите соотношение (5).

▶ **9.** [*15*] Переведите следующие *восьмеричные* числа в *шестнадцатеричные*, используя шестнадцатеричные цифры `0`, `1`, ..., `9`, `A`, `B`, `C`, `D`, `E`, `F`: *12*; *5655*; *2550276*; *76545336*; *3726755*.

10. [*M22*] Обобщите соотношение (5) для систем со смешанным основанием, как в соотношении (9).

11. [*22*] Разработайте алгоритм для вычисления суммы чисел $(a_n \ldots a_1 a_0)_{-2}$ и $(b_n \ldots$ $\ldots b_1 b_0)_{-2}$, дающий результат в виде $(c_{n+2} \ldots c_1 c_0)_{-2}$, с помощью системы счисления по основанию -2.

12. [*23*] Разработайте алгоритм перевода (a) числа, записанного в прямом двоичном коде $\pm(a_n \ldots a_0)_2$, в негадвоичное представление $(b_{n+1} \ldots b_0)_{-2}$ и (b) негадвоичного представления $(b_{n+1} \ldots b_0)_{-2}$ в прямой двоичный код $\pm(a_{n+1} \ldots a_0)_2$.

▶ **13.** [*M21*] Существуют числа, имеющие два различных разложения в бесконечную десятичную дробь, например $2.3599999\ldots = 2.3600000\ldots$. Единственно ли представление чисел в *негадесятичной* (по основанию -10) системе счисления или существуют также вещественные числа по этому основанию с двумя различными бесконечными разложениями?

14. [*14*] Умножьте $(11321)_{2i}$ само на себя в мнимочетверичной системе, используя метод, описанный выше.

15. [*M24*] Как выглядят множества $S = \{\sum_{k\geq 1} a_k b^{-k} \mid a_k$ допустимая цифра$\}$, аналогичные множеству, представленному на рис. 1, в негадесятичной и мнимочетверичной системах счисления?

16. [*M24*] Постройте алгоритм сложения 1 с $(a_n \ldots a_1 a_0)_{i-1}$ в системе счисления по основанию $i-1$.

17. [*M30*] Может показаться странным, что в качестве основания в системе счисления берется число $i-1$, а не аналогичное, но более простое число $i+1$. Всякое ли комплексное число $a+bi$ допускает "двоичное" представление в позиционной системе по основанию $i+1$?

18. [*HM32*] Покажите, что множество S, представленное на рис. 1, есть замкнутое множество, содержащее некоторую окрестность начала координат. (Следовательно, любое комплексное число допускает "двоичное" представление по основанию $i-1$.)

▶ **19.** [*23*] (Дэвид У. Матула (David W. Matula).) Пусть D — множество целых чисел в системе с основанием b, для которых уравнение $x \equiv j$ (по модулю b) имеет точно одно решение при $0 \leq j < b$. Докажите, что все целые числа m (положительные, отрицательные и нуль) могут быть представлены в виде $m = (a_n \ldots a_0)_b$, где все a_j принадлежат множеству D, тогда и только тогда, когда все целые числа в интервале $l \leq m \leq u$ также представимы в этом виде, где $l = -\max\{a \mid a \in D\}/(b-1)$ и $u = -\min\{a \mid a \in D\}/(b-1)$. Например, $D = \{-1, 0, \ldots, b-2\}$ удовлетворяет данным условиям для всех $b \geq 3$. [*Указание.* Постройте алгоритм, формирующий подходящее представление.]

20. [*HM28*] (Дэвид У. Матула.) Рассмотрим десятичную систему счисления, в которой вместо $\{0, 1, \ldots, 9\}$ используются числа $D = \{-1, 0, 8, 17, 26, 35, 44, 53, 62, 71\}$. Из результата

упр. 19 следует (как и из упр. 18), что все действительные числа могут быть представлены бесконечными десятичными дробями с помощью цифр из множества D.

В упр. 13 отмечалось, что некоторые числа в обычной десятичной системе имеют два представления. (a) Найдите действительное число, которое имеет *более* двух D— десятичных представлений. (b) Покажите, что ни одно действительное число не может иметь бесконечного множества D— десятичных представлений. (c) Покажите, что два или более D (десятичных представлений) имеют бесконечно много цифр.

▶ **21.** [*M22*] (К. Э. Шеннон (C. E. Shannon).) Может ли произвольное вещественное число (положительное, отрицательное или нуль) быть представлено в "уравновешенной десятичной" системе, т. е. в виде $\sum_{k\le n} a_k 10^k$ для некоторого целого числа n и некоторой последовательности $a_n, a_{n-1}, a_{n-2}, \ldots$, где любое a_k — это одно из десяти чисел $\{-4\frac{1}{2}, -3\frac{1}{2}, -2\frac{1}{2}, -1\frac{1}{2}, -\frac{1}{2}, \frac{1}{2}, 1\frac{1}{2}, 2\frac{1}{2}, 3\frac{1}{2}, 4\frac{1}{2}\}$? (Хотя нуль и не является одной из допустимых цифр, мы неявно полагаем, что $a_{n+1}, a_{n+2}, \ldots$ — нули.) Найдите в этой системе счисления все представления нуля и все представления единицы.

22. [*HM25*] Пусть $\alpha = -\sum_{m\ge 1} 10^{-m^2}$. Докажите, что для заданного $\epsilon > 0$ и любого вещественного числа x существует такое "десятичное" представление этого числа, что $0 < |x - \sum_{k=0}^{n} a_k 10^k| < \epsilon$, где каждое из чисел a_k может принимать только одно из трех значений: 0, 1 или α. (В этом представлении отрицательные степени 10 не используются.)

23. [*HM30*] Пусть множество D есть множество b вещественных чисел, таких, что любое положительное число имеет представление $\sum_{k\le n} a_k b^k$, где все $a_k \in D$. В упр. 20 показано, что существует много чисел, не имеющих *единственного* представления. Тем не менее докажите, что множество T всех таких чисел имеет меру нуль, если $0 \in D$. Покажите, что этот вывод не нуждается в доказательстве, если $0 \notin D$.

24. [*M35*] Найдите бесконечно много различных множеств D, которые состоят из десяти неотрицательных целых чисел, удовлетворяющих следующим трем условиям: (i) $\gcd(D) = 1$; (ii) $0 \in D$; (iii) любое положительное вещественное число может быть представлено в виде $\sum_{k\le n} a_k 10^k$ для всех $a_k \in D$*.

25. [*M25*] (С. А. Кук (S. A. Cook).) Пусть b, u и v — целые положительные числа, причем $b \ge 2$ и $0 < v < b^m$. Покажите, что представление числа u/v в системе счисления по основанию b не содержит последовательности из m цифр, равных $b-1$, нигде справа от разделяющей точки. (Согласно общепринятому соглашению в стандартном представлении по основанию b не допускаются бесконечные последовательности цифр $(b-1)$.)

▶ **26.** [*HM30*] (Н. С. Мендельсон (N. S. Mendelsohn).) Пусть $\langle \beta_n \rangle$ — последовательность вещественных чисел, определенная для всех целых n, $-\infty < n < \infty$, таких, что

$$\beta_n < \beta_{n+1}; \qquad \lim_{n\to\infty} \beta_n = \infty; \qquad \lim_{n\to-\infty} \beta_n = 0.$$

Пусть $\langle c_n \rangle$ — произвольная последовательность положительных целых чисел, также определенная для всех целых n, $-\infty < n < \infty$. Скажем, что число x допускает "обобщенное представление", если существует такое целое n и такая последовательность целых чисел $a_n, a_{n-1}, a_{n-2}, \ldots$, что $x = \sum_{k\le n} a_k \beta_k$, где $a_n \ne 0$, $0 \le a_k \le c_k$ и $a_k < c_k$ для бесконечного множества k.

Покажите, что каждое положительное вещественное число x допускает ровно одно обобщенное представление тогда и только тогда, когда

$$\beta_{n+1} = \sum_{k\le n} c_k \beta k \qquad \text{для всех } n.$$

---

* Здесь и далее "gcd" обозначает "наибольший общий делитель" (greatest common divisor). — *Прим. перев.*

(Следовательно, системы со смешанным целочисленным основанием обладают этим свойством. Наиболее общими системами такого типа являются системы со смешанным основанием, у которых $\beta_1 = (c_0 + 1)\beta_0$, $\beta_2 = (c_1 + 1)(c_0 + 1)\beta_0$, $\ldots$, $\beta_{-1} = \beta_0/(c_{-1} + 1)$, $\ldots$.)

27. [*M21*] Покажите, что любое ненулевое число имеет единственное "знакопеременное двоичное представление"

$$2^{e_0} - 2^{e_1} + \cdots + (-1)^t 2^{e_t},$$

где $e_0 < e_1 < \cdots < e_t$.

▶ **28.** [*M24*] Покажите, что любое неотрицательное комплексное число вида $a + bi$, где a и b — целые числа, обладает единственным "периодическим двоичным представлением"

$$(1+i)^{e_0} + i(1+i)^{e_1} - (1+i)^{e_2} - i(1+i)^{e_3} + \cdots + i^t(1+i)^{e_t},$$

где $e_0 < e_1 < \cdots < e_t$ (ср. с упр. 27).

29. [*M35*] (Н. Г. де Брейн (N. G. de Bruijn).) Пусть S_0, S_1, S_2, $\ldots$ — множества неотрицательных целых чисел; говорят, что совокупность $\{S_0, S_1, S_2, \ldots\}$ обладает свойством B, если любое неотрицательное целое число n может быть единственным способом записано в виде

$$n = s_0 + s_1 + s_2 + \cdots, \qquad s_j \in S_j.$$

(Свойство B означает, что $0 \in S_j$ для всех j, поскольку $n = 0$ может быть представлено только как $0+0+0+\cdots$.) Любая система счисления со смешанным основанием $b_0, b_1, b_2, \ldots$ дает пример совокупности множеств, удовлетворяющих свойству B, если положить $S_j = \{0, B_j, \ldots, (b_j - 1)B_j\}$, где $B_j = b_0 b_1 \ldots b_{j-1}$. В таком случае представление $n = s_0 + s_1 + s_2 + \cdots$ очевидным образом соответствует представлению (9) этого числа по смешанному основанию. Далее, если совокупность $\{S_0, S_1, S_2, \ldots\}$ A_0, A_1, A_2, $\ldots$ обладает свойством B, то, каково бы ни было разбиение A_0, A_1, A_2, $\ldots$ неотрицательных целых чисел (т. е. $A_0 \cup A_1 \cup A_2 \cup \cdots = \{0, 1, 2, \ldots\}$ и $A_i \cap A_j = \emptyset$ при $i \neq j$, некоторые из множеств A_j могут быть пустыми), этим свойством обладает и полученная из нее путем "стягивания" совокупность $\{T_0, T_1, T_2, \ldots\}$, где множество T_j состоит из всех сумм вида $\sum_{i \in A_j} s_i$, взятых по всевозможным выборкам элементов $s_i \in S_i$.

Докажите, что *любая* последовательность $\{T_0, T_1, T_2, \ldots\}$, удовлетворяющая свойству B, может быть получена посредством "стягивания" некоторой совокупности $\{S_0, S_1, S_2, \ldots\}$, соответствующей системе счисления по смешанному основанию.

30. [*M39*] (Н. Г. де Брейн.) Пример системы счисления по основанию -2 показывает, что любое целое число (положительное, отрицательное или нуль) имеет единственное представление в виде

$$(-2)^{e_1} + (-2)^{e_2} + \cdots + (-2)^{e_t}, \qquad e_1 > e_2 > \cdots > e_t \geq 0, \qquad t \geq 0.$$

Назначение данного упражнения — несколько обобщить это свойство.

a) Пусть последовательность целых чисел b_0, b_1, b_2, $\ldots$ такова, что любое целое число n допускает единственное представление в виде

$$n = b_{e_1} + b_{e_2} + \cdots + b_{e_t}, \qquad e_1 > e_2 > \cdots > e_t \geq 0, \qquad t \geq 0.$$

(Данная последовательность $\langle b_n \rangle$ называется бинарным базисом.) Покажите, что найдется такое значение индекса j, что b_j нечетно, а для всех $k \neq j$ числа b_j четны.

b) Докажите, что бинарный базис $\langle b_n \rangle$ всегда может быть преобразован в последовательность вида d_0, $2d_1$, $4d_2$, $\ldots = \langle 2^n d_n \rangle$, где каждое из чисел d_k нечетно.

c) Докажите, что если каждое из чисел d_0, d_1, d_2, $\ldots$ из п. (b) равно ± 1, то последовательность $\langle b_n \rangle$ образует бинарный базис тогда и только тогда, когда существует бесконечно много d_j, равных $+1$, и бесконечно много d_j, равных -1.

d) Докажите, что последовательность $7, -13\cdot 2, 7\cdot 2^2, -13\cdot 2^3, \ldots, 7\cdot 2^{2k}, -13\cdot 2^{2k+1}, \ldots$ является бинарным базисом, и найдите представление числа $n = 1$.

▸ **31.** [*M35*] Одно обобщение представления чисел в обратном двоичном коде, известное как "2-адические числа", было предложено в работе K. Hensel, *Crelle* **127** (1904), 51–84. (В действительности К. Гензель предложил *p-адические числа* для любого простого числа p.) 2-адическое число можно рассматривать как двоичное число

$$u = (\ldots u_3 u_2 u_1 u_0 . u_{-1} \ldots u_{-n})_2,$$

представление которого бесконечно продолжается влево и лишь на конечное количество знаков вправо от разделяющей точки. Сложение, вычитание и умножение 2-адических чисел выполняются в соответствии с алгоритмом обычных арифметических операций, которые, в принципе, допускают возможность неограниченного продолжения влево. Например,

$$7 = (\ldots 000000000000111)_2 \qquad \tfrac{1}{7} = (\ldots 110110110110111)_2$$
$$-7 = (\ldots 111111111111001)_2 \qquad -\tfrac{1}{7} = (\ldots 001001001001001)_2$$
$$\tfrac{7}{4} = (\ldots 000000000000001.11)_2 \qquad \tfrac{1}{10} = (\ldots 110011001100110.1)_2$$
$$\sqrt{-7} = (\ldots 100000010110101)_2 \quad \text{или} \quad (\ldots 011111101001011)_2 .$$

Здесь число 7 — обычное число "семь" в двоичном представлении, а -7 — обратный код, неограниченно продолженный влево. Легко проверить, что обычная процедура сложения двоичных чисел дает $-7 + 7 = (\ldots 00000)_2 = 0$, если ее выполнение продолжать неограниченно долго. Значения $\frac{1}{7}$ и $-\frac{1}{7}$ представляют собой единственные 2-адические числа, которые после формального умножения на 7 дают соответственно 1 и -1. Значения $\frac{7}{4}$ и $\frac{1}{10}$ есть примеры 2-адических чисел, не являющихся 2-адическими "целыми", так как они имеют ненулевые биты справа от разделяющей точки. Приведенные два значения $\sqrt{-7}$, получающиеся одно из другого в результате перемены знака, являются единственными 2-адическими числами, которые после формального возведения в квадрат дают $(\ldots 111111111111001)_2$.

a) Докажите, что любое 2-адическое число u можно разделить на произвольное ненулевое 2-адическое число v, чтобы вычислить 2-адическое число w, удовлетворяющее равенству $u = vw$. (Следовательно, множество 2-адических чисел образует поле; см. раздел 4.6.1.)

b) Докажите, что 2-адическое представление рационального числа $-1/(2n+1)$, где n — положительное целое число, можно получить следующим образом. Сначала находим обычное двоичное разложение числа $+1/(2n+1)$, которое имеет вид периодической дроби $(0.\alpha\alpha\alpha\ldots)_2$ (α — некоторая строка из нулей и единиц). Тогда $(\ldots\alpha\alpha\alpha)_2$ будет 2-адическим представлением числа $-1/(2n+1)$.

c) Докажите, что 2-адическое представление числа u периодично (т. е. $u_{N+\lambda} = u_N$ для всех больших N при некотором $\lambda \geq 1$) тогда и только тогда, когда u рационально (т. е. $u = m/n$ для некоторых целых чисел m и n).

d) Докажите, что если n — целое число, то $\sqrt{n}$ является 2-адическим числом только в том случае, если для некоторого неотрицательного целого числа n оно удовлетворяет условию $n \bmod 2^{2k+3} = 2^{2k}$. (Таким образом, либо $n \bmod 8 = 1$, либо $n \bmod 32 = 4$, и т. д.)

32. [*M40*] (И. З. Руца (I. Z. Ruzsa).) Сформируйте бесконечно много целых чисел, в троичных представлениях которых используются только нули и единицы, а в четверичном представлении — только нули, единицы и двойки.

33. [*M40*] (Д. А. Кларнер (D. A. Klarner).) Пусть множество D — произвольное множество целых чисел, b — любое положительное целое число, а k_n — количество различных целых чисел, которые могут быть записаны как n-разрядные числа $(a_{n-1}\ldots a_1a_0)_b$ по основанию b с цифрами a_i в D. Докажите, что последовательность $\langle k_n\rangle$ удовлетворяет линейному рекуррентному соотношению, и поясните, как вычислить производящую функцию $\sum_n k_n z^n$. Проиллюстрируйте разработанный алгоритм, показав, что в случае, когда $b = 3$ и $D = \{-1, 0, 3\}$, число k_n есть число Фибоначчи.

▸ **34.** [*22*] (Г. В. Райтвайзнер (G. W. Reitwiesner), 1960.) Поясните, как представить заданное целое число n в виде $(\ldots a_2a_1a_0)_2$, где каждое из a_j есть -1, 0 либо 1, используя наименьшую ненулевую цифру.

4.2. АРИФМЕТИКА ЧИСЕЛ С ПЛАВАЮЩЕЙ ТОЧКОЙ

В ЭТОМ РАЗДЕЛЕ рассмотрены основные принципы выполнения арифметических операций над числами с "плавающей точкой" и проанализирован внутренний механизм таких вычислений. Вероятно, у многих читателей данная тема не вызовет слишком большого интереса либо потому, что в вычислительных машинах, на которых они работают, имеются встроенные команды операций над числами с плавающей точкой, либо потому, что нужные подпрограммы содержатся в операционной системе. Но не следует считать, что материал этого раздела относится исключительно к компетенции инженеров — конструкторов ЭВМ или узкого круга лиц, которые пишут системные подпрограммы для новых машин. *Каждый* грамотный программист должен иметь представление о том, что происходит при выполнении элементарных шагов арифметических операций над числами с плавающей точкой. Предмет этот совсем не так тривиален, как принято считать; в нем удивительно много интересного.

4.2.1. Вычисления с однократной точностью

А. Обозначение чисел с плавающей точкой. В разделе 4.1 были рассмотрены различные способы обозначения чисел с фиксированной точкой. При таком способе обозначения программист знает, где положено находиться разделяющей точке в числах, с которыми выполняются те или иные операции. В некоторых ситуациях при выполнении программы значительно удобнее сделать положение разделяющей точки динамически изменяющимся, иными словами, сделать точку "плавающей" и связать с каждым числом информацию о ее положении. Эта идея уже давно использовалась в научных расчетах, в особенности для представления очень больших чисел наподобие числа Авогадро $N = 6.02214 \times 10^{23}$ или таких очень малых чисел, как постоянная Планка $h = 6.6261 \times 10^{-27}$ эрг·с.

В этом разделе речь пойдет о *p-разрядных числах с плавающей точкой по основанию b с избытком q.* Такое число представляется парой величин (e, f), которой отвечает значение

$$(e, f) = f \times b^{e-q}. \tag{1}$$

Здесь e — целое число, изменяющееся в соответствующем интервале значений, а f — дробное число со знаком. Условимся, что

$$|f| < 1,$$

иными словами, разделяющая точка в позиционном представлении f находится в крайней слева позиции. Точнее говоря, соглашение о том, что мы имеем дело с p-разрядными числами, означает, что $b^p f$ — целое число и

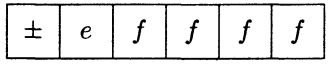
$$-b^p < b^p f < b^p. \tag{2}$$

Термин "двоичное число с плавающей точкой", как всегда, будет означать, что $b = 2$, термин "десятичное число с плавающей точкой" — что $b = 10$ и т. д. Используя 8-разрядные десятичные числа с плавающей точкой с избытком 50, можно, например, написать

$$\begin{aligned} &\text{число Авогадро} && N = (74, +.60221400); \\ &\text{постоянная Планка} && h = (24, +.66261000). \end{aligned} \tag{3}$$

Две компоненты, e и f, числа с плавающей точкой называются его *порядком* и *дробной частью* соответственно. (Иногда используются и другие названия, особенно "характеристика" и "мантисса"; однако слово "мантисса" для обозначения дробной части приводит к путанице в терминологии, так как этот термин употребляется совсем в другом смысле в теории логарифмов и, кроме того, английское слово "mantissa" означает "мало дающее добавление".)

В компьютере MIX числа с плавающей точкой имеют вид



(4)

Это представление с плавающей точкой по основанию b с избытком q, с четырьмя значащими "цифрами", где b есть размер байта (т. е. $b = 64$ или $b = 100$) и q равняется $\lfloor \frac{1}{2}b \rfloor$. Дробная часть равна $\pm\, f\, f\, f\, f$, а порядок и e находится в интервале $0 \le e < b$. Такое внутреннее представление — типичный пример соглашений, которые приняты в большинстве существующих компьютеров, хотя основание b здесь гораздо больше, чем обычно используемое.

B. Нормализованные вычисления. Число с плавающей точкой (e, f) является нормализованным, либо если наиболее значимая цифра в представлении f отлична от нуля, так что

$$1/b \le |f| < 1, \tag{5}$$

либо если $f = 0$, а e принимает наименьшее возможное значение. Чтобы установить, какое из двух нормализованных чисел с плавающей точкой имеет бóльшую величину, достаточно сравнить их порядки; только если порядки равны, нужно анализировать и дробные части.

Большинство ныне применяемых стандартных подпрограмм работает почти исключительно с нормализованными числами: предполагается, что входные значения для подпрограмм нормализованы, а результаты всегда нормализуются. При реализации этих соглашений в системных библиотеках мы теряем возможность представлять некоторые числа очень малой величины (например, значение $(0, .00000001)$ не может быть нормализовано без формирования отрицательного порядка), но мы выигрываем в скорости, единообразии и получаем возможность сравнительно легко ограничить относительную ошибку вычислений. (Арифметика ненормализованных чисел с плавающей точкой будет рассмотрена в разделе 4.2.2.)

Рассмотрим теперь арифметические операции над нормализованными числами с плавающей точкой подробнее. Попутно затронем и структуру подпрограмм, реализующих эти операции (предполагая, что в нашем распоряжении имеется компьютер без аппаратной реализации этих арифметических операций).

В стандартных подпрограммах для выполнения арифметических действий над числами с плавающей точкой, написанных на машинном языке, в очень большой степени используются крайне специфические особенности конкретной модели компьютера. Именно поэтому так мало сходства между двумя подпрограммами, скажем, сложения чисел с плавающей точкой, написанными для разных машин. Все же тщательный анализ большого числа подпрограмм как для двоичных, так и для десятичных компьютеров показывает, что в действительности данные программы имеют много общего, и обсуждение этой темы, вполне возможно, не зависит от конкретной машины.

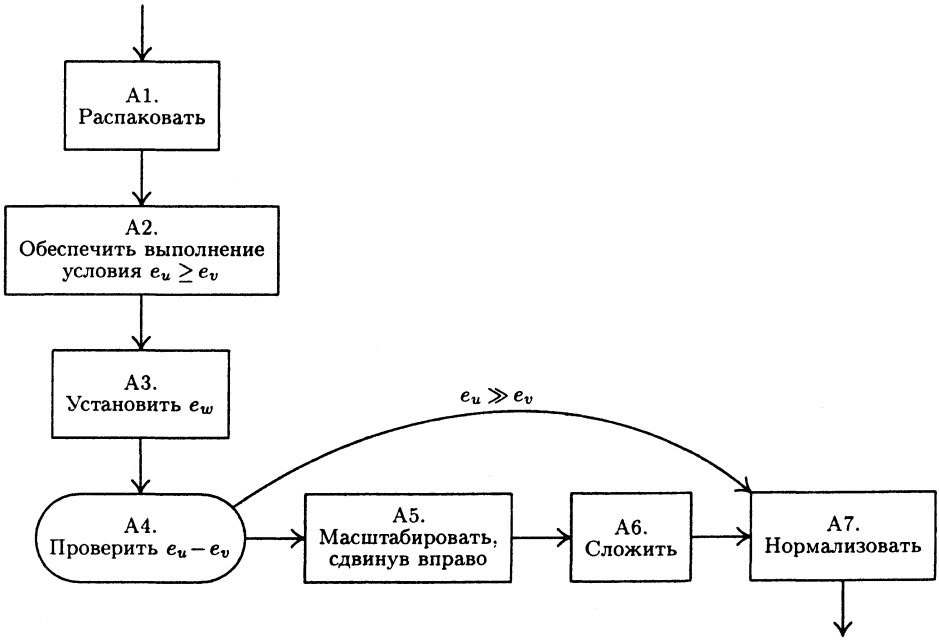


Рис. 2. Сложение чисел с плавающей точкой.

Первый (и наиболее трудный!) из алгоритмов, обсуждаемых в этом разделе, — это процедура сложения чисел с плавающей точкой:

$$(e_u, f_u) \oplus (e_v, f_v) = (e_w, f_w). \tag{6}$$

Ввиду того что арифметические действия над числами с плавающей точкой являются по самой своей сути приближенными, а не точными, для обозначения операций сложения, вычитания, умножения и деления с плавающей точкой здесь будут использоваться "округленные" символы

$$\oplus, \quad \ominus, \quad \otimes, \quad \oslash,$$

чтобы отличать приближенные операции от точных.

Идея, лежащая в основе сложения с плавающей точкой, довольно проста. Полагая, что $e_u \geq e_v$, формируем результат по принципу $e_w = e_u$, $f_w = f_u + f_v/b^{e_u - e_v}$ (таким образом, выравнивается положение разделяющих точек и соответственно положение разрядов слагаемых), а затем нормализуем результат. Может возникнуть несколько ситуаций, которые делают выполнение этого процесса нетривиальным; более точное описание метода дается в следующем алгоритме.

Алгоритм А (*Сложение чисел с плавающей точкой*). Для заданных p-разрядных нормализованных чисел с плавающей точкой $u = (e_u, f_u)$ и $v = (e_v, f_v)$ по основанию b с избытком q строится сумма $w = u \oplus v$. Данный алгоритм (рис. 2) можно использовать и для вычитания чисел с плавающей точкой, если v заменить на $-v$.

А1. [Распаковать.] Выделить порядок и дробную часть в представлениях для u и v.

A2. [Обеспечить выполнение условия $e_u \geq e_v$.] Если $e_u < e_v$, поменять местами u и v. (Во многих случаях удобнее совместить шаг A2 с шагом A1 или с каким-нибудь из последующих шагов.)

A3. [Установить e_w.] Установить $e_w \leftarrow e_u$.

A4. [Проверить $e_u - e_v$.] Если $e_u - e_v \geq p + 2$ (большая разница в порядках), установить $f_w \leftarrow f_u$ и перейти к шагу A7. (Так как предполагается, что u нормализовано, на этом выполнение алгоритма можно было бы и закончить, но часто полезно использовать операцию сложения с нулем для гарантированной нормализации любого, возможно, и ненормализованного, числа.)

A5. [Масштабировать, сдвинув вправо.] Сдвинуть f_v вправо на $e_u - e_v$ позиций, т. е. разделить f_v на $b^{e_u - e_v}$. [*Замечание.* Величина сдвига может достигать $p + 1$ разрядов, вследствие чего для выполнения следующего шага (сложения дробной части f_u с f_v) потребуется аккумулятор, способный хранить $2p+1$ цифр по основанию b справа от позиционной точки. Если такого вместительного аккумулятора нет, можно сократить сдвиг до $p + 2$ или $p + 3$ разрядов, но с соответствующими предосторожностями; подробности обсуждаются в упр. 5.]

A6. [Сложить.] Установить $f_w \leftarrow f_u + f_v$.

A7. [Нормализовать.] (В этот момент (e_w, f_w) представляет сумму u и v, но $|f_w|$ может содержать более p цифр и может быть больше единицы или меньше $1/b$.) Выполнить описываемый ниже алгоритм N, который нормализует и округлит (e_w, f_w), а также сформирует окончательный результат. ▮

Алгоритм N (*Нормализация*). "Грубый порядок" e и "сырая дробная часть" f приводятся к нормализованному виду с округлением при необходимости до p разрядов. В этом алгоритме (рис. 3) предполагается, что $|f| < b$.

N1. [Проверить f.] Если $|f| \geq 1$ (переполнение дробной части), перейти к шагу N4. Если $f = 0$, установить e равным его наименьшему значению и перейти к шагу N7.

N2. [f нормализовано?] Если $|f| \geq 1/b$, перейти к шагу N5.

N3. [Масштабировать, сдвинув влево.] Сдвинуть f на один разряд влево (т. е. умножить на b) и уменьшить e на 1. Возвратиться к шагу N2.

N4. [Масштабировать, сдвинув вправо.] Сдвинуть f вправо на один разряд (т. е. разделить на b) и увеличить e на 1.

N5. [Округлить.] Округлить f до p разрядов. (Это означает, что f изменяется до ближайшего кратного b^{-p}. Возможно, что $(b^p f) \bmod 1 = \frac{1}{2}$, т. е. имеется *два* ближайших кратных. Если b четно, то заменяем f ближайшим кратным b^{-p}, таким, что $b^p f' + \frac{1}{2}b$ нечетно (обозначим результат округления в таком случае через f'). Более подробное обсуждение аспектов округления приводится в разделе 4.2.2.) Важно отметить, что операция округления может привести к равенству $|f| = 1$ (переполнение при округлении); в такой ситуации следует вернуться к шагу N4.

N6. [Проверить e.] Если порядок e слишком велик, т. е. больше допустимой границы, это воспринимается, как выполнение условия *переполнения порядка*. Если e слишком мал, это воспринимается, как выполнение условия *исчезновения*

Рис. 3. Нормализация (e, f).

порядка. (Дополнительная информация по этому вопросу приводится ниже; эти ситуации интерпретируются обычно, как сигнал об ошибке, в том смысле, что результат не может быть представлен в виде нормализованного числа с плавающей точкой из требуемого интервала значений.)

N7. [Упаковать.] Объединить порядок e и дробную часть f для выдачи искомого результата. ▮

Несколько простых примеров сложения чисел с плавающей точкой рассматривается в упр. 4.

Приведенные ниже подпрограммы для сложения и вычитания на компьютере MIX чисел, имеющих форму (4), служат примером программной реализации алгоритмов A и N. Эти подпрограммы извлекают одно входное значение u по символическому адресу ACC, другое входное значение v извлекается из регистра A при входе в подпрограмму. Результат w одновременно появляется в регистре A и в поле ACC. Таким образом, последовательность команд

```
LDA A;  ADD B;  SUB C;  STA D,                    (7)
```

работающих с числами с фиксированной точкой, соответствовала бы такой последовательности команд, работающих с числами с плавающей точкой:

```
LDA A, STA ACC;  LDA B, JMP FADD;  LDA C, JMP FSUB;  STA D.    (8)
```

Программа A (*Сложение, вычитание и нормализация*). Следующая программа представляет собой подпрограмму, реализующую алгоритм A, причем она построена таким образом, что нормализующий фрагмент может использоваться другими

подпрограммами, которые будут рассмотрены ниже. Как в этой программе, так и во многих других программах данной главы, идентификатор OFLO именует подпрограмму, которая печатает сообщение о том, что индикатор переполнения машины MIX внезапно перешел в состояние "включено". Предполагается, что размер байта b кратен 4. В подпрограмме нормализации NORM предполагается, что rI2 $= e$ и rAX $= f$, где rA $= 0$ влечет за собой rX $= 0$ и rI2 $< b$.

```
00  BYTE  EQU  1(4:4)        Размер байта b.
01  EXP   EQU  1:1           Определение поля порядка.
02  FSUB  STA  TEMP          Подпрограмма вычитания чисел с плавающей точкой.
03        LDAN TEMP          Изменить знак операнда.
04  FADD  STJ  EXITF         Подпрограмма сложения чисел с плавающей точкой.
05        JOV  OFLO          Снятие блокировки переполнения.
06        STA  TEMP          TEMP ← v.
07        LDX  ACC           rX ← u.
08        CMPA ACC(EXP)      Шаги A1–A3 скомбинированы ниже.
09        JGE  1F            Переход, если e_v ≥ e_u.
10        STX  FU(0:4)       FU ← ± f f f f 0.
11        LD2  ACC(EXP)      rI2 ← e_w.
12        STA  FV(0:4)
13        LD1N TEMP(EXP)     rI1 ← −e_v.
14        JMP  4F
15  1H    STA  FU(0:4)       FU ← ± f f f f 0 (u, v меняются ролями).
16        LD2  TEMP(EXP)     rI2 ← e_w.
17        STX  FV(0:4)
18        LD1N ACC(EXP)      rI1 ← −e_v.
19  4H    INC1 0,2           rI1 ← e_u − e_v. (Шаг A4 не является необходимым.)
20  5H    LDA  FV            A5. Масштабировать, сдвинув вправо.
21        ENTX 0             Очистить rX.
22        SRAX 0,1           Сдвиг вправо на e_u − e_v позиций.
23  6H    ADD  FU            A6. Сложение.
24        JOV  N4            A7. Нормализация. Переход, если произошло
                               переполнение.
25        JXZ  NORM          Простой случай?
26        CMPA =0=(1:1)      f нормализовано?
27        JNE  N5            Если нормализовано, округлить его.
28        SRC  5             |rX| ↔ |rA|.
29        DECX 1             (rX положительно.)
30        STA  TEMP          (Операнды имеют противоположные знаки;
31        STA  HALF(0:0)       нужно уточнить состояние регистров
32        LDAN TEMP            перед округлением и нормализацией.)
33        ADD  HALF
34        ADD  HALF          Дополнить наименьшей значащей величиной.
35        SRC  4             Переход к подпрограмме нормализации.
36        JMP  N3A
37  HALF  CON  1//2          Половина размера слова (знак может изменяться).
38  FU    CON  0             Дробная часть f_u.
39  FV    CON  0             Дробная часть f_v.
```

```
40  NORM  JAZ  ZRO          N1. Проверить f.
41  N2    CMPA =0=(1:1)     N2. Нормализовано ли f?
42        JNE  N5           Перейти к шагу N5, если ведущий байт
                              отличен от нуля.
43  N3    SLAX 1            N3. Масштабировать, сдвинув влево.
44  N3A   DEC2 1            Уменьшить e на 1.
45        JMP  N2           Вернуться к шагу N2.
46  N4    ENTX 1            N4. Масштабировать, сдвинув вправо.
47        SRC  1            Выполнить сдвиг вправо и вставить "1"
                              с соответствующим знаком.
48        INC2 1            Увеличить e на 1.
49  N5    CMPA =BYTE/2=(5:5) N5. Округление.
50        JL   N6           |остаток| < (1/2)b?
51        JG   5F
52        JXNZ 5F           |остаток| > (1/2)b?
53        STA  TEMP         |остаток| = (1/2)b; округлить до нечетного.
54        LDX  TEMP(4:4)
55        JXO  N6           Перейти к шагу N6, если rX нечетно.
56  5H    STA  *+1(0:0)     Сохранить знак rA.
57        INCA BYTE         Добавить b^{-4} к |f| (знак может изменяться).
58        JOV  N4           Проверить переполнение из-за округления.
59  N6    J2N  EXPUN        N6. Проверка e. Исчезновение порядка, если e < 0.
60  N7    ENTX 0,2          N7. Упаковать. rX ← e.
61        SRC  1
62  ZRO   DEC2 BYTE         rI2 ← e − b.
63  8H    STA  ACC
64  EXITF J2N  *            Выход, если только не e ≥ b.
65  EXPOV HLT  2            Обнаружено переполнение порядка.
66  EXPUN HLT  1            Обнаружено исчезновение порядка.
67  ACC   CON  0            Аккумулятор для операций с плавающей точкой.
```

Довольно большой фрагмент кода программы (строки 25–37) включен в программу по той причине, что в MIX имеется 5-байтовый аккумулятор для сложения чисел со знаком, в то время как алгоритм A в общем случае требует для него $2p + 1 = 9$ разрядов. В действительности всю программу можно сократить почти вдвое, если пожертвовать хотя бы небольшой долей точности. Однако, как будет показано в следующем разделе, всегда лучше получать наибольшую возможную точность. В строке 55 используется нестандартная команда MIX, описанная в разделе 4.5.2. Время выполнения сложения и вычитания чисел с плавающей точкой зависит от нескольких факторов, анализируемых ниже, в разделе 4.2.4.

Рассмотрим теперь операции умножения и деления, которые выполняются проще, чем сложение, и довольно похожи одна на другую.

Алгоритм M (*Умножение или деление чисел с плавающей точкой*). По данным p-разрядным нормализованным числам с плавающей точкой $u = (e_u, f_u)$ и $v = (e_v, f_v)$ по основанию b с избытком q строится произведение $w = u \otimes v$ или частное $w = u \oslash v$.

M1. [Распаковать.] Выделить порядки и дробные части в представлениях u и v.

(Иногда удобно, хотя и необязательно, проверить в ходе выполнения этого шага, не равны ли операнды нулю.)

M2. [Выполнить операцию.] Установить

$$\begin{aligned} e_w &\leftarrow e_u + e_v - q, \qquad & f_w &\leftarrow f_u f_v && \text{для умножения;} \\ e_w &\leftarrow e_u - e_v + q + 1, & f_w &\leftarrow (b^{-1} f_u)/f_v && \text{для деления.} \end{aligned} \tag{9}$$

(Поскольку предполагается, что вводимые числа нормализованы, в результате получим, что либо $f_w = 0$, либо $1/b^2 \le |f_w| < 1$, либо возникнет ошибка "деление на нуль".) Здесь, если в этом есть необходимость, можно урезать f_w до $p+2$ или $p+3$ разрядов, как в упр. 5.

M3. [Нормализовать.] Применить к (e_w, f_w) алгоритм N с тем, чтобы нормализовать, округлить и упаковать результат. (*Замечание.* В этом случае нормализация выполняется проще ввиду того, что масштабирование посредством сдвига влево происходит не более одного раза и после деления переполнение не может возникнуть вследствие округления.) ▮

В подпрограммах для MIX, тексты которых приведены ниже, используются те же соглашения, что и в программе A. Эти подпрограммы служат примером машинной реализации алгоритма M.

Программа M (*Умножение и деление чисел с плавающей точкой*).

```
01  Q      EQU  BYTE/2       q есть половина размера байта.
02  FMUL   STJ  EXITF        Подпрограмма умножения чисел с плавающей точкой.
03         JOV  OFLO         Снятие блокировки переполнения.
04         STA  TEMP         TEMP ← v.
05         LDX  ACC          rX ← u.
06         STX  FU(0:4)      FU ← ± f f f f 0.
07         LD1  TEMP(EXP)
08         LD2  ACC(EXP)
09         INC2 -Q,1         rI2 ← e_u + e_v − q.
10         SLA  1
11         MUL  FU           Умножить f_u на f_v.
12         JMP  NORM         Нормализовать, округлить и выйти из подпрограммы.

13  FDIV   STJ  EXITF        Подпрограмма деления чисел с плавающей точкой.
14         JOV  OFLO         Снятие блокировки переполнения.
15         STA  TEMP         TEMP ← v.
16         STA  FV(0:4)      FV ← ± f f f f 0.
17         LD1  TEMP(EXP)
18         LD2  ACC(EXP)
19         DEC2 -Q,1         rI2 ← e_u − e_v + q.
20         ENTX 0
21         LDA  ACC
22         SLA  1            rA ← f_u.
23         CMPA FV(1:5)
24         JL   *+3          Переход, если |f_u| < |f_v|.
25         SRA  1            Иначе — масштабировать f_u вправо
26         INC2 1                  и увеличить rI2 на 1.
```

```
27          DIV  FV          Разделить.
28          JNOV NORM        Нормализовать, округлить и выйти из подпрограммы.
29  DVZRO HLT  3             Ненормализовано или делитель равен нулю.  ▮
```

Наиболее интересная особенность этой программы — подготовка к выполнению деления, осуществляемая командами в строках 23–26. Эти операции проводятся для того, чтобы обеспечить достаточную точность при округлении ответа. При $|f_u| < |f_v|$, непосредственно применив алгоритм M, можно сохранить результат в форме "$\pm 0 f f f f$" в регистре A, что сделает невозможным чистое округление без специального анализа остатка (он хранится в регистре X). Поэтому в такой ситуации программа вычисляет $f_w \leftarrow f_u/f_v$, гарантируя, что f_w либо равно нулю, либо во всех случаях нормализовано. Процедура округления может оперировать пятью значащими байтами, возможно, проверяя, не равен ли остаток нулю.

Иногда может потребоваться перевод из представления с фиксированной точкой в представление с плавающей точкой и обратно. При помощи описанного выше алгоритма нормализации легко получается программа перевода "из фиксированной в плавающую". Например, целое число переводится в форму с плавающей точкой с помощью следующей подпрограммы на языке MIX.

```
01  FLOT STJ  EXITF  Предполагаем, что rA = u есть целое число.
02        JOV  OFLO  Снятие блокировки переполнения.
03        ENT2 Q+5   Установить "грубый" порядок.                    (10)
04        ENTX 0
05        JMP  NORM  Нормализовать, округлить и выйти из подпрограммы.  ▮
```

Подпрограмма перевода "из плавающей в фиксированную" служит предметом упр. 14.

Отладка подпрограмм выполнения арифметических операций над числами с плавающей точкой — обычно довольно сложная задача из-за обилия различных случаев, которые нужно предусмотреть. Ниже перечислены распространенные ловушки, подстерегающие программиста, который занимается программами "плавающей арифметики".

1) *Потеря знака.* Во многих машинах (к MIX это не относится) команды сдвига регистров воздействуют на знаковый разряд, поэтому следует подробно анализировать операции сдвига, используемые при нормализации и изменении масштаба дробной части числа. Часто знак теряется и при появлении "минус нуля". (Например, операторы 30–34 программы A ответственны за установку знакового разряда регистра A. См. также упр. 6.)

2) *Невозможность правильного определения, что произошло — исчезновение или переполнение порядка.* Не следует проверять величину e_w *до окончания* операций округления и нормализации, так как предварительные проверки могут привести к ошибочным результатам. Исчезновение и переполнение порядка могут происходить и при выполнении сложения и вычитания, а не только при умножении и делении, и, несмотря на то что это событие довольно редкое, проверку необходимо проводить для каждого конкретного случая. Для того чтобы выполнить необходимые корректирующие операции после обнаружения исчезновения или переполнения, нужно заранее позаботиться о сохранении достаточной для этого информации.

К сожалению, некоторые программисты во многих случаях пренебрегают исчезновением порядка. Они просто полагают, что исчезающе малые результаты равны нулю без индикации ошибки. Часто это приводит к серьезной потере точности (а на самом деле к потере *всех* значащих цифр), что нарушает соглашения, принятые в операциях над числами с плавающей точкой. Таким образом, системная подпрограмма действительно должна информировать прикладного программиста об исчезновении порядка. Приравнивание результата к нулю допустимо только в отдельных случаях, когда результат должен складываться со значительно большей величиной. Если исчезновение порядка не фиксируется, то возникают таинственные ситуации, когда $(u \otimes v) \otimes w$ равно нулю, а $u \otimes (v \otimes w)$ нет, поскольку, скажем, умножение $u \otimes v$ приводит к исчезновению порядка, а $u \otimes (v \otimes w)$ вычисляется и без выхода порядков за пределы допустимого интервала. Подобным же образом можно найти пять таких положительных чисел a, b, c, d и y, что

$$\begin{aligned} (a \otimes y \oplus b) \oslash (c \otimes y \oplus d) &\approx \tfrac{2}{3}, \\ (a \oplus b \oslash y) \oslash (c \oplus d \oslash y) &= 1, \end{aligned} \tag{11}$$

если исчезновение порядка не фиксируется (см. упр. 9). Даже с учетом того, что подпрограммы арифметики с плавающей точкой не идеально точны, такие несуразные результаты, как (11), совсем уж неожиданны для случая, когда все числа a, b, c, d и y *положительны*! Исчезновение порядка обычно не предугадывается программистом, так что ему следует об этом сообщать*.

3) *Попадание "мусора".* При выполнении сдвига влево необходимо проследить, чтобы в освобождающиеся разряды справа не было введено чего-либо, отличного от нулей. Например, обратите внимание на команду `ENTX 0` в строке 21 программы A и "слишком легко забываемую" команду `ENTX 0` в строке 04 подпрограммы `FLOT` в (10). (Но было бы ошибкой очищать регистр X после строки 27 в подпрограмме деления.)

4) *Непредусмотренное переполнение при округлении.* Когда число наподобие .999999997 округляется до 8 цифр, происходит перенос влево от десятичной точки и результат сдвигается вправо. Многие ошибочно считали, что в ходе выполнения

* С другой стороны, нужно отметить, что современные языки программирования высокого уровня предоставляют программисту (или вовсе не предоставляют) весьма ограниченные возможности использования информации, содержащейся в стандартных программах арифметики с плавающей точкой. Подпрограммы для `MIX`, которые представлены в этом разделе, просто останавливают работу, обнаружив такую ситуацию, а это отнюдь не выход. Существует множество приложений, для которых исчезновение порядка относительно безвредно, и потому желательно найти способ, пользуясь которым программист смог бы просто и безопасно для приложения справиться с возникшей проблемой. Практика подстановки "втихомолку" нуля вместо результата с исчезнувшим порядком себя полностью дискредитировала, но существует альтернатива, которая в последнее время завоевывает все большую популярность. Ее суть в том, чтобы модифицировать самое определение формата представления чисел с плавающей точкой, допуская существование ненормализованной дробной части в случае, если порядок имеет минимальное допустимое значение. Эта идея "постепенной потери значимости" впервые была реализована в компьютере Electrologica X8; она лишь незначительно усложнила алгоритмы выполнения операций, но совершенно исключила возможность исчезновения порядка при сложении и вычитании. Рассмотрение этой идеи выходит за рамки данной книги, поэтому в простых формулах анализа относительной ошибки вычислений в разделе 4.2.2 появление постепенной потери значимости не учитывается. Тем не менее, используя формулы, подобные $\text{round}(x) = x(1-\delta)+\epsilon$, где $|\delta| < b^{1-p}/2$ и $|\epsilon| < b^{-p-q}/2$, можно показать, что формат с постепенной потерей значимости успешно срабатывает во многих важных случаях. (См. W. M. Kahan and J. Palmer, *ACM SIGNUM Newsletter* (October, 1979), 13–21.)

умножения переполнение при округлении невозможно, так как максимальное значение $|f_u f_v|$ равно $1-2b^{-p}+b^{-2p}$, а это число не может округлиться до 1. Ошибочность такого рассуждения продемонстрирована в упр. 11. Любопытно, что переполнение при округлении действительно *невозможно* при делении чисел с плавающей точкой (см. упр. 12).

Существует направление, представители которого утверждают, что можно безболезненно "округлять" .999999997 до .99999999, а не до 1.0000000, поскольку последний результат представляет все числа из интервала

$$[1.0000000 - 5 \times 10^{-8} \,..\, 1.0000000 + 5 \times 10^{-8}],$$

в то время как .99999999 представляет все числа из гораздо меньшего интервала

$$(.99999999 - 5 \times 10^{-9} \,..\, .99999999 + 5 \times 10^{-9}).$$

Хотя второй интервал и не содержит исходного числа .999999997, каждое число из второго интервала содержится в первом, так что последующие вычисления со вторым интервалом не менее точны, чем с первым. Но этот довод несовместим с математической идеологией арифметики с плавающей точкой, представленной ниже, в разделе 4.2.2.

5) *Округление до нормализации.* Неточность результата порождается и преждевременным округлением в неверных цифровых разрядах. Эта ошибка очевидна, когда округление производится слева от соответствующего разряда. Она также опасна в менее очевидном случае, когда округление сначала выполняется намного правее, а затем — в истинном разряде. По этой причине ошибочно осуществлять округление в ходе операции "сдвиг вправо" на шаге A5; исключением является случай, рассмотренный в упр. 5. (Однако специальный случай округления на шаге N5, а затем повторного округления уже после переполнения при округлении безобиден, потому что переполнение при округлении всегда приводит к значению ± 1.0000000, которое не меняется в результате последующей процедуры округления.)

6) *Невозможность сохранения достаточной точности в промежуточных вычислениях.* Детальный анализ точности арифметических операций с плавающей точкой, проводимый в следующем разделе, показывает, что нормализующие программы арифметики с плавающей точкой должны всегда обеспечивать максимальную точность подходящим образом округленного результата. Не должно быть никаких отступлений от этого принципа, даже в тех случаях, появление которых предельно маловероятно. Надлежащее число значащих цифр следует сохранять в ходе всех промежуточных вычислений, как реализовано в алгоритмах A и M.

C. Аппаратная реализация арифметических действий над числами с плавающей точкой. В арсенале почти каждой большой ЭВМ, предназначенной для научных расчетов, содержатся встроенные операции и команды арифметических операций над числами с плавающей точкой. К несчастью, в аппаратных реализациях таких команд обычно присутствуют некоторые дефекты, приводящие при определенных обстоятельствах к удручающе скверному поведению машины, и, надо надеяться, в будущем создатели вычислительной техники будут уделять больше внимания данному вопросу. Затраты на это очень малы, и соображения, представленные в следующем разделе, показывают, какой выигрыш может быть достигнут. Из того, что сегодня известно, следует, что для современных компьютеров не подходят вчерашние компромиссные решения.

Компьютер MIX, используемый в этой серии книг как пример "типичной" вычислительной машины, оснащается средством расширения для работы с числами в формате с плавающей точкой — арифметическим расширителем. Он доступен за небольшую дополнительную сумму и обеспечивает выполнение следующих шести команд на аппаратном уровне.

• FADD, FSUB, FMUL, FDIV, FLOT, FCMP (C = 1, 2, 3, 4, 5, 56 соответственно; F = 6). Содержимое регистра rA после выполнения команды FADD V такое же, как и содержимое rA после выполнения команд

STA ACC; LDA V; JMP FADD.

Здесь FADD — подпрограмма, которая уже появлялась выше в этом разделе, но оба операнда автоматически нормализуются непосредственно перед входом в подпрограмму, если они еще не были нормализованы. (Если во время предварительной нормализации, но не во время нормализации результата, возникает исчезновение порядка, вызывающая программа о нем не извещается.) Аналогичные замечания относятся к операциям FSUB, FMUL и FDIV. Содержимое регистра rA после выполнения операции FLOT совпадает с его же содержимым после выполнения команды JMP FLOT в подпрограмме (10). Содержимое rA не искажается командой FCMP V. Эта команда устанавливает индикатор сравнения в состояние LESS, EQUAL или GREATER в зависимости от того, будет ли содержимое rA "заметно меньше, чем", "примерно равно" или "заметно больше, чем" V, как обсуждается в следующем разделе. Работа этой команды в точности моделируется подпрограммой FCMP из упр. 4.2.2–17 с EPSILON в ячейке 0.

Ни одна из команд арифметики с плавающей точкой не воздействует ни на какой другой регистр, помимо rA. Если происходит переполнение или исчезновение порядка, то включается индикатор переполнения и указывается порядок результата по модулю размера байта. Попытка деления на нуль оставляет в регистре rA "мусор" (произвольное значение). Времена выполнения: $4u$, $4u$, $9u$, $11u$, $3u$ и $4u$ соответственно.

• FIX (C = 5; F = 7). Содержимое rA заменяется целым числом "round(rA)", округленным до ближайшего целого, как на шаге N5 алгоритма N. Однако, если этот результат слишком велик и не вмещается в разрядную сетку регистра, устанавливается индикатор переполнения и результат должен трактоваться как неопределенный. Время выполнения: $3u$.

Иногда полезно использовать операторы арифметики с плавающей точкой нестандартным образом. Например, если бы операция FLOT не была реализована как часть арифметического расширителя компьютера MIX, можно было бы легко обеспечить ее выполнение для 4-байтовых чисел, написав маленькую подпрограмму

```
FLOT STJ  9F
     SLA  1
     ENTX Q+4
     SRC  1                    (12)
     FADD =0=
9H   JMP  *
```

Эта программа не эквивалентна команде FLOT, так как в ней предполагается, что 1:1 байт регистра rA равен нулю; кроме того, она портит содержимое регистра rX.

В более общих ситуациях приходится прибегать ко всяким хитростям, потому что переполнение при округлении может происходить даже во время выполнения команды FLOT.

Аналогично предположим, что MIX имеет команду FADD, но не имеет FIX. Чтобы округлить число u, записанное в формате с плавающей точкой, до ближайшего целого числа с фиксированной точкой (причем известно, что число неотрицательно и займет не более трех байтов), в программе можно записать

```
FADD  FUDGE,
```

где в ячейке FUDGE содержится константа

+	Q+4	1	0	0	0

;

результат в rA будет иметь вид

+	Q+4	1	Округленное(u)

. (13)

D. История и библиография. Истоки арифметики чисел с плавающей точкой прослеживаются вплоть до вавилонян (около 1800 г. до н. э. или ранее), которые применяли арифметические операции над числами с плавающей точкой по основанию 60, но не имели обозначения для порядка. Нужный порядок всегда некоторым образом "подразумевался" тем, кто производил вычисления. По крайней мере в одном случае обнаружено, что у вавилонян получился ошибочный ответ, потому что сложение осуществлялось при неверно выполненном выравнивании операндов, однако это случалось весьма редко (см. O. Neugebauer, *The Exact Sciences in Antiquity* (Princeton, N. J.: Princeton University Press, 1952), 26–27). Другой пример раннего обращения к формату с плавающей точкой связан с именем греческого математика Аполлония (3 в. до н. э.), который, по-видимому, был первым, кто объяснил, как можно упростить умножение, по крайней мере в простых случаях, собирая степени 10 отдельно от их коэффициентов. Метод Аполлония обсуждается в труде Паппа Александрийского "Математическое собрание" (4 в. н. э.). После гибели вавилонской цивилизации представление с плавающей точкой существенным образом использовалось для формирования произведений и частных лишь много веков спустя, когда были открыты логарифмы (1600 г.) и вскоре после этого Отред (Oughtred) изобрел логарифмическую линейку (1630 г.). Примерно в тот же период было введено современное обозначение "x^n" для порядков; отдельные символы для квадрата x, куба x и т. д. использовались и раньше.

Арифметика с плавающей точкой была включена в конструкцию некоторых из самых ранних вычислительных машин. Это было независимо предложено Леонардо Торресом-и-Овьедо (Leonardo Torres y Quevedo) в Мадриде (1914 г.), Конрадом Цузе (Konrad Zuse) в Берлине (1936 г.) и Джорджем Стибицем (George Stibitz) в Нью-Джерси (1939 г.). В машине Цузе использовалось двоичное представление с плавающей точкой, которое он назвал полулогарифмической нотацией; он также реализовал соглашение относительно операций с некоторыми особыми величинами, подобными "∞" и "не определено". Первой американской вычислительной машиной, в которой появились средства для выполнения операций в формате с плавающей точкой, была Модель V (Bell Laboratories), за которой последовала гарвардская машина Марк II. Обе эти машины, спроектированные в 1944 году,

были релейными вычислительными устройствами. [См. B. Randell, *The Origins of Digital Computers* (Berlin: Springer, 1973), 100, 155, 163–164, 259–260; *Proc. Symp. Large-Scale Digital Calculating Machinery* (Harvard, 1947), 41–68, 69–79; *Datamation* **13** (April, 1967), 35–44 (May, 1967), 45–49; *Zeit. für angew. Math. und Physik* **1** (1950), 345–346.]

Использование двоичных чисел с плавающей точкой серьезно обсуждалось в 1944–1946 годах группой исследователей из Школы Мура (Moore School) в связи с планами создания первой *электронной* вычислительной машины, но оказалось, что на лампах электронную схему, реализующую арифметику с плавающей точкой, выполнить гораздо труднее, чем на реле. Конструкторы первых электронных машин поняли, что масштабирование — это целая проблема в программировании, но они чувствовали, что это всего лишь небольшая часть общей работы по программированию в те годы. Конечно, масштабирование в явном виде чисел с фиксированной запятой казалось вполне окупающим затраченные время и усилия, поскольку программист при этом вынужден был все время держать вычисления в поле зрения и заботиться об их точности. Далее конструкторы возражали, что при представлении чисел с плавающей точкой занимается ценное место в памяти, так как нужно хранить порядки. Кроме того, схемотехнические решения арифметики с плавающей точкой трудно приспособить к вычислениям с многократной точностью. [См. von Neumann's *Collected Works* **5** (New York: Macmillan, 1963), 43, 73–74.] Конечно же, в это время они создавали машину, которая была первой машиной с хранимой в памяти программой и второй электронной машиной, и им предстояло выбрать формат *либо* с фиксированной, *либо* с плавающей точкой, но не оба сразу. Они предвосхитили составление программ двоичной арифметики с плавающей точкой, и команды "сдвиг влево" и "сдвиг вправо" фактически были введены в эти машины, главным образом, с целью повышения их эффективности. Первой машиной, которая имела средства для выполнения арифметических операций с обоими форматами, была, по-видимому, ЭВМ, разработанная фирмой "Дженерал Электрик" [см. *Proc. 2nd Symp. Large-Scale Digital Calculating Machinery* (Cambridge, Mass.: Harvard University Press, 1951), 65–69].

Подпрограммы для работы в формате с плавающей точкой, интерпретирующие системы для ранних ЭВМ, были составлены Д. Дж. Уилером (D. J. Wheeler) и другими и впервые опубликованы в книге Wilkes, Wheeler, Gill, *The Preparation of Programs for an Electronic Digital Computer* (Reading, Mass.: Addison-Wesley, 1951) (см. подпрограммы A1–A11). Интересно отметить, что в ней описаны программы для *десятичного* представления с плавающей точкой, хотя использовалась двоичная ЭВМ; другими словами, числа представлялись в виде $10^e f$, а не $2^e f$, и поэтому для операций сдвига требовались умножения или деления на 10. На этой машине десятичное масштабирование выполнялось почти так же просто, как сдвиг, а десятичное представление значительно упрощало ввод-вывод.

Авторы большинства публикаций при описании деталей реализации подпрограмм арифметики ссылаются на технические отчеты многочисленных производителей компьютеров, но иногда встречаются ссылки на открытые источники. Помимо упомянутых выше работ, определенный исторический интерес представляют следующие: R. H. Stark and D. B. MacMillan, *Math. Comp.* **5** (1951), 86–92, в которой описана программа, "прошитая" на специальной сменной панели; D. McCracken,

Digital Computer Programming (New York: Wiley, 1957), 121–131; J. W. Carr III, *CACM* **2**, 5 (May, 1959), 10–15; W. G. Wadey, *JACM* **7** (1960), 129–139; D. E. Knuth, *JACM* **8** (1961), 119–128; O. Kesner, *CACM* **5** (1962), 269–271; F. P. Brooks and K. E. Iverson, *Automatic Data Processing* (New York: Wiley, 1963), 184–199. Дискуссия относительно арифметики с плавающей точкой с точки зрения разработчиков компьютеров представлена в работах S. G. Campbell, "Floating point operation" в *Planning a Computer System*, edited by W. Buchholz (New York: McGraw-Hill, 1962), 92–121, A. Padegs, *IBM Systems J.* **7** (1968), 22–29. Дополнительный список источников, в основном, имеющих отношение к анализу точности вычислений в формате с плавающей точкой, представлен в разделе 4.2.2.

Поистине революционные изменения в аппаратной реализации арифметики с плавающей точкой произошли, когда большинство изготовителей компьютеров приняли к исполнению стандарт ANSI/IEEE Standard 754 в конце 80-х годов. (См. *IEEE Micro* **4** (1984), 86–100; W. J. Cody, *Comp. Sci. и Statistics: Symp. on the Interface* **15** (1983), 133–139; W. M. Kahan, *Mini/Micro West-83 Conf. Record* (1983), Paper 16/1; D. Goldberg, *Computing Surveys* **23** (1991), 5–48, 413; W. J. Cody and J. T. Coonen, *ACM Trans. Math. Software* **19** (1993), 443–451.)

Компьютер `MMIX`*, который заменит* `MIX` *в следующем издании данной книги, будет полностью соответствовать этому стандарту.*

УПРАЖНЕНИЯ

1. [*10*] Как будут выглядеть число Авогадро и постоянная Планка (3), если их представить в виде четырехразрядных чисел с плавающей точкой по основанию 100 с избытком 50? (Именно таким было бы представление в машине `MIX`, как в (4), если бы размер байта равнялся 100.)

2. [*12*] Предположим, порядок e находится в интервале $0 \le e \le E$. Каковы наибольшее и наименьшее положительные значения, которые могут быть записаны как p-разрядные числа с плавающей точкой по основанию b с избытком q? Каковы наибольшее и наименьшее положительные значения, которые могут быть представлены в виде таких *нормализованных* чисел?

3. [*11*] (К. Цузе (K. Zuse), 1936.) Покажите, что, работая с нормализованными двоичными числами с плавающей точкой, можно немного увеличить точность без увеличения объема используемой памяти: p-разрядную дробную часть можно представлять при помощи всего лишь $p - 1$ разрядов машинного слова, если чуть-чуть уменьшить интервал значений порядка.

▸ **4.** [*16*] Пусть $b = 10$, $p = 8$. Какой результат даст алгоритм A для операции $(50, +.98765432) \oplus (49, +.33333333)$, для операции $(53, -.99987654) \oplus (54, +.10000000)$ и для операции $(45, -.50000001) \oplus (54, +.10000000)$?

▸ **5.** [*24*] Будем говорить, что $x \sim y$ (по отношению к данному основанию b), если x и y — действительные числа, удовлетворяющие следующим условиям:

$$\begin{aligned}
\lfloor x/b \rfloor &= \lfloor y/b \rfloor;\\
x \bmod b = 0 &\iff y \bmod b = 0;\\
0 < x \bmod b < \tfrac{1}{2}b &\iff 0 < y \bmod b < \tfrac{1}{2}b;\\
x \bmod b = \tfrac{1}{2}b &\iff y \bmod b = \tfrac{1}{2}b;\\
\tfrac{1}{2}b < x \bmod b < b &\iff \tfrac{1}{2}b < y \bmod b < b.
\end{aligned}$$

Докажите, что между шагами A5 и A6 алгоритма A можно, не изменяя результата, заменить f_v на $b^{-p-2}F_v$, где $F_v \sim b^{p+2}f_v$. (Если F_v — целое и b четно, эта операция, по сути, позволяет "урезать" f_v до $p+2$ разрядов, запоминая, был ли отброшен любой разряд, отличный от нуля. В результате может быть минимизирована длина регистра, необходимого для сложения на шаге A6.)

6. [*20*] Если результат выполнения команды FADD равен нулю, каким будет знак регистра rA (если следовать описанию операций арифметического расширителя компьютера MIX, представленному в этом разделе)?

7. [*27*] Проанализируйте арифметические операции с плавающей точкой с использованием уравновешенной тернарной нотации.

8. [*20*] Приведите примеры нормализованных восьмиразрядных десятичных чисел с плавающей точкой u и v, для которых сложение влечет за собой (a) исчезновение порядка, (b) переполнение порядка, если подразумевать, что для порядков справедливо соотношение $0 \le e < 100$.

9. [*M24*] (У. М. Кахан (W. M. Kahan).) Предположим, что исчезновение порядка приводит к присвоению результату значения "нуль" без какой-либо индикации ошибки. Используя восьмиразрядные десятичные числа с плавающей точкой с избытком нуль и порядком e в интервале $-50 \le e < 50$, найдите такие положительные значения a, b, c, d и y, для которых выполняются соотношения (11).

10. [*12*] Приведите пример нормализованных восьмиразрядных десятичных чисел с плавающей точкой u и v, в процессе сложения которых происходит переполнение при округлении.

▶ **11.** [*M20*] Приведите пример нормализованных восьмиразрядных десятичных чисел с плавающей точкой u и v, в процессе умножения которых происходит переполнение при округлении.

12. [*M25*] Докажите, что переполнение при округлении не может происходить в ходе выполнения фазы нормализации при делении чисел с плавающей точкой.

13. [*30*] Имея дело с "арифметикой интервалов", нежелательно округлять результаты вычислений в формате с плавающей точкой. Скорее, было бы желательно реализовать операции, подобные $\triangledown$ и $\triangle$, которые дают наиболее близкое представление границ сумм:

$$u \mathbin{\triangledown} v \le u + v \le u \mathbin{\triangle} v.$$

Как модифицировать алгоритмы, описанные в данном разделе, чтобы они подходили для этой цели?

14. [*25*] Напишите подпрограмму для MIX, которая работала бы с произвольным исходным числом в регистре A, необязательно нормализованным, и преобразовывала бы его в ближайшее целое в формате с фиксированной точкой (или обнаруживала, что число слишком велико по абсолютной величине, чтобы было возможно такое преобразование).

▶ **15.** [*28*] Разработайте подпрограмму для MIX, которая по заданному числу в формате с плавающей точкой u вычисляет $u \bmod 1$, а именно $u - \lfloor u \rfloor$, округленное до ближайшего числа в формате с плавающей точкой. Подпрограмма должна быть увязана с остальными подпрограммами этого раздела. Обратите внимание на то, что когда u — очень малое отрицательное число, $u \bmod 1$ должно быть округлено таким образом, чтобы результат был равен единице (хотя $u \bmod 1$ по определению всегда должно давать результат, *меньший* единицы как действительного числа).

16. [*HM21*] (Роберт Л. Смит (Robert L. Smith).) Разработайте алгоритм для вычисления действительной и мнимой частей комплексного числа $(a+bi)/(c+di)$ по заданным действительным числам в формате с плавающей точкой a, b, c и d. Постарайтесь избежать

вычисления $c^2 + d^2$, поскольку это может привести к переполнению порядка даже тогда, когда $|c|$ или $|d|$ приблизительно равно квадратному корню максимально возможного числа в формате с плавающей точкой.

17. [*40*] (Джон Кок (John Cocke).) Реализуйте идею расширения диапазона представления чисел в формате с плавающей точкой, определив однословное представление, в котором точность дробной части уменьшается по мере того, как увеличивается значение абсолютной величины порядка.

18. [*25*] Представим себе двоичный компьютер с 36-битовым форматом слова, в котором положительные двоичные числа в формате с плавающей точкой представлены в виде $(0e_1e_2\ldots e_8f_1f_2\ldots f_{27})_2$; здесь $(e_1e_2\ldots e_8)_2$ есть избыток $(10000000)_2$ порядка и $(f_1f_2\ldots f_{27})_2$ есть 27-битовая дробная часть. Отрицательные числа в формате с плавающей точкой представлены *двумя дополнениями* соответствующих положительных представлений (см. раздел 4.1). Таким образом, 1.5 имеет вид *201|600000000* в восьмеричных обозначениях, а -1.5 имеет вид *576|200000000*; восьмеричные представления 1.0 и -1.0 есть *201|400000000* и *576|400000000* соответственно. (Вертикальные черточки использованы здесь для отображения границы в машинном слове между порядком и дробной частью.) Учтите, что бит f_1 для нормализованного положительного числа всегда равен 1, в то время как для отрицательного он почти всегда равен нулю; исключениями являются представления чисел -2^k.

Предположим, что точный результат операции в формате с плавающей точкой имеет в восьмеричном представлении вид *572|740000000|01*; эта отрицательная 33-битовая дробная часть должна быть нормализована и округлена до 27 бит. Если сдвигать ее влево до тех пор, пока первый бит дробной части не станет равным нулю, получится *576|000000000|20*. Но это приведет к округлению до неправильного значения *576|000000000*; в данном случае возникла "перенормализация", поскольку правильный результат — *575|400000000*. С другой стороны, если начать (в какой-нибудь другой задаче) со значения *572|740000000|05* и остановиться до возникновения перенормализации, получится *575|400000000|50*. Этот результат округляется до ненормализованного числа *575|400000001*; последующая нормализация приведет к результату *576|000000002*, в то время как верный результат — *576|000000001*.

Придумайте простое, но правильное правило округления, которое разрешит эту дилемму для такой машины (но принятый формат с двумя дополнительными представлениями должен остаться в неприкосновенности).

19. [*24*] Каково время выполнения подпрограммы **FADD** в программе A в терминах, отображающих характеристики исходных данных? Каково максимальное время выполнения для любых исходных данных, которые не приводят к переполнению или потере значимости порядка?

Округленные числа всегда лгут.

— СЭМЮЭЛЬ ДЖОНСОН (SAMUEL JOHNSON) (1750)

Я буду говорить в округленных числах, не абсолютно точно, но не настолько далеко от истины, чтобы изменить реальный результат.

— ТОМАС ДЖЕФФЕРСОН (THOMAS JEFFERSON) (1824)

4.2.2. Точность арифметических операций с плавающей точкой

Вычисления над числами в формате с плавающей точкой неточны по самой своей природе, и программисту нетрудно столь неудачно организовать их выполнение, что полученные результаты будут почти полностью состоять из "шума". Одна из главных проблем численного анализа состоит в анализе точности результатов тех или иных численных методов; сюда же относится и проблема "степени доверия": мы не знаем, насколько правильны результаты вычислений на компьютере. Пользователи-новички решают эту проблему, доверяя компьютеру, как непогрешимому авторитету; они склонны считать, что все цифры напечатанного ответа являются значащими. У пользователей, лишенных этих иллюзий, подход прямо противоположный: они неизменно опасаются, что полученные результаты весьма далеки от истинных. Многие из серьезных математиков пытались строго проанализировать последовательность операций с плавающей точкой, но, обнаружив, что задача слишком сложна, удовлетворялись правдоподобными рассуждениями.

Полное исследование методов анализа ошибок выходит, разумеется, за рамки настоящей книги, однако некоторые из характеристик ошибок, возникающих при вычислениях в формате с плавающей точкой, мы здесь все-таки рассмотрим. Наша цель — выяснить, как выполнять операции с плавающей точкой таким образом, чтобы, сохраняя достаточно высокий уровень достоверности, упростить, насколько это возможно, анализ распространения ошибки.

Грубый (но зачастую вполне приемлемый) способ, с помощью которого можно охарактеризовать выполнение операций арифметики с плавающей точкой, основан на понятии значащих разрядов или *относительной ошибки*. Если точное вещественное число x в компьютере представляется посредством приближения $\hat{x} = x(1+\epsilon)$, то величина $\epsilon = (\hat{x}-x)/x$ называется относительной ошибкой приближения. Грубо говоря, при выполнении вычислений в формате с плавающей точкой операции умножения и деления не слишком увеличивают относительную ошибку, но вычитание почти равных величин (и сложение $u \oplus v$, где u почти равно $-v$) может увеличить ее значительно. Итак, общее эмпирическое правило таково: существенной потери точности можно ожидать от сложения и вычитания указанного вида, но не от умножения и деления. С другой стороны, ситуация довольно парадоксальная и ее нужно правильно воспринимать, поскольку "плохое" сложение и вычитание всегда выполняется с великолепной точностью! (См. упр. 25.)

Одним из следствий возможной ненадежности сложения в формате с плавающей точкой является нарушение закона ассоциативности:

$$(u \oplus v) \oplus w \neq u \oplus (v \oplus w) \qquad \text{для многих } u, v, w. \tag{1}$$

Например:

$$(11111113. \oplus -11111111.) \oplus 7.5111111 = 2.0000000 \oplus 7.5111111 = 9.5111111;$$
$$11111113. \oplus (-11111111. \oplus 7.5111111) = 11111113. \oplus -11111103. = 10.000000.$$

(Все примеры в этом разделе приводятся в восьмиразрядном десятичном формате с плавающей точкой с представлением порядков посредством прямого указания места расположения десятичной точки. Напомним, что, как и в разделе 4.2.1, символы $\oplus$, $\ominus$, $\otimes$, $\oslash$ используются для обозначения операций в формате с плавающей точкой, соответствующих точным операциям $+$, $-$, $\times$, $/$.)

В свете возможного невыполнения закона ассоциативности приведенное в начале этой главы замечание госпожи Ла Туш (La Touche), если его отнести к арифметике в формате с плавающей точкой, несет в себе большую долю здравого смысла. Математические обозначения наподобие "$a_1+a_2+a_3$" и "$\sum_{k=1}^{n} a_k$" по самому своему существу основаны на предположении об ассоциативности, так что программист должен быть особенно бдителен на сей счет, задаваясь постоянно вопросом, не предполагается ли неявно справедливость закона ассоциативности.

А. Аксиоматический подход. Хотя закон ассоциативности и не выполняется, закон коммутативности

$$u \oplus v = v \oplus u \tag{2}$$

должен выполняться; последний может служить серьезным концептуальным подспорьем при программировании и анализе программ. Это соображение подсказывает нам, что следует искать наиболее существенные законы, которым удовлетворяют операции $\oplus$, $\ominus$, $\otimes$ и $\oslash$. Далее, вполне резонным представляется следующее соображение: *программы арифметических операций в формате с плавающей точкой следует составлять таким образом, чтобы сохранить действие как можно бóльшего числа обычных математических законов.* Если сохраняется действие большего числа аксиом, то легче разрабатывать хорошие программы, к тому же обеспечивая их переносимость с одной модели компьютера на другую.

Рассмотрим теперь другие основные законы, которые сохраняются для нормализованных операций с плавающей точкой, описанных в предыдущем разделе. Прежде всего, имеем

$$u \ominus v = u \oplus -v; \tag{3}$$

$$-(u \oplus v) = -u \oplus -v; \tag{4}$$

$$u \oplus v = 0 \quad \text{тогда и только тогда, когда} \quad v = -u; \tag{5}$$

$$u \oplus 0 = u. \tag{6}$$

Из этих законов можно получить и другие тождества, например (см. упр. 1)

$$u \ominus v = -(v \ominus u). \tag{7}$$

Тождества (2)–(6) легко выводятся из алгоритмов, описанных в разделе 4.2.1. Следующее правило менее очевидно:

$$\text{если} \quad u \le v, \quad \text{то} \quad u \oplus w \le v \oplus w. \tag{8}$$

Вместо того чтобы попытаться доказать это правило, анализируя алгоритм 4.2.1А, вернемся к основным принципам, на которых этот алгоритм базируется. (Доказательство с помощью алгоритма отнюдь не всегда проще и легче формулируется, чем привычное нам математическое доказательство.) Идея состоит в том, что операции в формате с плавающей точкой должны удовлетворять следующим равенствам:

$$\begin{aligned} u \oplus v &= \text{round}(u + v), \qquad u \ominus v = \text{round}(u - v), \\ u \otimes v &= \text{round}(u \times v), \qquad u \oslash v = \text{round}(u \;/\; v), \end{aligned} \tag{9}$$

где $\text{round}(x)$ означает наиболее близкое приближение в формате с плавающей точкой к x, как определено в алгоритме 4.2.1N. Имеем

$$\text{round}(-x) = -\text{round}(x), \tag{10}$$

$$x \le y \quad \text{влечет} \quad \text{round}(x) \le \text{round}(y). \tag{11}$$

Из этих фундаментальных соотношений свойства (2)–(8) следуют немедленно. Теперь можно выписать еще несколько тождеств, вытекающих из указанных выше соотношений:

$$u \otimes v = v \otimes u, \qquad (-u) \otimes v = -(u \otimes v), \qquad 1 \otimes v = v;$$
$$u \otimes v = 0 \quad \text{тогда и только тогда, когда} \quad u = 0 \text{ или } v = 0;$$
$$(-u) \oslash v = u \oslash (-v) = -(u \oslash v);$$
$$0 \oslash v = 0, \qquad u \oslash 1 = u, \qquad u \oslash u = 1.$$

Если $u \le v$ и $w > 0$, то $u \otimes w \le v \otimes w$ и $u \oslash w \le v \oslash w$; также $w \oslash u \ge w \oslash v$, если $v \ge u > 0$. Если $u \oplus v = u + v$, то $(u \oplus v) \ominus v = u$; если $u \otimes v = u \times v \ne 0$, то $(u \otimes v) \oslash v = u$. Как видно, несмотря на природную "неточность" операций в формате с плавающей точкой, им присуща значительная регулярность, если все как следует продумать.

Все же в приведенном выше наборе тождеств, разумеется, бросается в глаза отсутствие нескольких известных законов алгебры; закон ассоциативности для умножения в формате с плавающей точкой выполняется не вполне точно, как будет видно из упр. 3. Что же касается закона дистрибутивности, связывающего операции $\otimes$ и $\oplus$, то он может нарушаться, и при этом довольно значительно. Пусть, например, $u = 20000.000$, $v = -6.0000000$ и $w = 6.0000003$; тогда

$$(u \otimes v) \oplus (u \otimes w) = -120000.00 \oplus 120000.01 = .010000000$$
$$u \otimes (v \oplus w) = 20000.000 \otimes .00000030000000 = .0060000000.$$

Таким образом,

$$u \otimes (v \oplus w) \ne (u \otimes v) \oplus (u \otimes w). \tag{12}$$

С другой стороны, действительно справедливо $b \otimes (v \oplus w) = (b \otimes v) \oplus (b \otimes w)$, когда b есть основание системы счисления, поскольку

$$\text{round}(bx) = b\,\text{round}(x). \tag{13}$$

(Строго говоря, в тождествах и неравенствах, которые рассматриваются в этом разделе, неявно подразумевается, что потеря значимости или переполнение порядка не возникает. Функция округления $\text{round}(x)$ не определена, когда $|x|$ слишком мало́ или слишком велико, и равенства, подобные (13), имеют место только в случае, когда обе части определены.)

Другой впечатляющий пример нарушения законов традиционной алгебры при работе с числами в формате с плавающей точкой — невыполнение фундаментального неравенства Коши

$$(x_1^2 + \cdots + x_n^2)(y_1^2 + \cdots + y_n^2) \ge (x_1 y_1 + \cdots + x_n y_n)^2.$$

Как это может произойти, продемонстрировано в упр. 7, причем в совершенно ординарном случае, когда $n = 2$, $x_1 = x_2 = 1$. Программисты-новички имеют привычку использовать для программы вычисления среднего квадратичного отклонения для ряда наблюдений формулу из справочника

$$\sigma = \sqrt{\left(n \sum_{1 \le k \le n} x_k^2 - \left(\sum_{1 \le k \le n} x_k\right)^2\right) \Big/ n(n-1)} \tag{14}$$

и часто при выполнении программы "нарываются" на попытку извлечения квадратного корня из отрицательного числа! Значительно лучший метод вычисления среднего значения и стандартного отклонения с учетом свойств операций в формате с плавающей точкой состоит в использовании рекуррентных формул

$$M_1 = x_1, \qquad M_k = M_{k-1} \oplus (x_k \ominus M_{k-1}) \oslash k, \tag{15}$$

$$S_1 = 0, \qquad S_k = S_{k-1} \oplus (x_k \ominus M_{k-1}) \otimes (x_k \ominus M_k) \tag{16}$$

для $2 \le k \le n$, где $\sigma = \sqrt{S_n/(n-1)}$. [См. B. P. Welford, *Technometrics* **4** (1962), 419–420.] При использовании этого метода S_n никогда не может быть отрицательным и можно избежать множества других серьезных проблем, возникающих при слишком доверчивом отношении к накоплению сумм, как показано в упр. 16. (О методах суммирования, обеспечивающих даже более высокую гарантированную точность, речь идет также в упр. 19.)

Даже если алгебраические законы выполняются не вполне строго, можно использовать описанные методы для определения того, насколько точно выполняется тот или иной закон. Когда $b^{e-1} \le x < b^e$, имеем $\text{round}(x) = x + \rho(x)$, где $|\rho(x)| \le \frac{1}{2}b^{e-p}$. Следовательно,

$$\text{round}(x) = x\big(1 + \delta(x)\big), \tag{17}$$

где относительная ошибка ограничена независимо от x:

$$|\delta(x)| = \frac{|\rho(x)|}{|x|} \le \frac{|\rho(x)|}{b^{e-1} + |\rho(x)|} \le \frac{\frac{1}{2}b^{e-p}}{b^{e-1} + \frac{1}{2}b^{e-p}} < \tfrac{1}{2}b^{1-p}. \tag{18}$$

Это неравенство можно использовать в качестве простого инструмента для оценки относительной погрешности вычислений, выполняемых с нормализованными числами в формате с плавающей точкой, поскольку $u \oplus v = (u+v)\big(1+\delta(u+v)\big)$ и т. д.

В качестве примера типичной процедуры оценки ошибки рассмотрим закон ассоциативности умножения. Как показано в упр. 3, $(u \otimes v) \otimes w$, вообще говоря, не равно $u \otimes (v \otimes w)$, но ситуация в данном случае намного лучше, чем в случае применения закона ассоциативности сложения (1) и закона дистрибутивности (12). На самом деле, имеем

$$(u \otimes v) \otimes w = \big((uv)(1+\delta_1)\big) \otimes w = uvw(1+\delta_1)(1+\delta_2),$$

$$u \otimes (v \otimes w) = u \otimes \big((vw)(1+\delta_3)\big) = uvw(1+\delta_3)(1+\delta_4)$$

для некоторых δ_1, δ_2, δ_3, δ_4 при условии, что не происходит переполнения или исчезновения порядка, причем $|\delta_j| < \frac{1}{2}b^{1-p}$ для каждого j. Следовательно,

$$\frac{(u \otimes v) \otimes w}{u \otimes (v \otimes w)} = \frac{(1+\delta_1)(1+\delta_2)}{(1+\delta_3)(1+\delta_4)} = 1 + \delta,$$

где

$$|\delta| < 2b^{1-p}/\big(1 - \tfrac{1}{2}b^{1-p}\big)^2. \tag{19}$$

В анализе точности очень часто появляется число b^{1-p}, которому дано специальное наименование — один *ulp*, что означает одну единицу в последнем разряде дробной части (Unit in the Last Place). Операции с числами в формате с плавающей точкой дают результат с точностью до половины ulp, и вычисление uvw посредством двух умножений в формате с плавающей точкой имеет точность около одного ulp

(если отбросить члены второго порядка). Следовательно, закон ассоциативности для умножения справедлив вплоть до двух ulp относительной ошибки.

Таким образом, показано, что $(u \otimes v) \otimes w$ приблизительно равно $u \otimes (v \otimes w)$, за исключением тех случаев, когда происходит исчезновение или переполнение порядка. Эта интуитивная идея "приблизительного равенства" заслуживает более подробного изучения; можно ли дать более точную формулировку этого утверждения?

Программист, использующий арифметические операции в формате с плавающей точкой, почти никогда не испытывает желания проверить, не выполняется ли точное равенство двух вычисленных значений, так как равенство является крайне маловероятным. Например, если используется рекуррентное соотношение

$$x_{n+1} = f(x_n),$$

о котором известно из литературы, что x_n стремится к некоторому пределу при $n \to \infty$, то, скорее всего, будет ошибкой продолжать вычисления, пока для некоторого n не выполнится равенство $x_{n+1} = x_n$, так как последовательность x_n может ввиду округления промежуточных результатов оказаться периодической с большим периодом. Разумно продолжать вычисления лишь до тех пор, пока для некоторого подходящим образом выбранного δ не станет справедливым неравенство $|x_{n+1} - x_n| < \delta$; но так как порядок величины x_n заранее неизвестен, еще лучше — дождаться выполнения неравенства

$$|x_{n+1} - x_n| \le \epsilon |x_n|, \tag{20}$$

где ϵ — число, которое должно быть выбрано заранее. Соотношение (20) позволяет другим способом выразить то, что числа x_{n+1} и x_n приблизительно равны; и наше обсуждение показывает, что при анализе вычислений над числами с плавающей точкой отношение "приблизительного равенства" было бы более полезно, чем традиционное отношение равенства, если только первое отношение удастся определить надлежащим образом.

Другими словами, тот факт, что строгое равенство величин в формате с плавающей точкой играет очень небольшую роль, приводит к необходимости ввода новой операции *сравнения величин с плавающей точкой*, назначение которой — упростить оценку относительных значений двух таких величин. Представляются подходящими следующие определения для чисел с плавающей точкой $u = (e_u, f_u)$ и $v = (e_v, f_v)$ по основанию b с избытком q:

$$u \prec v \quad (\epsilon) \quad \text{тогда и только тогда, когда} \quad v - u > \epsilon \max(b^{e_u - q}, b^{e_v - q}); \tag{21}$$

$$u \sim v \quad (\epsilon) \quad \text{тогда и только тогда, когда} \quad |v - u| \le \epsilon \max(b^{e_u - q}, b^{e_v - q}); \tag{22}$$

$$u \succ v \quad (\epsilon) \quad \text{тогда и только тогда, когда} \quad u - v > \epsilon \max(b^{e_u - q}, b^{e_v - q}); \tag{23}$$

$$u \approx v \quad (\epsilon) \quad \text{тогда и только тогда, когда} \quad |v - u| \le \epsilon \min(b^{e_u - q}, b^{e_v - q}). \tag{24}$$

Эти определения подходят как для нормализованных чисел, так и для ненормализованных. Согласно этим определениям для любой данной пары значений u и v может выполняться в точности одно из соотношений и $u \prec v$ (определенно меньше), $u \sim v$ (приблизительно равно) или $u \succ v$ (определенно больше). Отношение $u \approx v$ несколько более сильное, чем $u \sim v$, и его можно читать так: "u, по существу,

и то же самое неравенство выполняется, если поменять местами $(u \otimes v) \otimes w$ и $u \otimes (v \otimes w)$. Следовательно, в соответствии с (34) справедливо соотношение

$$(u \otimes v) \otimes w \approx u \otimes (v \otimes w) \quad (\epsilon) \tag{39}$$

для $\epsilon \geq 2\epsilon_0/(1 - \frac{1}{2}\epsilon_0)^2$. Например, при $b = 10$ и $p = 8$ можно взять $\epsilon = 0.00000021$.

Соотношения $\prec$, $\sim$, $\succ$ и $\approx$ полезны для численных алгоритмов, и поэтому разумно включить в состав системного программного обеспечения компьютера программы для сравнения чисел с плавающей точкой наряду с программами для выполнения над ними арифметических действий.

Теперь вновь перейдем к вопросу о нахождении *точных* соотношений, которым удовлетворяют операции над числами с плавающей точкой. Интересно отметить, что сложение и вычитание таких величин не полностью выпадают из поля зрения аксиоматики, так как они удовлетворяют нетривиальным тождествам, сформулированным в следующих теоремах.

Теорема А. *Пусть u и v — нормализованные числа с плавающей точкой. Тогда*

$$\big((u \oplus v) \ominus u\big) + \big((u \oplus v) \ominus \big((u \oplus v) \ominus u\big)\big) = u \oplus v \tag{40}$$

при условии, что не происходит переполнения или исчезновения порядка.

Это довольно громоздкое тождество можно переписать в следующем более простом виде. Положим

$$\begin{aligned} u' &= (u \oplus v) \ominus v, \qquad & v' &= (u \oplus v) \ominus u; \\ u'' &= (u \oplus v) \ominus v', \qquad & v'' &= (u \oplus v) \ominus u'. \end{aligned} \tag{41}$$

Интуитивно ясно, что u' и u'' должны быть приближениями к u, а v' и v'' — приближениями к v. Теорема А утверждает, что

$$u \oplus v = u' + v'' = u'' + v'. \tag{42}$$

Это более сильное утверждение, нежели тождество

$$u \oplus v = u' \oplus v'' = u'' \oplus v', \tag{43}$$

являющееся следствием округления (42).

Доказательство. Будем говорить, что t является *остаточным членом* x (по модулю b^e), если

$$t \equiv x \ (\text{по модулю } b^e), \qquad |t| \leq \tfrac{1}{2}b^e. \tag{44}$$

Таким образом, $x - \text{round}(x)$ всегда равно остаточному члену x. Доказательство теоремы А в значительной мере основывается на следующих простых умозаключениях, доказанных в упр. 11.

Лемма Т. *Если t есть остаточный член числа в формате с плавающей точкой x, то $x \ominus t = x - t$.* ▮

Пусть $w = u \oplus v$. Теорема А становится тривиальной, когда $w = 0$. Умножив все переменные на подходящие степени b, можно, не теряя общности, предположить, что $e_w = p$. Тогда $u + v = w + r$, где r есть остаточный член $u + v$ (по модулю 1). Далее, $u' = \text{round}(w - v) = \text{round}(u - r) = u - r - t$, где t есть остаточный член $u - r$ (по модулю b^e) и $e = e_{u'} - p$.

Если $e \le 0$, то $t \equiv u - r \equiv -v$ (по модулю b^e). Следовательно, t есть остаточный член $-v$ и $v'' = \text{round}(w - u') = \text{round}(v + t) = v + t$, что доказывает (40). Если $e > 0$, то $|u - r| \ge b^p - \frac{1}{2}$, а поскольку $|r| \le \frac{1}{2}$, имеем $|u| \ge b^p - 1$. Из этого следует, что u есть целое число, так что r — остаточный член v (по модулю 1). Если $u' = u$, то $t = -r$ является остаточным членом $-v$. В противном случае соотношение $\text{round}(u - r) \ne u$ влечет за собой $|u| = b^p - 1$, $|r| = \frac{1}{2}$, $|u'| = b^p$, $t = r$; опять же, t — остаточный член $-v$. ▮

Теорема A выявляет некое свойство регулярности операции сложения в формате с плавающей точкой, но она не представляется уж очень полезным результатом. Следующая теорема гораздо более существенна.

Теорема B. *В предположениях теоремы A и при условии* (41) *справедливо тождество*

$$u + v = (u \oplus v) + \big((u \ominus u') \oplus (v \ominus v'')\big). \tag{45}$$

Доказательство. Рассматривая каждый из случаев, возникших при доказательстве теоремы A, мы неизменно обнаруживаем, что $u \ominus u' = u - u'$, $v \ominus v'' = v - v''$ и $(u - u') \oplus (v - v'') = (u - u') + (v - v'')$. Значит, (45) следует из теоремы A. Если учесть принятые в предыдущем доказательстве обозначения, эти соотношения окажутся эквивалентными следующим:

$$\text{round}(t + r) = t + r, \qquad \text{round}(t) = t, \qquad \text{round}(r) = r. \tag{46}$$

В упр. 12 рассматривается теорема для особого случая, когда $|e_u - e_v| \ge p$. Иначе $u+v$ имеет не более $2p$ значащих разрядов, и можно легко показать, что $\text{round}(r) = r$. Если теперь $e > 0$, доказательство теоремы A показывает, что $t = -r$ или $t = r = \pm\frac{1}{2}$. Если $e \le 0$, имеем $t + r \equiv u$ и $t \equiv -v$ (по модулю b^e); этого достаточно для доказательства того, что $t + r$ и t при округлении не изменяются (округляются до "самих себя"), обеспечивая выполнение неравенств $e_u \ge e$ и $e_v \ge e$. Но либо $e_u < 0$, либо $e_v < 0$ будут противоречить нашей гипотезе о том, что $|e_u - e_v| < p$, поскольку $e_w = p$. ▮

Теорема B дает *в явном виде формулу для получения разности* между $u + v$ и $u \oplus v$ в терминах величин, которые можно вычислить, непосредственно используя пять арифметических операций в формате с плавающей точкой. Если основание системы счисления b равно 2 или 3, можно улучшить этот результат и получить точные значения корректирующих членов, используя всего две арифметические операции в формате с плавающей точкой и одно сравнение абсолютных величин в формате с фиксированной точкой.

Теорема C. *Если* $b \le 3$ *и* $|u| \ge |v|$, *то*

$$u + v = (u \oplus v) + \big(u \ominus (u \oplus v)\big) \oplus v. \tag{47}$$

Доказательство. Следуя соглашениям, принятым при доказательстве предыдущих теорем, желательно показать, что $v \ominus v' = r$. Достаточно показать, что $v' = w - u$, поскольку из (46) затем последует $v \ominus v' = \text{round}(v - v') = \text{round}(u + v - w) = \text{round}(r) = r$.

Фактически нужно доказать (47) для любых $b \le 3$ и $e_u \ge e_v$. Если $e_u \ge p$, то r есть остаточный член v (по модулю 1). Значит, $v' = w \ominus u = v \ominus r = v - r = w - u$,

что и следовало доказать. Если $e_u < p$, должно оказаться, что $e_u = p - 1$ и $w - u$ кратно b^{-1}. Таким образом, результат будет совпадать с округленным значением, если его абсолютная величина меньше, чем $b^{p-1} + b^{-1}$. Поскольку $b \le 3$, мы, конечно же, получим $|w - u| \le |w - u - v| + |v| \le \frac{1}{2} + (b^{p-1} - b^{-1}) < b^{p-1} + b^{-1}$. Таким образом, доказательство завершено. ▮

В доказательствах теорем A, B и C нет ссылок на точное определение округления (x) в подозрительных случаях, когда x равно точно половине интервала между последовательными числами в формате с плавающей точкой. Как бы не разрешилась данная ситуация, это не отразится на истинности использованных в процессе доказательств.

Не существует правила округления на все случаи жизни. Например, обычно желательно иметь специальное правило для округления прибыли, облагаемой налогом. Но для большинства расчетов наилучшим следует считать предлагаемый в алгоритме 4.2.1N вариант, который "настаивает" на том, чтобы наименее значимый разряд был всегда четным (или всегда нечетным) в случаях, допускающих неоднозначность трактовки правил округления. Реализовать аппаратно такое правило — отнюдь не тривиальная задача. Однако существуют весьма серьезные практические соображения в пользу ее решения, поскольку такие неоднозначные ситуации часто возникают совершенно неожиданно и двойственное решение приводит к существенно отрицательным результатам. Например, рассмотрим действия в десятичной системе и будем считать, что остаток 5 всегда округляется в бóльшую сторону. Тогда, если $u = 1.0000000$ и $v = 0.55555555$, получим $u \oplus v = 1.5555556$. Если в формате с плавающей точкой из этого результата вычесть v, то получится $u' = 1.0000001$. Сложение и вычитание v из u' дает 1.0000002, а в следующий раз получим 1.0000003 и т. д. Таким образом, результат будет постоянно увеличиваться, хотя складывается и вычитается одно и то же число.

Это явление, называемое *дрейфом*, не возникнет, если использовать стабилизирующее правило округления, базирующееся на приоритете наименее значимого разряда.

Теорема D. $\big(((u \oplus v) \ominus v) \oplus v\big) \ominus v = (u \oplus v) \ominus v$.

Например, если $u = 1.2345679$ и $v = -0.23456785$, то

$$u \oplus v = 1.0000000, \qquad (u \oplus v) \ominus v = 1.2345678,$$
$$((u \oplus v) \ominus v) \oplus v = 0.99999995, \qquad \big(((u \oplus v) \ominus v) \oplus v\big) \ominus v = 1.2345678.$$

Доказательство для любых u и v, как нам кажется, потребует еще более скрупулезного анализа частных случаев, чем в теоремах, рассмотренных выше (см. ссылки на литературу, которые приведены ниже). ▮

Теорема D справедлива и в отношении "округленных до четного", и в отношении "округленных до нечетного", и возникает вопрос, какой же из вариантов выбрать. Когда основание системы счисления b нечетно, подозрительных случаев возникнуть не может, кроме как при делении в формате с плавающей точкой, а здесь округление, в общем-то, не имеет значения. Если же основание системы счисления

четно, есть смысл придерживаться следующего правила: "Округлять до четного, если $b/2$ нечетно, и округлять до нечетного, если $b/2$ четно". Наименее значимый разряд в дробной части числа в формате с плавающей точкой часто образуется в результате округления остатка при последовательно выполняемых операциях, и это правило позволяет, насколько это возможно, избежать формирования значения $b/2$ в наименее значимом разряде. На практике применение этого правила приводит к тому, что выделяется некоторая память для подозрительных операций округления, чтобы последовательные округления имели тенденцию к формированию результата, не вызывающего неоднозначного толкования. Например, если округлять до нечетного в десятичной системе, повторяющиеся операции округления числа 2.44445 с "укорачиванием" на один разряд дадут результат в виде последовательности 2.4445, 2.445, 2.45, 2.5, 3; если же округлять до четного, подобные ситуации не возникнут, хотя повторяющееся округление чисел наподобие 2.5454 приведет к почти такой же ошибке. [См. Roy A. Keir, *Inf. Proc. Letters* **3** (1975), 188–189.] Иногда предпочтение отдается округлению до четного в любом случае, и в результате наименее значимый разряд оказывается равным 0 чаще. В упр. 23 продемонстрировано это преимущество округления до четного. Но, в конце концов, ни один из альтернативных вариантов не имеет решающего преимущества над другими. К счастью, обычно в качестве основания системы счисления используется $b = 2$ или $b = 10$, и тогда все согласны, что правило округления до четного является наилучшим.

Читатель, который заинтересуется деталями доказательства изложенных выше положений, обнаружит существенное упрощение задачи, которое предоставляет простое правило $u \oplus v = \text{round}(u + v)$. Если бы подпрограмма сложения в формате с плавающей точкой не следовала этому правилу в любом, даже в самом редком, случае, доказательство стало бы существенно более сложным и, возможно, его вообще не удалось бы сформулировать.

Теорема B не выполняется, если вместо округления используется арифметическое "усечение", т. е. если положить $u \oplus v = \text{trunc}(u + v)$ и $u \ominus v = \text{trunc}(u - v)$, где $\text{trunc}(x)$ для действительных положительных x есть наибольшее число в формате с плавающей точкой, которое $\leq x$. Исключение из теоремы B возникло бы, следовательно, в случаях, подобных $(20, +.10000001) \oplus (10, -.10000001) = (20, +.10000000)$, когда разность между $u + v$ и $u \oplus v$ не может быть точно выражена в виде числа в формате с плавающей точкой. Также исключением был бы случай, подобный $12345678 \oplus .012345678$, если он может встретиться в практике решения конкретного круга вычислительных задач.

Многие думают, что, поскольку арифметика в формате с плавающей точкой неточна по самой своей природе, не будет никакой беды в том, чтобы, если это окажется удобным, в некоторых довольно редких случаях выполнять вычисления чуть менее точно. Такая политика сберегает несколько центов при проектировании аппаратуры компьютера или небольшую часть общего времени выполнения подпрограммы, но проведенный выше анализ показывает, что подобный подход ошибочен. Можно сэкономить до 5% времени выполнения подпрограммы FADD и программы 4.2.1A, а также около 25% занимаемого ими пространства в оперативной памяти, если позволить себе вольность и некорректно осуществить округление в редких случаях, но гораздо лучше оставить эти программы в прежнем виде. И дело здесь не в "погоне за битами"; на карту поставлено нечто более важное и фундаментальное:

подпрограммы работы с числами должны давать результаты, которые максимально удовлетворяют простым общепринятым математическим законам. Ключевая формула $u \oplus v = \text{round}(u + v)$, например, выражает некое свойство "регулярности", и, таким образом, решается вопрос, стоит ли проводить математический анализ вычислительных алгоритмов. Не располагая какими-либо свойствами симметрии, составляющими фундамент дальнейших рассуждений, было бы крайне неудобно доказывать интересные результаты. *Получать удовольствие от работы с привычным инструментом — это, конечно, одно из важнейших условий успешной работы.*

B. Арифметические действия над ненормализованными числами с плавающей точкой. К практике нормализации всех чисел в формате с плавающей точкой можно относиться двояко: либо благосклонно воспринимать ее как попытку получения максимально возможной точности, достижимой для данной разрядности, либо рассматривать ее как потенциально опасную политику в том смысле, что она "соблазняет" нас считать результаты более точными, чем есть на самом деле. Когда, нормализовав результат операции $(1, +.31428571) \ominus (1, +.31415927)$, мы получаем $(-2, +.12644000)$, теряется информация о возможно большей неточности результата. Такая информация сохранилась бы, если бы ответ остался в виде $(1, +.00012644)$.

При решении любой задачи, связанной с вычислениями, часто оказывается, что входные данные известны с точностью, меньшей, чем точность, обеспечиваемая форматом с плавающей точкой. Например, значения числа Авогадро и постоянной Планка с восемью значащими разрядами неизвестны, и было бы удобнее обозначать их соответственно через

$$(27, +.00060221) \qquad \text{и} \qquad (-23, +.00066261),$$

а не через $(24, +.60221400)$ и $(-26, +.66261000)$. Было бы прекрасно, если бы можно было задавать входные данные для каждой задачи в ненормализованной форме, из которой можно было бы почерпнуть информацию об их реальной точности, и если бы в выходных данных содержалась информация о том, какова точность результата. К несчастью, это ужасно сложная проблема, хотя использование ненормализованной арифметики и может помочь получить некоторые сведения такого рода. Например, можно говорить с большой степенью уверенности, что произведение числа Авогадро и постоянной Планка равно $(1, +.00039903)$, а их сумма равна $(27, +.00060221)$. (Данный пример приведен не для того, чтобы привести к мысли о том, будто сумме или произведению этих фундаментальных констант можно приписать какой-либо важный физический смысл, а чтобы продемонстрировать, что можно сохранить некоторую информацию о точности результата вычислений над неточными величинами, когда исходные операнды не зависят один от другого.)

Правила ненормализованной арифметики просты: пусть l_u — количество ведущих нулей в дробной части числа $u = (e_u, f_u)$, так что l_u есть наибольшее целое число $\le p$, для которого $|f_u| < b^{-l_u}$. Тогда сложение и вычитание выполняются точно так, как в алгоритме 4.2.1A, но все сдвиги влево, масштабирующие дробную часть, опускаются. Умножение и деление выполняются так же, как в алгоритме 4.2.1M, но результат сдвинут вправо или влево, так что его дробная часть будет начинаться в точности с $\max(l_u, l_v)$ нулей. По существу, те же правила на протяжении многих лет использовались и для традиционных вычислений вручную.

Таким образом, для вычислений с ненормализованными числами имеем

$$e_{u\oplus v},\ e_{u\ominus v} = \max(e_u, e_v) + (0 \text{ или } 1), \tag{48}$$

$$e_{u\otimes v} = e_u + e_v - q - \min(l_u, l_v) - (0 \text{ или } 1), \tag{49}$$

$$e_{u\oslash v} = e_u - e_v + q - l_u + l_v + \max(l_u, l_v) + (0 \text{ или } 1). \tag{50}$$

Если результатом должен быть нуль, то в качестве результата вычислений выдается ненормализованный нуль (часто его называют величиной порядка нуль). Это означает, что в действительности результат может быть и не равен нулю, но нам просто не известен ни один его значащий разряд! Для арифметических операций над ненормализованными числами применявшаяся ранее методика анализа ошибок принимает несколько другой вид. Введем теперь новую величину

$$\delta_u = \tfrac{1}{2} b^{e_u - q - p} \qquad \text{для } u = (e_u, f_u). \tag{51}$$

Она зависит от представления числа u, а не от его значения $b^{e_u - q} f_u$. Правило округления гласит:

$$|u \oplus v - (u + v)| \le \delta_{u\oplus v}, \qquad |u \ominus v - (u - v)| \le \delta_{u\ominus v},$$
$$|u \otimes v - (u \times v)| \le \delta_{u\otimes v}, \qquad |u \oslash v - (u \;/\; v)| \le \delta_{u\oslash v}.$$

Эти неравенства применимы как к нормализованным, так и к ненормализованным величинам; основное различие между двумя типами анализа ошибок состоит в том, что порядок результата каждой из операций (соотношения (48)–(50)) определяется по-другому.

Отношения $\prec$, $\sim$, $\succ$ и $\approx$, определенные выше в настоящем разделе, сохраняют смысл и для ненормализованных чисел. В качестве примера использования этих отношений докажем приближенный закон ассоциативности для сложения ненормализованных величин по аналогии с (39):

$$(u \oplus v) \oplus w \approx u \oplus (v \oplus w) \quad (\epsilon) \tag{52}$$

для подходящим образом выбранного ϵ. Имеем

$$\begin{aligned} |(u \oplus v) \oplus w - (u + v + w)| &\le \big|(u \oplus v) \oplus w - \big((u \oplus v) + w\big)\big| + |u \oplus v - (u + v)| \\ &\le \delta_{(u\oplus v)\oplus w} + \delta_{u\oplus v} \\ &\le 2\delta_{(u\oplus v)\oplus w}. \end{aligned}$$

Аналогичная формула справедлива и для $|u \oplus (v \oplus w) - (u + v + w)|$. Но поскольку $e_{(u\oplus v)\oplus w} = \max(e_u, e_v, e_w) + (0,\ 1 \text{ или } 2)$, то $\delta_{(u\oplus v)\oplus w} \le b^2 \delta_{u\oplus(v\oplus w)}$. Следовательно, (52) верно при $\epsilon \ge b^{2-p} + b^{-p}$. С точки зрения закона ассоциативности сложение ненормализованных величин не вызывает столько ошибок, как сложение нормализованных.

Следует подчеркнуть, что арифметика ненормализованных величин ни в коей мере не может служить панацеей; иногда указываемая при таком способе выполнения вычислений точность больше действительной (например, сложение большого числа малых приблизительно одинаковых по абсолютной величине чисел или нахождение x^n для большого n). Существует еще больше примеров, когда в такой арифметике дается невысокая точность, в то время как при выполнении операций над нормализованными величинами в действительности получались бы гораздо более точные результаты. Существует важная причина, по которой ни один прямой метод

анализа ошибок "по одной операции" не может считаться удовлетворительным, а именно: операнды обычно не являются независимыми. Это означает, что ошибки имеют тенденцию к компенсации или усилению одна другой необычным образом. Предположим, например, что x приблизительно равно $1/2$ и что у этого значения есть приближение $y = x+\delta$ с абсолютной ошибкой δ. Чтобы вычислить произведение $x(1-x)$, можно найти $y(1-y)$; если $x = \frac{1}{2} + \epsilon$, то получим $y(1-y) = x(1-x) - 2\epsilon\delta - \delta^2$. Таким образом, абсолютная ошибка существенно уменьшилась — перед ней появился коэффициент $2\epsilon + \delta$! Это всего лишь один из случаев, когда умножение приближенных значений операндов, не являющихся независимыми, может привести к довольно точному результату. Более очевидный пример — вычисление $x \ominus x$, которое может быть выполнено с абсолютной точностью независимо от того, насколько ошибочным было приближение x, которое используется в качестве операнда.

Дополнительная информация, которую предоставляет арифметика ненормализованных величин, часто может быть более важна, чем информация, потерянная при выполнении обширных вычислений в таком формате, но, как обычно, необходимо соблюдать осторожность при ее использовании. Примеры правильного использования арифметики ненормализованных величин проанализированы Р. Л. Эшенхерстом (R. L. Ashenhurst) и Н. Метрополисом (N. Metropolis) в сборнике *Computers и Computing, AMM*, Slaught Memorial Papers **10** (February, 1965), 47–59, Н. Метрополисом в журнале *Numer. Math.* **7** (1965), 104–112, и Р. Л. Эшенхерстом в сборнике *Error in Digital Computation* **2**, edited by L. B. Rall (New York: Wiley, 1965), 3–37. Надлежащие методы вычисления стандартных математических функций с представлением как входных, так и выходных данных в ненормализованной форме были описаны Р. Л. Эшенхерстом в *JACM* **11** (1964), 168–187. Расширение ненормализованной арифметики, которое предусматривает сохранение информации о том, что определенные операнды известны *точно*, проанализировано Н. Метрополисом в *IEEE Trans.* **C-22** (1973), 573–576.

С. Арифметика интервалов. Другой подход к проблеме оценки погрешности связан с так называемой арифметикой интервалов или диапазонов, которая предусматривает учет в процессе выполнения вычислений строгих границ — верхней и нижней — для каждого числа. Так, например, если известно, что $u_0 \le u \le u_1$ и $v_0 \le v \le v_1$, в записи в форме интервалов это будет иметь вид $u = [u_0 \,..\, u_1]$, $v = [v_0 \,..\, v_1]$. Сумма $u \oplus v$ есть $[u_0 \bigtriangledown v_0 \,..\, u_1 \bigtriangleup v_1]$, где $\bigtriangledown$ означает нижнюю сумму в формате с плавающей точкой, т. е. наибольшее представимое в данном формате число, меньшее или равное действительному значению суммы, а $\bigtriangleup$ определяется аналогично (см. упр. 4.2.1–13). Далее, $u \ominus v = [u_0 \bigtriangledown v_1 \,..\, u_1 \bigtriangleup v_0]$; если u_0 и v_0 положительны, то $u \otimes v = [u_0 \bigtriangledown v_0, u_1 \bigtriangleup v_1]$, $u \oslash v = [u_0 \bigtriangledown v_1 \,..\, u_1 \bigtriangleup v_0]$. Например, число Авогадро и постоянная Планка в обозначениях арифметики интервалов будут иметь такой вид:

$$N = \big[(24, +.60221331) \,..\, (24, +.60221403)\big],$$
$$h = \big[(-26, +.66260715) \,..\, (-26, +.66260795)\big].$$

Сумма и произведение этих величин будут выражены следующим образом:

$$N \oplus h = \big[(24, +.60221331) \,..\, (24, +.60221404)\big],$$
$$N \otimes h = \big[(-2, +.39903084) \,..\, (-2, +.39903181)\big].$$

Если попытаться выполнить деление на $[v_0 \,..\, v_1]$, причем $v_0 < 0 < v_1$, весьма вероятно возникновение деления на нуль. Поскольку самой философией арифметики интервалов предусматривается строгая оценка ошибки вычислений, в этом случае обязательно должно формироваться сообщение об ошибке деления на нуль. Однако переполнение и потеря значимости не рассматриваются как фатальные ошибки, если поддерживаются специальные соглашения, которые более подробно рассматриваются в упр. 24.

Вычисления с использованием арифметики интервалов лишь вдвое больше по объему, чем те же вычисления в обычном формате. В то же время данный метод обеспечивает надежную оценку ошибки. Учитывая сложность математического анализа ошибок, такую цену можно признать вполне приемлемой. Поскольку в процессе вычислений промежуточные значения часто зависят одно от другого, полученные с помощью арифметики интервалов оценки результатов часто слишком пессимистичны; если вы собираетесь использовать ее, измените итерационные численные методы решения привычных задач. Хотя перспективы эффективного использования арифметики интервалов выглядят довольно хорошо, нужно приложить определенные усилия, направленные на расширение ее функциональных возможностей и, насколько это возможно, обеспечение "дружественности" по отношению к пользователю.

D. История и библиография. В классическом трактате о методах вычислений в десятичной системе счисления *Leçons d'Arithmétique* (Paris: Colin, 1894), принадлежащем перу Жюля Таннери (Jules Tannery), утверждается, что положительные числа должны округляться в сторону большего, если первый отбрасываемый разряд равен или больше 5. Поскольку ровно половина десятичных цифр равна или больше 5, это, как ему казалось, будет приводить к округлению в бóльшую сторону в среднем ровно в половине случаев, так что в результате ошибки (округления. — *Прим. ред.*) компенсируются. Идея округлять до четного в неоднозначных случаях впервые, кажется, была упомянута Джеймсом Б. Скáрборо (James B. Scarborough) в первом издании его пионерской книги *Numerical Mathematical Analysis* (Baltimore: Johns Hopkins Press, 1930), 2; во втором издании (1950 г.) он развил ранние примечания по этому поводу, утверждая, что "для любого думающего человека должно быть очевидным, что, когда отбрасывается 5, предыдущий разряд должен быть увеличен на 1 только в *половине* случаев"; как средство достижения этого он рекомендовал округлять до четного.

Впервые анализ арифметических операций в формате с плавающей точкой был выполнен Ф. Л. Бауэром (F. L. Bauer) и К. Замельсоном (K. Samelson), *Zeitschrift für angewandte Math. und Physik* **4** (1953), 312–316. Следующая публикация появилась лишь пятью годами позже: J. W. Carr III, *CACM* **2**, 5 (May, 1959), 10–15. (См. также P. C. Fischer, *Proc. ACM Nat. Meeting* **13** (1958), Paper 39.) В книге J. H. Wilkinson, *Rounding Errors in Algebraic Processes* (Englewood Cliffs: Prentice-Hall, 1963) показано, как применять методы анализа ошибок отдельных арифметических операций к анализу ошибок в задачах с большим числом вычислений (см. также его монографию *The Algebraic Eigenvalue Problem* (Oxford: Clarendon Press, 1965)).

Обзор других ранних работ, касающихся точности вычислений в формате с плавающей точкой, приведен в двух важных статьях, которые можно настоятельно рекомендовать для дальнейшей проработки: W. M. Kahan, *Proc. IFIP Congress* (1971), **2**, 1214–1239, и R. P. Brent, *IEEE Trans.* **C-22** (1973), 601–607. В обеих статьях

содержатся весьма полезная теоретическая часть и демонстрация ее приложения на практике.

Введенные в этом разделе отношения $\prec$, $\sim$, $\succ$, $\approx$ имеют аналогию с идеями, сформулированным в работе A. Van Wijngaarden, *BIT* **6** (1966), 66–81. Приведенные выше теоремы A и B навеяны некоторыми работами в этой области Оле Меллера (Ole Moller), *BIT* **5** (1965), 37–50, 251–255. Теорема C взята из работы T. J. Dekker, *Numer. Math.* **18** (1971), 224–242. Расширение и уточнение всех трех теорем опубликовано в работе S. Linnainmaa, *BIT* **14** (1974), 167–202. У. М. Кахан (W. M. Kahan) сформулировал теорему D в неопубликованных заметках; полное ее доказательство и комментарии приведены в работе J. F. Reiser, D. E. Knuth, *Inf. Proc. Letters* **3** (1975), 84–87, 164.

Использовать арифметику ненормализованных чисел с плавающей точкой рекомендовали Ф. Л. Бауэр и К. Замельзон в упомянутой выше статье; независимо ее использовал Дж. В. Карр (J. W. Carr III) из Мичиганского университета (1953 г.). Несколькими годами позже был спроектирован компьютер MANIAC III с аппаратной реализацией арифметики обоих типов (см. R. L. Ashenhurst и N. Metropolis, *JACM* **6** (1959), 415–428, *IEEE Trans.* **EC-12** (1963), 896–901; R. L. Ashenhurst, *Proc. Spring Joint Computer Conf.* **21** (1962), 195–202). Среди других других ранних работ, касающихся ненормализованной арифметики, нужно также упомянуть статьи H. L. Gray, C. Harrison, Jr., *Proc. Eastern Joint Computer Conf.* **16** (1959), 244–248, и W. G. Wadey, *JACM* **7** (1960), 129–139.

О ранних исследованиях в области арифметики интервалов речь идет в работах A. Gibb, *CACM* **4** (1961), 319–320; B. A. Chartres, *JACM* **13** (1966), 386–403, и книге *Interval Analysis* by Ramon E. Moore (Prentice-Hall, 1966). Последующий "расцвет" в этой области приводится в более поздней книге Мура (Moore) *Methods and Applications of Interval Analysis* (SIAM, 1979).

Расширение языка Pascal, допускающее использование переменных наподобие "interval", разработано в Университете Карлсруэ в начале 80-х годов.

Описание этого языка, который включает и множество других функций, ориентированных на научные расчеты, приведено в работе *Pascal-SC* (Academic Press, 1987); авторы — Бохлендер (Bohlender), Ульрих (Ullrich), Вольф фон Гуденберг (Wolff von Gudenberg) и Ролл (Rall).

Книга Ulrich Kulisch, *Grundlagen des numerischen Rechnens: Mathematische Begründung der Rechnerarithmetik* (Mannheim: Bibl. Inst., 1976) полностью посвящена исследованию систем арифметики в формате с плавающей точкой. (См. также статью Кулиша (Kulisch) в журнале *IEEE Trans.* **C-26** (1977), 610–621, и его более позднюю работу в соавторстве с У. Л. Миранкером (W. L. Miranker), озаглавленную *Computer Arithmetic in Theory и Practice* (New York: Academic Press, 1981).)

Прекрасный обзор наиболее свежих работ, касающихся анализа точности выполнения расчетов в формате с плавающей точкой, появился в книге N. J. Higham, *Accuracy и Stability of Numerical Algorithms* (Philadelphia: SIAM, 1996).

УПРАЖНЕНИЯ

Замечание. Если не оговорено противное, предполагаются действия над нормализованными числами в формате с плавающей точкой.

1. [*M18*] Докажите, что тождество (7) следует из соотношений (2)–(6).

2. [*M20*] Используя тождества (2)–(8), докажите, что $(u \oplus x) \oplus (v \oplus y) \geq u \oplus v$, каковы бы ни были $x \geq 0$ и $y \geq 0$.

3. [*M20*] Найдите восьмиразрядные десятичные числа с плавающей точкой u, v и w, для которых

$$u \otimes (v \otimes w) \neq (u \otimes v) \otimes w,$$

причем ни при одном из этих вычислений не происходит ни переполнения, ни исчезновения порядка.

4. [*10*] Можно ли найти числа с плавающей точкой u, v и w, для которых при вычислении $u \otimes (v \otimes w)$ происходило бы исчезновение порядка, а при вычислении $(u \otimes v) \otimes w$ не происходило?

5. [*M20*] Выполняется ли для всех чисел с плавающей точкой u и $v \neq 0$ равенство $u \oslash v = u \otimes (1 \oslash v)$ таким образом, что не возникает ни переполнения, ни исчезновения порядка?

6. [*M22*] Для каждого из следующих двух соотношений выясните, выполняется ли оно тождественно для всех чисел u в формате с плавающей точкой: (a) $0 \ominus (0 \ominus u) = u$; (b) $1 \oslash (1 \oslash u) = u$.

7. [*M21*] Пусть $u^{(2)}$ означает $u \otimes u$. Найдите такие бинарные числа с плавающей точкой u и v, для которых $(u \oplus v)^{(2)} > 2(u^{(2)} + v^{(2)})$.

▶ **8.** [*20*] Пусть $\epsilon = 0.0001$; какое из соотношений

$$u \prec v \quad (\epsilon), \qquad u \sim v \quad (\epsilon), \qquad u \succ v \quad (\epsilon), \qquad u \approx v \quad (\epsilon)$$

выполняется для следующих пар восьмиразрядных десятичных чисел с плавающей точкой с избытком 0?

a) $u = (1, +.31415927)$, $v = (1, +.31416000)$;
b) $u = (0, +.99997000)$, $v = (1, +.10000039)$;
c) $u = (24, +.60221400)$, $v = (27, +.00060221)$;
d) $u = (24, +.60221400)$, $v = (31, +.00000006)$;
e) $u = (24, +.60221400)$, $v = (28, +.00000000)$.

9. [*M22*] Докажите утверждение (33) и объясните, почему его нельзя усилить до $u \approx w$ $(\epsilon_1 + \epsilon_2)$.

▶ **10.** [*M25*] (У. М. Кахан (W. M. Kahan).) На некотором компьютере неправильно выполняется округление при арифметических операциях над числами в формате с плавающей точкой, и фактически программа умножения принимает во внимание только первые p значащих разрядов $2p$-разрядного произведения $f_u f_v$. (Таким образом, если $f_u f_v < 1/b$, из-за последующей нормализации наименее значимый разряд в $u \otimes v$ всегда оказывается равным нулю.) Покажите, что это приводит к утрате монотонности умножения, т. е. что существуют такие положительные нормализованные числа с плавающей точкой u, v и w, что на этом компьютере $u < v$, но $u \otimes w > v \otimes w$.

11. [*M20*] Докажите лемму T.

12. [*M24*] Докажите теорему B и (46), если $|e_u - e_v| \geq p$.

▶ **13.** [*M25*] Некоторые языки программирования (и даже некоторые компьютеры) оперируют только числами в формате с плавающей точкой, не выделяя в отдельную группу точные операции над целыми числами. Если желательны и операции над целыми числами, можно, естественно, представить их в формате с плавающей точкой. Тогда, если операции арифметики с плавающей точкой удовлетворяют базовым соотношениям (9), можно считать, что все операции выполняются точно в том смысле, что, если операнды представлены p значащими разрядами, точными будут p значащих разрядов. Далее, до тех пор, пока

можно быть уверенным, что числа не слишком велики, можно складывать, вычитать и умножать целые числа и считать при этом, что округление не сказывается на точности.

Но предположим, что программисту необходимо определить, является ли m точно кратным n, когда m и $n \neq 0$ оба целые. Далее предположим, что в нашем распоряжении имеется подпрограмма для вычисления величины $\text{round}(u \bmod 1) = u \pmod 1$ для любого заданного числа u в формате с плавающей точкой, как в упр. 4.2.1–15. В качестве подходящего способа определения, является ли m кратным n, можно было бы проверить выполнение соотношения $(m \oslash n) \pmod 1 = 0$, подразумевая наличие указанной выше подпрограммы. Но, возможно, в некоторых случаях эта проверка не даст адекватного результата из-за ошибок округления при выполнении вычислений в формате с плавающей точкой.

Отыщите подходящие условия, которые нужно наложить на диапазон значений целых чисел $n \neq 0$ и m, такие, что m является кратным n тогда и только тогда, когда $(m \oslash n) \pmod 1 = 0$. Другими словами, покажите, что, если m и n не слишком велики, эта проверка дает верный результат.

14. [*M27*] Найдите подходящее значение ϵ, такое, что $(u \otimes v) \otimes w \approx u \otimes (v \otimes w)$ (ϵ), если выполняется *ненормализованное* умножение. (Этим обобщается соотношение (39), поскольку ненормализованное умножение нормализованных операндов u, v и w в точности повторяет результат нормализованного умножения.)

▸ **15.** [*M24*] (Г. Бьёрк (H. Björk).) Действительно ли вычисленная средняя точка интервала всегда находится между крайними точками? (Другими словами, действительно ли из $u \leq v$ следует, что $u \leq (u \oplus v) \oslash 2 \leq v$?)

16. [*M28*] (a) Чему равно $(\cdots((x_1 \oplus x_2) \oplus x_3) \oplus \cdots \oplus x_n)$ при использовании восьмиразрядного представления в формате с плавающей точкой, если $n = 10^6$ и $x_k = 1.1111111$ для любых k? (b) Что произойдет, если использовать выражение (14) для вычисления стандартного отклонения на множестве этих конкретных чисел x_k? Что произойдет, если вместо этого выражения использовать формулы (15) и (16)? (c) Докажите, что $S_k \geq 0$ в (16) при любых $x_1, \ldots, x_k$.

17. [*28*] Разработайте подпрограмму `FCMP` для `MIX`, сравнивающую число в формате с плавающей точкой u, которое находится по адресу `ACC`, с числом в формате с плавающей точкой v, которое находится в регистре A и устанавливает в индикаторе сравнения значение `LESS`, `EQUAL` или `GREATER` соответственно результату $u \prec v$, $u \sim v$ или $u \succ v$ (ϵ). Здесь ϵ хранится по адресу `EPSILON` в виде неотрицательного значения с разделяющей точкой, фиксированной слева от старшего разряда слова. Предполагается, что входные значения нормализованы.

18. [*M40*] Существует ли в арифметике ненормализованных чисел подходящее значение ϵ, такое, что $u \otimes (v \oplus w) \approx (u \otimes v) \oplus (u \otimes w)$ (ϵ)?

▸ **19.** [*M30*] (У. М. Кахан.) Проанализируйте следующие процедуры суммирования $x_1, \ldots, x_n$ в формате с плавающей точкой:

$$s_0 = c_0 = 0;$$
$$y_k = x_k \ominus c_{k-1}, \quad s_k = s_{k-1} \oplus y_k, \quad c_k = (s_k \ominus s_{k-1}) \ominus y_k, \qquad \text{где } 1 \leq k \leq n.$$

Будем считать, что относительная ошибка в этих операциях определяется по формулам

$$y_k = (x_k - c_{k-1})(1 + \eta_k), \qquad s_k = (s_{k-1} + y_k)(1 + \sigma_k),$$
$$c_k = ((s_k - s_{k-1})(1 + \gamma_k) - y_k)(1 + \delta_k),$$

где $|\eta_k|, |\sigma_k|, |\gamma_k|, |\delta_k| \leq \epsilon$. Докажите, что $s_n = \sum_{k=1}^{n}(1 + \theta_k)x_k$, где $|\theta_k| \leq 2\epsilon + O(n\epsilon^2)$. [Теорема C утверждает, что, если $b = 2$ и $|s_{k-1}| \geq |y_k|$, имеем точное равенство $s_{k-1} + y_k = s_k - c_k$. Но в данном упражнении желательно получить оценку, которая справедлива *даже*

в том случае, когда результат операции в формате с плавающей точкой округлен не очень аккуратно, при единственном предположении, что каждая операция дает ограниченную относительную ошибку.]

20. [*25*] (С. Линненмаа (S. Linnainmaa).) Найдите все u и v, для которых $|u| \geq |v|$, а соотношение (47) не выполняется.

21. [*M35*] (Т. Дж. Деккер (T. J. Dekker).) Теорема С показывает, как выполнить "точное" сложение чисел в формате с плавающей точкой. Покажите, как выполнить *"точное" умножение* чисел в формате с плавающей точкой: выразите произведение uv в форме $w + w'$, где w и w' вычислены на основе заданных чисел в формате с плавающей точкой u и v с использованием только операций $\oplus$, $\ominus$ и $\otimes$.

22. [*M30*] Может ли возникнуть дрейф при умножении и/или делении в формате с плавающей точкой? Рассмотрите последовательность $x_0 = u$, $x_{2n+1} = x_{2n} \otimes v$, $x_{2n+2} = x_{2n+1} \oslash v$ при заданных u и $v \neq 0$. Каково значение наибольшего индекса k, при котором возможно выполнение условия $x_k \neq x_{k+2}$?

▸ **23.** [*M26*] Докажите или опровергните следующее утверждение: для всех чисел u в формате с плавающей точкой справедливо равенство $u \ominus (u \bmod 1) = \lfloor u \rfloor$.

24. [*M27*] Рассмотрим множество всех интервалов $[u_l \,..\, u_r]$, где u_l и u_r есть либо отличные от нуля числа в формате с плавающей точкой, либо специальные символы $+0$, -0, $+\infty$, $-\infty$. Каждый интервал должен иметь $u_l \leq u_r$, и $u_l = u_r$ допускается только при условии, что u_l конечно и отлично от нуля. Интервал $[u_l \,..\, u_r]$ вмещает все числа x в формате с плавающей точкой, такие, что $u_l \leq x \leq u_r$, причем, мы полагаем, что

$$-\infty < -x < -0 < +0 < +x < +\infty$$

для всех возможных x. (Таким образом, $[1\,..\,2]$ означает $1 \leq x \leq 2$; $[+0\,..\,1]$ означает $0 < x \leq 1$; $[-0\,..\,1]$ означает $0 \leq x \leq 1$; $[-0\,..\,+0]$ представляет единственное значение 0; $[-\infty\,..\,+\infty]$ вмещает все.) Покажите, как определить соответствующие арифметические операции на всех этих интервалах, не принимая во внимание индикаторов переполнения или потери значимости, кроме как при делении на число из интервала, содержащего нуль.

▸ **25.** [*15*] Когда речь заходит о точности выполнения арифметических операций в формате с плавающей точкой, ошибки зачастую приписываются "отказам", которые возникают при вычитании близких по значению величин. Но, если u и v почти равны, разность $u \ominus v$ получается безо всяких ошибок. Что же тогда подразумевается под "отказами"?

26. [*M21*] Дано: u, u', v и v' — положительные числа в формате с плавающей точкой, причем $u \sim u'$ (ϵ) и $v \sim v'$ (ϵ). Предполагая использование нормализованной арифметики, докажите, что существует малое значение ϵ', такое, что $u \oplus v \sim u' \oplus v'$ (ϵ').

27. [*M27*] (У. М. Кахан.) Докажите, что $1 \oslash (1 \oslash (1 \oslash u)) = 1 \oslash u$ для любых $u \neq 0$.

28. [*HM30*] (Г. Дж. Диамонд (H. G. Diamond).) Предположим, $f(x)$ есть жестко возрастающая функция на некотором интервале $[x_0 \,..\, x_1]$, и пусть $g(x)$ — обратная ей функция. (Например, f и g могут быть "exp" и "ln" или "tan" и "arctan".) Если x — число в формате с плавающей точкой, такое, что $x_0 \leq x \leq x_1$, то пусть $\hat{f}(x) = \text{round}(f(x))$ и, если y — другое число, такое, что $f(x_0) \leq y \leq f(x_1)$, пусть $\hat{g}(y) = \text{round}(g(y))$. Далее, пусть $h(x) = \hat{g}(\hat{f}(x))$, если только оно определено. Хотя $h(x)$ и не всегда равно x из-за округления, можно рассчитывать, что $h(x)$ очень близко к x.

Докажите, что если точность b^p как минимум равна 3 и если f — жестко вогнутая или жестко выпуклая (т. е. $f''(x)$ имеет тот же знак для всех x, принадлежащих интервалу $[x_0 \,..\, x_1]$), то многократное применение h будет *устойчивым* в том смысле, что

$$h(h(h(x))) = h(h(x))$$

для всех x, таких, что обе части этого равенства определены. Другими словами, не должно быть никакого "дрейфа", если подпрограмма правильно запрограммирована.

▸ **29.** [*M25*] Приведите пример, чтобы показать, что $b^p \geq 3$ есть необходимое условие для предыдущего упражнения.

▸ **30.** [*M30*] (У. М. Кахан.) Пусть $f(x) = 1 + x + \cdots + x^{106} = (1 - x^{107})/(1 - x)$ для $x < 1$ и пусть $g(y) = f((\frac{1}{3} - y^2)(3 + 3.45y^2))$ при $0 < y < 1$. Вычислите $g(y)$ на разных карманных калькуляторах для $y = 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$ и объясните, почему все результаты, которые при этом получены, различны. (Поскольку в большинстве современных калькуляторов округление выполняется неправильно, результаты могут стать для вас большим сюрпризом. Обратите внимание, что $g(\epsilon) = 107 - 10491.35\epsilon^2 + 659749.9625\epsilon^4 - 30141386.26625\epsilon^6 + O(\epsilon^8)$.)

31. [*M25*] (У. Кулиш (U. Kulisch).) При вычислении полинома $2y^2 + 9x^4 - y^4$ для $x = 408855776$ и $y = 708158977$ с использованием стандартного пакета подпрограмм арифметики с плавающей точкой с двойной точностью (53 бит) получен результат $\approx -3.7 \times 10^{19}$. Вычисление по альтернативной формуле $2y^2 + (3x^2 - y^2)(3x^2 + y^2)$ дает $\approx +1.0 \times 10^{18}$. Правильный ответ, однако, — точно 1.0. Объясните, как строить аналогичные примеры нестабильности вычислений.

*4.2.3. Вычисления с удвоенной точностью

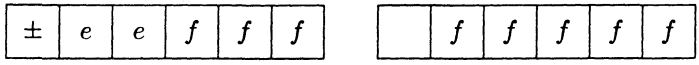
До сих пор речь шла об арифметических операциях над числами с однократной точностью в формате с плавающей точкой. Это, по существу, означает, что числа в формате с плавающей точкой, с которыми мы имели дело, могли храниться в одном машинном слове. Если вычисления в таком формате не обеспечивают достаточной для приложения точности, ее можно увеличить программными средствами, используя для представления каждого числа два или больше слов памяти.

Хотя проблема вычислений с высокой точностью будет в общем виде рассмотрена в разделе 4.3, имеет смысл отдельно проанализировать возможности формата с удвоенной точностью представления. Специальные методы, которые используются при работе с таким форматом, практически непригодны для большей точности. Кроме того, вычисления с удвоенной точностью разумно считать темой, имеющей самостоятельное значение, так как это первый шаг за пределы однократной точности и, работая в этом формате, можно удовлетворительно решать многие задачи, не требующие чрезвычайно высокой точности.

Материал настоящего раздела был актуален, когда автор работал над первым изданием этой книги в начале 60-х годов. Но с тех пор компьютеры были значительно усовершенствованы и причины, по которым использовались операции с удвоенной точностью, ушли в прошлое. Таким образом, на сегодняшний день данный материал представляет скорее исторический интерес. В планируемом четвертом издании этой книги раздел 4.2.1 будет называться "Нормализованные вычисления", а в настоящем разделе, 4.2.3, будут рассматриваться "необычные числа". В новой редакции данного раздела внимание будет сосредоточено на специфических аспектах стандарта ANSI/IEEE Standard 754: представлении необычных числовых величин наподобие бесконечности и неопределенности (так называемых NaNs). (См. ссылки на литературу в конце раздела 4.2.1.) Тем не менее давайте все-таки бросим последний взгляд на идеи, которые кажутся устаревшими, хотя бы для того, чтобы извлечь из них урок на будущее.

Арифметические операции с удвоенной точностью требуются чаще всего при работе с числами в формате с плавающей точкой и значительно реже — при работе с числами в формате с фиксированной точкой. Исключением является, пожалуй, лишь область статистических расчетов, в которой для вычисления сумм квадратов и перекрестных произведений обычно используются числа с фиксированной точкой и имеет смысл выполнять операции с удвоенной точностью. Поскольку вычисления с удвоенной точностью для чисел с фиксированной точкой проще, чем с плавающей, наш анализ будет ограничен этим последним случаем.

Удвоенная точность очень часто желательна для увеличения не только точности дробных частей чисел в формате с плавающей точкой, но и интервала изменения порядка. Таким образом, в этом разделе речь пойдет о следующем двухсловном представлении чисел с плавающей точкой в компьютере MIX:



(1)

Здесь используются 2 байта для хранения порядка и 8 байт для дробной части. Порядок имеет избыток $b^2/2$, где b — размер байта. Знак находится в слове, содержащем более значимые разряды; знак другого слова принято полностью игнорировать.

Наш анализ арифметики с удвоенной точностью будет в значительной степени машинно-ориентированным, потому что, только рассматривая задачи, которые возникают при разработке программ, можно правильно воспринять предмет. Поэтому для понимания данного раздела важно внимательно изучить приводимые ниже программы для MIX.

Здесь мы постараемся "отдалиться" от идеалистических целей достижения точности, продекларированных в двух предыдущих разделах; рассматриваемые ниже процедуры *не включают* округления результатов и допускают наличие некоторой ошибки. Пользователи не рискуют слишком уж доверять такого рода подпрограммам. Существуют весьма основательные причины блокировать все возможные источники ошибок при вычислениях с однократной точностью, но теперь мы сталкиваемся с иной ситуацией, и вот почему.

(а) Дополнительные фрагменты программ, требующиеся для обеспечения правильности округления во всех случаях, довольно объемны, так что, скажем, программа, реализующая такой алгоритм, занимала бы раза в два больше места и требовала бы для выполнения раза в полтора больше времени. Сравнительно легко можно разработать прекрасные программы для вычисления с однократной точностью, но при вычислениях с удвоенной точностью лицом к лицу мы сталкиваемся с ограниченными возможностями компьютеров*. Аналогичная ситуация возникает и в отношении других подпрограмм, выполняющих вычисления в формате с плавающей точкой. Нельзя ожидать от подпрограммы вычисления косинуса определения точного значения $\mathrm{round}(\cos x)$ для всех x, поскольку это фактически невозможно. Вместо этого подпрограмма вычисления косинуса должна выдавать наименьшую возможную относительную ошибку при всех возможных значениях x, которую можно получить за разумное время вычисления. Конечно, программист должен стараться насколько это возможно обеспечить выполнение математических

* Автор, очевидно, имеет в виду компьютеры — современники первого издания этой книги. — *Прим. перев.*

законов, существующих в отношении вычисляемых функций, например

$$\text{(cos)}\,(-x) = \text{(cos)}\,x; \quad |\text{(cos)}\,x| \le 1; \quad \text{(cos)}\,x \ge \text{(cos)}\,y \text{ при } 0 \le x \le y < \pi.$$

(b) Программы вычислений с однократной точностью — это "хлеб насущный" на каждый день, который должен быть доступен каждому, кто хочет работать с арифметикой с плавающей точкой. А удвоенная точность обычно используется в ситуациях, когда "чистые" результаты не так уж и важны. Разница между вычислениями с семью или восемью значащими разрядами достаточно заметна, а разница между вычислениями с 15 или 16 разрядами нас не очень беспокоит. Вычисления с удвоенной точностью чаще всего используются на промежуточных этапах решения сложной задачи с однократной точностью, а потому в полном спектре возможностей таких процедур нет необходимости.

(c) Весьма поучительно проанализировать эти процедуры с тем, чтобы увидеть, насколько серьезной может быть ошибка, поскольку источники ошибок типичны и для "плохих" подпрограмм, работающих с однократной точностью (см. упр. 7 и 8).

Рассмотрим теперь с этой точки зрения операции сложения и вычитания. Вычитание, естественно, заменяется сложением с инвертированием знака второго слагаемого. Для выполнения сложения используется раздельное сложение менее значимых и более значимых половин, обеспечивая при этом, разумеется, надлежащую обработку переноса.

Сложности возникают, однако, вследствие использования прямого кода: можно, сложив менее значимые половины, получить неверный знак (эта ситуация возникает, когда знаки операндов противоположны и менее значимая половина меньшего операнда больше менее значимой половины большего операнда). Простейшее решение — заранее определить правильный знак. Поэтому на шаге A2 алгоритма 4.2.1A нужно положить не только условие $e_u \ge e_v$, но и $|u| \ge |v|$. Тогда можно быть уверенным в том, что знак результата будет знаком u. В прочих отношениях программа сложения с удвоенной точностью очень похожа на программу с однократной точностью, но все выполняется дважды.

Программа A (*Сложение с удвоенной точностью*). Подпрограмма DFADD складывает число удвоенной точности с плавающей точкой v, имеющее формат (1), с числом удвоенной точности в формате с плавающей точкой u. Причем предполагается, что v перед началом выполнения программы занесено в rAX (регистры A и X), а u первоначально хранится в ячейках ACC и ACCX. Результат появляется одновременно в rAX и в (ACC, ACCX). Подпрограмма DFSUB вычитает v из u при соблюдении тех же соглашений.

Предполагается, что оба входных операнда нормализованы и результат также нормализован. Заключительный фрагмент программы представляет собой процедуру нормализации в формате с удвоенной точностью, используемую и в других подпрограммах этого раздела. В упр. 5 показано, как можно существенно усовершенствовать эту программу.

```
01  ABS    EQU  1:5      Определение поля для абсолютного значения.
02  SIGN   EQU  0:0      Определение поля для знака.
03  EXPD   EQU  1:2      Поле порядка для формата с удвоенной точностью.
04  DFSUB  STA  TEMP     Вычитание с удвоенной точностью.
05         LDAN TEMP     Изменить знак v.
```

```
06  DFADD  STJ   EXITDF        Сложение с удвоенной точностью.
07         CMPA  ACC(ABS)      Сравнить |v| с |u|.
08         JG    1F
09         JL    2F
10         CMPX  ACCX(ABS)
11         JLE   2F
12  1H     STA   ARG           Если |v| > |u|, произвести взаимную замену u ↔ v.
13         STX   ARGX
14         LDA   ACC
15         LDX   ACCX
16         ENT1  ACC           (ACC и ACCX находятся в
17         MOVE  ARG(2)            последовательных ячейках.)
18  2H     STA   TEMP
19         LD1N  TEMP(EXPD)    rI1 ← −e_v.
20         LD2   ACC(EXPD)     rI2 ← e_u.
21         INC1  0,2           rI1 ← e_u − e_v.
22         SLAX  2             Удалить порядок.
23         SRAX  1,1           Масштабирование посредством сдвига вправо.
24         STA   ARG           0 v1 v2 v3 v4
25         STX   ARGX                  v5 v6 v7 v8 v9.
26         STA   ARGX(SIGN)    Запомнить верный знак v в обеих половинах.
27         LDA   ACC           (Известно, что u имеет знак результата.)
28         LDX   ACCX          rAX ← u.
29         SLAX  2             Удалить порядок.
30         STA   ACC           u1 u2 u3 u4 u5.
31         SLAX  4
32         ENTX  1
33         STX   EXPO          EXPO ← 1 (см. ниже).
34         SRC   1             1 u5 u6 u7 u8.
35         STA   1F(SIGN)      См. комментарии в тексте.
36         ADD   ARGX(0:4)     Сложить 0 v5 v6 v7 v8.
37         SRAX  4
38  1H     DECA  1             Восстановить после занесения 1 (знак варьируется).
39         ADD   ACC(0:4)      Сложить более значимые половины.
40         ADD   ARG           (Переполнения быть не может.)

41  DNORM  JANZ  1F            Подпрограмма нормализации.
42         JXNZ  1F            f_w в rAX, e_w = EXPO + rI2.
43  DZERO  STA   ACC           Если f_w = 0, присвоить e_w ← 0.
44         JMP   9F
45  2H     SLAX  1             Нормализовать влево.
46         DEC2  1
47  1H     CMPA  =0=(1:1)      Ведущий байт нулевой?
48         JE    2B
49         SRAX  2             (Округление опущено.)
50         STA   ACC
51         LDA   EXPO          Вычислить порядок результата.
52         INCA  0,2
53         JAN   EXPUND        Он отрицателен?
54         STA   ACC(EXPD)
55         CMPA  =1(3:3)=      Он занимает более двух байтов?
```

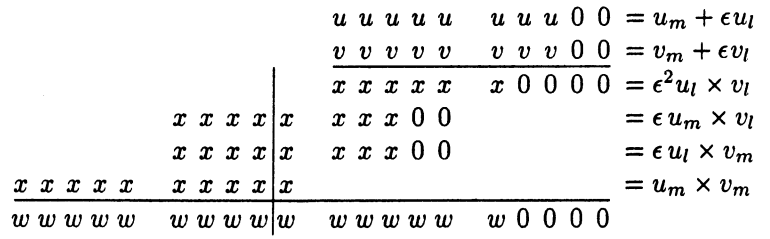


Рис. 4. Умножение с удвоенной точностью чисел, имеющих дробные части длиной 8 байт.

```
56          JL   8F
57  EXPOVD  HLT  20
58  EXPUND  HLT  10
59  8H      LDA  ACC        Переслать ответ в rA.
60  9H      STX  ACCX
61  EXITDF  JMP  *          Выход из подпрограммы.
62  ARG     CON  0
63  ARGX    CON  0
64  ACC     CON  0          Аккумулятор для чисел с плавающей точкой.
65  ACCX    CON  0
66  EXPO    CON  0          Часть "грубого" порядка.  ▮
```

Когда в этой подпрограмме складываются менее значимые половины, в то слово, о котором известно, что оно имеет верный знак, слева вставляется дополнительный разряд "1". После выполнения сложения этот байт может равняться 0, 1 или 2 в зависимости от обстоятельств и все эти три случая обрабатываются одновременно. (Ср. данный фрагмент с довольно громоздким методом формирования дополнения, который использовался в программе 4.2.1A.)

Стоит отметить, что после выполнения команды в строке 40 регистр A может оказаться равным нулю, но аккумулятор будет содержать правильный знак, который должен быть приписан результату, если регистр X не равен нулю (это есть следствие способа, с помощью которого в компьютере MIX определяется знак результата, равного нулю). Поменяй мы местами строки 39 и 40, программа давала бы неверный результат, хотя обе команды имеют один и тот же код "ADD"!

Рассмотрим теперь умножение с удвоенной точностью. Произведение имеет четыре компоненты, схематически представленные на рис. 4. Так как нам нужны только восемь крайних слева байтов, можем игнорировать разряды, расположенные справа от вертикальной черты на диаграмме. В частности, это означает, что нет необходимости даже вычислять произведение двух менее значимых половин.

Программа M (*Умножение с удвоенной точностью*). Соглашения относительно входных и выходных данных для этой подпрограммы те же, что и для программы A.

```
01  BYTE    EQU  1(4:4)        Размер байта.
02  QQ      EQU  BYTE*BYTE/2   Избыток порядка удвоенной точности.
03  DFMUL   STJ  EXITDF        Умножение с удвоенной точностью.
04          STA  TEMP
05          SLAX 2             Удалить порядок.
06          STA  ARG           v_m.
```

```
07        STX  ARGX
08        LDA  TEMP(EXPD)
09        ADD  ACC(EXPD)
10        STA  EXPO          EXPO ← e_u + e_v.
11        ENT2 -QQ           rI2 ← −QQ.
12        LDA  ACC
13        LDX  ACCX
14        SLAX 2             Удалить порядок.
15        STA  ACC           u_m.
16        STX  ACCX          u_l.
17        MUL  ARGX          u_m × v_l.
18        STA  TEMP
19        LDA  ARG(ABS)
20        MUL  ACCX(ABS)     |v_m × u_l|.
21        SRA  1             0 x x x x.
22        ADD  TEMP(1:4)     (Переполнение невозможно.)
23        STA  TEMP
24        LDA  ARG
25        MUL  ACC           v_m × u_m
26        STA  TEMP(SIGN)    Верный знак результата.
27        STA  ACC           Теперь приготовиться к сложению
28        STX  ACCX                всех частичных произведений.
29        LDA  ACCX(0:4)     0 x x x x.
30        ADD  TEMP          (Переполнение невозможно.)
31        SRAX 4
32        ADD  ACC           (Переполнение невозможно.)
33        JMP  DNORM         Нормализовать и выйти из подпрограммы. ▮
```

Обратите внимание на аккуратное обращение со знаками в этой программе, а также на тот факт, что диапазон представления порядков позволяет вычислять порядок результата при помощи индексного регистра. Программа M, вероятно, несколько "хромает" в отношении точности, так как она использует только информацию, расположенную на рис. 4 слева от вертикальной черты, а это может привести к ошибке до двух единиц в наименее значимом байте. Как достичь несколько большей точности, вы узнаете из упр. 4.

Программа деления с удвоенной точностью чисел в формате с плавающей точкой — самая трудная или, по крайней мере, самая страшная на вид из всех программ, которые до сих пор рассматривались в этой главе. На самом деле она не такая уж сложная, если разобраться в принципе ее работы. Запишем пару чисел, участвующих в делении, в виде $(u_m + \epsilon u_l)/(v_m + \epsilon v_l)$, где ϵ — величина, обратная размеру машинного слова, причем предполагается, что число v_m нормализовано. Эту дробь можно разложить в ряд:

$$\begin{aligned}\frac{u_m+\epsilon u_l}{v_m+\epsilon v_l} &= \frac{u_m+\epsilon u_l}{v_m}\left(\frac{1}{1+\epsilon(v_l/v_m)}\right)\\ &= \frac{u_m+\epsilon u_l}{v_m}\left(1-\epsilon\left(\frac{v_l}{v_m}\right)+\epsilon^2\left(\frac{v_l}{v_m}\right)^2-\cdots\right).\end{aligned}\tag{2}$$

Так как $0 \le |v_l| < 1$ и $1/b \le |v_m| < 1$, получим $|v_l/v_m| < b$ и можно пренебречь ошибкой, связанной с отбрасыванием членов порядка ϵ^2. Наш метод состоит, таким образом, в том, чтобы вычислить $w_m + \epsilon w_l = (u_m + \epsilon u_l)/v_m$, а затем вычесть из результата ϵ раз $w_m v_l / v_m$.

В приведенной ниже программе команды строк 27–32 выполняют нижнюю половину сложения с удвоенной точностью, используя иной метод получения нужного знака, чем в программе A.

Программа D (*Деление с удвоенной точностью*). Эта программа работает при тех же предположениях, что и программы A и M.

```
01  DFDIV  STJ   EXITDF          Деление с удвоенной точностью.
02         JOV   OFLO            Убедиться, что индикатор переполнения
                                     выключен.
03         STA   TEMP
04         SLAX  2               Удалить порядок.
05         STA   ARG             v_m.
06         STX   ARGX            v_l.
07         LDA   ACC(EXPD)
08         SUB   TEMP(EXPD)
09         STA   EXPO            EXPO ← e_u − e_v.
10         ENT2  QQ+1            rI2 ← QQ + 1.
11         LDA   ACC
12         LDX   ACCX
13         SLAX  2               Удалить порядок.
14         SRAX  1               (См. алгоритм 4.2.1M.)
15         DIV   ARG             Если произошло переполнение, оно будет
                                     обнаружено ниже.
16         STA   ACC             w_m.
17         SLAX  5               Использовать остаток для дальнейшего деления.
18         DIV   ARG
19         STA   ACCX            ±w_l.
20         LDA   ARGX(1:4)
21         ENTX  0
22         DIV   ARG(ABS)        rA ← ⌊|b^4 v_l/v_m|⌋/b^5.
23         JOV   DVZROD          Вызывает ли деление переполнение?
24         MUL   ACC(ABS)        rAX ← |w_m v_l/bv_m| (приблизительно).
25         SRAX  4               Умножить на b и сохранить
26         SLC   5                  ведущий байт в rX.
27         SUB   ACCX(ABS)       Вычесть |w_l|.
28         DECA  1               Принудительно установить знак "минус".
29         SUB   WM1
30         JOV   *+2             Если переполнения нет, сделать еще
31         INCX  1                  один перенос в старшую половину.
32         SLC   5               (Теперь rA ≤ 0.)
33         ADD   ACC(ABS)        rA ← |w_m| − |rA|.
34         STA   ACC(ABS)        (Теперь rA ≥ 0.)
35         LDA   ACC             rA ← w_m с правильным знаком.
36         JMP   DNORM           Нормализовать и выйти из подпрограммы.
37  DVZROD HLT   30              Ненормализованный или нулевой делитель.
```

```
38  1H      EQU  1(1:1)
39  WM1     CON  1B-1,BYTE-1(1:1)  Размер слова минус единица.  I
```

Ниже приведена таблица с приближенными значениями средних времен выполнения для рассмотренных программ вычислений с удвоенной точностью в сравнении с соответствующими характеристиками подпрограмм вычислений с однократной точностью из раздела 4.2.1.

	Однократная точность	Удвоенная точность
Сложение	$45.5u$	$84u$
Вычитание	$49.5u$	$88u$
Умножение	$48u$	$109u$
Деление	$52u$	$126.5u$

Дополнительная информация относительно обобщений методов из этого раздела для вычислений с утроенной точностью в формате с плавающей точкой приводится в работе Y. Ikebe, *CACM* **8** (1965), 175–177.

УПРАЖНЕНИЯ

1. [*16*] Попытайтесь вручную реализовать методику деления с удвоенной точностью числа 180 000 на 314 159, полагая, что $\epsilon = \frac{1}{1000}$. (Положите $(u_m, u_l) = (.180, .000)$ и $(v_m, v_l) = (.314, .159)$ и найдите частное с помощью метода, описанного в тексте после формулы (2).)

2. [*20*] Стоит ли вставлять между строками 30 и 31 программы M команду "`ENTX 0`" с тем, чтобы предотвратить нежелательное влияние на точность результатов информации, оставшейся в регистре X?

3. [*M20*] Объясните, почему при выполнении программы M не может произойти переполнения.

4. [*22*] Как следовало бы изменить программу M, чтобы достичь повышения точности за счет сдвига вертикальной линии, показанной на рис. 4, на одну позицию вправо? Перечислите все необходимые изменения и определите, как изменится при этом время выполнения.

▸ **5.** [*24*] Как следовало бы изменить программу A, чтобы повысить точность за счет перехода к аккумулятору размером 9 байт вместо 8 байт справа от разделяющей точки? Перечислите все необходимые изменения и определите, как изменится при этом время выполнения.

6. [*23*] Предположим, что в одной и той же основной программе используются и подпрограммы с удвоенной точностью из этого раздела, и подпрограммы с однократной точностью из раздела 4.2.1. Разработайте подпрограмму, которая переводит число из формата с однократной точностью в формат с удвоенной точностью соответственно (1). Разработайте также другую подпрограмму, которая переводит число в формате с удвоенной точностью в число в формате с однократной точностью (или сообщает о переполнении или исчезновении порядка, если преобразование невозможно).

▸ **7.** [*M30*] Оцените точность вычислений в подпрограммах этого раздела, работающих с форматом удвоенной точности, подыскав подходящие постоянные δ_1, δ_2 и δ_3, которые могут служить верхними границами для относительных ошибок

$$\left|((u \oplus v) - (u + v))/(u + v)\right|, \qquad \left|((u \otimes v) - (u \times v))/(u \times v)\right|,$$
$$\left|((u \oslash v) - (u/v))/(u/v)\right|.$$

8. [*M28*] Оцените (в смысле упр. 7) точность усовершенствованных подпрограмм вычислений с удвоенной точностью из упр. 4 и 5.

9. [*M42*] В работе T. J. Dekker, *Numer. Math.* **18** (1971), 224–242, предлагается альтернативный подход к вычислениям с удвоенной точностью, полностью базирующийся на методах вычислений с однократной точностью. Например, теорема 4.2.2C утверждает, что $u+v=w+r$, где $w=u\oplus v$ и $r=(u\ominus w)\oplus v$, если $|u|\geq|v|$, а основание системы счисления равно 2. Здесь $|r|\leq|w|/2^p$, так что пару (w,r) можно рассматривать в качестве версии $u+v$ с удвоенной точностью. Для сложения двух таких пар $(u,u')\oplus(v,v')$, где $|u'|\leq|u|/2^p$ и $|v'|\leq|v|/2^p$, $|u|\geq|v|$, Деккер предложил точно вычислить сначала $u+v=w+r$, затем — приближенное значение остатка $s=(r\oplus v')\oplus u'$ и наконец вернуть в вызывающую программу в качестве результата значение $(w\oplus s,(w\ominus(w\oplus s))\oplus s)$.

Проанализируйте точность и эффективность такого подхода в случае, когда он используется рекурсивно для вычислений с учетверенной точностью.

4.2.4. Распределение чисел в формате с плавающей точкой

Чтобы дать усредненную картину поведения алгоритмов, использующих арифметические операции с числами в формате с плавающей точкой (и, в частности, чтобы определить среднее время выполнения таких алгоритмов), потребуется некоторая статистическая информация, позволяющая подсчитать частоту появления различных случаев. В этом разделе будут проанализированы эмпирические и теоретические свойства распределения чисел в формате с плавающей точкой.

A. Программы сложения и вычитания. Время выполнения операций сложения и вычитания чисел с плавающей точкой в значительной мере зависит от исходной разности порядков, а также от числа необходимых шагов нормализации (влево или вправо). До сих пор неизвестна хорошая теоретическая модель, которая предсказывала бы ожидаемые характеристики, но мы располагаем результатами обширных эмпирических исследований, проведенных Д. У. Суини [D. W. Sweeney, *IBM Systems J.* **4** (1965), 31–42].

При помощи специальной программы трассировки Суини проследил выполнение шести "типовых" программ вычислений большой сложности, отобранных в различных вычислительных лабораториях, и тщательно исследовал каждую операцию сложения и вычитания чисел с плавающей точкой. В процесс сбора этих данных было вовлечено более 250 000 таких операций. Примерно одной из каждых десяти команд, выполненных тестируемыми программами, была либо `FADD`, либо `FSUB`.

Поскольку вычитание есть не что иное, как сложение, в котором второй операнд взят с противоположным знаком, все такого рода статистические данные можно рассматривать как относящиеся только к сложению. Результаты, полученные Суини, можно подытожить следующим образом.

Одно из двух слагаемых равнялось нулю примерно в 9% всех случаев, и обычно это был аккумулятор (`ACC`). Остальные 91% случаев распределились примерно поровну между одинаковыми и противоположными знаками операндов, а также примерно поровну между случаями, когда $|u|\leq|v|$ и когда $|v|\leq|u|$. Вычисленный результат равнялся нулю примерно в 1.4% случаев.

Поведение разности между порядками приблизительно описывалось распределениями вероятностей, приведенными в табл. 1, для разных значений основания b.

Таблица 1

ЭМПИРИЧЕСКИЕ ДАННЫЕ
О СООТНОШЕНИЯХ МЕЖДУ ОПЕРАНДАМИ
ПЕРЕД ВЫПОЛНЕНИЕМ СЛОЖЕНИЯ

$\lvert e_u - e_v \rvert$	$b = 2$	$b = 10$	$b = 16$	$b = 64$
0	0.33	0.47	0.47	0.56
1	0.12	0.23	0.26	0.27
2	0.09	0.11	0.10	0.04
3	0.07	0.03	0.02	0.02
4	0.07	0.01	0.01	0.02
5	0.04	0.01	0.02	0.00
Свыше 5	0.28	0.13	0.11	0.09
В среднем	3.1	0.9	0.8	0.5

Таблица 2

ЭМПИРИЧЕСКИЕ ДАННЫЕ
О НОРМАЛИЗАЦИИ ПОСЛЕ ВЫПОЛНЕНИЯ СЛОЖЕНИЯ

	$b = 2$	$b = 10$	$b = 16$	$b = 64$
Сдвиг вправо на 1 разряд	0.20	0.07	0.06	0.03
Никакого сдвига	0.59	0.80	0.82	0.87
Сдвиг влево на 1 разряд	0.07	0.08	0.07	0.06
Сдвиг влево на 2 разряда	0.03	0.02	0.01	0.01
Сдвиг влево на 3 разряда	0.02	0.00	0.01	0.00
Сдвиг влево на 4 разряда	0.02	0.01	0.00	0.01
Сдвиг влево на 5 и более разрядов	0.06	0.02	0.02	0.02

(Строка "Свыше 5" в таблице содержит, по сути, все случаи, когда один операнд был равен нулю, но в строку "В среднем" они не включены.)

Когда u и v имеют одинаковые знаки и нормализованы, при вычислении суммы $u + v$ либо требуется осуществить сдвиг на один разряд *вправо* (в связи с переполнением дробной части), либо вообще не требуется выполнять сдвиги в процессе нормализации. Когда u и v имеют противоположные знаки, при нормализации происходит сдвиг *влево* на нуль или более разрядов. В табл. 2 представлены данные наблюдений за распределением количества сдвигов. В последнюю строку вошли все случаи, когда результат был равен нулю. Среднее число сдвигов влево в процессе нормализации равнялось примерно 0.9 для $b = 2$, около 0.2 — для $b = 10$ или 16 и около 0.1 — для $b = 64$.

B. Дробные части. Дальнейший анализ программ, работающих с числами в формате с плавающей точкой, можно проводить на основе *статистического распределения дробных частей* случайно выбранных нормализованных чисел. Здесь обнаруживаются весьма удивительные факты и имеется интересная теория, объясняющая эти феномены.

Для удобства временно предположим, что мы имеем дело с числами в десятичной системе счисления; последующие рассуждения легко распространяются на любое другое положительное основание b. Предположим, что дано "случайное" положительное нормализованное число $(e, f) = 10^e \cdot f$. Поскольку f нормализовано,

его ведущий (наиболее значимый) разряд равен 1, 2, 3, 4, 5, 6, 7, 8 или 9, и, казалось бы, естественно предположить, что каждая из этих цифр встречается в качестве ведущей приблизительно в $1/9$ всех случаев. Однако на практике картина оказалась совершенно иной. Например, ведущий разряд равен 1 более чем в 30% случаев!

Один из способов проверить это утверждение состоит в том, чтобы обратиться к таблице физических постоянных (наподобие скорости света, гравитационной постоянной и т. п.) из какого-либо справочника. Заглянув, например, в *Handbook of Mathematical Functions* (U. S. Dept of Commerce, 1964), можно обнаружить, что у 8 из 28 различных физических постоянных, приведенных в табл. 2.3, а это примерно 29% случаев, ведущий разряд равен 1. Значения $n!$ в десятичной системе при $1 \le n \le 100$ включают ровно 30 элементов, начинающихся с 1; то же самое относится к десятичным значениям 2^n и F_n при $1 \le n \le 100$. Можно было бы заглянуть также в отчеты о переписи населения или в "Календарь фермера" (но не в телефонный справочник).

В те не такие уж далекие времена, когда не было карманных калькуляторов, в таблицах логарифмов, которые часто использовались, как правило, первые страницы были довольно грязными, потрепанными, а последние оставались сравнительно чистыми и, на первый взгляд, нетронутыми. По-видимому, впервые это явление было отмечено в печати американским астрономом Саймоном Ньюкомбом [Simon Newcomb, *Amer. J. Math.* **4** (1881), 39–40], который привел разумные доводы в пользу того, что цифра d встречается в качестве ведущей с вероятностью $\log_{10}(1+1/d)$. Тот же закон распределения много лет спустя был эмпирически открыт Ф. Бенфордом [F. Benford, *Proc. Amer. Philosophical Soc.* **78** (1938), 551–572], который сообщил о результатах 20 229 наблюдений, взятых из других источников.

Для того чтобы "прочувствовать" этот закон ведущего разряда, давайте внимательно посмотрим, как записываются числа в формате с плавающей точкой. Если взять произвольное положительное число u, то его дробная часть будет определяться по формуле $10f_u = 10^{(\log_{10} u) \bmod 1}$. Следовательно, ведущий разряд в ней будет меньше, чем d, тогда и только тогда, когда

$$(\log_{10} u) \bmod 1 < \log_{10} d. \tag{1}$$

Далее, если имеется какое-либо "случайное" положительное число U, выбранное из совокупности, которая существует в природе, то можно ожидать, что числа $(\log_{10} U) \bmod 1$ будут равномерно распределены между нулем и единицей или по крайней мере их распределение будет достаточно близко к равномерному. (Аналогично мы ожидаем, что так же равномерно распределены величины $U \bmod 1$, $U^2 \bmod 1$, $\sqrt{U+\pi} \bmod 1$ и т. д. Мы уверены, что колесо рулетки беспристрастно, в основном, по этой же причине.) Значит, из неравенства (1) следует, что единица будет ведущей цифрой с вероятностью $\log_{10} 2 \approx 30.103\%$, двойка — с вероятностью $\log_{10} 3 - \log_{10} 2 \approx 17.609\%$, и вообще, если r — произвольное вещественное число, заключенное между 1 и 10, приблизительно в $\log_{10} r$ всех случаев должно выполняться неравенство $10f_U \le r$.

Поскольку ведущие цифры, в основном, небольшие, наиболее распространенная методика оценки "средней ошибки" при вычислениях в формате с плавающей точкой становится неверной. Относительная ошибка вследствие округления обычно оказывается несколько большей, чем ожидается.

Разумеется, можно справедливо утверждать, что приведенные выше эвристические доводы не доказывают сформулированного закона. Они только указывают правдоподобные причины того, что поведение ведущих цифр именно таково, каково оно есть на самом деле. Интересный подход к анализу значений ведущих разрядов был предложен Р. Хэммингом (R. Hamming). Пусть $p(r)$ — вероятность того, что $10f_U \le r$, где $1 \le r \le 10$ и f_U — нормализованная дробная часть случайным образом выбранного нормализованного числа U. Если говорить о случайных величинах в реальном мире, то можно заметить, что они измеряются в произвольных единицах, и если изменить, скажем, определение метра или грамма, то многие из фундаментальных физических постоянных будут иметь другое значение. Предположим поэтому, что все числа во Вселенной внезапно оказались умноженными на некоторый постоянный множитель c; наша "Вселенная" случайных величин с плавающей точкой должна после этого преобразования остаться, по существу, неизменной, так что вероятности $p(r)$ не должны измениться.

Умножение всех чисел на c имеет тот эффект, что $(\log_{10} U) \bmod 1$ превращается в $(\log_{10} U + \log_{10} c) \bmod 1$. Выведем формулы, описывающие искомое распределение. Можно считать, что $1 \le c \le 10$. По определению

$$p(r) = \Pr\big((\log_{10} U) \bmod 1 \le \log_{10} r\big).$$

Согласно сделанному ранее предположению имеем также

$$\begin{aligned} p(r) &= \Pr\big((\log_{10} U + \log_{10} c) \bmod 1 \le \log_{10} r\big) \\ &= \begin{cases} \Pr\big((\log_{10} U \bmod 1) \le \log_{10} r - \log_{10} c \\ \qquad\qquad \text{или } (\log_{10} U \bmod 1) \ge 1 - \log_{10} c\big), & \text{если } c \le r; \\ \Pr\big((\log_{10} U \bmod 1) \le \log_{10} r + 1 - \log_{10} c \\ \qquad\qquad \text{и } (\log_{10} U \bmod 1) \ge 1 - \log_{10} c\big), & \text{если } c \ge r; \end{cases} \\ &= \begin{cases} p(r/c) + 1 - p(10/c), & \text{если } c \le r; \\ p(10r/c) - p(10/c), & \text{если } c \ge r. \end{cases} \end{aligned} \tag{2}$$

Продолжим теперь функцию $p(r)$ вне интервала $1 \le r \le 10$, положив $p(10^n r) = p(r) + n$. Тогда, заменив $10/c$ на d, можно записать последнее соотношение в (2) в виде

$$p(rd) = p(r) + p(d). \tag{3}$$

Если верно предположение об инвариантности распределения относительно умножения на произвольный постоянный множитель, то соотношение (3) должно выполняться для всех $r > 0$ и $1 \le d \le 10$. Из того факта, что $p(1) = 0$ и $p(10) = 1$, теперь следует:

$$1 = p(10) = p\big((\sqrt[n]{10})^n\big) = p(\sqrt[n]{10}) + p\big((\sqrt[n]{10})^{n-1}\big) = \cdots = np(\sqrt[n]{10});$$

отсюда приходим к заключению, что для всех положительных целых m и n справедливо равенство $p(10^{m/n}) = m/n$. Если дополнительно потребовать, чтобы распределение p было непрерывным, то можно прийти к заключению, что $p(r) = \log_{10} r$, а это и есть искомый закон.

Хотя данный аргумент, возможно, и убедительнее предыдущих, он тоже в действительности не выдерживает придирчивой проверки, если строго придерживаться

общепринятого определения вероятности. Традиционный способ точной формулировки подобных аргументов подразумевает, что существует некое лежащее в основе рассматриваемого явления распределение чисел $F(u)$, такое, что вероятность того, что данное произвольное число U не превосходит u, равна $F(u)$; тогда интересующая нас вероятность равна

$$p(r) = \sum_m \big(F(10^m r) - F(10^m)\big), \tag{4}$$

где суммирование проводится по всем значениям $-\infty < m < \infty$. Наше предположение об инвариантности по отношению к масштабированию и о непрерывности приводит к заключению, что

$$p(r) = \log_{10} r.$$

Используя те же доводы, можно "доказать", что

$$\sum_m \big(F(b^m r) - F(b^m)\big) = \log_b r \tag{5}$$

при $1 \le r \le b$ для любого целого числа $b \ge 2$. Но *не существует* функции распределения F, которая удовлетворяла бы этому равенству для всех таких b и r (см. упр. 7).

Один из способов выхода из этой затруднительной ситуации состоит в рассмотрении логарифмического закона $p(r) = \log_{10} r$ лишь как очень хорошего *приближения* к истинному распределению. Возможно, истинное распределение изменяется при расширении Вселенной, делая с течением времени наше приближение все лучшим и лучшим; и, если заменить основание 10 произвольным основанием b, наше приближение будет тем менее точным (в любой данный момент времени), чем больше b. Другой довольно привлекательный способ решения дилеммы, связанный с отказом от традиционного понятия функции распределения, предложен Р. А. Рэйми [R. A. Raimi, *AMM* **76** (1969), 342–348].

Витиеватые рассуждения последнего абзаца, по-видимому, ни в коей мере нельзя признать удовлетворительным объяснением, так что следует весьма положительно отнестись к проводимым ниже вычислениям (где мы придерживаемся строгого математического канона и избегаем любых интуитивно понятных (и вдобавок к тому еще и парадоксальных) понятий теории вероятностей). Рассмотрим вместо распределения некоего воображаемого множества вещественных чисел распределение значений ведущих (старших значащих) разрядов *положительных целых* чисел. Исследование этого вопроса чрезвычайно интересно, и не только потому, что оно проливает некоторый свет на распределения вероятностей для данных, представленных в формате с плавающей точкой, но и потому, что оно служит весьма поучительным примером сочетания методов дискретной математики с методами исчисления бесконечно малых.

Во всех последующих рассуждениях r будет обозначать фиксированное вещественное число, $1 \le r \le 10$; мы попытаемся дать разумное определение $p(r)$ как "вероятности" того, что представление $10^{e_N} \cdot f_N$ "случайного" положительного целого числа N удовлетворяет неравенству $10 f_N < r$ в предположении о бесконечной точности представления.

Для начала попытаемся найти эту вероятность, используя методику предельного перехода, аналогично тому, как мы определяли "Pr" в разделе 3.5. Вот удобный способ перефразирования этого определения:

$$P_0(n) = \big[n = 10^e \cdot f, \text{ где } 10f < r\big] = \big[(\log_{10} n) \bmod 1 < \log_{10} r\big]. \tag{6}$$

Итак, последовательность $P_0(1), P_0(2), \ldots$ есть бесконечная последовательность нулей и единиц, причем единицы соответствуют случаям, вносящим вклад в значение вероятности, которую мы ищем. Можно попытаться "усреднить" эту последовательность, положив

$$P_1(n) = \frac{1}{n}\sum_{k=1}^{n} P_0(k). \tag{7}$$

Таким образом, если сформировать случайное целое число в интервале между 1 и n, используя методику главы 3, и преобразовать его в десятичный формат с плавающей точкой (e, f), то вероятность того, что $10f < r$, и будет в точности равна $P_1(n)$. Естественно принять $\lim_{n\to\infty} P_1(n)$ в качестве искомой "вероятности" $p(r)$. Именно так и было сделано в определении 3.5A.

Но в данном случае этот предел не существует. Рассмотрим, например, подпоследовательность

$$P_1(s),\ P_1(10s),\ P_1(100s),\ \ldots,\ P_1(10^n s),\ \ldots,$$

где s — некоторое вещественное число, $1 \le s \le 10$. Если $s \le r$, то имеем

$$\begin{aligned} P_1(10^n s) &= \frac{1}{10^n s}\big(\lceil r\rceil - 1 + \lceil 10r\rceil - 10 + \cdots + \lceil 10^{n-1}r\rceil - 10^{n-1} + \lfloor 10^n s\rfloor + 1 - 10^n\big) \\ &= \frac{1}{10^n s}\big(r(1+10+\cdots+10^{n-1}) + O(n) + \lfloor 10^n s\rfloor - 1 - 10 - \cdots - 10^n\big) \\ &= \frac{1}{10^n s}\big(\tfrac{1}{9}(10^n r - 10^{n+1}) + \lfloor 10^n s\rfloor + O(n)\big). \end{aligned} \tag{8}$$

При $n \to \infty$ функция $P_1(10^n s)$ стремится, следовательно, к предельному значению $1 + (r-10)/9s$. Те же вычисления справедливы и для $s > r$, если заменить $\lfloor 10^n s\rfloor + 1$ на $\lceil 10^n r\rceil$. Тогда получим предельное значение $10(r-1)/9s$ при $s \ge r$. [См. J. Franel, *Naturforschende Gesellschaft, Vierteljahrsschrift* **62** (Zürich, 1917), 286–295.]

Другими словами, последовательность $\langle P_1(n)\rangle$ содержит подпоследовательности $\langle P_1(10^n s)\rangle$, предел которых возрастает от $(r-1)/9$ до $10(r-1)/9r$, а затем убывает до $(r-1)/9$ по мере того, как s возрастает от 1 до r, а затем от r до 10. Отсюда видно, что последовательность $P_1(n)$ не имеет предела при $n \to \infty$ и что значения $P_1(n)$ при больших n нельзя считать слишком хорошим приближением к пределу $\log_{10} r$, на который мы рассчитывали!

Так как $P_1(n)$ не стремится к некоторому пределу, можно попытаться еще раз использовать ту же идею, что и в (7), для "усреднения" этого аномального поведения нашей последовательности. Вообще, положим

$$P_{m+1}(n) = \frac{1}{n}\sum_{k=1}^{n} P_m(k). \tag{9}$$

Тогда $P_{m+1}(n)$ будет иметь тенденцию к более правильному поведению, нежели $P_m(n)$. Попытаемся подтвердить это количественными вычислениями. Опыт, приобретенный при рассмотрении частного случая, когда $m = 0$, подсказывает, что имеет смысл рассмотреть подпоследовательность $P_{m+1}(10^n s)$. Таким способом можно доказать следующий результат.

Лемма Q. *Для произвольного целого числа $m \ge 1$ и произвольного вещественного числа $\epsilon > 0$ найдутся такие функции $Q_m(s)$, $R_m(s)$ и такое целое число $N_m(\epsilon)$, что при $n > N_m(\epsilon)$ и $1 \le s \le 10$ выполняются неравенства*

$$|P_m(10^n s) - Q_m(s) - R_m(s)[s > r]| < \epsilon. \tag{10}$$

Далее, функции $Q_m(s)$ и $R_m(s)$ удовлетворяют соотношениям

$$\begin{aligned} Q_m(s) &= \frac{1}{s}\Big(\frac{1}{9}\int_1^{10} Q_{m-1}(t)\,dt + \int_1^{s} Q_{m-1}(t)\,dt + \frac{1}{9}\int_r^{10} R_{m-1}(t)\,dt\Big); \\ R_m(s) &= \frac{1}{s}\int_r^{s} R_{m-1}(t)\,dt; \\ Q_0(s) &= 1, \qquad R_0(s) = -1. \end{aligned} \tag{11}$$

Доказательство. Рассмотрим функции $Q_m(s)$ и $R_m(s)$, определенные формулами (11), и положим

$$S_m(t) = Q_m(t) + R_m(t)[t > r]. \tag{12}$$

Докажем лемму индукцией по m.

Сначала отметим, что $Q_1(s) = \big(1 + (s-1) - (10-r)/9\big)/s = 1 + (r-10)/9s$ и $R_1(s) = (r-s)/s$. Из (8) найдем, что $|P_1(10^n s) - S_1(s)| = O(n)/10^n$; это доказывает лемму при $m = 1$.

При $m > 1$ имеем

$$P_m(10^n s) = \frac{1}{s}\Bigg(\sum_{0\le j<n} \frac{1}{10^{n-j}} \sum_{10^j \le k < 10^{j+1}} \frac{1}{10^j} P_{m-1}(k) + \sum_{10^n \le k \le 10^n s} \frac{1}{10^n} P_{m-1}(k)\Bigg).$$

Необходимо оценить эту величину. Разность

$$\left|\sum_{10^j \le k \le 10^j q} \frac{1}{10^j} P_{m-1}(k) - \sum_{10^j \le k \le 10^j q} \frac{1}{10^j} S_{m-1}\Big(\frac{k}{10^j}\Big)\right| \tag{13}$$

по индукции меньше $q\epsilon$, когда $1 \le q \le 10$ и $j > N_{m-1}(\epsilon)$. А поскольку функция $S_{m-1}(t)$ непрерывна и потому интегрируема по Риману, разность

$$\left|\sum_{10^j \le k \le 10^j q} \frac{1}{10^j} S_{m-1}\Big(\frac{k}{10^j}\Big) - \int_1^q S_{m-1}(t)\,dt\right| \tag{14}$$

меньше ϵ для всех j, бóльших некоторого числа N, которое не зависит от q, как следует из определения интегрируемости. N можно выбрать бóльшим, чем $N_{m-1}(\epsilon)$.

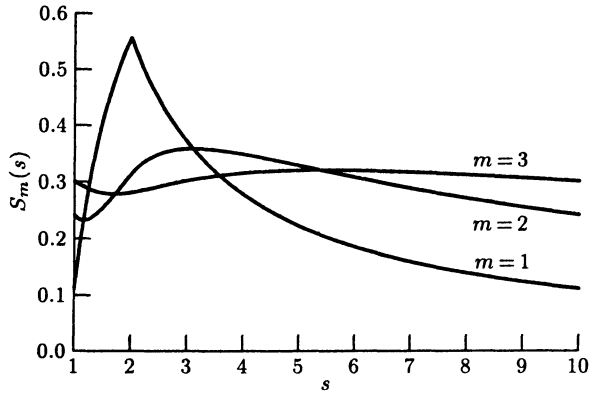


Рис. 5. Вероятность того, что старший значащий разряд равен 1.

Следовательно, при $n > N$ разность

$$\left| P_m(10^n s) - \frac{1}{s}\left(\sum_{0\le j<n} \frac{1}{10^{n-j}} \int_1^{10} S_{m-1}(t)\,dt + \int_1^s S_{m-1}(t)\,dt \right) \right| \tag{15}$$

ограничена величиной $\sum_{j=0}^{N}(M/10^{n-j}) + \sum_{N<j<n}(11\epsilon/10^{n-j}) + 11\epsilon$, если M служит верхней границей для суммы (13) + (14), которая имеет смысл при всех положительных целых j. Наконец, сумма $\sum_{0\le j<n}(1/10^{n-j})$, фигурирующая в (15), равна $(1 - 1/10^n)/9$ Поэтому разность

$$\left| P_m(10^n s) - \frac{1}{s}\left(\frac{1}{9}\int_1^{10} S_{m-1}(t)\,dt + \int_1^s S_{m-1}(t)\,dt \right) \right|$$

может быть сделана меньше, скажем, 20ϵ при достаточно больших n. Сопоставляя этот вывод с (10) и (11), видим, что доказательство завершено. ∎

Основной смысл леммы Q — утверждение о существовании предела

$$\lim_{n\to\infty} P_m(10^n s) = S_m(s). \tag{16}$$

Далее из леммы следует, что предел

$$\lim_{n\to\infty} P_m(n),$$

который мог бы быть нашей желанной "вероятностью", не существует ни для какого m, поскольку функция $S_m(s)$ не остается постоянной при изменении s. Представление о создавшейся ситуации дает рис. 5, на котором изображены значения $S_m(s)$ для малых m и $r = 2$.

Хотя функции $S_m(s)$ и не постоянны, так что у $P_m(n)$ не существует предела, из рис. 5 видно, что уже для $m = 3$ значение $S_m(s)$ всегда остается очень близким к $\log_{10} 2 \approx 0.30103$. Следовательно, есть серьезные основания полагать, что функция $S_m(s)$ очень близка к $\log_{10} r$ для всех больших m и что последовательность функций $\langle S_m(s)\rangle$ равномерно сходится к постоянной функции $\log_{10} r$.

Интересно доказать это предположение, вычислив в явном виде $Q_m(s)$ и $R_m(s)$ для всех m, что и делается в доказательстве следующей теоремы.

Теорема F. *Пусть $S_m(s)$ — предел, определенный в* (16). *Для всякого $\epsilon > 0$ найдется такое число $N(\epsilon)$, что*

$$|S_m(s) - \log_{10} r| < \epsilon \qquad \text{для } 1 \le s \le 10, \tag{17}$$

где $m > N(\epsilon)$.

Доказательство. На основании леммы Q этот результат может быть доказан, если показать, что существует такое число M, зависящее от ϵ, что для всех $1 \le s \le 10$ и всех $m > M$ справедливы неравенства

$$|Q_m(s) - \log_{10} r| < \epsilon \qquad \text{и} \qquad |R_m(s)| < \epsilon. \tag{18}$$

Нетрудно найти решение рекуррентного уравнения (11) для R_m. В самом деле, имеем $R_0(s) = -1$, $R_1(s) = -1 + r/s$, $R_2(s) = -1 + (r/s)\big(1 + \ln(s/r)\big)$ и в общем случае

$$R_m(s) = -1 + \frac{r}{s}\left(1 + \frac{1}{1!}\ln\frac{s}{r} + \cdots + \frac{1}{(m-1)!}\Big(\ln\frac{s}{r}\Big)^{m-1}\right). \tag{19}$$

Для значений s из указанного интервала эта функция равномерно сходится к

$$-1 + (r/s)\exp\big(\ln(s/r)\big) = 0.$$

Для Q_m рекуррентная формула (11) принимает вид

$$Q_m(s) = \frac{1}{s}\left(c_m + 1 + \int_1^s Q_{m-1}(t)\,dt\right), \tag{20}$$

где

$$c_m = \frac{1}{9}\left(\int_1^{10} Q_{m-1}(t)\,dt + \int_r^{10} R_{m-1}(t)\,dt\right) - 1. \tag{21}$$

Решение рекуррентного уравнения (20) также можно найти без труда; необходимо выписать выражения для нескольких первых членов, сообразить, какова общая формула, и доказать ее по индукции. Получим, что

$$Q_m(s) = 1 + \frac{1}{s}\left(c_m + \frac{1}{1!}c_{m-1}\ln s + \cdots + \frac{1}{(m-1)!}c_1(\ln s)^{m-1}\right). \tag{22}$$

Остается только вычислить коэффициенты c_m, которые в соответствии с формулами (19), (21) и (22) удовлетворяют соотношениям

$$\begin{aligned} c_1 &= (r-10)/9; \\ c_{m+1} &= \frac{1}{9}\left(c_m \ln 10 + \frac{1}{2!}c_{m-1}(\ln 10)^2 + \cdots + \frac{1}{m!}c_1(\ln 10)^m \right. \\ &\qquad \left. + r\left(1 + \frac{1}{1!}\ln\frac{10}{r} + \cdots + \frac{1}{m!}\Big(\ln\frac{10}{r}\Big)^m\right) - 10\right). \end{aligned} \tag{23}$$

Эта последовательность кажется очень сложной, однако в действительности ее можно легко исследовать при помощи производящих функций. Положим

$$C(z) = c_1 z + c_2 z^2 + c_3 z^3 + \cdots;$$

тогда в соответствии с равенством $10^z = 1 + z\ln 10 + (1/2!)(z\ln 10)^2 + \cdots$ приходим к заключению, что

$$\begin{aligned} c_{m+1} &= \frac{1}{10}c_{m+1} + \frac{9}{10}c_{m+1} \\ &= \frac{1}{10}\Big(c_{m+1} + c_m\ln 10 + \cdots + \frac{1}{m!}c_1(\ln 10)^m\Big) + \frac{r}{10}\Big(1 + \cdots + \frac{1}{m!}\Big(\ln\frac{10}{r}\Big)^m\Big) - 1 \end{aligned}$$

есть коэффициент при z^{m+1} в разложении функции

$$\frac{1}{10}C(z)10^z + \frac{r}{10}\Big(\frac{10}{r}\Big)^z\Big(\frac{z}{1-z}\Big) - \frac{z}{1-z}. \tag{24}$$

Это условие выполняется для всех значений m, так что значение выражения (24) должно равняться $C(z)$. Получаем в явном виде формулу

$$C(z) = \frac{-z}{1-z}\left(\frac{(10/r)^{z-1} - 1}{10^{z-1} - 1}\right). \tag{25}$$

Чтобы завершить анализ, необходимо проанализировать асимптотические свойства коэффициентов $C(z)$. Множитель в скобках в выражении (25) при $z \to 1$ стремится к $\ln(10/r)/\ln 10 = 1 - \log_{10} r$, откуда следует, что

$$C(z) + \frac{1 - \log_{10} r}{1 - z} = R(z) \tag{26}$$

есть аналитическая функция комплексной переменной z в круговой области

$$|z| < \left|1 + \frac{2\pi\imath}{\ln 10}\right|.$$

В частности, разложение функции $R(z)$ сходится при $z = 1$, так что ее коэффициенты стремятся к нулю. Это доказывает, что коэффициенты функции $C(z)$ ведут себя, как коэффициенты функции $(\log_{10} r - 1)/(1 - z)$, так что

$$\lim_{m\to\infty} c_m = \log_{10} r - 1.$$

Наконец, сопоставляя этот результат с выражением (22), получаем, что на отрезке $1 \le s \le 10$ функция $Q_m(s)$ равномерно стремится к

$$1 + \frac{\log_{10} r - 1}{s}\Big(1 + \ln s + \frac{1}{2!}(\ln s)^2 + \cdots\Big) = \log_{10} r. \quad \blacksquare$$

Итак, посредством прямого вычисления доказан логарифмический закон для целых чисел, причем одновременно обнаружено, что, хотя этот закон и служит очень хорошим приближением для описания усредненного поведения, он никогда не выполняется в точности.

Приведенные выше доказательства леммы Q и теоремы F — это несколько упрощенные и усиленные методы, описанные в работе Б. Дж. Флехингер (B. J. Flehinger, *AMM* **73** (1966), 1056–1061). Многие авторы исследовали распределение значений в ведущих разрядах, показав, что логарифмический закон является хорошим приближением к таким функциям распределения. Более полный обзор имеющейся по этому вопросу литературы читатель найдет в работах Ральфа А. Рэйми (Ralph A. Raimi,

AMM **83** (1976), 521–538) и Петера Шатта (Peter Schatte, *J. Information Processing and Cybernetics* **24** (1988), 443–455).

В упр. 17 рассматривается подход к определению распределения вероятностей, при котором для целых чисел логарифмический закон выдерживается точно. Далее в упр. 18 демонстрируется, что *любое* разумное определение распределения вероятностей на целых числах должно приводить к логарифмическому закону, если оно (определение) применимо к вероятности появления значения ведущего разряда.

Конечно, при работе в формате с плавающей точкой используются, в основном, нецелые числа, а мы анализировали целые числа, в первую очередь, потому, что этот предмет более знаком и анализ выполняется существенно проще. Если рассматривать произвольные действительные числа, то теоретические результаты получить сложнее. Однако постепенно накапливаются доказательства, что имеет место та же статистика в том смысле, что повторяющиеся вычисления с действительными числами будут почти всегда иметь тенденцию ко все большему приближению к логарифмическому закону по отношению к дробным частям. Например, Петер Шатт показал (Peter Schatte, *Zeitschrift für angewandte Math. und Mechanik* **53** (1973), 553–565), что при весьма слабом ограничении функция распределения произведения независимых случайных действительных переменных, имеющих один закон распределения, стремится к логарифмическому закону. Сумма таких переменных ведет себя так же, но только для повторяющегося усреднения. Аналогичные результаты были получены в работе J. L. Barlow, E. H. Bareiss, *Computing* **34** (1985), 325–347.

УПРАЖНЕНИЯ

1. [*13*] Если u и v — десятичные числа в формате с плавающей точкой, *имеющие один и тот же знак*, то каково согласно табл. 1 и 2 приближенное значение вероятности того, что при вычислении значения $u \oplus v$ произойдет переполнение дробной части?

2. [*42*] Проведите дальнейшие эксперименты со сложением и вычитанием чисел в формате с плавающей точкой для подтверждения и уточнения данных, приведенных в табл. 1 и 2.

3. [*15*] Найдите, исходя из логарифмического закона, вероятность того, что две начальные цифры десятичного числа в формате с плавающей точкой будут "23".

4. [*M18*] В тексте раздела отмечено, что начальные страницы интенсивно используемых таблиц логарифмов потрепаны в большей степени, нежели последние. Какие страницы были бы самыми потрепанными, если бы мы работали с таблицей *антилогарифмов*, т. е. с таблицей, которая для данного значения $\log_{10} x$ указывает значение x?

▸ **5.** [*M20*] Предположим, что вещественное число U равномерно распределено в интервале $0 < U < 1$. Каково распределение значений наиболее значимого разряда U?

6. [*23*] Если бы одно двоичное слово содержало $n+1$ бит, то можно было бы использовать p бит для представления дробной части двоичных чисел с плавающей точкой, один бит — для знака и $n-p$ бит — для порядка. Это означает, что интервал изменения представимых значений, т. е. отношение наибольшего положительного нормализованного значения к наименьшему, равен, по существу, $2^{2^{n-p}}$. То же машинное слово можно было бы использовать и для представления *шестнадцатеричных* чисел в формате с плавающей точкой, выделив $p+2$ бит для дробной части $((p+2)/4$ шестнадцатеричных цифр) и $n-p-2$ бит для порядка. Тогда интервал изменения значений был бы $16^{2^{n-p-2}} = 2^{2^{n-p}}$, т. е. тем же, что и раньше, причем с бóльшим числом битов в дробной части. Может сложиться впечатление, что получено нечто из ничего, однако условие нормализации в случае основания 16 слабее

в том смысле, что дробная часть может содержать нули в трех старших значимых битах. Таким образом, не все $p+2$ бит "значащие".

Исходя из логарифмического закона, выясните, какова вероятность того, что дробная часть положительного нормализованного шестнадцатеричного числа в формате с плавающей точкой имеет в точности 0, 1, 2 и 3 нулевых наиболее значимых битов? Основываясь на материале, изложенном в этом разделе, рассмотрите вопрос о достоинствах шестнадцатеричной системы в сравнении с двоичной.

7. [*HM28*] Докажите, что не существует функции распределения $F(u)$, удовлетворяющей соотношению (5) для каждого целого числа $b \geq 2$ и для всех вещественных значений r из интервала $1 \leq r \leq b$.

8. [*HM23*] Выполняется ли соотношение (10) при $m = 0$ для соответствующим образом выбранного $N_0(\epsilon)$?

9. [*HM25*] (П. Дьяконис (P. Diaconis).) Пусть $P_1(n)$, $P_2(n)$, ... — некоторая последовательность функций, определенных повторяющимся усреднением функции $P_0(n)$ в соответствии с формулой (9). Докажите, что $\lim_{m\to\infty} P_m(n) = P_0(1)$ для всех фиксированных n.

▶ **10.** [*HM28*] В тексте раздела показано, что $c_m = \log_{10} r - 1 + \epsilon_m$, где ϵ_m стремится к нулю при $m \to \infty$. Найдите следующий член в асимптотическом разложении для c_m.

11. [*M15*] Докажите, что если U — случайная величина, распределенная по логарифмическому закону, то этим же свойством будет обладать и величина $1/U$.

12. [*HM25*] (Р. У. Хэмминг (R. W. Hamming).) Цель этого упражнения — показать, что результат умножения в формате с плавающей точкой соответствует логарифмическому закону лучше, чем сомножители. Пусть U и V — случайные нормализованные положительные числа в формате с плавающей точкой, имеющие независимое распределение с функциями плотности вероятности $f(x)$ и $g(x)$ соответственно. Тогда $f_u \leq r$ и $f_v \leq s$ с вероятностью $\int_{1/b}^{r}\int_{1/b}^{s} f(x)g(y)\,dy\,dx$, где $1/b \leq r, s \leq 1$. Пусть $h(x)$ — функция плотности вероятности дробной части $U \times V$ (неокругленной). Определим *меру отклонения* $A(f)$ функции плотности вероятности f как максимальную относительную ошибку

$$A(f) = \max_{1/b\leq x\leq 1} \left| \frac{f(x) - l(x)}{l(x)} \right|,$$

где $l(x) = 1/(x \ln b)$ есть плотность логарифмической функции распределения.

Докажите, что $A(h) \leq \min(A(f), A(g))$. (В частности, если некоторый множитель имеет логарифмическое распределение, то и произведение будет иметь такое распределение.)

▶ **13.** [*M20*] В алгоритме умножения чисел с плавающей точкой (алгоритм 4.2.1М) число сдвигов влево, требуемых при нормализации, равно нулю или единице в зависимости от того, будет ли $f_u f_v \geq 1/b$. Предполагая, что операнды распределены независимо по логарифмическому закону, найдите вероятность того, что для нормализации результата не потребуется ни одного сдвига влево.

▶ **14.** [*HM30*] Пусть U и V — случайные нормализованные положительные числа в формате с плавающей точкой, имеющие дробные части с независимым распределением по логарифмическому закону, и пусть p_k — вероятность того, что разность их порядков равна k. Предполагая, что распределение порядков не зависит от распределения дробных частей, выведите формулу, которая через основание b и величины $p_0, p_1, p_2, \ldots$ выражает вероятность того, что при выполнении сложения $U \oplus V$ происходит "переполнение дробной части". Сравните результат с результатом упр. 1 (округление игнорировать).

15. [*HM28*] Пусть $U, V, p_0, p_1, \ldots$ те же, что и в упр. 14, и пусть используется арифметика по основанию 10. Покажите, что независимо от значений $p_0, p_1, p_2, \ldots$ сумма $U \oplus V$ *не будет*

точно подчиняться логарифмическому закону и в действительности вероятность того, что старшая цифра суммы $U \oplus V$ равна 1, всегда строго *меньше* $\log_{10} 2$.

16. [*HM28*] (П. Дьяконис.) Пусть $P_0(n)$ равно 0 или 1 для каждого n. Определите "вероятности" $P_{m+1}(n)$ с помощью повторяющегося усреднения, как в формуле (9). Покажите, что если $\lim_{n\to\infty} P_1(n)$ не существует, то не существует и $\lim_{n\to\infty} P_m(n)$ для любых m. [*Указание.* Докажите, что $a_n \to 0$, если только имеем $(a_1+\cdots+a_n)/n \to 0$ и $a_{n+1} \le a_n + M/n$ для некоторой фиксированной константы $M > 0$.]

▶ **17.** [*HM25*] (М. Цуджи (M. Tsuji).) Другой способ определения значения вероятности $\Pr(S(n))$ состоит в вычислении величины $\lim_{n\to\infty}(H_n^{-1}\sum_{k=1}^{n}[S(k)]/k)$; можно показать, что эта *гармоническая вероятность* существует и равна $\Pr(S(n))$, если только последняя существует в соответствии с определением 3.5А. Докажите, что гармоническая вероятность выражения "$(\log_{10} n) \bmod 1 < r$" существует и равна r. (Таким образом, значения начальных разрядов целых чисел *в точности* удовлетворяют логарифмическому закону в этом смысле.)

▶ **18.** [*HM30*] Пусть $P(S)$ есть некоторая функция с действительными значениями, определенная на множествах S положительных целых чисел, но необязательно на всех таких множествах, и удовлетворяющая следующим довольно слабым аксиомам.

i) Если $P(S)$ и $P(T)$ определены и $S \cap T = \emptyset$, то $P(S \cup T) = P(S) + P(T)$.
ii) Если $P(S)$ определена, то $P(S+1) = P(S)$, где $S+1 = \{n+1 \mid n \in S\}$.
iii) Если $P(S)$ определена, то $P(2S) = \frac{1}{2}P(S)$, где $2S = \{2n \mid n \in S\}$.
iv) Если S есть множество *всех* положительных целых чисел, то $P(S) = 1$.
v) Если $P(S)$ определена, то $P(S) \ge 0$.

Предположим далее, что $P(L_a)$ определено для всех положительных целых чисел a, где L_a есть множество всех положительных целых чисел, для которых десятичное представление начинается с a:

$$L_a = \{n \mid 10^m a \le n < 10^m(a+1) \text{ для некоторого целого } m\}\,.$$

(В этом определении m может быть отрицательным, например 1 есть элемент из L_{10}, но не из L_{11}.) Докажите, что $P(L_a) = \log_{10}(1 + 1/a)$ для всех целых чисел $a \ge 1$.

19. [*HM25*] (Р. Л. Данкэн (R. L. Duncan).) Докажите, что значения ведущих разрядов в числах Фибоначчи подчиняются логарифмическому закону для дробных частей: $\Pr(10f_{F_n} < r) = \log_{10} r$.

20. [*HM40*] Сформулируйте более строго выражение (16), найдя асимптотическое поведение зависимости $P_m(10^n s) - S_m(s)$ при $n \to \infty$.

4.3. АРИФМЕТИКА МНОГОКРАТНОЙ ТОЧНОСТИ

Рассмотрим теперь операции над числами произвольно высокой точности. Для простоты изложения будем считать, что имеются в виду целые числа, а не числа, разделяющая точка которых находится внутри числа.

4.3.1. Классические алгоритмы

В этом разделе будут рассмотрены алгоритмы реализации следующих операций:

a) сложение и вычитание n-разрядных целых чисел с получением n-разрядного результата и разряда переноса;
b) умножение m-разрядного целого числа на n-разрядное целое число с получением $(n+m)$-разрядного результата;
c) деление $(n+m)$-разрядного целого числа на n-разрядное целое число с получением $(m+1)$-разрядного частного и n-разрядного остатка.

Эти алгоритмы можно назвать *классическими*, так как само слово "алгоритм" на протяжении столетий использовалось в связи с реализацией вычислительных процессов. Термин "n-разрядное целое число" означает любое неотрицательное целое число, меньшее b^n, где b есть основание обычной позиционной системы счисления, в которой представляются числа; такие числа в этой системе записываются с использованием не более чем n "разрядов".

Классические алгоритмы для целых чисел можно очевидным образом распространять на числа с разделяющей точкой внутри числа или числа в формате с плавающей точкой, заданные с повышенной точностью (так же, как в машине MIX арифметические операции, определенные для целых чисел, распространяются на эти более общие случаи).

В настоящем разделе будут рассмотрены алгоритмы выполнения перечисленных операций (a)–(c) над целыми числами, представляемыми в позиционной системе по основанию b, где b — заданное целое число, равное или большее 2. Таким образом, эти алгоритмы представляют собой достаточно общие определения арифметических процессов, и в этом качестве они не связаны ни с какой конкретной вычислительной машиной. Тем не менее рассуждения будут некоторым образом машинно-ориентированными, поскольку нас, в основном, интересуют эффективные методы выполнения при помощи компьютера вычислений с высокой точностью. Хотя приведенные примеры ориентированы на гипотетический компьютер MIX, по существу, те же рассуждения применимы почти для любой другой машины.

Для понимания сути чисел повышенной точности наиболее существенно то, что их можно рассматривать как числа, записанные в системе счисления по основанию w, где w — размер слова. Например, целое число, заполняющее 10 машинных слов в памяти компьютера, размер слова которой — $w = 10^{10}$, имеет 100 десятичных разрядов. Однако мы будем рассматривать его как 10-разрядное число по основанию 10^{10}. Такой подход обосновывается теми же соображениями, что и переход, скажем, от двоичной системы счисления к шестнадцатеричной; мы просто группируем биты (см. выражение 4.1–(5)).

С учетом этих соглашений будем рассматривать следующие элементарные операции:

a_0) сложение и вычитание одноразрядных целых чисел с получением одноразрядного результата и разряда переноса;

b_0) умножение одноразрядного целого числа на одноразрядное с получением двухразрядного результата;

c_0) деление двухразрядного целого числа на одноразрядное, которое обеспечивает получение частного как одноразрядного целого числа и одноразрядного остатка.

После надлежащей установки размера слова, если это необходимо, названные операции будут доступны для выполнения почти на каждом компьютере. Поэтому сформулируем перечисленные выше алгоритмы (a), (b) и (c) в терминах элементарных операций (a_0), (b_0) и (c_0).

В связи с тем, что целые числа повышенной точности воспринимаются как числа по основанию b, полезно представить себе соответствующую ситуацию при $b = 10$, считая при этом, что арифметические операции выполняются вручную. Тогда операция (a_0) аналогична запоминанию таблицы сложения, операция (b_0) аналогична запоминанию таблицы умножения, а операция (c_0) — это, по существу, запоминание "обращенной" таблицы умножения. Более сложные операции (a), (b) и (c) над числами высокой точности могут быть теперь реализованы на основе простых операций сложения, вычитания, умножения и деления в столбик, которым учат детей в начальной школе. Фактически большинство алгоритмов, которые будут рассматриваться ниже, — не что иное, как механическое воспроизведение знакомых операций, выполняемых при помощи карандаша и бумаги. Конечно, алгоритмы придется формулировать гораздо более тщательно, чем в начальной школе. Кроме того, необходимо стремиться минимизировать используемую машинную память и время, затрачиваемое на выполнение программ.

Во избежание скучных рассуждений и громоздких обозначений будем с самого начала полагать, что все используемые числа здесь *неотрицательны*. Дополнительные меры, необходимые для вычисления знаков и т. д., довольно очевидны, хотя при работе с числами в виде дополнения на компьютере, на котором не используется представление больших чисел в прямом коде, необходимо соблюдать осторожность. Эти вопросы будут рассмотрены в конце раздела.

Начнем со сложения, с, безусловно, очень простой, но все же заслуживающей внимательного изучения операции, так как те же идеи встречаются и в других алгоритмах.

Алгоритм A (*Сложение неотрицательных целых чисел*). По заданным неотрицательным n-разрядным целым числам по основанию b $(u_{n-1}\ldots u_1u_0)_b$ и $(v_{n-1}\ldots v_1v_0)_b$ этот алгоритм формирует их сумму $(w_nw_{n-1}\ldots w_1w_0)_b$. Здесь w_n — разряд переноса, всегда равный 0 или 1.

A1. [Начальная установка.] Присвоить $j \leftarrow 0$, $k \leftarrow 0$. (Переменная j будет пробегать позиции различных разрядов, а переменная k будет следить за переносами на каждом шаге.)

A2. [Сложить разряды.] Присвоить $w_j \leftarrow (u_j+v_j+k) \bmod b$ и $k \leftarrow \lfloor (u_j+v_j+k)/b \rfloor$. (По индукции, распространяемой на вычисления, всегда выполняется неравенство

$$u_j + v_j + k \le (b-1) + (b-1) + 1 < 2b.$$

Следовательно, k присваивается значение 0 или 1 в зависимости от того, произошел (1) или не произошел (0) перенос, т. е. выполняется присвоение $k \leftarrow [u_j + v_j + k \geq b]$.)

A3. [Цикл по j.] Увеличить j на единицу. Если теперь $j < n$, то вернуться к шагу A2, иначе — присвоить $w_n \leftarrow k$ и завершить выполнение алгоритма. ▮

По поводу формального доказательства корректности алгоритма A обратитесь к упр. 4.

Программа для компьютера MIX, реализующая эту процедуру сложения, могла бы выглядеть так.

Программа A (*Сложение неотрицательных целых чисел*). Пусть $\texttt{LOC}(u_j) \equiv \texttt{U}+j$, $\texttt{LOC}(v_j) \equiv \texttt{V}+j$, $\texttt{LOC}(w_j) \equiv \texttt{W}+j$, rI1 $\equiv j-n$, rA $\equiv k$, размер слова $\equiv b$, N $\equiv n$.

```
01      ENN1 N          1       A1. Начальная установка. j ← 0.
02      JOV  OFLO       1       Сбросить индикатор переполнения.
03   1H ENTA 0       N+1-K      k ← 0.
04      J1Z  3F         N+1-K   Если j = n, то перейти к шагу A3.
05   2H ADD  U+N,1      N       A2. Сложить разряды.
06      ADD  V+N,1      N
07      STA  W+N,1      N
08      INC1 1          N       A3. Цикл по j. j ← j + 1.
09      JNOV 1B         N       Если переполнения нет, присвоить k ← 0.
10      ENTA 1          K       Иначе — присвоить k ← 1.
11      J1N  2B         K       Если j < n, то перейти к шагу A2.
12   3H STA  W+N        1       Запомнить последний перенос в w_n. ▮
```

Время выполнения этой программы равно $10N + 6$ циклам независимо от числа переносов K. Величина K подробно анализируется в конце этого раздела.

Возможны многочисленные модификации алгоритма A, но лишь некоторые из них упоминаются в упражнениях к этому разделу. Главу, посвященную обобщениям данного алгоритма, можно было бы назвать "Разработка схем сложения для цифровой вычислительной машины".

Задача вычитания аналогична задаче сложения, но есть и заслуживающие внимания отличия.

Алгоритм S (*Вычитание неотрицательных целых чисел*). По данным неотрицательным n-разрядным целым числам $(u_{n-1} \ldots u_1 u_0)_b \geq (v_{n-1} \ldots v_1 v_0)_b$, записанным по основанию b, этот алгоритм находит неотрицательную разность $(w_{n-1} \ldots w_1 w_0)_b$.

S1. [Начальная установка.] Присвоить $j \leftarrow 0$, $k \leftarrow 0$.

S2. [Вычитание разрядов.] Присвоить $w_j \leftarrow (u_j - v_j + k) \bmod b$ и $k \leftarrow \lfloor (u_j - v_j + k)/b \rfloor$. (Другими словами, k присваивается -1 или 0 в зависимости от того, происходит ли заимствование, т. е. оказывается ли, что $u_j - v_j + k < 0$. Что касается вычисления w_j, примем во внимание тот факт, что должно выполняться неравенство $-b = 0 - (b-1) + (-1) \leq u_j - v_j + k \leq (b-1) - 0 + 0 < b$. Следовательно, $0 \leq u_j - v_j + k + b < 2b$ и это полагается в описываемой ниже программной реализации.)

S3. [Цикл по j.] Увеличить j на единицу. Если теперь $j < n$, то вернуться к шагу S2; в противном случае завершить выполнение алгоритма. (Когда выполнение алгоритма завершается, должно получиться $k = 0$; условие $k = -1$ соответствует единственному случаю, когда $(v_{n-1} \ldots v_1 v_0)_b > (u_{n-1} \ldots u_1 u_0)_b$, что противоречит сделанному раньше предположению; см. упр. 12.) ▮

При реализации алгоритма в MIX-программе удобнее вместо значения k хранить значение $1+k$, так что на шаге S2 можно вычислить величину $u_j - v_j + (1+k) + (b-1)$. (Напомним, что b — размер машинного слова.) Проиллюстрируем алгоритм следующей программой.

Программа S (*Вычитание неотрицательных целых чисел*). Эта программа аналогична программе A, но в ней rA $\equiv 1+k$. Как и в других программах этого раздела, в ячейке WM1 хранится константа $b-1$ — максимально допустимое значение, которое может храниться в машинном слове MIX (см. программу 4.2.3D, строки 38–39).

```
01      ENN1 N          1       S1. Начальная установка.  j ← 0.
02      JOV  OFLO       1       Сбросить индикатор переполнения.
03  1H  J1Z  DONE     K+1       Если j = n, завершить выполнение программы.
04      ENTA 1          K       Присвоить k ← 0.
05  2H  ADD  U+N,1      N       S2. Вычесть разряды.
06      SUB  V+N,1      N       Вычислить u_j − v_j + k + b.
07      ADD  WM1        N
08      STA  W,1        N       (Возможен минус нуль.)
09      INC1 1          N       S3. Цикл по j.  j ← j + 1.
10      JOV  1B         N       Если произошло переполнение, то присвоить k ← 0.
11      ENTA 0          N − K   В противном случае присвоить k ← −1.
12      J1N  2B         N − K   Если j < n, то вернуться к шагу S2.
13      HLT  5                  (Ошибка, v > u.)  ▮
```

Время выполнения этой программы равно $12N+3$ циклов, что немного больше, чем для программы A.

Читатель, возможно, поинтересуется, не лучше ли было бы разработать одну комбинированную программу для сложения-вычитания вместо двух разных алгоритмов A и S. Но анализ программ показывает, что в общем случае для реализации этих операций предпочтительнее использовать две разные программы, чтобы внутренний вычислительный цикл выполнялся настолько быстро, насколько это возможно.

Наша следующая задача — умножение. Здесь идеи, использованные при построении алгоритма A, получают дальнейшее развитие.

Алгоритм M (*Умножение неотрицательных целых чисел*). По заданным неотрицательным n-разрядным целым числам $(u_{n-1} \ldots u_1 u_0)_b$ и $(v_{n-1} \ldots v_1 v_0)_b$ по основанию b этот алгоритм строит их произведение $(w_{m+n-1} \ldots w_1 w_0)_b$. (Общепринятый метод умножения при помощи карандаша и бумаги основан на нахождении сначала частичных произведений $(u_{m-1} \ldots u_1 u_0) \times v_j$ для $0 \le j < n$, а затем — суммы этих произведений, сдвинутых на соответствующее число масштабирующих разрядов. Но на компьютере предпочтительнее выполнять сложение параллельно умножению, как это и делается в данном алгоритме.)

Таблица 1

УМНОЖЕНИЕ 914 НА 84.

Шаг	i	j	u_i	v_j	t	w_4	w_3	w_2	w_1	w_0
M5	0	0	4	4	16	.	.	0	0	6
M5	1	0	1	4	05	.	.	0	5	6
M5	2	0	9	4	36	.	.	6	5	6
M6	3	0	.	4	36	.	3	6	5	6
M5	0	1	4	8	37	.	3	6	7	6
M5	1	1	1	8	17	.	3	7	7	6
M5	2	1	9	8	76	.	6	7	7	6
M6	3	1	.	8	76	7	6	7	7	6

M1. [Начальная установка.] Присвоить всем параметрам $w_{m-1}, w_{m-2} \ldots, w_0$ значения "нуль". Присвоить $j \leftarrow 0$. (Если не обнулить все параметры $w_{m-1}, \ldots, w_0$, можно получить более общий алгоритм с результатом

$$(w_{m+n-1} \ldots w_0)_b \leftarrow (u_{m-1} \ldots u_0)_b \times (v_{n-1} \ldots v_0)_b + (w_{m-1} \ldots w_0)_b.$$

Этот алгоритм умножения и сложения часто используется в приложениях*.

M2. [Нулевой множитель?] Если $v_j = 0$, то присвоить $w_{j+m} \leftarrow 0$ и перейти к шагу M6. (Эта проверка может сэкономить время, если достаточно велика вероятность того, что v_j равно нулю; в противном случае проверку можно опустить без ущерба для корректности алгоритма.)

M3. [Начальная установка i.] Присвоить $i \leftarrow 0$, $k \leftarrow 0$.

M4. [Умножить и сложить.] Присвоить сначала $t \leftarrow u_i \times v_j + w_{i+j} + k$, а затем — $w_{i+j} \leftarrow t \bmod b$ и $k \leftarrow \lfloor t/b \rfloor$. (Здесь разряд переноса k всегда будет принадлежать интервалу $0 \le k < b$; см. ниже.)

M5. [Цикл по i.] Увеличить i на единицу. Если после этого $i < m$, то вернуться к шагу M4; в противном случае присвоить $w_{j+m} \leftarrow k$.

M6. [Цикл по j.] Увеличить j на единицу. Если после этого $j < n$, то вернуться к шагу M2; в противном случае завершить выполнение алгоритма. ▮

Работа алгоритма M для случая, когда $b = 10$, проиллюстрирована в табл. 1, в которой показано состояние вычислительного процесса в начале шагов M5 и M6. Доказательство корректности алгоритма M приведено в упр. 14.

Два неравенства,

$$0 \le t < b^2, \qquad 0 \le k < b, \tag{1}$$

являются критическими для эффективной реализации этого алгоритма, так как они накладывают ограничения на размер регистра, необходимый для вычислений. Их можно доказать по индукции, так как если в начале шага M4 справедливо неравенство $k < b$, то

$$u_i \times v_j + w_{i+j} + k \le (b-1) \times (b-1) + (b-1) + (b-1) = b^2 - 1 < b^2$$

Следующая MIX-программа демонстрирует те моменты, которые нужно учитывать при реализации алгоритма M на компьютере. Шаг M4 можно было бы

* Его часто называют "умножение с накоплением". — *Прим. перев.*

запрограммировать проще, если бы на компьютере имелась команда "умножить и сложить" или если бы в нем был аккумулятор удвоенной длины для сложения.

Программа M (*Умножение неотрицательных целых чисел*). Эта программа аналогична программе A. rI1 $\equiv i - m$, rI2 $\equiv j - n$, rI3 $\equiv i + j$, CONTENTS(CARRY) $\equiv k$.

```
01       ENT1 M-1          1          M1. Начальная установка.
02       JOV  OFLO         1          Сбросить индикатор переполнения.
03       STZ  W,1          M          w_rI1 ← 0.
04       DEC1 1            M
05       J1NN *-2          M          Повторить при m > rI1 ≥ 0.
06       ENN2 N            1          j ← 0.
07   1H  LDX  V+N,2         N          M2. Нулевой множитель?
08       JXZ  8F           N          Если v_j = 0, присвоить w_{j+m} ← 0 и перейти
                                        к шагу M6.
09       ENN1 M            N − Z      M3. Начальная установка i. i ← 0.
10       ENT3 N,2          N − Z      (i + j) ← j.
11       ENTX 0            N − Z      k ← 0.
12   2H  STX  CARRY        (N − Z)M   M4. Умножить и сложить.
13       LDA  U+M,1        (N − Z)M
14       MUL  V+N,2        (N − Z)M   rAX ← u_i × v_j.
15       SLC  5            (N − Z)M   Произвести взаимную замену содержимого
                                        регистров rA ↔ rX.
16       ADD  W,3          (N − Z)M   Прибавить w_{i+j} к нижней половине.
17       JNOV *+2          (N − Z)M   Переполнение произошло?
18       INCX 1               K       Если произошло, то разряд переноса, равный 1,
                                        занести в верхнюю половину.
19       ADD  CARRY        (N − Z)M   Прибавить k к нижней половине.
20       JNOV *+2          (N − Z)M   Переполнение произошло?
21       INCX 1               K'      Если произошло, то разряд переноса, равный 1,
                                        занести в верхнюю половину.
22       STA  W,3          (N − Z)M   w_{i+j} ← t mod b.
23       INC1 1            (N − Z)M   M5. Цикл по i. i ← i + 1.
24       INC3 1            (N − Z)M   (i + j) ← (i + j) + 1.
25       J1N  2B           (N − Z)M   Возврат к шагу M4, если i < m, rX = ⌊t/b⌋.
26   8H  STX  W+M+N,2          N      Присвоить w_{j+m} ← k.
27       INC2 1               N       M6. Цикл по j. j ← j + 1.
28       J2N  1B              N       Повторять до тех пор, пока не станет j = n.  ∎
```

Время выполнения программы M зависит от числа разрядов M множителя $\boldsymbol{u}$, числа разрядов N множимого v, числа нулей Z в множителе, и числа переносов K и K', возникающих при выполнении сложения в нижней половине произведения в ходе вычисления t. Если ограничить значения K и K' разумной (хотя и несколько пессимистической) величиной $\frac{1}{2}(N-Z)M$, то время выполнения программы составит примерно $28MN + 4M + 10N + 3 - Z(28M + 3)$ циклов. Если исключить шаг M2, то время выполнения программы составит $28MN + 4M + 7N + 3$ циклов, так что включение этого шага выгодно лишь при условии, что плотность числа нулей в множителе равна $Z/N > 3/(28M + 3)$. Если множители выбираются случайным образом, то следует ожидать, что среднее значение этой плотности будет около $1/b$, что очень мало. Поэтому включать шаг M2, вообще говоря, невыгодно, даже если число b мало́.

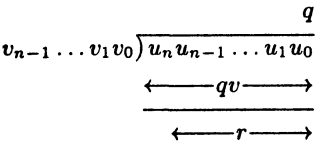


Рис. 6. Цель: быстро определить q.

Алгоритм М — не самый быстрый способ умножения, если множители m и n велики, хотя он имеет преимущество (он прост). Более быстрые, но более сложные методы рассматриваются в разделе 4.3.3; уже при $m = n = 4$ можно перемножать числа быстрее, чем при помощи алгоритма М.

Последний из алгоритмов, которые рассматриваются в этом разделе, — деление в столбик $(m+n)$-разрядного целого числа на n-разрядное целое число. При обычном делении в столбик вручную в процессе вычисления необходимо строить догадки, что требует от вычисляющего определенного искусства. Поэтому необходимо либо совсем исключить этот "творческий процесс" из алгоритма, либо развить некую теорию, позволяющую строго его формализовать.

Самый беглый анализ обычного процесса деления в столбик показывает, что вся задача разбивается на более простые шаги, каждый из которых состоит в делении $(n+1)$-разрядного числа u на n-разрядное число v, где $0 \le u/v < b$. Остаток r после выполнения каждого шага меньше v, поэтому величину rb+(следующий разряд делителя) можно использовать как новое число u на следующем шаге. Например, если ставится задача разделить 3 142 на 53, то сначала разделим 314 на 53 и получим 5 в качестве частного и 49 в остатке; затем разделим 492 на 53 и получим 9 и 15 в остатке. Таким образом, окончательно получаем в качестве частного 59 и в остатке 15. Ясно, что эта идея подходит и для общего случая, так что поиски подходящего алгоритма деления сводятся к следующей задаче (рис. 6).

> *Пусть* $u = (u_nu_{n-1}\ldots u_1u_0)_b$ *и* $v = (v_{n-1}\ldots v_1v_0)_b$ *— представленные по основанию* b *неотрицательные целые числа, такие, что* $u/v < b$. *Найти алгоритм для определения числа* $q = \lfloor u/v \rfloor$.

Можно заметить, что условие $u/v < b$ равносильно условию $u/b < v$, так что $\lfloor u/b \rfloor < v$. Это просто условие $(u_nu_{n-1}\ldots u_1)_b < (v_{n-1}v_{n-2}\ldots v_0)_b$. Далее, если записать $r = u - qv$, искомое число q будет единственным целым числом, таким, что $0 \le r < v$.

Наиболее простой подход к решению поставленной задачи состоит в том, чтобы высказать какую-нибудь догадку относительно величины q, основываясь на наиболее значимых разрядах u и v. Не ясно, достаточно ли надежен такой подход, но он заслуживает исследования. Итак, положим

$$\hat{q} = \min\left(\left\lfloor \frac{u_nb + u_{n-1}}{v_{n-1}} \right\rfloor,\ b-1\right). \tag{2}$$

Согласно этой формуле $\hat{q}$ получается делением двух первых значащих разрядов числа u на первый значащий разряд числа v. Если результат равен b или больше b, заменяем его на $b-1$.

Примечательно, что $\hat{q}$ всегда оказывается очень хорошим приближением к искомому ответу q, если только v_{n-1} относительно велико. Прежде чем анализировать степень близости $\hat{q}$ к q, докажем, что $\hat{q}$ не может быть слишком мало́.

Теорема A. *$\hat{q} \geq q$ (в принятых выше обозначениях).*

Доказательство. Так как $q \leq b-1$, теорема, безусловно, верна при $\hat{q} = b-1$. В противном случае имеем $\hat{q} = \lfloor (u_n b + u_{n-1})/v_{n-1} \rfloor$, поэтому $\hat{q}v_{n-1} \geq u_n b + u_{n-1} - v_{n-1} + 1$. Отсюда следует, что

$$\begin{aligned} u - \hat{q}v &\leq u - \hat{q}v_{n-1}b^{n-1} \\ &\leq u_n b^n + \cdots + u_0 - (u_n b^n + u_{n-1}b^{n-1} - v_{n-1}b^{n-1} + b^{n-1}) \\ &= u_{n-2}b^{n-2} + \cdots + u_0 - b^{n-1} + v_{n-1}b^{n-1} < v_{n-1}b^{n-1} \leq v. \end{aligned}$$

Поскольку $u - \hat{q}v < v$, мы должны получить $\hat{q} \geq q$. ▮

Докажем теперь, что на практике $\hat{q}$ не может быть намного больше, чем q. Предположим, что $\hat{q} \geq q+3$. Тогда

$$\hat{q} \leq \frac{u_n b + u_{n-1}}{v_{n-1}} = \frac{u_n b^n + u_{n-1}b^{n-1}}{v_{n-1}b^{n-1}} \leq \frac{u}{v_{n-1}b^{n-1}} < \frac{u}{v - b^{n-1}}.$$

(Случай, когда $v = b^{n-1}$, невозможен из-за того, что если $v = (100\ldots0)_b$, то $q = \hat{q}$.) Далее, так как $q > (u/v) - 1$, то

$$3 \leq \hat{q} - q < \frac{u}{v - b^{n-1}} - \frac{u}{v} + 1 = \frac{u}{v}\left(\frac{b^{n-1}}{v - b^{n-1}}\right) + 1.$$

Поэтому

$$\frac{u}{v} > 2\left(\frac{v - b^{n-1}}{b^{n-1}}\right) \geq 2(v_{n-1} - 1).$$

Таким образом, поскольку $b - 4 \geq \hat{q} - 3 \geq q = \lfloor u/v \rfloor \geq 2(v_{n-1} - 1)$, то $v_{n-1} < \lfloor b/2 \rfloor$. Это доказывает следующую теорему.

Теорема B. *Если $v_{n-1} \geq \lfloor b/2 \rfloor$, то $\hat{q} - 2 \leq q \leq \hat{q}$.* ▮

Наиболее важным моментом этой теоремы является то, что *постулируемое в теореме выражение не зависит от b*. Не важно, насколько велико b: пробное частное $\hat{q}$ никогда не отличается от истинного более чем на 2.

Условие $v_{n-1} \geq \lfloor b/2 \rfloor$ очень похоже на условие нормализации; действительно, оно в точности совпадает с условием нормализации для двоичных компьютеров. Один из простых способов обеспечения достаточно большого значения v_{n-1} — умножить оба числа u и v на $\lfloor b/(v_{n-1}+1) \rfloor$. Это не изменит значения u/v, не увеличит количество разрядов в v и, как доказывается в упр. 23, всегда приведет к достаточно большому новому значению v_{n-1}. (Другой путь нормализации делимого рассматривается в упр. 28.)

Вооружившись теперь всеми этими сведениями, можно разработать требуемый алгоритм деления в столбик (рис. 7). В нем на шаге D3 используется несколько усовершенствованный способ выбора числа $\hat{q}$, гарантирующий, что $q = \hat{q}$ или $\hat{q} - 1$. На практике этот улучшенный метод выбора числа $\hat{q}$ почти всегда дает точное решение.

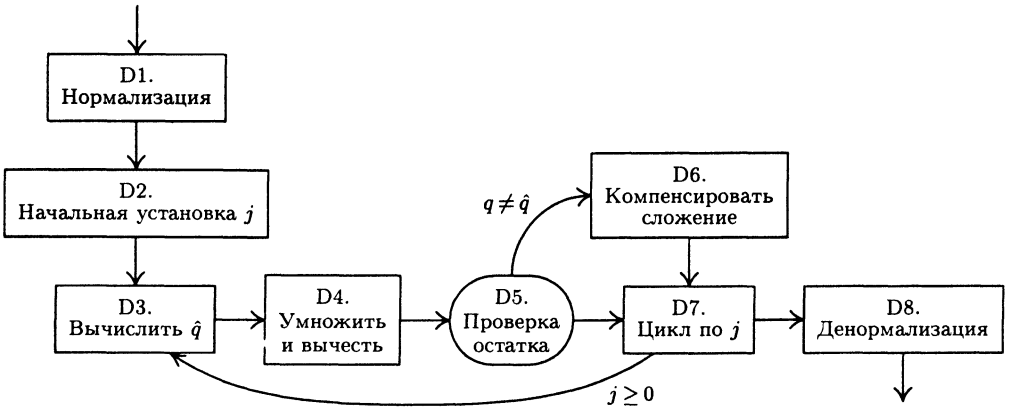


Рис. 7. "Длинное" деление.

Алгоритм D (*Деление неотрицательных целых чисел*). По данным неотрицательным целым числам $u = (u_{m+n-1} \ldots u_1 u_0)_b$ и $v = (v_{n-1} \ldots v_1 v_0)_b$, представленным по основанию b, в предположении, что $v_{n-1} \neq 0$ и $n > 1$, находим в системе счисления по основанию b частное $\lfloor u/v \rfloor = (q_m q_{m-1} \ldots q_0)_b$ и остаток $u \bmod v = (r_{n-1} \ldots r_1 r_0)_b$. (При $n = 1$ следует пользоваться более простым алгоритмом из упр. 16.)

D1. [Нормализация.] Присвоить $d \leftarrow \lfloor b/(v_{n-1}+1) \rfloor$. Затем присвоить $(u_{m+n} u_{m+n-1} \ldots u_1 u_0)_b$ значение $(u_{m+n-1} \ldots u_1 u_0)_b$, умноженное на d. Точно так присвоить $(v_{n-1} \ldots v_1 v_0)_b$ значение $(v_{n-1} \ldots v_1 v_0)_b$, умноженное на d. (Обратите внимание на введение нового разряда u_{m+n} слева от u_{m+n-1}; если $d = 1$, то все, что необходимо выполнить на этом шаге, — присвоить $u_{m+n} \leftarrow 0$. На двоичном компьютере может оказаться, что предпочтительнее вместо предлагаемого здесь значения взять в качестве d некоторую степень 2; подойдет любое значение d, приводящее к выполнению неравенства $v_{n-1} \geq \lfloor b/2 \rfloor$. См. также упр. 37.)

D2. [Начальная установка j.] Присвоить $j \leftarrow m$. (Цикл по j, который состоит из шагов D2–D7, по существу, представляет собой процесс деления $(u_{j+n} \ldots u_{j+1} u_j)_b$ на $(v_{n-1} \ldots v_1 v_0)_b$, дающий один разряд частного q_j; см. рис. 6.)

D3. [Вычислить $\hat{q}$.] Присвоить $\hat{q} \leftarrow \lfloor (u_{j+n} b + u_{j+n-1})/v_{n-1} \rfloor$, и пусть $\hat{r}$ — остаток, $(u_{j+n} b + u_{j+n-1}) \bmod v_{n-1}$. Проверить выполнение неравенства $\hat{q} = b$ или $\hat{q} v_{n-2} > b\hat{r} + u_{j+n-2}$. Если оно удовлетворяется, то уменьшить $\hat{q}$ на 1, увеличить $\hat{r}$ на v_{n-1} и повторить эту проверку при $\hat{r} < b$. (В ходе проверки обнаруживается, и притом очень быстро, большинство случаев, когда пробное значение $\hat{q}$ на единицу больше истинного, и исключаются *все* случаи, когда $\hat{q}$ больше истинного на два; см. упр. 19–21.)

D4. [Умножить и вычесть.] Заменить $(u_{j+n} u_{j+n-1} \ldots u_j)_b$ на

$$(u_{j+n} u_{j+n-1} \ldots u_j)_b \; - \; \hat{q} (v_{n-1} \ldots v_1 v_0)_b.$$

Этот шаг (аналогичный шагам M3–M5 алгоритма M) состоит из простой операции умножения на одноразрядное число, скомбинированной с вычитанием. Значения разрядов $(u_{j+n}, u_{j+n-1}, \ldots, u_j)$ всегда должны быть положительными; если на этом шаге получен отрицательный результат, то в качестве истинного результата должно остаться $(u_{j+n} u_{j+n-1} \ldots u_j)_b$ плюс b^{n+1}, а именно —

представление истинного значения в виде дополнения до b, причем следует запомнить "заимствование" слева, из старшего разряда.

D5. [Проверка остатка.] Присвоить $q_j \leftarrow \hat{q}$. Если результат шага D4 был отрицательным, перейти к шагу D6; в противном случае перейти к шагу D7.

D6. [Компенсировать сложение.] (Как показано в упр. 21, вероятность того, что данный шаг понадобится, очень мала (порядка $2/b$). Поэтому до его выполнения следует проанализировать данные.) Уменьшить q_j на 1 и добавить $(0v_{n-1}\ldots v_1v_0)_b$ к $(u_{j+n}u_{j+n-1}\ldots u_{j+1}u_j)_b$. (При этом произойдет перенос в разряд, находящийся слева от u_{j+n}, но им следует пренебречь, так как перенос погашается "заимствованием" из того же разряда, произведенным на шаге D4.)

D7. [Цикл по j.] Уменьшить j на единицу. Если теперь $j \geq 0$, то вернуться к шагу D3.

D8. [Денормализация.] Теперь $(q_m\ldots q_1q_0)_b$ есть искомое частное, а для получения искомого остатка достаточно $(u_{n-1}\ldots u_1u_0)_b$ разделить на d. ∎

При реализации алгоритма D в виде MIX-программы нужно обратить внимание на несколько интересных моментов.

Программа D (*Деление неотрицательных целых чисел*). Соглашения о начальном состоянии для этой программы аналогичны соглашениям для программы A; rI1 $\equiv i - n$, rI2 $\equiv j$, rI3 $\equiv i + j$.

```
001  D1 JOV  OFLO            1        D1. Нормализация.
...                                   (См. упр. 25.)
039  D2 ENT2 M               1        D2. Начальная установка j. j ← m.
040     STZ  V+N             1        Присвоить v_n ← 0 для удобства выполнения
                                        шага D4.
041  D3 LDA  U+N,2(1:5)      M+1      D3. Вычислить q̂.
042     LDX  U+N-1,2         M+1      rAX ← u_{j+n}b + u_{j+n-1}.
043     DIV  V+N-1           M+1      rA ← ⌊rAX/v_{n-1}⌋.
044     JOV  1F              M+1      Переход, если частное = b.
045     STA  QHAT            M+1      q̂ ← rA.
046     STX  RHAT            M+1      r̂ ← u_{j+n}b + u_{j+n-11} − q̂v_{n-1}
047     JMP  2F              M+1           = (u_{j+n}b + u_{j+n-1}) mod v_{n-1}.
048  1H LDX  WM1                      rX ← b − 1.
049     LDA  U+N-1,2                  rA ← u_{j+n-1}. (Здесь u_{j+n} = v_{n-1}.)
050     JMP  4F
051  3H LDX  QHAT            E
052     DECX 1               E        Уменьшить q̂ на 1.
053     LDA  RHAT            E        Соответственно установить r̂.
054  4H STX  QHAT            E        q̂ ← rX.
055     ADD  V+N-1           E        rA ← r̂ + v_{n-1}.
056     JOV  D4              E        (Если r̂ окажется ≥ b, то q̂v_{n-2} будет < r̂b.)
057     STA  RHAT            E        r̂ ← rA.
058     LDA  QHAT            E
059  2H MUL  V+N-2           M+E+1
060     CMPA RHAT            M+E+1    Проверить q̂v_{n-2} ≤ r̂b + u_{j+n-2}.
061     JL   D4              M+E+1
062     JG   3B              E
```

```
063      CMPX U+N-2,2
064      JG   3B                                  Если нет, то q̂ слишком велико.
065  D4  ENTX 1              M+1                  D4. Умножить и сложить.
066      ENN1 N              M+1                  i ← 0.
067      ENT3 0,2            M+1                  (i+j) ← j.
068  2H  STX  CARRY      (M+1)(N+1)               (Здесь 1 − b < rX ≤ +1.)
069      LDAN V+N,1      (M+1)(N+1)
070      MUL  QHAT       (M+1)(N+1)               rAX ← −q̂v_i.
071      SLC  5          (M+1)(N+1)               Взаимный обмен rA ↔ rX.
072      ADD  CARRY      (M+1)(N+1)               Добавить вклад от
073      JNOV *+2        (M+1)(N+1)                   разрядов справа плюс 1.
074      DECX 1              K                    Если сумма ≤ −b, то перенос −1.
075      ADD  U,3        (M+1)(N+1)               Прибавить u_{i+j}.
076      ADD  WM1        (M+1)(N+1)               Прибавить b − 1, чтобы получить знак "+".
077      JNOV *+2        (M+1)(N+1)               Если переполнения нет, то перенос −1.
078      INCX 1              K'                   rX ≡ перенос + 1.
079      STA  U,3        (M+1)(N+1)               u_{i+j} ← rA (возможен минус нуль).
080      INC1 1          (M+1)(N+1)
081      INC3 1          (M+1)(N+1)
082      J1NP 2B         (M+1)(N+1)               Повторить для 0 ≤ i ≤ n.
083  D5  LDA  QHAT           M+1                  D5. Проверка остатка.
084      STA  Q,2            M+1                  Присвоить q_j ← q̂.
085      JXP  D7             M+1                  (Здесь rX = 0 или 1, так как v_n = 0.)
086  D6  DECA 1                                   D6. Компенсирующее сложение.
087      STA  Q,2                                 Присвоить q_j ← q̂ − 1.
088      ENN1 N                                   i ← 0.
089      ENT3 0,2                                 (i+j) ← j.
090  1H  ENTA 0                                   (Этот фрагмент, по существу, —
                                                      программа A.)
091  2H  ADD  U,3
092      ADD  V+N,1
093      STA  U,3
094      INC1 1
095      INC3 1
096      JNOV 1B
097      ENTA 1
098      J1NP 2B
099  D7  DEC2 1              M+1                  D7. Цикл по j.
100      J2NN D3             M+1                  Повторять для m ≥ j ≥ 0.
101  D8  ···                                      (См. упр. 26.)  ▮
```

Обратите внимание, как легко реализуются в машине казавшиеся довольно сложными вычисления и выбор вариантов на шаге D3. Отметим также, что программа для выполнения шага D4 аналогична программе M с той лишь оговоркой, что здесь применяются еще и идеи программы S.

Для оценки времени выполнения программы D следует рассмотреть значения параметров M, N, E, K и K', представленных в программе. (Эти параметры не учитывают ситуации, которые могут возникнуть с очень малой вероятностью; например, можно предполагать, что команды в строках программы 048–050, 063, 064

и фрагмент, соответствующий шагу D6, никогда не выполняются.) Здесь $M+1$ — количество слов в частном, N — количество слов в делителе, E — число, показывающее, сколько раз $\hat{q}$ уменьшается на шаге D3, K и K' — число установок "разрядов переноса" в ходе выполнения цикла "умножение-вычитание". Если предположить, что $K+K'$ приблизительно равно $(N+1)(M+1)$ и что E приблизительно равно $\frac{1}{2}M$, можно получить, что общее время выполнения программы приблизительно равно $30MN + 30N + 89M + 111$ циклов плюс дополнительно $67N + 235M + 4$ циклов, если $d > 1$. (В этих расчетах учитывались фрагменты программ из упр. 25 и 26.) При больших M и N это время только на 7% больше времени, которое потребуется программе M, чтобы перемножить частное и делитель.

Если основание системы счисления b сравнительно мало́, так что b^2 меньше, чем размер машинного слова, процесс деления может быть ускорен за счет того, что исключаются условия попадания значений отдельных разрядов промежуточного результата в интервал $[0\,..\,b)$ (см. D. M. Smith, *Math. Comp.* **65** (1996), 157–163).

Дополнительные комментарии к алгоритму D включены в упражнения в конце этого раздела.

При отладке программ реализации арифметических операций многократной точности для проверки результатов выполнения программы деления можно использовать программы умножения и сложения. Для тестирования иногда бывают полезны следующие числа:

$$(t^m - 1)(t^n - 1) = t^{m+n} - t^n - t^m + 1.$$

При $m < n$ такие числа записываются по основанию t в виде

$$\underbrace{(t-1)\quad\ldots\quad(t-1)}_{m-1\text{ разрядов}}\quad(t-2)\quad\underbrace{(t-1)\quad\ldots\quad(t-1)}_{n-m\text{ разрядов}}\quad\underbrace{0\quad\ldots\quad0}_{m-1\text{ разрядов}}\quad 1,$$

например $(10^3 - 1)(10^8 - 1) = 99899999001$. Для программы D необходимо также опробовать некоторые ситуации, когда вступают в действие редко работающие фрагменты программы; некоторые из фрагментов, вероятно, остались бы непроверенными даже после миллиона случайных тестов (см. упр. 22).

Теперь, после ознакомления с принципами работы с числами, представленными в прямом коде, посмотрим, какой подход следует избрать для решения тех же задач в случае, когда используется компьютер с представлением чисел в виде дополнений. Для дополнительного и обратного кодов по основанию 2 в качестве основания b лучше всего брать *половину* размера слова. Таким образом, для 32-битового машинного слова в вышеприведенных алгоритмах использовалось бы основание $b = 2^{31}$. Знаковый бит всех слов, кроме наиболее значимого слова в представлении с многократной точностью, будет равен нулю, поэтому в ходе выполнения машинных операций умножения и деления аномальных коррекций знака не будет. Фактически по самой сути представления в виде дополнения все слова, кроме наиболее значимого, рассматриваются как неотрицательные. Например, при 8-битовом слове число, имеющее в дополнительном двоичном коде вид

11011111 1111110 1101011

(в котором знаковый бит указан только для наиболее значимого слова), следует понимать как

$$-2^{21} + (1011111)_2 \cdot 2^{14} + (1111110)_2 \cdot 2^7 + (1101011)_2.$$

С другой стороны, в некоторых двоичных компьютерах с дополнительным кодом также предусмотрена арифметика без знака. Например, пусть x и y — 32-битовые операнды. Компьютер может воспринимать их как числа в дополнительном коде в интервале $-2^{31} \le x, y < 2^{31}$ или как беззнаковые числа в интервале $0 \le x, y < 2^{32}$. Если не обращать внимания на переполнение, то 32-битовое число, равное сумме $(x+y) \bmod 2^{32}$, будет одинаковым в любом из этих представлений, но если изменить интервал, то в определенных ситуациях может произойти переполнение. Если в компьютере предусмотрена простая операция вычисления бита переноса $\lfloor (x+y)/2^{32} \rfloor$ в беззнаковой интерпретации и выдается 64-битовый результат для беззнаковых 32-битовых целых чисел, то вместо основания $b = 2^{31}$ в алгоритмах для арифметики с высокой точностью можно использовать основание $b = 2^{32}$.

Представление в виде дополнения позволяет проще выполнять сложение чисел со знаком, так как программа для сложения n-разрядных неотрицательных целых чисел может использоваться для сложения произвольных n-разрядных целых чисел. Знак появляется только в первом слове, поэтому менее значимые слова можно складывать независимо от действительного знака числа. (В случае использования обратного кода для представления числа особое внимание следует обратить на перенос из крайнего слева (старшего) разряда. Он должен быть добавлен к наименее значимому слову и, возможно, передан дальше влево*.) Аналогично вычитание чисел со знаком в таком представлении выполняется несколько проще. С другой стороны, кажется, умножение и деление легче всего производить над неотрицательными величинами, предварительно выполнив операции дополняющего преобразования, чтобы оба операнда были неотрицательными. Можно также при помощи некоторых специальных приемов избежать этих преобразований и работать непосредственно с отрицательными числами в виде дополнений. Нетрудно показать, как это можно было бы осуществить в случае умножения чисел с удвоенной точностью. Однако при этом необходимо следить, чтобы не было замедления во время выполнения операций во внутренних циклах подпрограмм, когда требуется высокая точность.

Обратимся теперь к анализу величины K, появляющейся в программе A, т. е. числа переносов, происходящих при сложении двух n-разрядных чисел. Хотя величина K и не влияет на общее время выполнения программы A, от нее зависит время работы "двойников" программы A, связанных с представлением чисел в виде дополнения. К тому же ее анализ интересен сам по себе как замечательное применение метода производящих функций.

Предположим, что u и v — независимые случайные n-разрядные целые числа, равномерно распределенные в интервале $0 \le u, v < b^n$. Пусть p_{nk} — вероятность того, что при сложении u и v произошло ровно k переносов u при этом один из переносов произошел в наиболее значимом разряде. Нетрудно видеть, что для всех

---

* Это так называемый "циклический перенос". — *Прим. перев.*

k и n

$$p_{0k} = 0, \qquad p_{(n+1)(k+1)} = \frac{b+1}{2b} p_{nk} + \frac{b-1}{2b} q_{nk},$$
$$q_{0k} = \delta_{0k}, \qquad q_{(n+1)k} = \frac{b-1}{2b} p_{nk} + \frac{b+1}{2b} q_{nk}. \tag{3}$$

Это следует из того, что $(b-1)/2b$ есть вероятность того, что $u_{n-1} + v_{n-1} \geq b$, и $(b+1)/2b$ есть вероятность того, что $u_{n-1} + v_{n-1} + 1 \geq b$, где u_{n-1} и v_{n-1} — независимые равномерно распределенные в интервале $0 \leq u_{n-1}, v_{n-1} < b$ целые числа.

Построим производящие функции

$$P(z,t) = \sum_{k,n} p_{nk}\, z^k t^n, \qquad Q(z,t) = \sum_{k,n} q_{nk}\, z^k t^n. \tag{4}$$

Из равенств (3) следуют основные соотношения

$$P(z,t) = zt\left(\frac{b+1}{2b} P(z,t) + \frac{b-1}{2b} Q(z,t)\right),$$
$$Q(z,t) = 1 + t\left(\frac{b-1}{2b} P(z,t) + \frac{b+1}{2b} Q(z,t)\right).$$

Эти два уравнения легко разрешаются относительно $P(z,t)$ и $Q(z,t)$. Если положить

$$G(z,t) = P(z,t) + Q(z,t) = \sum_n G_n(z) t^n,$$

где $G_n(z)$ — производящая функция для общего числа переносов при сложении n-разрядных чисел, то получим

$$G(z,t) = (b - zt)/p(z,t), \quad \text{где } p(z,t) = b - \tfrac{1}{2}(1+b)(1+z)t + zt^2. \tag{5}$$

Заметим, что $G(1,t) = 1/(1-t)$ в полном соответствии с тем фактом, что $G_n(1)$ должно равняться 1 (как сумма вероятностей всех возможных событий). Взяв частные производные от (5) по z, получаем

$$\frac{\partial G}{\partial z} = \sum_n G_n'(z) t^n = \frac{-t}{p(z,t)} + \frac{t(b-zt)(b+1-2t)}{2p(z,t)^2};$$
$$\frac{\partial^2 G}{\partial z^2} = \sum_n G_n''(z) t^n = \frac{-t^2(b+1-2t)}{p(z,t)^2} + \frac{t^2(b-zt)(b+1-2t)^2}{2p(z,t)^3}.$$

Положим теперь $z = 1$ и разложим $P(z,t)$ и $Q(z,t)$ на элементарные дроби:

$$\sum_n G_n'(1) t^n = \frac{t}{2}\left(\frac{1}{(1-t)^2} - \frac{1}{(b-1)(1-t)} + \frac{1}{(b-1)(b-t)}\right),$$
$$\sum_n G_n''(1) t^n = \frac{t^2}{2}\left(\frac{1}{(1-t)^3} - \frac{1}{(b-1)^2(1-t)} + \frac{1}{(b-1)^2(b-t)} + \frac{1}{(b-1)(b-t)^2}\right).$$

Отсюда следует, что среднее число переносов, т. е. математическое ожидание величины K, равно

$$G_n'(1) = \frac{1}{2}\left(n - \frac{1}{b-1}\left(1 - \left(\frac{1}{b}\right)^n\right)\right), \tag{6}$$

а дисперсия равна

$$G_n''(1) + G_n'(1) - G_n'(1)^2 = \frac{1}{4}\left(n + \frac{2n}{b-1} - \frac{2b+1}{(b-1)^2} + \frac{2b+2}{(b-1)^2}\left(\frac{1}{b}\right)^n - \frac{1}{(b-1)^2}\left(\frac{1}{b}\right)^{2n}\right). \tag{7}$$

Таким образом, при сделанных допущениях число переносов несколько меньше, чем $\frac{1}{2}n$.

История и библиография. Раннюю историю описанных в этом разделе классических алгоритмов предоставляем читателю в качестве интересной темы для самостоятельного изучения. Здесь же будет прослежена лишь история их внедрения на современных компьютерах.

Использование числа 10^n в качестве основания системы счисления применительно к умножению больших чисел на калькуляторе обсуждалась Д. Н. Лемером (D. N. Lehmer) и Дж. Р. Баллантином (J. P. Ballantine) [*AMM* **30** (1923), 67–69]. Арифметика с удвоенной точностью для компьютеров впервые была рассмотрена Дж. фон Нейманом (J. von Neumann) и Г. Г. Голдстайном (H. H. Goldstine) во введении к руководству по программированию, впервые опубликованному в 1947 году [J. von Neumann, *Collected Works* **5**, 142–151]. Теоремы A и B, изложенные выше, принадлежат Д. А. Поупу (D. A. Pope) и М. Л. Стейну (M. L. Stein) [*CACM* **3** (1960), 652–654]. В их статье приведена также библиография более ранних работ, посвященных арифметике с удвоенной точностью. Другие способы выбора пробного частного $\hat{q}$ рассмотрены А. Г. Коксом (A. G. Cox) и Г. А. Лютером (H. A. Luther) в *CACM* **4** (1961), 353 [деление на $v_{n-1}+1$ вместо v_{n-1}], а также М. Л. Стейном в *CACM* **7** (1964), 472–474 [деление на v_{n-1} или $v_{n-1}+1$ в зависимости от величины v_{n-2}]. Е. В. Кришнамурти (E. V. Krishnamurthy) [*CACM* **8** (1965), 179–181] показал, что исследование остатка от деления с однократной точностью позволяет усилить теорему B. Кришнамурти и Нанди (Nandi) [*CACM* **10** (1967), 809–813] предложили способ замены нормализации и денормализации в алгоритме D вычислением $\hat{q}$, которое базируется на анализе нескольких ведущих разрядов операндов. Интересный анализ оригинального алгоритма Поупа и Стейна выполнили Г. Э. Коллинз (G. E. Collins) и Д. Р. Муссер (D. R. Musser) [*Information Processing Letters* **6** (1977), 151–155].

Предлагались и другие методы деления.

1) "Деление по Фурье" [J. Fourier, *Analyse des Équations Déterminées* (Paris, 1831), §2.21]. В этом методе, часто используемом в калькуляторах, каждый новый разряд, по существу, получается посредством увеличения на каждом шаге точности представления делимого и делителя. Довольно большое количество тестов, выполненных автором, показали, что этот метод хуже описанного выше метода "деления и коррекции", но в некоторых приложениях деление по Фурье вполне приемлемо. (См. статью Д. Г. Лемера в *AMM* **33** (1926), 198–206, и книгу Дж. В. Успенского (J. V. Uspensky) *Theory of Equations* (New York: McGraw-Hill, 1948), 159–164.)

2) В ранних вычислительных машинах, в которых не было команды деления с однократной точностью, для вычисления обратной величины числа широко использовался "метод Ньютона". Его идея состоит в нахождении некоторого начального приближения x_0 к числу $1/v$ и выполнении дальнейших вычислений по формуле

$x_{n+1} = 2x_n - vx_n^2$. Этот метод обеспечивает быструю сходимость к $1/v$, так как из равенства $x_n = (1-\epsilon)/v$ следует, что $x_{n+1} = (1-\epsilon^2)/v$. Порядок сходимости равен трем, т. е. решение можно найти по формуле

$$\begin{aligned} x_{n+1} &= x_n + x_n(1-vx_n) + x_n(1-vx_n)^2 \\ &= x_n\left(1 + (1-vx_n)(1+(1-vx_n))\right), \end{aligned}$$

заменив на каждом шаге ϵ на $O(\epsilon^3)$. Аналогичные формулы получены для четвертого порядка сходимости (см. P. Rabinowitz, *CACM* **4** (1961), 98). Для выполнения вычислений с очень большими числами метод Ньютона со вторым порядком точности (с последующим умножением на u) может действительно оказаться намного быстрее алгоритма D, если на каждом шаге повысить точность вычисления x_n и, кроме того, воспользоваться программами быстрого умножения из раздела 4.3.3 (см. алгоритм 4.3.3R). Несколько родственных итеративных схем рассмотрено в статье E. B. Кришнамурти, опубликованной в журнале *IEEE Trans.* **C-19** (1970), 227–231.

3) Ряд методов деления основан на разложении числа $\frac{u}{v+\epsilon}$ в ряд

$$\frac{u}{v+\epsilon} = \frac{u}{v}\left(1 - \left(\frac{\epsilon}{v}\right) + \left(\frac{\epsilon}{v}\right)^2 - \left(\frac{\epsilon}{v}\right)^3 + \cdots\right).$$

(См. статью Г. Г. Лофлина (H. H. Laughlin) в журнале *AMM* **37** (1930), 287–293.) Мы уже применяли эту идею для вычислений с удвоенной точностью (см. выражение 4.2.3–(2)).

Помимо упомянутых работ, отметим следующие статьи, посвященные арифметике с многократной точностью. Программы арифметики с высокой точностью в формате с плавающей точкой, использующие обратный код, описаны в работе A. H. Stroud, D. Secrest, *Comp. J.* **6** (1963), 62–66. Имеется описание подпрограмм повышенной точности, предназначенных для использования в программах, написанных на языках FORTRAN (I. Blum, *CACM* **8** (1965), 318–320) и ALGOL (M. Tienari, Suokonautio, *BIT* **6** (1966), 332–338). Арифметические операции над целыми числами с *неограниченной* точностью с привлечением методов связного распределения памяти были элегантно реализованы Г. Э. Коллинзом (G. E. Collins) [*CACM* **9** (1966), 578–589]. Для ознакомления с более обширным набором операций над числами с многократной точностью, включая логарифмы и тригонометрические функции, обратитесь к работам R. P. Brent, *ACM Trans. Math. Software* **4** (1978), 57–81; D. M. Smith, *ACM Trans. Math. Software* **17** (1991), 273–283.

Достижения человечества в технике вычислений традиционно оцениваются количеством десятичных разрядов числа π, известным на данный момент истории. В разделе 4.1 упоминалось о нескольких ранних достижениях в решении этой задачи. В 1719 году Тома Фанте де Ланьи (Thomas Fantet de Lagny) вычислил число π до 127 десятичных знаков [*Mémoires Acad. Sci. Paris* (1719), 135–145; в 113-м знаке была допущена типографская ошибка].

Когда были изобретены более совершенные формулы для вычисления числа π, знаменитому вычислителю из Гамбурга Захариусу Дазе (Zacharias Dase) в 1844 году потребовалось менее двух месяцев для вычисления 200 правильных десятичных знаков числа π [*Crelle* **27** (1844), 198]. После этого в 1853 году Уильям Шэнкс

(William Shanks) опубликовал 607 десятичных знаков числа π. Он продолжал свои вычисления, пока в 1873 году не определил 707 правильных десятичных знаков π. [См. W. Shanks, *Contributions to Mathematics* (London, 1853); *Proc. Royal Soc. London* **21** (1873), 318–319; **22** (1873), 45–46; J. C. V. Hoffmann, *Zeit. für math. und naturwiss. Unterricht* **26** (1895), 261–264.]

Значение числа π с точностью до 707 знаков в течение многих лет широко использовалось в книгах, пока Д. Ф. Фергюсон (D. F. Ferguson) не заметил в 1945 году, что в нем имеется несколько ошибок, начиная с 528-го десятичного знака [*Math. Gazette* **30** (1946), 89–90]. В 1949 году в рамках проведения Дня труда (Labor Day) Г. Райтвайзнер (G. Reitwiesner) с сотрудниками затратил 70 часов машинного времени на ЭВМ ENIAC для вычисления 2 037 правильных десятичных цифр числа π [*Math. Tables and Other Aids to Comp.* **4** (1950), 11–15]. Ф. Гениус (F. Genuys) в 1958 году получил 10 000 цифр после 100 минут вычислений на IBM 704 [*Chiffres* **1** (1958), 17–22]. Вскоре после этого Д. Шэнкс (D. Shanks) (не путать с Уильямом!) и Дж. У. Ренч (J. W. Wrench) опубликовали 100 000 цифр, полученных после почти 8 часов вычислений на ЭВМ IBM 7090 и еще 4.5 часов проверки [*Math. Comp.* **16** (1962), 76–99]. В результате проверки ими была обнаружена случайная ошибка в схеме, устраненная при повторном счете. В 1973 году, затратив 24 часа машинного времени на ЭВМ CDC 7600, Жан Гилу (Jean Guilloud) и Мартин Буйе (Martine Bouyer) из французского Центра по атомной энергии вычислили один миллион цифр числа π [см. A. Shibata, *Surikagaku* **20** (1982), 65–73]. Самое поразительное, что за семь лет до этого д-р И. Дж. Мэтрикс (Dr. I. J. Matrix) правильно предсказал, что миллионная цифра числа π должна равняться 5 [Martin Gardner, *New Mathematical Diversions* (Simon and Schuster, 1966), приложение к гл. 8]. Миллиардный барьер был преодолен в 1989 году Григорием В. Чудновским (Gregory V. Chudnovsky), Давидом В. Чудновским* (David V. Chudnovsky) и независимо Ясумаса Канада (Yasumasa Kanada) и Иошиаки Тамура (Yoshiaki Tamura). В 1991 году после 250 часов вычислений на лично разработанном параллельном компьютере Чудновские расширили свой результат до двух миллиардов цифр. [См. Richard Preston, *The New Yorker* **68**, 2 (2 March 1992), 36–67. Новая формула, примененная Чудновскими, описана в *Proc. Nat. Acad. Sci.* **86** (1989), 8178–8182.] В июле 1997 года Ясумаса Канада и Дэисуке Такахаши (Daisuke Takahashi), используя два независимых метода, вычислили более 51.5 миллиардов цифр, что потребовало соответственно 29.0 и 37.1 часов вычислений на компьютере HITACHI SR2201 с 1024 процессорами. Будем ждать новых рекордов в связи с переходом в новое тысячелетие.

В этом разделе мы ограничились методами выполнения арифметических операций, которые используются при программировании компьютеров. Существуют многочисленные алгоритмы для *аппаратной* реализации арифметических операций, которые представляют определенный интерес, но, по-видимому, неприменимы к машинным программам для работы с числами повышенной точности. (См., например, G. W. Reitwiesner, "Binary Arithmetic", *Advances in Computers* **1** (New York: Academic Press, 1960), 231–308; O. L. MacSorley, *Proc. IRE* **49** (1961), 67–91; G. Metze, *IRE Trans.* **EC-11** (1962), 761–764; H. L. Garner, "Number Systems and Arithmetic", *Advances in Computers* **6** (New York: Academic Press, 1965), 131–194.)

---

* Г. В. Чудновский и Д. В. Чудновский — выпускники Киевского университета. — *Прим. ред.*

В статье А. Эдельмана (A. Edelman), опубликованной в журнале *SIAM Review* **39** (1997), 54–67, описан неизвестный, но очень поучительный "прокол" в чипе Pentium разработки 1994 года, связанный с реализацией программы деления. Минимально достижимое при аппаратной реализации время выполнения операций сложения и умножения исследовалось в работах S. Winograd, *JACM* **12** (1965), 277–285, **14** (1967), 793–802, R. P. Brent, *IEEE Trans.* **C-19** (1970), 758–759, и R. W. Floyd, *FOCS* **16** (1975), 3–5 (см. также раздел 4.3.3E).

УПРАЖНЕНИЯ

1. [*42*] Изучите раннюю историю классических алгоритмов выполнения арифметических операций по оригинальным произведениям, скажем, Сунь Цю, аль-Хорезми, аль-Уклидиси, Фибоначчи (Fibonacci) и Роберта Рекорде (Robert Recorde), и как можно точнее перескажите их методы на строгом языке алгоритмов.

2. [*15*] Обобщите алгоритм A таким образом, чтобы он осуществлял сложение в столбик, вычисляя суммы m неотрицательных n-разрядных целых чисел. (Предположите, что $m \leq b$.)

3. [*21*] Разработайте MIX-программу, реализующую алгоритм из упр. 2, и оцените время ее выполнения как функцию от m и n.

▶ **4.** [*M21*] Приведите формальное доказательство корректности алгоритма A, основываясь на методе индуктивных утверждений, описанном в разделе 1.2.1.

5. [*21*] Алгоритм A выполняет сложение двух вводимых чисел, двигаясь справа налево, но иногда данные более доступны для считывания слева направо. Разработайте алгоритм, который выдает тот же ответ, что и алгоритм A, но порождает разряды результата слева направо и, если происходит перенос, делающий предыдущее значение неверным, возвращается назад, чтобы изменить предыдущее значение. [*Замечание.* В ранних индусских и арабских манускриптах, посвященных сложению слева направо, используется именно такой способ сложения. Вероятно, причиной тому послужила ориентация алгоритма A на выполнение операций слева направо на абаках. Алгоритм сложения справа налево стал логическим продолжением этого алгоритма благодаря работам аль-Уклидиси, возможно, потому, что арабы пишут справа налево.]

▶ **6.** [*22*] Разработайте алгоритм, который выполняет сложение слева направо (как в упр. 5), но никогда не запоминает разряд результата, пока существует возможность его изменения из-за последующих переносов. Уже занесенные значения результата не должны изменяться после запоминания очередного значения любого разряда. [*Указание.* Следите за количеством последовательных разрядов $(b-1)$, которые еще не хранятся в результате.] Алгоритм такого вида удобен, например, в ситуации, если вводимые и выводимые числа считываются и записываются на магнитную ленту или если они появляются в простом линейном списке.

7. [*M26*] Определите среднее число случаев, когда выполнение алгоритма из упр. 5 приведет к необходимости возврата при переносе и изменению k разрядов частичного ответа для $k = 1, 2, \ldots, n$. (Предполагается, что оба входных числа имеют независимые и равномерные распределения на интервале 0 и $b^n - 1$.)

8. [*M26*] Разработайте программу для MIX, реализующую алгоритм из упр. 5, и определите среднее время ее работы, исходя из ожидаемого числа переносов, которое подсчитано в тексте.

▶ **9.** [*21*] Обобщите алгоритм A так, чтобы получившийся алгоритм складывал два n-разрядных числа в системе счисления со *смешанным основанием* $b_0, b_1, \ldots$ (справа налево).

Таким образом, наименее значимый разряд расположен в интервале от 0 до $b_1 - 1$ и т. д. (см. формулу 4.1–(9)).

10. [*18*] Будет ли правильно работать программа B, если поменять местами команды в строках 06 и 07, а также в строках 05 и 06?

11. [*10*] Разработайте алгоритм сравнения двух неотрицательных n-разрядных целых чисел $u = (u_{n-1} \ldots u_1 u_0)_b$ и $v = (v_{n-1} \ldots v_1 v_0)_b$, который определяет, какое из неравенств, $u < v$, $u = v$ или $u > v$, выполняется.

12. [*16*] В алгоритме S предполагается, что заранее известно, какой из двух исходных операндов больше. Даже если информация отсутствует, все равно можно начать и так или иначе выполнить вычитание, но при этом обнаружится, что лишнее "заимствование" сохранилось до самого конца алгоритма. Постройте другой алгоритм, в котором можно было бы использовать (если в конце работы алгоритма S имеется "заимствование") дополнение $(w_{n-1} \ldots w_1 w_0)_b$ и поэтому вычислять абсолютное значение разности между u и v.

13. [*21*] Разработайте программу для MIX, которая умножает $(u_{n-1} \ldots u_1 u_0)_b$ на v, где v — произвольное число, представляемое с однократной точностью (т. е. $0 \le v < b$), и получает в результате $(w_n \ldots w_1 w_0)_b$. Каково время ее выполнения?

▶ **14.** [*M22*] Приведите формальное доказательство корректности алгоритма M, основываясь на методе индуктивных утверждений, который описан в разделе 1.2.1 (см. упр. 4).

15. [*M20*] Если необходимо сформировать произведение двух n-разрядных дробных частей чисел $(.u_1 u_2 \ldots u_n)_b \times (.v_1 v_2 \ldots v_n)_b$, а в качестве результата получить только n-разрядное приближение $(.w_1 w_2 \ldots w_n)_b$, можно при помощи алгоритма M вычислить $2n$-разрядный результат, который затем округлить до требуемого приближения. Но на это потребуется вдвое больше вычислительных затрат, чем необходимо для достижения приемлемой точности, так как учет произведения $u_i v_j$ при $i + j > n + 2$ оказывает лишь незначительное влияние на результат.

Оцените максимальную ошибку, которая может возникнуть, если произведения $u_i v_j$ при $i + j > n + 2$ в ходе умножения будут полагаться равными нулю вместо того, чтобы вычисляться.

▶ **16.** [*20*] (*Короткое деление.*) Постройте алгоритм деления неотрицательного n-разрядного целого числа $(u_{n-1} \ldots u_1 u_0)_b$ на v, где v — целое число, заданное с однократной точностью (т. е. $0 < v < b$), получив частное $(w_{n-1} \ldots w_1 w_0)_b$ и остаток r.

17. [*M20*] В обозначениях рис. 6 предположим, что $v_{n-1} \ge \lfloor b/2 \rfloor$. Покажите, что если $u_n = v_{n-1}$, то должно выполняться либо равенство $q = b - 1$, либо равенство $q = b - 2$.

18. [*M20*] В обозначениях рис. 6 покажите, что если $q' = \lfloor (u_n b + u_{n-1})/(v_{n-1} + 1) \rfloor$, то $q' \le q$.

▶ **19.** [*M21*] В обозначениях рис. 6 пусть $\hat{q}$ — некоторое приближение к q и $\hat{r} = u_n b + u_{n-1} - \hat{q} v_{n-1}$. Предположим, что $v_{n-1} > 0$. Покажите, что если $\hat{q} v_{n-2} > b\hat{r} + u_{n-2}$, то $q < \hat{q}$. [*Указание.* Усовершенствуйте доказательство теоремы A, исследовав влияние величины v_{n-2}.]

20. [*M22*] Используя обозначения и предположения из упр. 19, покажите, что если $\hat{q} v_{n-2} \le b\hat{r} + u_{n-2}$, то $\hat{q} = q$ или $q = \hat{q} - 1$.

▶ **21.** [*M23*] В обозначениях из упр. 19 и 20 покажите, что если $v_{n-1} \ge \lfloor b/2 \rfloor$ и $\hat{q} v_{n-2} \le b\hat{r} + u_{n-2}$, но $\hat{q} \ne q$, то $u \bmod v \ge (1 - 2/b)v$. (Последнее событие происходит с вероятностью, приблизительно равной $2/b$, так что если b есть размер машинного слова, то в алгоритме D (за крайне редкими исключениями) должно выполняться равенство $q_j = \hat{q}$.)

▶ **22.** [*24*] Приведите пример деления четырехразрядного числа на трехразрядное, для которого необходимо включение в алгоритм D шага D6, если основание системы счисления — 10.

23. [*M23*] Докажите, что для заданных целых чисел v и u, таких, что $1 \le v < b$, всегда выполняются неравенства $\lfloor b/2 \rfloor \le v\lfloor b/(v+1) \rfloor < (v+1)\lfloor b/(v+1) \rfloor \le b$.

24. [*M20*] Используя закон распределения наиболее значимых разрядов, описанный в разделе 4.2.4, получите приближенную формулу вероятности того, что $d = 1$ в алгоритме D. (При $d = 1$ можно, разумеется, опустить бóльшую часть вычислений на шагах D1 и D8.)

25. [*26*] Напишите программу для MIX, реализующую шаг D1 алгоритма D, что необходимо для завершения программы D.

26. [*21*] Напишите программу для MIX, реализующую шаг D8 алгоритма D, что необходимо для завершения программы D.

27. [*M20*] Докажите, что в начале шага D8 алгоритма D ненормализованный остаток $(u_{n-1} \ldots u_1 u_0)_b$ всегда является точным кратным d.

28. [*M30*] (A. Svoboda, *Stroje na Zpracování Informací* **9** (1963), 25–32.) Обозначим $v = (v_{n-1} \ldots v_1 v_0)_b$ для любого целого основания b при $v_{n-1} \ne 0$. Выполним следующие действия.

N1. Если $v_{n-1} < b/2$, умножим v на $\lfloor (b+1)/(v_{n-1}+1) \rfloor$. Обозначим результат этого шага через $(v_n v_{n-1} \ldots v_1 v_0)_b$.

N2. Если $v_n = 0$, присвоим $v \leftarrow v + (1/b)\lfloor b(b - v_{n-1})/(v_{n-1}+1) \rfloor v$, и пусть результатом этого шага будет $(v_n v_{n-1} \ldots v_0 . v_- \ldots)_b$. Будем повторять шаг N2 до тех пор, пока не получим $v_n \ne 0$.

Докажите, что шаг N2 выполнится не более трех раз и что в конце вычислений всегда будет $v_n = 1$ и $v_{n-1} = 0$.

[*Замечание.* Если оба числа u и v умножить на указанные выше константы, значение частного u/v не изменится, а делитель примет вид $(10 v_{n-2} \ldots v_0 . v_{-1} v_{-2} v_{-3})_b$. Такой вид делителя очень удобен, так как в обозначениях алгоритма D в начале шага D3, если $(u_{j+n+1}, u_{j+n}) = (1, 0)$, в качестве пробного делителя берется $\hat{q} = u_{j+n}$ или $\hat{q} = b - 1$.]

29. [*15*] Докажите или опровергните следующее утверждение: в начале шага D7 алгоритма D равенство $u_{j+n} = 0$ выполняется всегда.

▸ **30.** [*22*] В случае ограниченного объема памяти при выполнении некоторых алгоритмов, описанных в этом разделе, для ввода и вывода информации желательно отводить одни и те же ячейки памяти. Можно ли при выполнении алгоритма A или S хранить числа w_0, w_1, ..., w_{n-1} и u_0, ..., u_{n-1} или v_0, ..., v_{n-1} в одних и тех же ячейках памяти? Можно ли допустить, чтобы при выполнении алгоритма D значения частного q_0, ..., q_m занимали те же ячейки памяти, что и u_n, ..., u_{m+n}? Допустимо ли перекрытие ячеек памяти, используемых при выполнении алгоритма M для хранения входных и выходных данных?

31. [*28*] Пусть основание системы счисления $b = 3$ и $u = (u_{m+n-1} \ldots u_1 u_0)_3$, $v = (v_{n-1} \ldots v_1 v_0)_3$ — целые числа, заданные в *уравновешенной троичной* системе счисления (см. раздел 4.1), причем $v_{n-1} \ne 0$. Напишите алгоритм деления u на v, вычисляя остаток, абсолютное значение которого не должно превышать $\frac{1}{2}|v|$. Попробуйте найти алгоритм, достаточно эффективный для аппаратной реализации на компьютере со встроенной уравновешенной троичной арифметикой.

32. [*M40*] Предположим, что основание системы $b = 2i$, а числа u и v — комплексные числа, представленные в мнимочетверичной системе счисления. Постройте несколько алгоритмов деления u на v с возможным вычислением остатка и сравните их эффективность.

33. [*M40*] Составьте алгоритм для извлечения квадратного корня, аналогичный алгоритму D и методу извлечения квадратного корня, который используется при вычислениях вручную.

34. [*40*] Разработайте набор машинных подпрограмм для выполнения четырех арифметических операций над произвольными целыми числами без ограничений их величины, но с учетом ограничения общего объема оперативной памяти компьютера. (Используйте связное распределение памяти так, чтобы время на поиск места для хранения результата совсем не тратилось.)

35. [*40*] Разработайте набор подпрограмм для "плавающей арифметики десятикратной точности", используя девятиразрядное представление чисел с плавающей точкой по основанию b с избытком 0, где b равно длине машинного слова, и выделяя для порядка полное слово. Таким образом, каждое число в формате с плавающей точкой записывается в 10 словах памяти и общее масштабирование выполняется посредством сдвигов машинных слов целиком вместо сдвигов внутри слов.)

36. [*M25*] Поясните, как с высокой точностью вычислить $\ln \phi$ по заданному с соответствующей точностью приближению числа ϕ, используя только операции сложения и вычитания многократной точности и деления на короткие числа.

▸ **37.** [*20*] (Ю. Саламин (E. Salamin).) Объясните, как в алгоритме D при его реализации на двоичном компьютере запретить нормализацию и денормализацию, не изменяя порядка попыток вычисления разрядов частного, если число d представлено по степеням 2. (Как можно на шаге D3 вычислить $\hat{q}$, если на шаге D1 не была выполнена нормализация?)

38. [*M35*] Предположим, что u и v — целые числа в интервале $0 \le u, v < 2^n$. Предложите способ вычисления среднего геометрического $\lfloor \sqrt{uv} + \frac{1}{2} \rfloor$ при помощи $O(n)$ операций сложения, вычитания и сравнения $(n+2)$-битовых чисел. [*Указание.* Объедините классические методы умножения и извлечения корней, используя "конвейер".]

39. [*25*] (Д. Бейли (D. Bailey), П. Борвейн (P. Borwein) и С. Плуфф (S. Plouffe), 1996.) Поясните, как вычислить n-й бит двоичного представления числа π, не зная предыдущих $n-1$ бит, используя тождество

$$\pi = \sum_{k \ge 0} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right)$$

и выполняя $O(n \log n)$ арифметических операций над $O(\log n)$-битовыми целыми числами. (Полагаем, что двоичные разряды числа π не содержат слишком длинных строк из нулей или единиц.)

40. [*M24*] Иногда возникает необходимость в делении числа u на число v, когда заранее известно, что деление выполнится без остатка. Покажите, что если u есть $2n$-разрядное число и v является n-разрядным числом, таким, что $u \bmod v = 0$, то можно сэкономить около 75% объема вычислений в алгоритме D, вычислив сначала половину частного, начиная слева направо, а затем другую половину — справа налево.

▸ **41.** [*M26*] Во многих приложениях арифметики с высокой точностью требуются повторные вычисления основания n-разрядного числа w, которое изначально было представлено по основанию b. Эти вычисления могут быть ускорены при помощи приема, предложенного Петером Л. Монтгомери (Peter L. Montgomery) [*Math. Comp.* **44** (1985), 519–521]. Он выполнял вычисление остатка справа налево вместо общепринятого направления вычислений слева направо.

a) Для заданных чисел $u = \pm(u_{m+n-1} \ldots u_1 u_0)_b$, $w = (w_{n-1} \ldots w_1 w_0)_b$ и числа w', такого, что $w_0 w' \bmod b = 1$, покажите, как вычислить $v = \pm(v_{n-1} \ldots v_1 v_0)_b$, чтобы выполнялось соотношение $b^m v \bmod w = u \bmod w$.

b) Для заданных n-разрядных целых чисел со знаком u, v и w с $|u|, |v| < w$ и заданного w', как в алгоритме (a), покажите, как вычислить n-разрядное целое число t, такое, что $|t| < w$ и $b^n t \equiv uv$ (по модулю w).

c) Как алгоритмы (a) и (b) упрощают выполнение арифметических операций по модулю w?

42. [*HM35*] Обозначим через P_{nk} вероятность того, что для заданных m и b выполняется $\lfloor (u_1 + \cdots + u_m)/b^n \rfloor = k$, где $u_1, \ldots, u_m$ — случайные n-разрядные целые числа, представленные в системе счисления по основанию b. (Это распределение события w_n в алгоритме сложения в столбик из упр. 2.) Покажите, что $P_{nk} = \frac{1}{m!}\left\langle {m \atop k} \right\rangle + O(b^{-n})$, где $\left\langle {m \atop k} \right\rangle$ есть число Эйлера (см. раздел 5.1.3).

▶ **43.** [*22*] Оттенки серого цвета или компоненты цветовой гаммы в цифровом виде обычно представляются 8-битовыми числами u в интервале $[0\,..\,255]$, выраженными дробью $u/255$. По двум таким дробям $u/255$ и $v/255$ часто используемые алгоритмы графики приближенно вычисляют $w/255$, как ближайшее целое к $uv/255$. Докажите, что w может быть вычислено по формуле

$$t = uv + 128, \qquad w = \lfloor (\lfloor t/256 \rfloor + t)/256 \rfloor.$$

*4.3.2. Модулярная арифметика

Еще один интересный метод выполнения арифметических действий над большими целыми числами основан на простых положениях теории чисел. Идея этого метода состоит в том, чтобы оперировать не непосредственно числом u, а его "остатками" $u \bmod m_1$, $u \bmod m_2$, $\ldots$, $u \bmod m_r$, где $m_1, m_2, \ldots, m_r$ — "модули", не содержащие общих делителей (т. е. они взаимно просты). Примем для удобства везде в этом разделе следующие обозначения:

$$u_1 = u \bmod m_1, \qquad u_2 = u \bmod m_2, \qquad \ldots, \qquad u_r = u \bmod m_r. \tag{1}$$

Числа $(u_1, u_2, \ldots, u_r)$ можно легко вычислить путем деления числа u на простые целые числа v_k. Важно отметить, что при этом нет никакой потери информации (при условии, что u не очень велико), так как всегда, зная $(u_1, u_2, \ldots, u_r)$, можно восстановить u. Например, если $0 \le u < v \le 1000$, нельзя получить ($u \bmod 7$, $u \bmod 11$, $u \bmod 13$), которое равнялось бы ($v \bmod 7$, $v \bmod 11$, $v \bmod 13$). Это следствие китайской теоремы об остатках, рассматриваемой ниже. Поэтому $(u_1, u_2, \ldots, u_r)$ можно рассматривать как новый тип представления в компьютере — "модулярное представление" целого числа u*.

Преимущество модулярного представления заключается в том, что операции сложения, вычитания и умножения выполняются очень просто:

$$(u_1, \ldots, u_r) + (v_1, \ldots, v_r) = \big((u_1 + v_1) \bmod m_1, \ldots, (u_r + v_r) \bmod m_r\big), \tag{2}$$

$$(u_1, \ldots, u_r) - (v_1, \ldots, v_r) = \big((u_1 - v_1) \bmod m_1, \ldots, (u_r - v_r) \bmod m_r\big), \tag{3}$$

$$(u_1, \ldots, u_r) \times (v_1, \ldots, v_r) = \big((u_1 \times v_1) \bmod m_1, \ldots, (u_r \times v_r) \bmod m_r\big). \tag{4}$$

Для доказательства, к примеру, (4) достаточно показать, что для каждого модуля m_j выполняется равенство

$$uv \bmod m_j = (u \bmod m_j)(v \bmod m_j) \bmod m_j.$$

---

* В литературе на русском языке используется аналогичный по смыслу термин "представление в остаточных классах". Далее в переводе будет использована терминология оригинала. — *Прим. перев.*

Это равенство следует из основного положения элементарной теории чисел: $x \bmod m_j = y \bmod m_j$ тогда и только тогда, когда $x \equiv y$ (по модулю m_j). Далее, если $x \equiv x'$ и $y \equiv y'$, то $xy \equiv x'y'$ (по модулю m_j); отсюда следует $(u \bmod m_j)(v \bmod m_j) \equiv uv$ (по модулю m_j).

Основной недостаток модулярного представления состоит в том, что не так просто проверить, является ли $(u_1, \ldots, u_r)$ бóльшим, чем $(v_1, \ldots, v_r)$. Трудно также проверить, возникло ли переполнение в результате выполнения операций сложения, вычитания или умножения, но еще сложнее выполнять операцию деления. При выполнении такого рода операций в сочетании с операциями сложения, вычитания и умножения применение модулярной арифметики оправдано лишь в том случае, если имеются средства быстрого перехода к модулярному представлению и обратно. По этой причине переход от модулярного представления к позиционным системам счисления и наоборот является одной из основных тем данного раздела.

Операции сложения, вычитания и умножения, основанные на формулах (2)–(4), называются арифметикой остатков или *модулярной арифметикой*. Область чисел, над которыми можно выполнять операции модулярной арифметики, — это $m = m_1 m_2 \ldots m_r$ (произведение модулей). Если каждое из m_j близко к размеру машинного слова, можно оперировать n-разрядными числами, когда $r \approx n$. Отсюда следует, что при использовании модулярной арифметики общее время, затрачиваемое на выполнение операций сложения, вычитания и умножения с n-разрядными числами, по существу, пропорционально n (не учитывая время, затрачиваемое на переход к модулярному представлению и обратно). При выполнении операций сложения и вычитания применение модулярной арифметики не дает никаких преимуществ, однако при умножении может быть получена значительная выгода, поскольку время выполнения операции умножения традиционным методом, описанным в разделе 4.3.1, пропорционально n^2.

Кроме того, на компьютере, в котором предусмотрена возможность параллельного выполнения операций, применение модулярной арифметики дает значительное преимущество даже для сложения и вычитания. Операции, связанные с разными модулями, могут выполняться одновременно, так что получается реальное повышение скорости их выполнения. Поскольку для традиционного метода, описанного в предыдущем разделе, требуется выполнение переноса разрядов, традиционный метод не позволяет достичь подобного сокращения времени реализации арифметических операций. Возможно, когда-нибудь компьютеры с высокой степенью параллелизма позволят сделать одновременное выполнение операций обычным делом, и тогда модулярная арифметика приобретет важное значение для вычислений в реальном времени, т. е. в условиях, когда для конкретной задачи требуется дать быстрый ответ с высокой точностью. (При работе на компьютерах с высокой степенью параллелизма часто предпочтительнее одновременно выполнять k *отдельных* программ вместо того, чтобы выполнять *одну* программу в k раз быстрее, поскольку последний подход сложнее в реализации, но никакого повышения эффективности использования компьютера не дает. Вычисления в реальном времени являются тем исключением, при котором параллелизм, присущий модулярной арифметике, приобретает большое значение.)

Теперь проанализируем фундаментальное положение, лежащее в основе модулярного представления чисел.

Теорема C (*Китайская теорема об остатках*). *Пусть* $m_1, m_2, \ldots, m_r$ — положительные целые попарно простые числа, т. е.

$$m_j \perp m_k \qquad \text{при } j \neq k. \tag{5}$$

Пусть $m = m_1 m_2 \ldots m_r$ *и пусть также* a, u_1, u_2, $\ldots$, u_r — *целые числа. Тогда существует ровно одно целое число* u, *удовлетворяющее условиям*

$$a \le u < a + m \qquad \text{и} \qquad u \equiv u_j \ (\text{по модулю } m_j) \quad \text{при } 1 \le j \le r. \tag{6}$$

Доказательство. Если $u \equiv v$ (по модулю m_j) при $1 \le j \le r$, то $u - v$ кратно m_j для всех j. Тогда из условия (5) следует, что $u - v$ кратно $m = m_1 m_2 \ldots m_r$. Значит, уравнение (6) имеет *не более* одного решения. Для завершения доказательства необходимо показать, что существует *по меньшей мере* одно решение. Это можно сделать двумя простыми способами.

Способ 1 ("Неконструктивное" доказательство). Так как u принимает m различных значений $a \le u < a + m$, наборы из r чисел $(u \bmod m_1, \ldots, u \bmod m_r)$ также должны принимать m различных значений (поскольку уравнение (6) имеет не более одного решения). Но имеется ровно $m_1 m_2 \ldots m_r$ возможных r-наборов $(v_1, \ldots, v_r)$, таких, что $0 \le v_j < m_j$. Поэтому каждый r-набор должен встречаться точно один раз и должно существовать такое значение u, для которого $(u \bmod m_1, \ldots$ $\ldots, u \bmod m_r) = (u_1, \ldots, u_r)$.

Способ 2 ("Конструктивное" доказательство). Можно найти числа M_j при $1 \le j \le r$, такие, что

$$M_j \equiv 1 \ (\text{по модулю } m_j) \qquad \text{и} \qquad M_j \equiv 0 \ (\text{по модулю } m_k) \ \text{ для } \ k \neq j. \tag{7}$$

Действительно, из (5) следует, что m_j и m/m_j взаимно просты, а потому согласно теореме Эйлера (упр. 1.2.4–28) можно взять

$$M_j = (m/m_j)^{\varphi(m_j)}. \tag{8}$$

Теперь число

$$u = a + \big((u_1 M_1 + u_2 M_2 + \cdots + u_r M_r - a) \bmod m\big) \tag{9}$$

удовлетворяет всем условиям (6). ∎

Частный случай этой теоремы был сформулирован китайским математиком Сунь Цу, который предложил правило, названное "тай-йен" ("большое обобщение"). Дата написания его работы точно не установлена; предположительно — между 280 и 473 г. н. э. Дальнейшее развитие эта задача получила в работах математиков средневековой Индии, предложивших методы *куттака (kuṭṭaka)* (см. раздел 4.5.2). Однако в общем виде теорема C впервые была сформулирована и доказана Чин Чжу-Шао в работе *Shu Shu Chiu Chang* (1247), в которой рассмотрен также случай, когда модули могут иметь общие множители (как в упр. 3). [См. J. Needham, *Science and Civilization in China* **3** (Cambridge University Press, 1959), 33–34, 119–120; Y. Li and S. Du, *Chinese Mathematics* (Oxford: Clarendon, 1987), 92–94, 105, 161–166; K. Shen, *Archive for History of Exact Sciences* **38** (1988), 285–305.] Многочисленные исследования, посвященные этой теории, обобщены Л. Ю. Диксоном (L. E. Dickson) в книге *History of the Theory of Numbers* **2** (Carnegie Inst. of Washington, 1920), 57–64.

Как следует из теоремы C, модулярное представление можно использовать для чисел в любом соответствующем интервале $m = m_1 m_2 \ldots m_r$ целых чисел. Например, можно в (6) взять $a = 0$ и работать только с неотрицательными целыми числами u, меньшими m. С другой стороны, при выполнении операций сложения и вычитания, так же, как и умножения, обычно удобнее всего предположить, что все модули $m_1, m_2, \ldots, m_r$ являются нечетными числами, так что и $m = m_1 m_2 \ldots m_r$ тоже нечетное, и работать с целыми числами из интервала

$$-\frac{m}{2} < u < \frac{m}{2}, \tag{10}$$

симметричного относительно нуля.

Для выполнения основных операций, перечисленных в (2)–(4), необходимо вычислить $(u_j + v_j) \bmod m_j$, $(u_j - v_j) \bmod m_j$ и $u_j v_j \bmod m_j$ при $0 \le u_j, v_j < m_j$. Если m_j — число однократной точности, то лучше всего формировать $u_j v_j \bmod m_j$ путем умножения и последующего деления. Что касается операций сложения и вычитания, то здесь ситуация проще, так как не требуется деление; удобно рассматривать следующие формулы:

$$(u_j + v_j) \bmod m_j = u_j + v_j - m_j[u_j + v_j \ge m_j]. \tag{11}$$

$$(u_j - v_j) \bmod m_j = u_j - v_j + m_j[u_j < v_j]. \tag{12}$$

(См. раздел 3.2.1.1.) Поскольку желательно, чтобы w было как можно бóльшим, проще всего принять m_1 наибольшим нечетным числом, соответствующим машинному слову, в качестве m_2 принять наибольшее нечетное число $< m_1$, взаимно простое с m_1, а в качестве m_3 — наибольшее нечетное число $< m_2$, взаимно простое как с m_1, так и с m_2, и т. д., пока не наберется столько m_j, сколько будет достаточно для образования нужного m. Способы эффективного определения, являются ли два числа взаимно простыми, рассматриваются в разделе 4.5.2.

В качестве простого примера предположим, что имеется десятичный компьютер со словом, содержащим две цифры, так что размер слова равен 100. Тогда в результате только что описанной процедуры получаем

$$m_1 = 99, \quad m_2 = 97, \quad m_3 = 95, \quad m_4 = 91, \quad m_5 = 89, \quad m_6 = 83 \tag{13}$$

и т. д.

При работе на двоичных компьютерах иногда желательно выбирать модули m_j иным образом:

$$m_j = 2^{e_j} - 1. \tag{14}$$

Другими словами, значение каждого модуля на единицу меньше очередной степени 2. Такой выбор значения модуля m_j зачастую упрощает выполнение основных арифметических операций, ибо выполнять вычисления с числами, представленными по модулю $2^{e_j} - 1$, несколько проще, чем с числами, представленными в обратном коде. После того как значения модулей выбраны таким образом, полезно несколько ослабить условие $0 \le u_j < m_j$ и потребовать только, чтобы

$$0 \le u_j < 2^{e_j}, \qquad u_j \equiv u \ (\text{по модулю } 2^{e_j} - 1). \tag{15}$$

Таким образом, значение $u_j = m_j = 2^{e_j} - 1$ принимается в качестве оптимального вместо $u_j = 0$, поскольку это, с одной стороны, не влияет на справедливость теоремы C, а с другой — означает, что u_j может быть любым e_j-битовым двоичным

числом. При таком допущении операции сложения и вычитания по модулю m_j выполняются следующим образом:

$$u_j \oplus v_j = \big((u_j + v_j) \bmod 2^{e_j}\big) + [u_j + v_j \geq 2^{e_j}]. \tag{16}$$

$$u_j \otimes v_j = (u_j v_j \bmod 2^{e_j}) \ \oplus \ \lfloor u_j v_j / 2^{e_j} \rfloor. \tag{17}$$

(Здесь $\oplus$ и $\otimes$ указывают на действия, которые с учетом условия (15) должны быть выполнены с отдельными компонентами $(u_1, \ldots, u_r)$ и $(v_1, \ldots, v_r)$ при сложении или умножении соответственно.) При вычитании можно пользоваться и соотношением (12). Можно также использовать условие

$$u_j \ominus v_j = \big((u_j - v_j) \bmod 2^{e_j}\big) - [u_j < v_j]. \tag{18}$$

Эти операции могут быть эффективно выполнены, даже если 2^{e_j} больше машинного слова компьютера, так как совсем просто вычислить остаток положительного числа по модулю степени 2 или разделить число на степень 2. В (17) имеем сумму "верхней половины" и "нижней половины" произведения, как в разделе 3.2.1.1–8.

Для работы с модулями вида $2^{e_j} - 1$ необходимо знать, при каких условиях число $2^e - 1$ является взаимно простым с числом $2^f - 1$. К счастью, для этого существует очень простое правило:

$$\gcd(2^e - 1, 2^f - 1) = 2^{\gcd(e,f)} - 1. \tag{19}$$

Данная формула утверждает, в частности, что sl $2^e - 1$ и $2^f - 1$ взаимно просты тогда и только тогда, когда взаимно просты e и f. Уравнение (19) следует из алгоритма Евклида и тождества

$$(2^e - 1) \bmod (2^f - 1) = 2^{e \bmod f} - 1. \tag{20}$$

(См. упр. 6.) Поэтому на компьютере с длиной слова 2^{32} можно выбрать $m_1 = 2^{32} - 1$, $m_2 = 2^{31} - 1$, $m_3 = 2^{29} - 1$, $m_4 = 2^{27} - 1$, $m_5 = 2^{25} - 1$, что обеспечивает эффективность сложения, вычитания и умножения целых чисел в интервале вплоть до $m_1 m_2 m_3 m_4 m_5 > 2^{143}$.

Как мы уже заметили, операции преобразования в модулярное представление и обратно очень важны. Модулярное представление $(u_1, \ldots, u_r)$ для заданного числа u может быть получено посредством деления u на m_1, ..., m_r с запоминанием остатков. В случае, когда $u = (v_m v_{m-1} \ldots v_0)_b$, возможно применение более подходящего способа, который состоит в том, чтобы, используя модулярную арифметику, вычислить полином

$$\big(\ldots (v_m b + v_{m-1}) b + \cdots\big)\, b + v_0.$$

Если $b = 2$ и модули m_j имеют специальный вид $2^{e_j} - 1$, оба подхода сводятся к совсем простому способу. Рассмотрим двоичное представление числа u с блоками e_j бит:

$$u = a_t A^t + a_{t-1} A^{t-1} + \cdots + a_1 A + a_0, \tag{21}$$

где $A = 2^{e_j}$ и $0 \leq a_k < 2^{e_j}$ при $0 \leq k \leq t$. Тогда

$$u \equiv a_t + a_{t-1} + \cdots + a_1 + a_0 \ \text{(по модулю } 2^{e_j} - 1), \tag{22}$$

поскольку $A \equiv 1$. Поэтому u_j вычисляются, как и в (16), путем сложения e_j-битовых чисел $a_t \oplus \cdots \oplus a_1 \oplus a_0$. Этот процесс аналогичен уже знакомому процессу

"выбрасывания девяток", который использовался для определения $u \bmod 9$ в случае, когда u выражалось в десятичной системе.

Обратный переход от модулярного представления к позиционной системе счисления выполняется немного сложнее. В связи с этим интересно отметить, как изучение способов вычисления приводит к пересмотру критериев оценок математических доказательств. В теореме С утверждается, что возможен переход от $(u_1, \ldots, u_r)$ к u, и приводятся два доказательства. Первое из рассмотренных доказательств считается классическим; оно основывается лишь на самых простых понятиях, а именно:

i) любое число, кратное m_1, m_2, ... и m_r, должно быть кратным $m_1 m_2 \ldots m_r$, если числа m_j попарно взаимно просты;

ii) если m предметов поместить в m ящиков так, чтобы ни в каком ящике не было двух предметов одновременно, то в каждом ящике должно быть по одному предмету.

Согласно традиционным понятиям математической эстетики это, несомненно, наилучший способ доказательства теоремы С. Но с точки зрения вычислительной он никуда не годится! Это все равно что сказать: "Попробуйте перебирать $u = a$, $a+1, \ldots$, пока не найдете значение, для которого $u \equiv u_1$ (по модулю m_1), ..., $u \equiv u_r$ (по модулю m_r)".

Второй способ доказательства теоремы С более конкретен. Он показывает, как вычислить r новых констант $M_1, \ldots, M_r$ и получить решение, выражаемое через данные константы, с помощью формулы (9). В этом доказательстве используются более сложные понятия (например, теорема Эйлера), но с вычислительной точки зрения оно гораздо более удовлетворительно, поскольку константы $M_1, \ldots, M_r$ определяются только один раз. С другой стороны, определение констант M_j при помощи уравнения (8) является нетривиальной задачей, так как вычисление Эйлеровой φ-функции в общем случае требует факторизации, т. е. разложения чисел m_j на простые сомножители. Существует много способов вычисления M_j, лучших, чем использование (8). В связи со сказанным можно снова подчеркнуть разницу между математической элегантностью и вычислительной эффективностью. Но даже после нахождения M_j при помощи лучшего из возможных способов можно столкнуться с фактом, что M_j слишком велико. Таким образом, использование (9) приводит к большому числу вычислительных операций с высокой точностью, а именно этого нам хотелось бы избежать прежде всего.

Поэтому, чтобы найти действительно пригодный для практического применения метод перехода от $(u_1, \ldots, u_r)$ к u, необходимо иметь *лучшее* доказательство теоремы С. Такое доказательство предложено в 1958 году Х. Л. Гарнером (H. L. Garner). Оно основано на использовании $\binom{r}{2}$ констант c_{ij} для $1 \le i < j \le r$, где

$$c_{ij}\, m_i \equiv 1 \ \text{(по модулю } m_j). \tag{23}$$

Константы c_{ij} легко вычисляются при помощи алгоритма Евклида, так как алгоритм 4.5.2X для заданных i и j позволяет определить a и b, такие, что $am_i + bm_j = \gcd(m_i, m_j) = 1$, и можно положить $c_{ij} = a$. Простой метод определения c_{ij} для модулей специального вида $2^{e_j} - 1$ приведен в упр. 6.

Так как c_{ij} удовлетворяет условию (23), можно выполнить присвоения

$$\begin{aligned}
&v_1 \leftarrow u_1 \bmod m_1,\\
&v_2 \leftarrow (u_2 - v_1)\, c_{12} \bmod m_2,\\
&v_3 \leftarrow \big((u_3 - v_1)\, c_{13} - v_2\big)\, c_{23} \bmod m_3,\\
&\quad\vdots\\
&v_r \leftarrow \big(\ldots((u_r - v_1)\, c_{1r} - v_2)\, c_{2r} - \cdots - v_{r-1}\big)\, c_{(r-1)r} \bmod m_r.
\end{aligned} \tag{24}$$

Тогда число

$$u = v_r m_{r-1} \ldots m_2 m_1 + \cdots + v_3 m_2 m_1 + v_2 m_1 + v_1 \tag{25}$$

будет удовлетворять условиям

$$0 \le u < m, \qquad u \equiv u_j \text{ (по модулю } m_j) \quad \text{для } 1 \le j \le r. \tag{26}$$

(См. упр. 8; другой способ записи формул (24), не требующий такого большого количества констант, приведен в упр. 7.) Формула (25) — это *представление по смешанному основанию* числа u, которое можно перевести в двоичный либо десятичный формат, используя методы, описанные в разделе 4.4. Если интервал $0 \le u < m$ не является необходимым, то после завершения процесса перевода к нему можно добавить (или вычесть из него) соответствующее число, кратное m. Преимущество метода, представленного в (24), состоит в том, что для вычисления v_j используется только арифметика по модулю m_j, которая уже встроена в алгоритмы этого класса. Более того, соотношения (24) позволяют выполнять вычисления параллельно. Можно начать с присвоения $(v_1, \ldots, v_r) \leftarrow (u_1 \bmod m_1, \ldots, u_r \bmod m_r)$, затем в момент j при $1 \le j < r$ сразу же получить $v_k \leftarrow (v_k - v_j)\, c_{jk} \bmod m_k$ для $j < k \le r$. Другой способ вычисления представления числа по смешанному основанию, обеспечивающий возможность достижения параллелизма, рассматривается в интересной статье А. С. Френкеля (A. S. Fraenkel), опубликованной в журнале *Proc. ACM Nat. Conf.* **19** (Philadelphia, 1964), E1.4. Важно отметить, что представление (25) по смешанному основанию пригодно для сравнения величин двух чисел, заданных в модулярном представлении. Так, если известно, что $0 \le u < m$ и $0 \le u' < m$, можно сказать, будет ли $u < u'$, если сначала выполнить переход к $(v_1, \ldots, v_r)$ и к $(v'_1, \ldots, v'_r)$, а затем в соответствии с лексикографическим правилом проверить выполнение неравенств $v_r < v'_r$ или $v_r = v'_r$ и $v_{r-1} < v'_{r-1}$ и т. д. Нет необходимости переходить к двоичной или десятичной системе счисления, если нужно всего лишь выяснить, будет ли $(u_1, \ldots, u_r)$ меньше, чем $(u'_1, \ldots, u'_r)$.

Операции сравнения двух чисел или определения знака числа при модулярном представлении интуитивно понятны и очень просты, поэтому можно было бы ожидать, что удастся найти значительно более легкий способ выполнения такого сравнения, чем переход к представлению со смешанным основанием. Но из приводимой ниже теоремы следует, что шансов на поиск существенно более легкого метода мало, поскольку величина числа в модулярном представлении существенным образом зависит от всех битов всех остатков $(u_1, \ldots, u_r)$.

Теорема S (Николаш Сабо (Nicholas Szabó), 1961). *Используя введенные выше обозначения, предположим, что* $m_1 < \sqrt{m}$, *и пусть* L *— произвольное значение,*

удовлетворяющее неравенству

$$m_1 \le L \le m - m_1. \tag{27}$$

Пусть g — произвольная функция, такая, что ряд $\{g(0), g(1), \ldots, g(m_1 - 1)\}$ содержит меньше значений, чем m_1. Тогда существуют такие числа u и v, что

$$g(u \bmod m_1) = g(v \bmod m_1), \qquad u \bmod m_j = v \bmod m_j \quad \text{при } 2 \le j \le r; \tag{28}$$

$$0 \le u < L \le v < m. \tag{29}$$

Доказательство. Так как согласно предположению должны существовать числа $u \ne v$, удовлетворяющие (28), g должно принимать одинаковые значения для двух различных остатков. Пусть (u, v) — пара таких значений, удовлетворяющая равенству (28) при $0 \le u < v < m$, для которых u минимально. Поскольку $u' = u - m_1$ и $v' = v - m_1$ также удовлетворяют равенству (28), в силу минимальности u мы должны получить $u' < 0$. Отсюда следует, что $u < m_1 \le L$, и, если неравенство (29) не выполняется, получаем $v < L$. Но $v > u$, а $v - u$ кратно $m_2 \ldots m_r = m/m_1$, так что $v \ge v - u \ge m/m_1 > m_1$. Поэтому, если неравенство (29) для (u, v) не выполняется, оно будет удовлетворяться для пары $(u'', v'') = (v - m_1,\ u + m - m_1)$. ▮

Разумеется, подобный результат может быть доказан для любого m_j вместо m_1; можно было бы также заменить неравенство (29) условием $a \le u < a + L \le v < a + m$, внеся при этом незначительные изменения в доказательство. Таким образом, теорема S показывает, что многие простые функции не могут использоваться для определения области, которой принадлежит число в модулярном представлении.

Напомним теперь главные моменты, которые рассматривались в этом разделе. Применение модулярной арифметики может дать значительное преимущество в приложениях, в которых основная доля вычислений приходится на точное умножение (или возведение в степень) больших чисел в сочетании со сложением и вычитанием, но в которых очень редко появляется необходимость в делении либо сравнении чисел *или не нужно проверять, не "выходят" ли промежуточные результаты за пределы области.* (Важно не забывать последней оговорки; существуют методы проверки принадлежности числа данной области (один из них рассмотрен в упр. 12), но они настолько сложны, что сводят на нет все преимущества модулярной арифметики.) Некоторые приложения вычислений с применением модулярной арифметики рассмотрены Х. Такахаси (H. Takahasi) и Й. Ишибаши (Y. Ishibashi) в работе, опубликованной в *Information Proc. in Japan* **1** (1961), 28–42.

Примером такого приложения является точное решение линейных уравнений с рациональными коэффициентами. По различным причинам в этом случае удобно предположить, что модули $m_1, m_2, \ldots, m_r$ являются простыми числами; линейные уравнения могут решаться независимо по каждому модулю m_j. Эта процедура была подробно исследована И. Борошем (I. Borosh) и А. С. Френкелем (A. S. Fraenkel) [*Math. Comp.* **20** (1966), 107–112] и в дальнейшем усовершенствована А. С. Френкелем и Д. Левенталем (D. Loewenthal) [*J. Res. National Bureau of Standards* **75B** (1971), 67–75]. При помощи этого метода на компьютере CDC 1604 меньше чем за 20 мин было получено 9 независимых решений системы из 111 линейных уравнений со 120-ю неизвестными. Та же процедура полезна и для решения систем линейных

уравнений, когда коэффициенты представлены в формате с плавающей точкой, а матрица коэффициентов плохо обусловлена. В таком виде коэффициенты трактуются как точные рациональные числа. Модульный способ дает более быстрый метод вычисления *истинных* результатов быстрее, чем традиционные методы могут дать *приближенный* ответ! Последующие достижения в этой области опубликованы в работе M. T. McClellan, *JACM* **20** (1973), 563–588. Исследования, посвященные ограничениям в применении этого метода, опубликованы в работе E. H. Bareiss, *J. Inst. Math. and Appl.* **10** (1972), 68–104.

Опубликованная литература по модулярной арифметике ориентирована, главным образом, на разработку компьютеров, так как свойства свободного переноса модулярной арифметики делают ее привлекательной с точки зрения быстродействующих операций. Эта идея впервые была опубликована А. Свободой (A. Svoboda) и М. Валахом (M. Valach) в чехословацком журнале *Stroje na Zpracování Informací* (*Information Processing Machines*) **3** (1955), 247–295, а затем независимо — Х. Л. Гарнером (H. L. Garner) [*IRE Trans.* **EC-8** (1959), 140–147]. Использование модулей $2^{e_j}-1$ было предложено А. С. Френкелем в работе *JACM***8** (1961), 87–96, а некоторые преимущества модулей такого вида продемонстрированы А. Шёнхаге (A. Schönhage) [*Computing* **1** (1966), 182–196]. Дополнительную информацию и исчерпывающую библиографию по этому вопросу можно найти в книге N. S. Szabó, R. I. Tanaka *Residue Arithmetic and its Applications to Computer Technology* (New York: McGraw-Hill, 1967). В 1968 году опубликована книга И. Я. Акинского и Д. И. Юдицкого, в которую включена глава, посвященная комплексным модулям [см. *Rev. Roumaine de Math. Pures et Appl.* **15** (1970), 159–160]*. Обсуждение модулярной арифметики будет продолжено в разделе 4.3.3В.

Сообщение на доске объявлений гласило, что он находится в комнате 423, но при взгляде на план системы нумерации, с виду логичной, складывалось впечатление, будто ее разрабатывал либо лунатик, либо математик.

— РОБЕРТ БЭРНАРД (ROBERT BARNARD), *The Case of the Missing Brontë* (1983)

УПРАЖНЕНИЯ

1. [*20*] Найдите все целые числа u, удовлетворяющие всем следующим условиям: $u \bmod 7 = 1$, $u \bmod 11 = 6$, $u \bmod 13 = 5$, $0 \le u < 1000$.

2. [*M20*] Будет ли теорема С по-прежнему справедливой, если допустить, что $a, u_1, u_2, \ldots, u_r$ и u — произвольные вещественные числа (а не только целые)?

▶ **3.** [*M26*] (*Обобщенная китайская теорема об остатках.*) Пусть $m_1, m_2, \ldots, m_r$ — положительные целые числа, m — наименьшее общее кратное чисел $m_1, m_2, \ldots, m_r$ и пусть $a, u_1, u_2, \ldots, u_r$ — произвольные целые числа. Докажите, что имеется точно одно целое число u, удовлетворяющее соотношениям

$$a \le u < a + m, \qquad u \equiv u_j \pmod{m_j}, \qquad 1 \le j \le r,$$

при условии, что

$$u_i \equiv u_j \pmod{\gcd(m_i, m_j)}, \qquad 1 \le i < j \le r,$$

и что если последнее условие не выполняется, то такого целого числа u не существует.

---

* См. Акинский И. Я. и Юдицкий Д. И. *Машинная арифметика в остаточных классах.* — М.: Сов. радио, 1968. — 440 с. (1970). — *Прим. перев.*

4. [*20*] Продолжите процесс, представленный в (13). Чему будут равны m_7, m_8, m_9, ...?

▶ **5.** [*M23*] Предположим, что метод, определенный в (13), продолжен до того момента, когда уже нельзя выбрать ни одного нового числа m_j. Получим ли мы этим "вульгарным" методом наибольшее возможное значение произведения $m_1 m_2 \dots m_r$, такое, что все m_j будут положительными целыми числами, меньшими 100, и попарно взаимно простыми?

6. [*M22*] Пусть e, f и g — неотрицательные целые числа.

a) Покажите, что $2^e \equiv 2^f$ (по модулю $2^g - 1$) в том и только в том случае, когда $e \equiv f$ (по модулю g).

b) Для заданных $e \bmod f = d$ и $ce \bmod f = 1$ докажите тождество

$$((1 + 2^d + \cdots + 2^{(c-1)d}) \cdot (2^e - 1)) \bmod (2^f - 1) = 1.$$

(Таким образом, получена сравнительно простая формула для величины, обратной $2^e - 1$, по модулю $2^f - 1$, как это и требуется в (23).)

▶ **7.** [*M21*] Покажите, что (24) можно переписать следующим образом:

$$\begin{aligned}
v_1 &\leftarrow u_1 \bmod m_1,\\
v_2 &\leftarrow (u_2 - v_1)\, c_{12} \bmod m_2,\\
v_3 &\leftarrow (u_3 - (v_1 + m_1 v_2))\, c_{13} c_{23} \bmod m_3,\\
&\vdots\\
v_r &\leftarrow (u_r - (v_1 + m_1(v_2 + m_2(v_3 + \cdots + m_{r-2} v_{r-1}) \dots)))\, c_{1r} \dots c_{(r-1)r} \bmod m_r.
\end{aligned}$$

Если это сделать, то вместо $r(r-1)/2$ констант c_{ij}, как в (24), потребуется только $r - 1$ констант $C_j = c_{1j} \dots c_{(j-1)j} \bmod m_j$. Оцените с точки зрения вычислений на компьютере достоинства и недостатки настоящего варианта формулы (24) по сравнению с ее исходным вариантом.

8. [*M21*] Докажите, что число u, определенное формулами (24) и (25), удовлетворяет условию (26).

9. [*M20*] Покажите, как перейти от значений $v_1, \dots, v_r$, фигурирующих в уравнении (25) и представленных в нем в системе со смешанным основанием, обратно к исходным остаткам $u_1, \dots, u_r$, используя для вычислений значений u_j только арифметические операции вычисления остатка по модулю m_j.

10. [*M25*] Целое число u, принадлежащее симметричной области (10), можно было бы представить при помощи чисел $u_1, \dots, u_r$, таких, что $u \equiv u_j$ (по модулю m_j), удовлетворяющих условию $-m_j/2 < u_j < m_j/2$ вместо условия $0 \le u_j < m_j$, как указано в тексте раздела. Рассмотрите процедуры модулярной арифметики для такого симметричного представления, включая процедуру перевода (24).

11. [*M23*] Допустим, все числа m_j — нечетные и известно, что $u = (u_1, \dots, u_r)$ — четное число при $0 \le u < m$. Используя модулярную арифметику, найдите достаточно быстрый способ вычисления $u/2$.

Обратите внимание на тождество $2t \cdot \frac{m+1}{2} \equiv t$ (по модулю m). В общем случае, если v и m являются взаимно простыми, можно найти (по алгоритму Евклида) число $v' = (v'_1, \dots, v'_r)$, такое, что $vv' \equiv 1$ (по модулю m). Далее, если известно, что u кратно v, то $u/v = uv'$ вычисляем при помощи модулярного умножения. Если v не является взаимно простым с m, то деление выполняется значительно сложнее.

12. [*M10*] Докажите, что, если $0 \le u, v < m$, модулярное сложение чисел u и v приведет к переполнению (т. е. полученное число будет находиться за пределами допустимой области) тогда и только тогда, когда сумма меньше числа u. (Таким образом, проблема обнаружения переполнения эквивалентна проблеме сравнения.)

▸ **13.** [*M25*] (*Автоморфы.*) Десятичное n-разрядное число $x > 1$ математиками-шутниками называется автоморфом, если последние n цифр числа x^2 равны x. К примеру, 9 376 есть 4-разрядный автоморф, так как $9\,376^2 = 87\,909\,376$. [См. *Scientific American* **218** (January, 1968), 125.]

a) Докажите, что n-разрядное число $x > 1$ есть автоморф тогда и только тогда, когда $x \bmod 5^n = 0$ (или 1) и $x \bmod 2^n = 1$ (или 0) соответственно. (Таким образом, если $m_1 = 2^n$ и $m_2 = 5^n$, то в (7) только числа M_1 и M_2 являются n-разрядными автоморфами.)

b) Докажите, что если x есть n-разрядный автоморф, то $(3x^2 - 2x^3) \bmod 10^{2n}$ является $2n$-разрядным автоморфом.

c) Пусть известно, что $cx \equiv 1$ (по модулю y). Найдите простую формулу для числа c', являющегося функцией c и x, но не y, которая (формула) имеет вид $c'x^2 \equiv 1$ (по модулю y^2).

▸ **14.** [*M30*] (*Мерзкое умножение.*) По определению циклическая свертка $(x_0, x_1, \ldots, x_{n-1})$ и $(y_0, y_1, \ldots, y_{n-1})$ есть $(z_0, z_1, \ldots, z_{n-1})$, где

$$z_k = \sum_{i+j \equiv k \text{ (по модулю } n)} x_i y_j \qquad \text{для } 0 \le k < n.$$

Эффективные алгоритмы для циклической свертки будут рассмотрены в разделах 4.3.3 и 4.6.4.

Рассмотрим q-битовые целые числа u и v, которые представлены в виде

$$u = \sum_{k=0}^{n-1} u_k 2^{\lfloor kq/n \rfloor}, \qquad v = \sum_{k=0}^{n-1} v_k 2^{\lfloor kq/n \rfloor},$$

где $0 \le u_k, v_k < 2^{\lfloor (k+1)q/n \rfloor - \lfloor kq/n \rfloor}$. (Такое представление является смесью оснований $2^{\lfloor q/n \rfloor}$ и $2^{\lceil q/n \rceil}$.) Используя подходящую циклическую свертку, предложите хороший способ поиска представления числа

$$w = (uv) \bmod (2^q - 1).$$

[*Указание.* Не бойтесь использовать вычисления в формате с плавающей точкой.]

*4.3.3. Насколько быстро можно выполнять умножение

Для умножения m-разрядного числа на n-разрядное традиционным методом (алгоритм 4.3.1M) требуется приблизительно cmn операций, где c — константа. Для простоты в этом разделе предположим, что $m = n$, и обсудим следующий вопрос: *Для любого ли обычного вычислительного алгоритма умножения двух n-разрядных чисел время выполнения пропорционально n^2 по мере увеличения n?* (В этом вопросе под термином "обычный" понимается алгоритм, воспринимающий в качестве входа число n и два произвольных n-разрядных числа в позиционной интерпретации и на выходе дающий произведение этих чисел также в позиционной интерпретации. Безусловно, если бы можно было для каждого значения n выбрать свой алгоритм, вопрос не представлял бы интереса, так как для любого конкретного значения n умножение можно было бы выполнить, просто отыскав результат в некоторой огромной таблице. Под термином "вычислительный алгоритм" понимается алгоритм, пригодный для применения на цифровом компьютере, подобном MIX, а время выполнения — это время, затраченное на таком компьютере на получение результата по такому алгоритму.)

А. Цифровые методы. Ответ на поставленный вопрос звучит довольно неожиданно — "Нет". И в самом деле, нетрудно понять, почему. Для удобства будем далее полагать, что мы оперируем числами, выраженными в двоичной системе счисления. Два $2n$-битовых числа $u = (u_{2n-1} \ldots u_1 u_0)_2$ и $v = (v_{2n-1} \ldots v_1 v_0)_2$ можно записать в виде

$$u = 2^n U_1 + U_0, \qquad v = 2^n V_1 + V_0, \tag{1}$$

где $U_1 = (u_{2n-1} \ldots u_n)_2$ — "наиболее значимая половина" числа u и $U_0 = (u_{n-1} \ldots u_0)_2$ — "наименее значимая половина" числа u. Аналогично $V_1 = (v_{2n-1} \ldots v_n)_2$ и $V_0 = (v_{n-1} \ldots v_0)_2$. Тогда

$$uv = (2^{2n} + 2^n) U_1 V_1 + 2^n (U_1 - U_0)(V_0 - V_1) + (2^n + 1) U_0 V_0. \tag{2}$$

Эта формула сводит задачу умножения $2n$-битовых чисел к трем операциям умножения n-битовых чисел $U_1 V_1$, $(U_1 - U_0)(V_0 - V_1)$ и $U_0 V_0$ и выполнению некоторых простых операций сдвига и сложения. Формула (2) пригодна и для умножения чисел с удвоенной точностью, когда требуется получить результат с учетверенной точностью. На многих компьютерах это реализуется немного быстрее, чем умножение традиционными методами. Но главное преимущество формулы (2) заключается в том, что ее можно использовать для определения рекурсивного процесса умножения, который значительно быстрее при больших n уже знакомого метода, имеющего время выполнения порядка n^2. Если $T(n)$ — время, затрачиваемое на выполнение умножения n-битовых чисел, то для некоторой константы c имеем

$$T(2n) \le 3T(n) + cn, \tag{3}$$

так как в правой части формулы (2) требуется выполнить только три операции умножения плюс некоторые операции сдвига и сложения. Из соотношения (3) по индукции следует, что

$$T(2^k) \le c(3^k - 2^k), \qquad k \ge 1, \tag{4}$$

если выбрать константу c достаточно большой, чтобы данное неравенство выполнялось при $k = 1$; поэтому имеем

$$T(n) \le T\big(2^{\lceil \lg n \rceil}\big) \le c\big(3^{\lceil \lg n \rceil} - 2^{\lceil \lg n \rceil}\big) < 3c \cdot 3^{\lg n} = 3cn^{\lg 3}. \tag{5}$$

Из соотношения (5) видно, что время порядка n^2, затрачиваемое на выполнение операции умножения, можно сократить до величины порядка $n^{\lg 3} \approx n^{1.585}$, так что рекурсивный метод при больших n обеспечивает гораздо более высокую скорость, чем традиционный. В упр. 18 рассмотрено применение этого метода.

(Похожий, но немного более сложный метод умножения со временем выполнения порядка $n^{\lg 3}$ был впервые предложен А. Карацубой в *ДАН СССР* **145** (1962), 293–294. Любопытно, что эта идея, по-видимому, до 1962 года не была известна; нет сведений о том, чтобы кто-нибудь из "вундеркиндов-счетчиков", прославившихся своими способностями умножать в уме большие числа, применял когда-либо подобный метод, хотя аналог формулы (2) для десятичной системы счисления, казалось бы, дает довольно легкий способ умножения в уме восьмизначных чисел.)

В пределе при n, стремящемся к бесконечности, время выполнения можно сократить еще больше, если учесть, что рассмотренный только что метод является

частным случаем $r = 1$ более общего метода, который для произвольного фиксированного r дает

$$T\big((r+1)n\big) \le (2r+1)T(n) + cn. \tag{6}$$

Этот более общий метод можно получить следующим образом. Разобьем

$$u = (u_{(r+1)n-1} \dots u_1 u_0)_2 \qquad \text{и} \qquad v = (v_{(r+1)n-1} \dots v_1 v_0)_2$$

на $r+1$ частей

$$u = U_r 2^{rn} + \dots + U_1 2^n + U_0, \qquad v = V_r 2^{rn} + \dots + V_1 2^n + V_0, \tag{7}$$

где каждое U_j и каждое V_j является n-битовым числом. Рассмотрим полиномы

$$U(x) = U_r x^r + \dots + U_1 x + U_0, \qquad V(x) = V_r x^r + \dots + V_1 x + V_0 \tag{8}$$

и положим

$$W(x) = U(x)V(x) = W_{2r} x^{2r} + \dots + W_1 x + W_0. \tag{9}$$

Так как $u = U(2^n)$ и $v = V(2^n)$, получаем $uv = W(2^n)$, поэтому при известных коэффициентах W_k в $W(x)$ можно легко вычислить uv. Задача заключается в поиске хорошего способа вычисления этих коэффициентов в $W(x)$, требующем только $2r+1$ умножений n-битовых чисел и несколько последующих операций, время выполнения которых пропорционально n. Это может быть достигнуто посредством вычисления

$$U(0)V(0) = W(0), \quad U(1)V(1) = W(1), \quad \dots, \quad U(2r)V(2r) = W(2r). \tag{10}$$

Коэффициенты полинома степени $2r$ могут быть выражены в виде линейной комбинации значений этого полинома в $2r+1$ различных точках. Время, необходимое для выполнения этой операции, пропорционально n или меньше. (В действительности произведения $U(j)V(j)$ не являются в строгом смысле произведениями n-битовых чисел, но являются произведениями $(n+t)$-битовых чисел, где t есть фиксированное значение, зависящее от r. Программа умножения $(n+t)$-битовых чисел строится легко. Для ее выполнения требуется лишь $T(n) + c_1 n$ операций, где $T(n)$ — количество операций, необходимых для умножения n разрядов, так как при фиксированном t два произведения t- и n-битовых чисел можно получить за $c_2 n$ операций.) Таким образом, получаем метод умножения, для которого выполняется неравенство (6).

Рассуждая так же, как при выводе неравенства (5), и учитывая неравенство (6), приходим к неравенству $T(n) \le c_3 n^{\log_{r+1}(2r+1)} < c_3 n^{1+\log_{r+1} 2}$. Итак, доказана следующая теорема.

Теорема А. *Для любого $\epsilon > 0$ существуют такая постоянная $c(\epsilon)$ и такой алгоритм умножения, что число элементарных операций $T(n)$, которые необходимо выполнить, чтобы перемножить два n-битовых числа, удовлетворяет оценке*

$$T(n) < c(\epsilon) n^{1+\epsilon}. \qquad \blacksquare \tag{11}$$

Данная теорема — это еще не тот результат, который нам нужен. Для практических целей он неудовлетворителен, так как метод резко усложняется, когда $\epsilon \to 0$ (т. е. $r \to \infty$). Это приводит к столь быстрому росту $c(\epsilon)$, что приходится иметь дело с очень большими значениями n прежде, чем будут внесены какие-либо существенные улучшения в соотношение (5). Теорема неудовлетворительна и с

теоретической точки зрения, так как в ней не в полной мере используется лежащий в ее основе полиномиальный метод. Если предположить, что r *варьируется* вместе с n, то, выбирая по мере увеличения n все бóльшие и бóльшие значения r, можно получить лучший результат. Эта идея предложена А. Л. Тоомом [*ДАН СССР* **150** (1963), 496–498]. Тоом использовал ее для доказательства того факта, что при возрастающем n можно построить автомат для умножения n-разрядных чисел, состоящий из довольно малого числа компонентов.

Позднее С. А. Кук (S. A. Cook) в работе *On the Minimum Computation Time of Functions* (Thesis, Harvard University, 1966), 51–77, показал, как применить идею Тоома для ускорения работы компьютерных программ.

Прежде чем продолжить обсуждение алгоритма Тоома-Кука, рассмотрим простой пример перехода от $U(x)$ и $V(x)$ к коэффициентам функции $W(x)$. На этом примере нельзя ощутить эффективность метода, поскольку используемые в нем числа слишком малы. Но пример демонстрирует полезные упрощения, которые можно применять и в общем случае. Предположим, что нужно умножить $u = 1234$ на $v = 2341$ или в двоичной системе счисления число

$$u = (0100\ 1101\ 0010)_2 \text{ на число } v = (1001\ 0010\ 0101)_2. \tag{12}$$

Пусть $r = 2$. Полиномы $U(x)$ и $V(x)$ в (8) имеют вид

$$U(x) = 4x^2 + 13x + 2, \qquad V(x) = 9x^2 + 2x + 5.$$

Отсюда находим для $W(x) = U(x)V(x)$:

$$\begin{array}{lllll} U(0) = 2, & U(1) = 19, & U(2) = 44, & U(3) = 77, & U(4) = 118; \\ V(0) = 5, & V(1) = 16, & V(2) = 45, & V(3) = 92, & V(4) = 157; \\ W(0) = 10, & W(1) = 304, & W(2) = 1980, & W(3) = 7084, & W(4) = 18526. \end{array} \tag{13}$$

Теперь нужно найти пять коэффициентов полинома $W(x)$, используя пять последних величин.

Чтобы найти коэффициенты полинома $W(x) = W_{m-1}x^{m-1} + \cdots + W_1x + W_0$ при заданных значениях $W(0), W(1), \ldots, W(m-1)$, можно воспользоваться одним интересным алгоритмом. Сначала запишем

$$W(x) = a_{m-1}\,x^{\underline{m-1}} + a_{m-2}\,x^{\underline{m-2}} + \cdots + a_1x^{\underline{1}} + a_0, \tag{14}$$

где $x^{\underline{k}} = x(x-1)\ldots(x-k+1)$, а коэффициенты a_j неизвестны. Полиномы вида (14) обладают важным свойством:

$$W(x+1) - W(x) = (m-1)a_{m-1}\,x^{\underline{m-2}} + (m-2)a_{m-2}\,x^{\underline{m-3}} + \cdots + a_1;$$

отсюда по индукции получаем, что для всех $k \ge 0$

$$\begin{aligned} &\frac{1}{k!}\left(W(x+k) - \binom{k}{1}W(x+k-1) + \binom{k}{2}W(x+k-2) - \cdots + (-1)^kW(x)\right) \\ &\quad = \binom{m-1}{k}a_{m-1}\,x^{\underline{m-1-k}} + \binom{m-2}{k}a_{m-2}\,x^{\underline{m-2-k}} + \cdots + \binom{k}{k}a_k. \end{aligned} \tag{15}$$

Обозначив левую часть (15) через $(1/k!)\,\Delta^k W(x)$, замечаем, что

$$\frac{1}{k!}\Delta^k W(x) = \frac{1}{k}\left(\frac{1}{(k-1)!}\Delta^{k-1}W(x+1) - \frac{1}{(k-1)!}\Delta^{k-1}W(x)\right)$$

и $(1/k!)\,\Delta^k W(0) = a_k$. Таким образом, коэффициенты a_j можно вычислить при помощи очень простого метода, который иллюстрируется здесь на примере полинома $W(x)$, определенного соотношениями (13).

$$\begin{array}{rrlll} 10 & & & & \\ & 294 & & & \\ 304 & & 1382/2 = \ 691 & & \\ & 1676 & & 1023/3 = 341 & \\ 1980 & & 3428/2 = 1714 & & 144/4 = 36 \\ & 5104 & & 1455/3 = 485 & \\ 7084 & & 6338/2 = 3169 & & \\ & 11442 & & & \\ 18526 & & & & \end{array} \qquad (16)$$

Крайняя слева колонка этой таблицы состоит из заданных значений $W(0), W(1), \ldots$ $\ldots, W(4)$. Чтобы вычислить элементы в k-й колонке, нужно найти разность между соседними элементами предыдущей колонки и разделить эти разности на k. Коэффициенты a_j — это верхние числа в колонках, так что $a_0 = 10$, $a_1 = 294, \ldots, a_4 = 36$; отсюда имеем

$$\begin{aligned} W(x) &= 36x^4 + 341x^3 + 691x^2 + 294x^1 + 10 \\ &= \big(((36(x-3)+341)(x-2)+691)(x-1)+294\big)x + 10. \end{aligned} \qquad (17)$$

В общем случае можно записать

$$W(x) = \big(\ldots((a_{m-1}(x-m+2)+a_{m-2})(x-m+3)+a_{m-3})(x-m+4)+\cdots+a_1\big)x + a_0.$$

Данная формула показывает, каким образом с помощью коэффициентов a_m можно вычислить коэффициенты $W_{m-1}, \ldots, W_1, W_0$.

$$\begin{array}{|l|l|l|l|l} \hline 36 & 341 & & & \\ & -3\cdot 36 & & & \\ \hline 36 & 233 & 691 & & \\ & -2\cdot 36 & -2\cdot 233 & & \\ \hline 36 & 161 & 225 & 294 & \\ & -1\cdot 36 & -1\cdot 161 & -1\cdot 225 & \\ \hline 36 & 125 & 64 & 69 & 10 \\ \hline \end{array} \qquad (18)$$

В этой таблице числа, расположенные ниже горизонтальных линий, являются соответственно коэффициентами полиномов

$$\begin{aligned} &a_{m-1}, \\ &a_{m-1}(x-m+2)+a_{m-2}, \\ &\big(a_{m-1}(x-m+2)+a_{m-2}\big)(x-m+3)+a_{m-3} \qquad \text{и т. д.} \end{aligned}$$

Согласно данной таблице имеем

$$W(x) = 36x^4 + 125x^3 + 64x^2 + 69x + 10, \qquad (19)$$

так что ответом будет $1234 \cdot 2341 = W(16) = 2888794$, где $W(16)$ вычисляется с помощью операций сложения и сдвига. В разделе 4.6.4 рассмотрено обобщение этого метода вычисления коэффициентов.

Из основного тождества для чисел Стирлинга (см. упр. 1.2.6–(45)),

$$x^n = \left\{ {n \atop n} \right\} x^{\underline{n}} + \cdots + \left\{ {n \atop 1} \right\} x^{\underline{1}} + \left\{ {n \atop 0} \right\},$$

видно, что, если коэффициенты полинома $W(x)$ неотрицательны, таковыми же будут и числа a_j. В этом случае *все промежуточные результаты в вышеприведенных вычислениях неотрицательны.* Данное обстоятельство еще больше упрощает алгоритм умножения Тоома-Кука, который теперь будет рассмотрен подробно. (Нетерпеливые читатели могут перелистать страницы подраздела С.)

Алгоритм Т (*Умножение с высокой точностью двоичных чисел*). Для заданных положительного целого числа n и двух неотрицательных n-битовых целых чисел u и v этот алгоритм (рис. 8) формирует их $2n$-битовое произведение w. Для хранения длинных чисел, представляющих промежуточные результаты выполнения алгоритма, используются четыре вспомогательных стека.

Стеки U, V	Временное хранение $U(j)$ и $V(j)$ на шаге Т4
Стек C	Сомножители и управляющие коды
Стек W	Сохранение величин $W(j)$

Эти стеки могут содержать либо двоичные числа, либо специальные управляющие символы, называемые "код-1", "код-2" и "код-3". Алгоритм формирует также дополнительную таблицу чисел q_k, r_k, построенную таким образом, что ее можно хранить в запоминающем устройстве как линейный список, единственный указатель которого, перемещающийся по списку в обоих направлениях, может использоваться для выбора нужного текущего элемента из этого списка.

(Стеки C и W используются для контроля рекуррентного механизма алгоритма умножения довольно простым способом, который является частным случаем общей процедуры, рассматриваемой в главе 8.)

Т1. [Вычислить таблицы q, r.] Очистить стеки U, V, C и W и присвоить

$$k \leftarrow 1, \qquad q_0 \leftarrow q_1 \leftarrow 16, \qquad r_0 \leftarrow r_1 \leftarrow 4, \qquad Q \leftarrow 4, \qquad R \leftarrow 2.$$

Если теперь $q_{k-1} + q_k < n$, присвоить

$$k \leftarrow k+1, \qquad Q \leftarrow Q + R, \qquad R \leftarrow \lfloor \sqrt{Q} \rfloor, \qquad q_k \leftarrow 2^Q, \qquad r_k \leftarrow 2^R$$

и повторять эту операцию до тех пор, пока не выполнится условие $q_{k-1} + q_k \geq n$. (*Примечание.* Для вычисления $R \leftarrow \lfloor \sqrt{Q} \rfloor$ нет необходимости в извлечении корня квадратного, так как при $(R+1)^2 \leq Q$ можно просто присвоить $R \leftarrow R+1$, а если $(R+1)^2 > Q$, то нужно оставить R неизменным (см. упр. 2). На этом шаге строятся такие последовательности.

$k =$	0	1	2	3	4	5	6	...
$q_k =$	2^4	2^4	2^6	2^8	2^{10}	2^{13}	2^{16}	...
$r_k =$	2^2	2^2	2^2	2^2	2^3	2^3	2^4	...

Рис. 8. Алгоритм Тоома-Кука умножения с высокой точностью.

При умножении 70 000-битовых чисел этот шаг закончился бы при значении $k = 6$, так как $70000 < 2^{13} + 2^{16}$.)

T2. [Занести u, v в стек.] Занести "код-1" в стек C, затем поместить u и v в стек C как числа, каждое из которых содержит ровно $q_{k-1} + q_k$ бит.

T3. [Проверить уровень рекурсии.] Уменьшить k на 1. Если $k = 0$, то в вершине стека C находится в данный момент два 32-битовых числа u и v. Извлечь из стека эти числа, присвоить $w \leftarrow uv$, пользуясь встроенной программой для умножения 32-битовых чисел, и перейти к шагу T10. Если $k > 0$, то присвоить $r \leftarrow r_k$, $q \leftarrow q_k$, $p \leftarrow q_{k-1} + q_k$ и перейти к шагу T4.

T4. [Разбить на $r + 1$ частей.] Число, находящееся в вершине стека C, будем рассматривать как список из $r + 1$ чисел, каждое из которых состоит из q бит, $(U_r \dots U_1 U_0)_{2^q}$. (В вершине стека C находится $(r + 1)q = (q_k + q_{k+1})$-битовое число.) Для $j = 0, 1, \dots, 2r$ вычислим p-битовые числа

$$\big(\dots(U_r j + U_{r-1})j + \cdots + U_1\big)j + U_0 = U(j)$$

и поместим эти значения в стек U. (На самом дне стека U сейчас находится $U(0)$, затем следует $U(1)$ и т. д., а на вершине стека — $U(2r)$. Согласно упр. 3 имеем

$$U(j) \le U(2r) < 2^q\big((2r)^r + (2r)^{r-1} + \cdots + 1\big) < 2^{q+1}(2r)^r \le 2^p.)$$

Далее извлекаем из стека C значения $U_r \dots U_1 U_0$.

Теперь на вершине стека C находится другой список $r+1$ q-битовых чисел $V_r \dots V_1 V_0$, а p-битовые числа

$$\big(\dots(V_r j + V_{r-1})j + \cdots + V_1\big)j + V_0 = V(j)$$

таким же образом должны быть помещены в стек V. После завершения этих действий переместить из стека C числа $V_r \dots V_1 V_0$.

T5. [Выполнить рекурсию.] Величины

$$\text{код-2},\ V(2r),\ U(2r),\ \text{код-3},\ V(2r-1),\ U(2r-1),\ \ldots,$$
$$\text{код-3},\ V(1),\ U(1),\ \text{код-3},\ V(0),\ U(0)$$

поместить последовательно в стек C, одновременно освобождая стеки U и V. "Код-4" поместить в стек W. Вернуться к шагу T3.

T6. [Сохранить одно произведение.] (В этот момент алгоритм умножения сформировал в w одно из произведений $W(j) = U(j)V(j)$.) Поместить w в стек W. (Число w содержит $2(q_k + q_{k-1})$ бит.) Вернуться к шагу T3.

T7. [Найти все a.] Присвоить $r \leftarrow r_k$, $q \leftarrow q_k$, $p \leftarrow q_{k-1} + q_k$. (Сейчас в стек W последовательно от дна к вершине помещена последовательность чисел, оканчивающаяся $W(0), W(1), \ldots, W(2r)$, где числа $W(j)$ представлены как $2p$-битовые.)

Теперь для $j = 1, 2, 3, \ldots, 2r$ выполнить следующий цикл: для $t = 2r, 2r-1, 2r-2, \ldots, j$ присвоить $W(t) \leftarrow \big(W(t) - W(t-1)\big)/j$. (В цикле j должно увеличиваться, а t — уменьшаться. Величина $\big(W(t) - W(t-1)\big)/j$ всегда будет неотрицательным целым числом, занимающим $2p$ бит (см. табл. (16).)

T8. [Найти все $W(j)$.] Для $j = 2r-1, 2r-2, \ldots, 1$ выполнить следующий цикл: для $t = j, j+1, \ldots, 2r-1$ присвоить $W(t) \leftarrow W(t) - jW(t+1)$. (В цикле j должно уменьшаться, а t — увеличиваться. Результатом этой операции снова будет неотрицательное целое число, занимающее $2p$ бит; см. табл. (18).)

T9. [Сформировать ответ.] Сформировать в w $2(q_k + q_{k+1})$-битовое целое число

$$\big(\ldots\big(W(2r)2^q + W(2r-1)\big)2^q + \cdots + W(1)\big)2^q + W(0).$$

Удалить $W(2r)$, $\ldots$, $W(0)$ из стека W.

T10. [Возвратиться.] Присвоить $k \leftarrow k+1$. Извлечь число из вершины стека C. Если это "код-3", перейти к шагу T6. Если это "код-2", поместить w в стек W и перейти к шагу T7. Закончить выполнение алгоритма, если это "код-1" (w является искомым результатом.) ∎

Оценим время выполнения $T(n)$ алгоритма T в некоторых единицах, называемых циклами, т. е. в количестве элементарных машинных операций. На выполнение шага T1 уходит $O(q_k)$ циклов даже в том случае, когда число q_k представлено в виде длинной цепочки из q_k бит, за которой следует некоторый разграничитель, так как $q_k + q_{k-1} + \cdots + q_0$ равно $O(q_k)$. Шаг T2 выполняется за $O(q_k)$ циклов.

Обозначим через t_k объем вычислений, которые необходимо выполнить, чтобы от шага T3 перейти к шагу T10 для некоторого конкретного значения k (после того, как в начале шага T3 значение k было уменьшено на 1). На выполнение шага T3 требуется не более $O(q)$ циклов. На шаге T4 выполняется r умножений p-битовых чисел на $\lg(2r)$-битовых числа и r сложений p-битовых чисел. Все эти операции повторяются $O(r^2 q \log r)$ раз. Таким образом, общее количество циклов равно $O(r^2 q \log r)$. На шаге T5 выполняется перемещение $4r+2$ p-битовых чисел, и на каждой итерации перемещение повторяется $2r+1$ раз. Для выполнения рекурсии, которая появляется, когда алгоритм повторяет сам себя (возвращаясь к шагу T3), необходимо по t_{k-1} циклов на каждой из итераций $2r+1$. На шаге T7

требуется выполнить $O(r^2)$ вычитаний p-битовых чисел и делений p-битовых чисел на $\lg(2r)$-битовые числа, на что затрачивается $O(r^2 q \log r)$ циклов, как и на шаге T8. Для шага T9 необходимо $O(rq)$ циклов, а на выполнение шага T10 время практически вовсе не затрачивается.

Суммируя, получаем $T(n) = O(q_k) + O(q_k) + t_{k-1}$, где (при $q = q_k$ и $r = r_k$) основной вклад во время выполнения алгоритма удовлетворяет равенству

$$\begin{aligned} t_k &= O(q) + O(r^2 q \log r) + O(rq) + (2r+1)O(q) + O(r^2 q \log r) \\ &\qquad + O(r^2 q \log r) + O(rq) + O(q) + (2r+1)t_{k-1} \\ &= O(r^2 q \log r) + (2r+1)t_{k-1}. \end{aligned}$$

Следовательно, найдется константа c такая, что

$$t_k \le c r_k^2 q_k \lg r_k + (2r_k + 1)t_{k-1}.$$

Чтобы завершить оценку t_k, докажем (довольно грубо), что для некоторой константы C

$$t_k \le C q_{k+1} 2^{2.5\sqrt{\lg q_{k+1}}}. \tag{20}$$

Выберем $C > 20c$, и пусть, кроме того, C настолько велико, что неравенство (20) справедливо для $k \le k_0$, где k_0 будет уточнено ниже. Далее, для $k > k_0$ положим $Q_k = \lg q_k$, $R_k = \lg r_k$. По индукции имеем

$$t_k \le c q_k r_k^2 \lg r_k + (2r_k+) C q_k 2^{2.5\sqrt{Q_k}} = C q_{k+1} 2^{2.5\sqrt{\lg q_{k+1}}} (\eta_1 + \eta_2),$$

где

$$\eta_1 = \frac{c}{C} R_k 2^{R_k - 2.5\sqrt{Q_{k+1}}} < \frac{1}{20} R_k 2^{-R_k} < 0.05,$$

$$\eta_2 = \left(2 + \frac{1}{r_k}\right) 2^{2.5(\sqrt{Q_k} - \sqrt{Q_{k+1}})} \to 2^{-1/4} < 0.85,$$

так как

$$\sqrt{Q_{k+1}} - \sqrt{Q_k} = \sqrt{Q_k + \lfloor \sqrt{Q_k} \rfloor} - \sqrt{Q_k} \to \tfrac{1}{2}$$

при $k \to \infty$. Отсюда следует, что можно найти k_0, такое, что $\eta_2 < 0.95$ для всех $k > k_0$. На этом доказательство по индукции неравенства (20) завершается.

С учетом полученного результата теперь можно оценить $T(n)$. Поскольку $n > q_{k-1} + q_{k-2}$, то $q_{k-1} < n$. Получаем

$$r_{k-1} = 2^{\lfloor \sqrt{\lg q_{k-1}} \rfloor} < 2^{\sqrt{\lg n}} \qquad \text{и} \qquad q_k = r_{k-1} q_{k-1} < n 2^{\sqrt{\lg n}}.$$

Таким образом,

$$t_{k-1} \le C q_k 2^{2.5\sqrt{\lg q_k}} < C n 2^{\sqrt{\lg n} + 2.5(\sqrt{\lg n} + 1)}$$

и, учитывая, что $T(n) = O(q_k) + t_{k-1}$, мы сформулировали следующую теорему.

Теорема B. *Существует константа c_0, такая, что время выполнения алгоритма T меньше, чем $c_0 n 2^{3.5\sqrt{\lg n}}$ циклов.* ▮

Поскольку $n2^{3.5\sqrt{\lg n}} = n^{1+3.5/\sqrt{\lg n}}$, этот результат существенно сильнее, чем теорема A. Несколько усложнив алгоритм и распространив эти идеи вплоть до очевидных ограничений (см. упр. 5), можно улучшить время выполнения, добившись оценки

$$T(n) = O(n2^{\sqrt{2\lg n}}\log n). \tag{21}$$

***B. Модулярный метод.** Существует еще один метод очень быстрого перемножения больших чисел, основанный на идеях модулярной арифметики, которые представлены в разделе 4.3.2. На первый взгляд, трудно поверить, что он может иметь какие-либо преимущества, так как алгоритм умножения, основанный на модулярной арифметике, кроме собственно операции умножения, должен включать процедуры выбора модуля и перевода чисел в модулярное представление и обратно. Несмотря на такие пугающие трудности А. Шёнхаге (A. Schönhage) обнаружил, что все эти операции можно очень быстро реализовать.

Чтобы лучше понять суть метода А. Шёнхаге, рассмотрим один частный случай — последовательность, определенную по правилам

$$q_0 = 1, \qquad q_{k+1} = 3q_k - 1, \tag{22}$$

так что $q_k = 3^k - 3^{k-1} - \cdots - 1 = \frac{1}{2}(3^k + 1)$. Исследуем процедуру, выполняющую умножение p_k-битовых чисел, где $p_k = (18q_k + 8)$, в терминах метода умножения p_{k-1}-битовых чисел. Итак, если известно, как умножать числа, состоящие из $p_0 = 26$ бит, описываемая ниже процедура покажет, как умножать числа из $p_1 = 44$, 98, 260 бит и т. д., увеличивая количество битов почти в три раза на каждом шаге.

При умножении p_k-битовых чисел идея состоит в использовании шести модулей:

$$\begin{aligned} m_1 &= 2^{6q_k-1} - 1, \quad & m_2 &= 2^{6q_k+1} - 1, \quad & m_3 &= 2^{6q_k+2} - 1, \\ m_4 &= 2^{6q_k+3} - 1, \quad & m_5 &= 2^{6q_k+5} - 1, \quad & m_6 &= 2^{6q_k+7} - 1. \end{aligned} \tag{23}$$

Эти модули взаимно просты согласно соотношению 4.3.2–(19), так как показатели степени в (23)

$$6q_k - 1, \quad 6q_k + 1, \quad 6q_k + 2, \quad 6q_k + 3, \quad 6q_k + 5, \quad 6q_k + 7 \tag{24}$$

всегда взаимно просты (см. упр. 6). При помощи шести модулей в (23) можно представлять числа вплоть до $m = m_1m_2m_3m_4m_5m_6 > 2^{36q_k+16} = 2^{2p_k}$, и поэтому при умножении p_k-битовых чисел u и v возможность переполнения совершенно исключена. Таким образом, при $k > 0$ можно использовать следующий метод.

a) Вычислить $u_1 = u \bmod m_1$, ..., $u_6 = u \bmod m_6$ и $v_1 = v \bmod m_1$, ..., $v_6 = v \bmod m_6$.

b) Умножить u_1 на v_1, u_2 на v_2, ..., u_6 на v_6. Эти числа состоят не более чем из $6q_k + 7 = 18q_{k-1} + 1 < p_{k-1}$ бит, поэтому операции умножения могут быть выполнены при помощи процедуры, используемой для умножения p_{k-1}-битовых чисел.

c) Вычислить $w_1 = u_1v_1 \bmod m_1$, $w_2 = u_2v_2 \bmod m_2$, ..., $w_6 = u_6v_6 \bmod m_6$.

d) Вычислить w, такое, чтобы выполнялось неравенство $0 \le w < m$, $w \bmod m_1 = w_1$, ..., $w \bmod m_6 = w_6$.

Пусть время, требуемое для выполнения этого процесса, равно t_k. Нетрудно заметить, что на выполнение операции (a) необходимо $O(p_k)$ циклов, ибо определение $u \bmod (2^e - 1)$ осуществляется совсем просто (подобно "выбрасыванию девяток"), как описано в разделе 4.3.2. Аналогично на операцию (b) уходит $O(p_k)$ циклов. Остается операция (d), которая, на первый взгляд, требует выполнения сложных вычислений. Но Шёнхаге нашел оригинальный способ выполнения операции (d) за $O(p_k \log p_k)$ циклов. В этом и состоит сущность предложенного метода. Как следствие имеем

$$t_k = 6t_{k-1} + O(p_k \log p_k).$$

Так как $p_k = 3^{k+2} + 17$, можно показать, что время, затрачиваемое на умножение n-битовых чисел, равно

$$T(n) = O(n^{\log_3 6}) = O(n^{1.631}). \tag{25}$$

(См. упр. 7.)

Хотя модулярный метод сложнее, чем описанная в начале этого раздела процедура, на выполнение которой требуется $O(n^{\lg 3})$ циклов, в действительности время, затрачиваемое на умножение согласно формуле (25), существенно меньше времени $O(n^2)$ на умножение n-битовых чисел. Таким образом, используя один из совершенно разных методов, рассмотренных выше, можно усовершенствовать классический метод умножения n-битовых чисел.

Теперь проанализируем упомянутую выше операцию (d). Предположим, что дан ряд положительных попарно взаимно простых целых чисел $e_1 < e_2 < \cdots < e_r$. Пусть

$$m_1 = 2^{e_1} - 1, \qquad m_2 = 2^{e_2} - 1, \qquad \ldots, \qquad m_r = 2^{e_r} - 1. \tag{26}$$

Пусть также даны числа $w_1, \ldots, w_r$, такие, что $0 \le w_j \le m_j$. Задача состоит в том, чтобы определить двоичное представление чисел w, удовлетворяющих условиям

$$\begin{gathered} 0 \le w < m_1 m_2 \ldots m_r, \\ w \equiv w_1 \text{ (по модулю } m_1), \qquad \ldots, \qquad w \equiv w_r \text{ (по модулю } m_r). \end{gathered} \tag{27}$$

Метод основан на использовании соотношений (24) и (25) из раздела 4.3.2. Вычислим сначала

$$w'_j = \left(\ldots((w_j - w'_1)\, c_{1j} - w'_2)\, c_{2j} - \cdots - w'_{j-1}\right) c_{(j-1)j} \bmod m_j \tag{28}$$

для $j = 2, \ldots, r$, где $w'_1 = w_1 \bmod m_1$. Затем вычислим

$$w = \left(\ldots(w'_r m_{r-1} + w'_{r-1})\, m_{r-2} + \cdots + w'_2\right) m_1 + w'_1. \tag{29}$$

В этом равенстве c_{ij} — такое число, что $c_{ij} m_i \equiv 1$ (по модулю m_j); числа c_{ij} должны быть определены по числам e_j.

При любых j для вычисления по формуле (28) требуется $\binom{r}{2}$ операций сложения по модулю m_j, на каждую из которых затрачивается $O(e_r)$ циклов, плюс $\binom{r}{2}$ операций умножения на c_{ij} по модулю m_j. Вычисление w по формуле (29) требует r операций сложения и r операций умножения на m_j. Но операция умножения на m_j выполняется легко, ибо это просто сложение, вычитание и сдвиг, поэтому ясно, что на выполнение вычислений по формуле (29) затрачивается $O(r^2 e_r)$ циклов. Как вскоре будет видно, каждая операция умножения на c_{ij} по модулю m_j требует

для выполнения только $O(e_r \log e_r)$ циклов, а потому *весь процесс перехода можно выполнить за* $O(r^2 e_r \log e_r)$ *циклов.*

После этого остается решить следующую задачу. Для заданных положительных целых чисел e и f $(e < f)$ и неотрицательного целого числа $u < 2^f$ найти значение $(cu) \bmod (2^f - 1)$, где число c таково, что $(2^e - 1)c \equiv 1$ (по модулю $2^f - 1$), причем вычисления должны быть выполнены за $O(f \log f)$ циклов. В ответе к упр. 4.3.2–6 приведена формула для c, которая наводит на мысль о необходимости использовать здесь ту же процедуру. Прежде всего найдем наименьшее положительное целое число b, такое, что

$$be \equiv 1 \ \text{(по модулю } f\text{)}. \tag{30}$$

При помощи алгоритма Евклида это можно сделать за $O\big((\log f)^3\big)$ циклов, так как данному алгоритму для обработки чисел e и f требуется $O(\log f)$ итераций и каждая итерация выполняется за $O\big((\log f)^2\big)$ циклов. Число b можно было бы найти и путем простого перебора, не изменяя общее время выполнения, а просто применяя $b = 1, 2$ и т. д. до тех пор, пока не будет удовлетворяться (30). Этот процесс существенно не сказывается на общем времени выполнения, поскольку на него потребовалось бы всего $O(f \log f)$ циклов. После того как b найдено, в силу упр. 4.3.2–6 имеем

$$c = c[b] = \left(\sum_{0 \le j < b} 2^{je} \right) \bmod (2^f - 1). \tag{31}$$

Прямого умножения $(cu) \bmod (2^f - 1)$ может оказаться недостаточно для решения задачи, потому что нам неизвестно, как умножить f-битовые числа общего вида за $O(f \log f)$ циклов. Но специальная форма числа c дает ключ к решению: двоичное представление числа c имеет некоторую регулярную структуру битов, и из формулы (31) видно, что число $c[2b]$ может быть простым способом получено из числа $c[b]$. Это свойство наводит на мысль, что можно быстро умножить число u на $c[b]$, если подходящим образом получить число $c[b]u$ за $\lg b$ шагов. Например, это можно выполнить следующим образом. Пусть в двоичной системе счисления число b имеет вид

$$b = (b_s \ldots b_2 b_1 b_0)_2.$$

Можно вычислить четыре последовательности чисел a_k, d_k, u_k, v_k, которые определены правилами

$$\begin{aligned} a_0 &= e, & a_k &= 2a_{k-1} \bmod f; \\ d_0 &= b_0 e, & d_k &= (d_{k-1} + b_k\, a_k) \bmod f; \\ u_0 &= u, & u_k &= (u_{k-1} + 2^{a_{k-1}} u_{k-1}) \bmod (2^f - 1); \\ v_0 &= b_0 u, & v_k &= (v_{k-1} + b_k\, 2^{d_{k-1}} u_k) \bmod (2^f - 1). \end{aligned} \tag{32}$$

Индукцией по k легко доказать, что

$$\begin{aligned} a_k &= (2^k e) \bmod f; & u_k &= (c[2^k]u) \bmod (2^f - 1); \\ d_k &= \big((b_k \ldots b_1 b_0)_2\, e\big) \bmod f; & v_k &= \big(c[(b_k \ldots b_1 b_0)_2]u\big) \bmod (2^f - 1). \end{aligned} \tag{33}$$

Следовательно, искомый результат $(c[b]u) \bmod (2^f - 1)$ будет равен v_s. Для вычисления a_k, d_k, u_k и v_k по a_{k-1}, d_{k-1}, u_{k-1}, v_{k-1} требуется $O(\log f) + O(\log f) +$

$O(f)+O(f)=O(f)$ циклов, поэтому весь объем вычислений можно выполнить, как и требовалось, за $s\,O(f)=O(f\log f)$ циклов.

Тщательный анализ оригинального метода, представленного формулами (32) и (33), принесет читателю много пользы. Подобные методы рассматриваются в разделе 4.6.3.

В работе Шёнхаге, опубликованной в *Computing* **1** (1966), 182–196, показано, что эти идеи могут быть распространены на операции умножения n-битовых чисел с использованием модулей $r\approx 2^{\sqrt{2\lg n}}$, в результате чего будет создан метод, аналогичный алгоритму T. Мы не будем здесь останавливаться на деталях этого метода, поскольку алгоритм T всегда имеет приоритет. К тому же сейчас будет рассмотрен еще лучший метод.

C. Умножение при помощи дискретного преобразования Фурье. Основной проблемой при умножении с высокой точностью является вычисление "свертки", например

$$u_rv_0+u_{r-1}v_1+\cdots+u_0v_r, \tag{34}$$

а между свертками и важным математическим понятием, называемым "преобразование Фурье", имеется тесная связь. Если $\omega=\exp(2\pi i/K)$ есть корень K-й степени из единицы, то можно определить одномерное преобразование Фурье последовательности комплексных чисел $(u_0,u_1,\ldots,u_{K-1})$ как последовательность $(\hat u_0,\hat u_1,\ldots,\hat u_{K-1})$, где

$$\hat u_s=\sum_{0\le t<K}\omega^{st}u_t,\qquad 0\le s<K. \tag{35}$$

Положив, что $(\hat v_0,\hat v_1,\ldots,\hat v_{K-1})$ — такое же преобразование Фурье для последовательности $(v_0,v_1,\ldots,v_{K-1})$, нетрудно заметить, что $(\hat u_0\hat v_0,\hat u_1\hat v_1,\ldots,\hat u_{K-1}\hat v_{K-1})$ будет преобразованием Фурье для $(w_0,w_1,\ldots,w_{K-1})$, где

$$\begin{aligned}w_r&=u_rv_0+u_{r-1}v_1+\cdots+u_0v_r+u_{K-1}v_{r+1}+\cdots+u_{r+1}v_{K-1}\\&=\sum_{i+j\equiv r\ (\text{по модулю } K)}u_iv_j.\end{aligned} \tag{36}$$

В случае, когда $K\ge 2n-1$ и $u_n=u_{n+1}=\cdots=u_{K-1}=v_n=v_{n+1}=\cdots=v_{K-1}=0$, числа w — это как раз то, что необходимо для операции умножения, поскольку в (36) члены $u_{K-1}v_{r+1}+\cdots+u_{r+1}v_{K-1}$ при $0\le r\le 2n-2$ исчезают. Другими словами, *преобразование от свертки есть обычное произведение преобразований компонентов.* Это частный случай идеи Тоома, связанной с использованием полиномов (см. выражения (9) и (10)), причем x заменяется корнями из единицы.

Для K, равного степени 2, дискретное преобразование Фурье (35) может быть получено очень быстро, если вычисления выполняются в определенной последовательности; именно таким образом выполняется обратное преобразование Фурье (определение чисел w из $\hat w$). Такое свойство преобразования Фурье было использовано в 1968 году Ф. Штрассеном (V. Strassen), который предложил способ умножения больших чисел, более быстрый, чем это было возможно с использованием любых известных к тому времени методов. Позже вместе с Шёнхаге он уточнил метод и опубликовал модифицированные алгоритмы в *Computing* **7** (1971), 281–292. Похожие идеи, но для произвольных целых чисел, были независимо от Ф. Штрассена

предложены Дж. М. Поллардом (J. M. Pollard), *Math. Comp.* **25** (1971), 365–374. Чтобы понять значимость их вклада в решение проблемы, рассмотрим для начала механизм быстрого преобразования Фурье. Для данной последовательности $K = 2^k$ комплексных чисел $(u_0, \ldots, u_{K-1})$ и данного комплексного числа

$$\omega = \exp(2\pi i/K) \tag{37}$$

последовательность $(\hat{u}_0, \ldots, \hat{u}_{K-1})$, определенная выражением (35), может быть быстро вычислена по следующей схеме. (Параметры s_j и t_j в этих формулах равны либо 0, либо 1, так что на каждый "проход" приходится 2^k элементарных вычислительных операций.)

Проход 0. Пусть $A^{[0]}(t_{k-1}, \ldots, t_0) = u_t$, где $t = (t_{k-1} \ldots t_0)_2$.

Проход 1. Присвоить $A^{[1]}(s_{k-1}, t_{k-2}, \ldots, t_0) \leftarrow$

$$A^{[0]}(0, t_{k-2}, \ldots, t_0) + \omega^{2^{k-1}s_{k-1}} A^{[0]}(1, t_{k-2}, \ldots, t_0).$$

Проход 2. Присвоить $A^{[2]}(s_{k-1}, s_{k-2}, t_{k-3}, \ldots, t_0) \leftarrow$

$$A^{[1]}(s_{k-1}, 0, t_{k-3}, \ldots, t_0) + \omega^{2^{k-2}(s_{k-2}s_{k-1})_2} A^{[1]}(s_{k-1}, 1, t_{k-3}, \ldots, t_0).$$

. . .

Проход k. Присвоить $A^{[k]}(s_{k-1}, \ldots, s_1, s_0) \leftarrow$

$$A^{[k-1]}(s_{k-1}, \ldots, s_1, 0) + \omega^{(s_0 s_1 \ldots s_{k-1})_2} A^{[k-1]}(s_{k-1}, \ldots, s_1, 1).$$

По индукции очень просто доказать, что

$$A^{[j]}(s_{k-1}, \ldots, s_{k-j}, t_{k-j-1}, \ldots, t_0) = \sum_{0 \le t_{k-1}, \ldots, t_{k-j} \le 1} \omega^{2^{k-j}(s_{k-j} \ldots s_{k-1})_2 (t_{k-1} \ldots t_{k-j})_2} u_t, \tag{38}$$

где $t = (t_{k-1} \ldots t_1 t_0)_2$, так что

$$A^{[k]}(s_{k-1}, \ldots, s_1, s_0) = \hat{u}_s, \qquad \text{где } s = (s_0 s_1 \ldots s_{k-1})_2. \tag{39}$$

(Важно отметить, что двоичные цифры числа s в конечном результате (39) обращены. В разделе 4.6.4 приводится дальнейший анализ такого рода преобразований.)

Прежде чем приступить к поиску обратного преобразования Фурье $(u_0, \ldots, u_{K-1})$ по значениям $(\hat{u}_0, \ldots, \hat{u}_{K-1})$, заметим, что "двойное преобразование" имеет вид

$$\begin{aligned} \hat{\hat{u}}_r &= \sum_{0 \le s < K} \omega^{rs} \hat{u}_s = \sum_{0 \le s,t < K} \omega^{rs} \omega^{st} u_t \\ &= \sum_{0 \le t < K} u_t \left(\sum_{0 \le s < K} \omega^{s(t+r)} \right) = K u_{(-r) \bmod K}, \end{aligned} \tag{40}$$

так как для j, кратных K, сумма геометрической прогрессии $\sum_{0 \le s < K} \omega^{sj}$ равна нулю. Поэтому обратное преобразование может быть получено точно так, как и прямое, за исключением того, что конечный результат делится на K и немного сдвинут. Возвращаясь к проблеме умножения целых чисел, предположим, что нужно вычислить произведение двух n-битовых целых чисел u и v. Будем оперировать (как и в алгоритме T) группами битов. Положим

$$2n \le 2^k l < 4n, \qquad K = 2^k, \qquad L = 2^l \tag{41}$$

и запишем

$$u = (U_{K-1} \dots U_1 U_0)_L, \qquad v = (V_{K-1} \dots V_1 V_0)_L, \tag{42}$$

полагая числа u и v K-разрядными числами по основанию L, так что каждый из разрядов U_j или V_j есть l-битовое целое число. Поскольку $2^{k-1}l \geq n$, ведущие разряды в U_j и V_j для всех $j \geq K/2$ равны нулю. Соответствующие значения для k и l будут выбраны позже, когда в процессе решения задачи прояснится общая ситуация и можно будет при выборе k и l учесть всю имеющуюся информацию.

Следующим шагом в процедуре умножения является вычисление преобразований Фурье $(\hat{u}_0, \dots, \hat{u}_{K-1})$ и $(\hat{v}_0, \dots, \hat{v}_{K-1})$ последовательностей $(u_0, \dots, u_{K-1})$ и $(v_0, \dots, v_{K-1})$, где по определению

$$u_t = U_t/2^{k+l}, \qquad v_t = V_t/2^{k+l}. \tag{43}$$

Для удобства масштабирование выполнено так, что любые u_t и v_t меньше 2^{-k}, а это, в свою очередь, обеспечивает то, что абсолютные значения $|\hat{u}_s|$ и $|\hat{v}_s|$ оказываются меньше 1 для всех s.

Здесь возникает очевидная проблема, связанная с тем, что комплексное число ω не может быть точно представлено в двоичной системе счисления. Как же вычислить достоверное преобразование Фурье? Провидению было угодно, чтобы результат получился правильным даже в случае, если в процессе вычислений предъявить весьма скромные требования к точности представления чисел. Оставим на время эту проблему и предположим, что вычисления выполняются с бесконечной точностью. Анализ необходимой точности будет приведен несколько ниже (через пару страниц).

Для полученных $\hat{u}_s$ и $\hat{v}_s$ положим $\hat{w}_s = \hat{u}_s \hat{v}_s$ $(0 \leq s < K)$ и определим обратное преобразование Фурье $(w_0, \dots, w_{K-1})$. Как было разъяснено выше, получим

$$w_r = \sum_{i+j=r} u_i v_j = \sum_{i+j=r} U_i V_j / 2^{2k+2l},$$

так что целые числа $W_r = 2^{2k+2l} w_r$ являются коэффициентами искомого произведения

$$u \cdot v = W_{K-2} L^{K-2} + \dots + W_1 L + W_0. \tag{44}$$

Поскольку $0 \leq W_r < (r+1)L^2 < KL^2$, каждое W_r содержит не более $k + 2l$ бит, поэтому нетрудно получить двоичное представление для известных величин W_k, если только k не слишком велико по сравнению с l. Например, пусть нужно умножить $u = 1234$ на $v = 2341$ при $k = 3$ и $l = 4$. Вычисление изображения $(\hat{u}_0, \dots, \hat{u}_7)$ по оригиналу u выполняется следующим образом (см. (12)).

$(r,s,t) =$	$(0,0,0)$	$(0,0,1)$	$(0,1,0)$	$(0,1,1)$	$(1,0,0)$	$(1,0,1)$	$(1,1,0)$	$(1,1,1)$
$2^7 A^{[0]}(r,s,t) =$	2	13	4	0	0	0	0	0
$2^7 A^{[1]}(r,s,t) =$	2	13	4	0	2	13	4	0
$2^7 A^{[2]}(r,s,t) =$	6	13	-2	13	$2+4i$	13	$2-4i$	13
$2^7 A^{[3]}(r,s,t) =$	19	-7	$-2+13i$	$-2-13i$	$\alpha+\beta$	$\alpha-\beta$	$\bar{\alpha}-\bar{\beta}$	$\bar{\alpha}+\bar{\beta}$

Здесь $\alpha = 2+4i$, $\beta = 13\omega$ и $\omega = (1+i)/\sqrt{2}$. Это дает нам колонку $\hat{u}_s$ в табл. 1. Данные в колонке $\hat{v}_s$ получаются из v аналогично. После этого находим $\hat{w}_s$, умножив $\hat{u}_s$ на $\hat{v}_s$. Выполняем еще одно преобразование, учитывая соотношение (40), и получаем w_s и W_s. Итак, мы получили свертку (19), но на этот раз используя комплексные

Таблица 1

УМНОЖЕНИЕ ПРИ ПОМОЩИ ДИСКРЕТНОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

s	$2^7\hat{u}_s$	$2^7\hat{v}_s$	$2^{14}\hat{w}_s$	$2^{14}\hat{\hat{w}}_s$	$2^{14}w_s = W_s$
0	19	16	304	80	10
1	$2+4i+13\omega$	$5+9i+2\omega$	$-26+64i+69\omega-125\bar{\omega}$	0	69
2	$-2+13i$	$-4+2i$	$-18-56i$	0	64
3	$2-4i-13\bar{\omega}$	$5-9i-2\bar{\omega}$	$-26-64i+125\omega-69\bar{\omega}$	0	125
4	-7	12	-84	288	36
5	$2+4i-13\omega$	$5+9i-2\omega$	$-26+64i-69\omega+125\bar{\omega}$	1000	0
6	$-2-13i$	$-4-2i$	$-18+56i$	512	0
7	$2-4i+13\bar{\omega}$	$5-9i+2\bar{\omega}$	$-26-64i-125\omega+69\bar{\omega}$	552	0

числа вместо того, чтобы оставаться верными методу, оперирующему только целыми числами.

Попробуем теперь оценить время, необходимое этому методу при больших числах, если использовать m-битовую арифметику в формате с фиксированной точкой. Из упр. 10 видно, что в процессе преобразования все величины $A^{[j]}$ будут меньше 1 из-за масштабирования (43). Следовательно, достаточно иметь дело только с дробными m-битовыми частями $(.a_{-1}\ldots a_{-m})_2$ для действительной и мнимой частей любых промежуточных результатов. Так как входные величины u_t и v_t являются действительными числами, можно упростить вычислительную процедуру, а именно — вместо $2K$ действительных значений на каждом шаге преобразований использовать только K значений (см. упр. 4.6.4–14). Чтобы не усложнять выкладки, далее эти тонкости учитывать не будем.

Прежде всего необходимо вычислить ω и ее степени. Для простоты сформируем таблицу значений $\omega^0, \ldots, \omega^{K-1}$. Положим

$$\omega_r = \exp(2\pi i/2^r) \tag{45}$$

таким, что $\omega_1 = -1$, $\omega_2 = i$, $\omega_3 = (1+i)/\sqrt{2}$, $\ldots$, $\omega_k = \omega$. Если $\omega_r = x_r + iy_r$, то имеем $\omega_{r+1} = x_{r+1} + iy_{r+1}$, где

$$x_{r+1} = \sqrt{\frac{1+x_r}{2}}, \qquad y_{r+1} = \frac{y_r}{2x_{r+1}}. \tag{46}$$

[См. S. R. Tate, *IEEE Transactions* **SP-43** (1995), 1709–1711.] На вычисление ω_1, $\omega_2, \ldots, \omega_k$ затрачивается по сравнению с остальными операциями относительно немного времени, так что вполне можно использовать любой стандартный способ извлечения квадратных корней. После ω_r можно легко вычислить все степени ω^j, учитывая, что

$$\omega^j = \omega_1^{j_{k-1}} \ldots \omega_{k-1}^{j_1}\omega_k^{j_0}, \qquad \text{если } j = (j_{k-1}\ldots j_1 j_0)_2. \tag{47}$$

Поскольку каждое значение ω^j получается как результат умножения ω_r не более чем k раз, ошибки вычислений при таком методе не накапливаются. Общее время вычисления всех значений ω^j равно $O(KM)$, где M — время выполнения операций умножения m-битовых комплексных чисел, поскольку на вычисление каждого из следующих значений ω^j по известному предыдущему требуется одна операция умножения. На выполнение последующих операций необходимо больше циклов, чем $O(KM)$, поэтому на вычисление степеней ω требуется сравнительно мало времени.

Каждое из трех преобразований Фурье состоит из k проходов, а каждый из проходов включает K операций вида $a \leftarrow b + \omega^j c$, так что общее время вычисления преобразований Фурье равно

$$O(kKM) = O(Mnk/l).$$

В итоге для вычисления двоичных разрядов $u \cdot v$ согласно (44) требуется $O\big(K(k+l)\big) = O(n + nk/l)$ машинных циклов. Если просуммировать все операции, получим, что общее время умножения n-битовых чисел u и v равняется $O(n) + O(Mnk/l)$ циклам.

Теперь, зная, какой должна быть величина M, посмотрим, какой должна быть точность вычисления промежуточных результатов m. Для простоты вместо поиска наилучших оценок точности будем рассматривать оценки с запасом, гарантирующие получение при помощи этого алгоритма требуемых результатов. Достаточно вычислить все значения ω^j в таком виде, чтобы приближения $(\omega^j)'$ удовлетворяли неравенству $|(\omega^j)'| \le 1$. Это условие легко обеспечить, если значения не округлять, а усекать до нуля, так как в (46) $x_{r+1}^2 + y_{r+1}^2 = (1 + x_r^2 + y_r^2 + 2x_r)/(2 + 2x_r)$. Все операции с m-битовыми числами в комплекснозначной арифметике с фиксированной точкой выполняются посредством замены точных вычислений вида $a \leftarrow b + \omega^j c$ приближенными:

$$a' \leftarrow \text{усечение}\big(b' + (\omega^j)' c'\big), \tag{48}$$

где b', $(\omega^j)'$ и c' — приближения, вычисленные предварительно. Все эти комплексные числа и их приближения по абсолютному значению ограничены единицей. Если $|b'-b| \le \delta_1$, $|(\omega^j)'-\omega^j| \le \delta_2$ и $|c'-c| \le \delta_3$, то нетрудно увидеть, что будет выполняться $|a' - a| < \delta + \delta_1 + \delta_2 + \delta_3$, где

$$\delta = |2^{-m} + 2^{-m} i| = 2^{1/2-m}, \tag{49}$$

поскольку имеем $\big|(\omega^j)'c' - \omega^j c\big| = \big|\big((\omega^j)' - \omega^j\big)c' + \omega^j(c' - c)\big| \le \delta_2 + \delta_3$, а δ превышает максимальную погрешность усечения. Вычисления приближений $(\omega^j)'$ начинаются с приближений ω_r' к числам, определенным в уравнениях (46), и можно считать, что вычисления в (46) выполняются с допустимой погрешностью, такой, что выполняется $|\omega_r' - \omega_r| < \delta$. Тогда из уравнения (47) следует, что $\big|(\omega^j)' - \omega^j\big| < (2k-1)\delta$ при всех j, поскольку ошибка обусловлена не более чем k приближениями и не более чем $k-1$ усечениями.

Если перед началом любого прохода быстрого преобразования Фурье ошибка не превышает ϵ, то операции этого прохода имеют вид (48), где $\delta_1 = \delta_3 = \epsilon$ и $\delta_2 = (2k-1)\delta$. Тогда ошибки после выполнения прохода не будут превышать $2\epsilon + 2k\delta$. Так как на нулевом проходе ошибок нет, по индукции можно показать, что после j-го прохода ошибка будет ограничена $(2^j - 1) \cdot 2k\delta$ и вычисленные значения $\hat{u}_s$ будут удовлетворять неравенству $|(\hat{u}_s)' - \hat{u}_s| < (2^k - 1) \cdot 2k\delta$. Аналогичное неравенство можно получить и для $(\hat{v}_s)'$; тогда

$$|(\hat{w}_s)' - \hat{w}_s| < 2(2^k - 1) \cdot 2k\delta + \delta < (4k2^k - 2k)\delta.$$

В процессе обратного преобразования накапливаются дополнительные ошибки, однако большинство из них подавляется при делении на $K = 2^k$. По этой же причине приходим к выводу, что вычисленные значения величин w_r будут удовлетворять неравенству

$$\big|(\hat{w}_r)' - \hat{w}_r\big| < 2^k(4k2^k - 2k)\delta + (2^k - 1)2k\delta; \quad |w_r' - w_r| < 4k2^k\delta. \tag{50}$$

Для округления $2^{2k+2l}w'_r$ до правильного целого числа W_r необходимо обеспечить достаточную точность вычислений, поэтому должно выполняться неравенство

$$2^{2k+2l+2+\lg k+k+1/2-m} \le \tfrac{1}{2}, \tag{51}$$

т. е. $m \ge 3k+2l+\lg k+7/2$. Для этого достаточно потребовать, чтобы выполнялось

$$k \ge 7 \qquad \text{и} \qquad m \ge 4k+2l. \tag{52}$$

Соотношения (41) и (52) могут быть использованы для определения таких значений параметров k, l и m, чтобы на операцию умножения затрачивалось $O(n)+O(Mnk/l)$ единиц времени, где M — время умножения m-битовых дробных частей.

Например, для компьютера MIX предположим, что перемножаются двоичные числа, каждое из которых представлено $n = 2^{13} = 8192$ бит. Можно выбрать значения параметров $k = 11$, $l = 8$, $m = 60$, так что требуемые операции с m-битовыми числами будут ни чем иным, как операциями умножения с числами в арифметике с удвоенной точностью. Поэтому необходимое время выполнения M операций умножения m-битовых комплексных чисел будет сравнительно малым. Если положить, к примеру, $k = l = 15$, то придется иметь дело с операциями утроенной точности, что выходит за пределы возможностей памяти машины MIX. На компьютере с бóльшими возможностями можно, положив $k = l = 27$ и $m = 144$, перемножить два гигабитовых числа.

Дальнейший анализ выбора значений параметров k, l и m приводит к удивительному выводу: *для большинства практических случаев можно считать M постоянным, а время выполнения процедуры умножения Шёнхаге-Штрассена пропорционально n.* Суть в том, что можно выбрать $k = l$ и $m = 6k$; такой выбор k приводит к тому, что время выполнения всегда меньше $\lg n$, поэтому потребуется не более чем ушестеренная точность вычислений, даже если n больше размера машинного слова. (В частности, n может быть больше размера индексного регистра, так что, возможно, числа u и v не поместятся в основной памяти.)

Таким образом, практическая сторона быстрого умножения решена, за исключением возможного усовершенствования процедуры выбора констант. Действительно, алгоритм свертки для произвольных целых чисел, приведенный в упр. 4.6.4–59, является лучшим практическим решением проблемы умножения с высокой точностью. Наш интерес к проблеме умножения больших чисел отчасти теоретический, поскольку интересно исследовать предельные ограничения на вычислительную сложность алгоритма. Итак, оставим на время практическую сторону вопроса и предположим, что число n чрезвычайно велико, возможно, значительно больше числа атомов во Вселенной. Можно считать m приблизительно равным $6\lg n$ и использовать для умножения m-битовых чисел тот же алгоритм рекурсивно. Время его выполнения будет удовлетворять выражению $T(n) = O\big(nT(\log n)\big)$. Следовательно,

$$T(n) \le C\,n(C\lg n)(C\lg\lg n)(C\lg\lg\lg n)\ldots, \tag{53}$$

где произведение продолжается до тех пор, пока множитель $\lg\ldots\lg n \le 2$.

Шёнхаге и Штрассен в своей работе показали, как улучшить эту теоретическую верхнюю границу до $O(n\log n\log\log n)$, используя *целые* числа ω, чтобы распространить быстрое преобразование Фурье на целые числа по модулю $2^\epsilon+1$. Эта верхняя граница применима к машине Тьюринга, т. е. к компьютерам с ограниченной памятью и конечным числом лент произвольной длины.

Для более мощного компьютера со случайной выборкой любого числа слов ограниченного размера Шёнхаге обратил внимание на то, что верхняя граница снижается до $O(n \log n)$. Теперь можно выбрать параметры $k = l$ и $m = 6k$ и появляется время для формирования полной таблицы всевозможных произведений xy при $0 \le x, y < 2^{\lceil m/12 \rceil}$. (Количество таких произведений равно 2^k или 2^{k+1}. После этого можно вычислить любой элемент таблицы за $O(k)$ шагов, добавив к нему одного из предшественников.) Следовательно, $O(k\,2^k) = O(n)$ шагов будет достаточно для вычисления. В этом случае M — время, необходимое для выполнения 12-разрядной арифметической операции по основанию $2^{\lceil m/12 \rceil}$, а отсюда следует, что $M = O(k) = O(\log n)$, так как 1-разрядное умножение можно выполнить по таблицам. (Время выборки слова из памяти считается пропорциональным количеству битов, содержащемуся в адресе слова.)

Более того, в 1979 году Шёнхаге обнаружил, что *машина с указателями* может выполнять умножение n-битовых чисел за $O(n)$ шагов (см. упр. 12).

Такие устройства (которые называются также машинами с модификацией хранения и связными автоматами) при $n \to \infty$ реализуют, как нам кажется, лучшие вычислительные модели, что было описано в разделе 2.6. Таким образом, можно сделать вывод о том, что умножение за $O(n)$ шагов возможно как на практике, так и теоретически.

В 1986 году Д. В. Чудновский, Г. В. Чудновский, М. М. Денно (M. M. Denneau) и С. Г. Юнис (S. G. Younis) сконструировали необычный компьютер общего назначения, названный Маленьким Ферма (Little Fermat), который мог быстро умножать большие числа. Компьютер оснащен аппаратными средствами быстрого выполнения арифметических операций по модулю $2^{256} + 1$ над 257-битовыми словами. Тогда свертка массивов, состоящих из 256 слов, может выполняться с помощью 256 умножений однословных чисел вместе с тремя дискретными преобразованиями, требующими только операций сложения, вычитания и сдвига. Это позволило, основываясь на конвейерном принципе организации цикла длительностью примерно 60 нс, перемножить два 10^6-битовых целых числа меньше чем за 0.1 с [*Proc. Third Int. Conf. on Supercomputing* **2** (1988), 498–499; *Contemporary Math.* **143** (1993), 136].

D. Деление. Теперь при наличии эффективных программ для умножения рассмотрим обратную проблему. Оказывается, что деление может быть выполнено так же быстро, как и умножение, с точностью до постоянного множителя.

Чтобы разделить n-битовое число u на n-битовое число v, можно сначала найти n-битовое приближение к числу $1/v$, затем умножить его на u, что даст приближение $\hat{q}$ к u/v, и наконец выполнить еще одно умножение для внесения небольшой коррекции в $\hat{q}$, чтобы убедиться, что выполняется неравенство $0 \le u - qv < v$. Исходя из сказанного, достаточно иметь эффективный алгоритм, который формировал бы по заданному n-битовому числу приближенное значение числа, обратного n-битовому числу. Это может быть реализовано следующим алгоритмом, который использует "метод Ньютона", рассмотренный в разделе 4.3.1.

Алгоритм R (*Получение обратной величины с высокой точностью*). Пусть число v имеет двоичное представление $v = (0.v_1v_2v_3\ldots)_2$, где $v_1 = 1$. Этот алгоритм вычисляет приближение z числа $1/v$, такое, что

$$|z - 1/v| \le 2^{-n}. \tag{54}$$

R1. [Начальное приближение.] Присвоить $z \leftarrow \frac{1}{4}\lfloor 32/(4v_1 + 2v_2 + v_3)\rfloor$ и $k \leftarrow 0$.

R2. [Итерация по Ньютону.] (Здесь имеем число z в двоичном виде $(xx.xx\ldots x)_2$ с $2^k + 1$ знаками после разделяющей точки и $z \le 2$.) При помощи программы быстрого умножения точно вычислить $z^2 = (xxx.xx\ldots x)_2$. После этого точно вычислить $V_k z^2$, где $V_k = (0.v_1v_2\ldots v_{2^{k+1}+3})_2$. Затем присвоить $z \leftarrow 2z - V_k z^2 + r$, где r, удовлетворяющее неравенству $0 \le r < 2^{-2^{k+1}-1}$, прибавляется при необходимости для округления z, чтобы z было кратным $2^{-2^{k+1}-1}$. И наконец присвоить $k \leftarrow k + 1$.

R3. [Завершено?] Если $2^k < n$, то вернуться к шагу R2; в противном случае выполнение алгоритма заканчивается. ▮

Этот алгоритм основывается на алгоритме, предложенном С. А. Куком (S. A. Cook). Похожий алгоритм использовался при разработке арифметического блока компьютера [см. Anderson, Earle, Goldschmidt, Powers, *IBM J. Res. Dev.* **11** (1967), 48–52]. Конечно, нужно тщательно проверять точность алгоритма R, так как он находится на грани того, чтобы быть некорректным. По индукции докажем, что в начале и в конце шага R2 выполняются неравенства

$$z \le 2 \qquad \text{и} \qquad |z - 1/v| \le 2^{-2^k}. \tag{55}$$

Обозначим через z_k значение z, вычисленное после k итераций на шаге R2, и пусть $\delta_k = 1/v - z_k$. При $k = 0$ имеем

$$\delta_0 = 1/v - 8/v' + (32/v' - \lfloor 32/v'\rfloor)/4 = \eta_1 + \eta_2,$$

где $v' = (v_1v_2v_3)_2$ и $\eta_1 = (v' - 8v)/vv'$. При этом параметры η_1 и η_2 удовлетворяют неравенствам $-\frac{1}{2} < \eta_1 \le 0$ и $0 \le \eta_2 < \frac{1}{4}$. Значит, $|\delta_0| < \frac{1}{2}$. Теперь предположим, что второе неравенство в (55) удовлетворяется при k. Тогда

$$\begin{aligned}
\delta_{k+1} = 1/v - z_{k+1} &= 1/v - z_k - z_k(1 - z_kV_k) - r \\
&= \delta_k - z_k(1 - z_kv) - z_k^2(v - V_k) - r \\
&= \delta_k - (1/v - \delta_k)v\delta_k - z_k^2(v - V_k) - r \\
&= v\delta_k^2 - z_k^2(v - V_k) - r.
\end{aligned}$$

Отсюда следует, что $0 \le v\delta_k^2 < \delta_k^2 \le (2^{-2^k})^2 = 2^{-2^{k+1}}$ и

$$0 \le z^2(v - V_k) + r < 4(2^{-2^{k+1}-3}) + 2^{-2^{k+1}-1} = 2^{-2^{k+1}},$$

так что $|\delta_{k+1}| \le 2^{-2^{k+1}}$. Осталось еще проверить первое неравенство в (55). Чтобы убедиться в том, что $z_{k+1} \le 2$, рассмотрим три случая:

a) $V_k = \frac{1}{2}$; тогда $z_{k+1} = 2$;

b) $V_k \ne \frac{1}{2} = V_{k-1}$; тогда $z_k = 2$, поэтому $2z_k - z_k^2V_k \le 2 - 2^{-2^{k+1}-1}$;

c) $V_{k-1} \ne \frac{1}{2}$; тогда $z_{k+1} = 1/v - \delta_{k+1} < 2 - 2^{-2^{k+1}} \le 2$, так как $k > 0$.

Время, затрачиваемое на выполнение алгоритма R, ограничено сверху количеством циклов, равным

$$2T(4n) + 2T(2n) + 2T(n) + 2T(\tfrac{1}{2}n) + \cdots + O(n),$$

где $T(n)$ — верхняя оценка времени, необходимого для выполнения операции умножения n-битовых чисел. Если для некоторой монотонно неубывающей функции $f(n)$ выражение для $T(n)$ имеет вид $nf(n)$, то

$$T(4n) + T(2n) + T(n) + \cdots < T(8n), \tag{56}$$

так что деление может быть выполнено со скоростью, сравнимой со скоростью умножения, с точностью до постоянного множителя.

Р. П. Брент (R. P. Brent) в работе *JACM* **23** (1976), 242–251, показал, что если для умножения n-битовых чисел затрачивается $M(n)$ единиц времени, то для функций вида $\log x$, $\exp x$ и $\arctan x$ значения с n значащими битами можно вычислить за $O\big(M(n)\log n\big)$ шагов.

Е. Умножение в реальном времени. Вполне естественно возникает вопрос, можно ли в действительности выполнить умножение n-битовых чисел точно за n шагов. Мы уже сократили время выполнения с n^2 до порядка n шагов, но, может быть, его удастся сократить еще больше — до абсолютного минимума? Если мысленно покинуть бренный мир и вообразить себя в мире компьютеров с неограниченным числом компонентов, действующих одновременно, то возможен положительный ответ на этот вопрос.

Линейная итерационная конфигурация автоматов — это ряд устройств M_1, M_2, M_3, ..., каждое из которых может на каждом шаге находиться в некотором конечном множестве состояний. Все машины M_2, M_3, ... имеют *одинаковые* схемы, и их состояние в момент $t+1$ является функцией их собственного состояния в момент t, а также состояний в момент t их соседок справа и слева. Первая машина M_1 несколько отличается от остальных. Ее состояние в момент $t+1$ есть функция ее собственного состояния и состояния в момент t машины M_2, а также состояния *на входе* в момент t. *Выход* линейной итерационной конфигурации — это некоторая функция состояний машины M_1.

Пусть $u = (u_{n-1}\ldots u_1u_0)_2$, $v = (v_{n-1}\ldots v_1v_0)_2$ и $q = (q_{n-1}\ldots q_1q_0)_2$ — двоичные числа и пусть $uv + q = w = (w_{2n-1}\ldots w_1w_0)_2$. Примечательно, что линейная итерационная конфигурация может быть построена независимо от n, и, если в моменты $0, 1, 2, \ldots$ ввести (u_0, v_0, q_0), (u_1, v_1, q_1), (u_2, v_2, q_2), ..., то в моменты $1, 2, 3, \ldots$ на ее выходе будет w_0, w_1, w_2, ...

Это свойство можно сформулировать в терминах языка конструирования компьютеров, сказав, что можно сконструировать один модуль интегральной схемы, обладающий следующим свойством: если последовательно так смонтировать достаточно много подобных чипов, чтобы каждый из них соединялся только со своим соседом слева и справа, то результирующая схема выдаст $2n$-битовое произведение n-битовых чисел точно через $2n$ тактовых импульсов.

В основе этой конструкции лежит следующая идея. В момент 0 машина M_1 обрабатывает (u_0, v_0, q_0), поэтому в момент 1 она способна выдать результат $(u_0v_0 + q_0) \bmod 2$. Затем она обрабатывает (u_1, v_1, q_1) и в момент 2 может выдать результат $(u_0v_1 + u_1v_0 + q_1 + k_1) \bmod 2$, где k_1 — "перенос" влево, выполненный на предыдущем шаге. После этого машина обрабатывает (u_2, v_2, q_2) и выдает результат $(u_0v_2 + u_1v_1 + u_2v_0 + q_2 + k_2) \bmod 2$. Кроме того, ее состояние регистрирует значения u_2 и v_2 с тем, чтобы машина M_2 смогла обрабатывать эти величины в момент 3 и чтобы M_2 в момент 4 могла вычислить значение u_2v_2 для M_1. Машина M_1 дает

Таблица 2

УМНОЖЕНИЕ В ЛИНЕЙНОЙ ИТЕРАЦИОННОЙ КОНФИГУРАЦИИ

Время	Вход		Модуль M_1					Модуль M_2					Модуль M_3				
	u_j v_j	q_j	c	x_0 y_0	x_1 y_1	x y	z_2 z_1 z_0	c	x_0 y_0	x_1 y_1	x y	z_2 z_1 z_0	c	x_0 y_0	x_1 y_1	x y	z_2 z_1 z_0
0	1 1	1	0	0 0	0 0	0 0	0 0 0	0	0 0	0 0	0 0	0 0 0	0	0 0	0 0	0 0	0 0 0
1	1 1	1	1	1 1	0 0	0 0	0 1 0	0	0 0	0 0	0 0	0 0 0	0	0 0	0 0	0 0	0 0 0
2	1 1	0	2	1 1	1 1	0 0	1 0 0	0	0 0	0 0	0 0	0 0 0	0	0 0	0 0	0 0	0 0 0
3	0 0	1	3	1 1	1 1	1 1	0 1 1	0	0 0	0 0	0 0	0 0 1	0	0 0	0 0	0 0	0 0 0
4	1 1	0	3	1 1	1 1	0 0	1 0 1	1	1 1	0 0	0 0	0 0 1	0	0 0	0 0	0 0	0 0 0
5	0 0	0	3	1 1	1 1	1 1	0 1 1	2	1 1	0 0	0 0	0 0 1	0	0 0	0 0	0 0	0 0 0
6	0 0	0	3	1 1	1 1	0 0	1 0 0	3	1 1	0 0	1 1	0 1 0	0	0 0	0 0	0 0	0 0 0
7	0 0	0	3	1 1	1 1	0 0	0 0 0	3	1 1	0 0	0 0	0 1 0	1	1 1	0 0	0 0	0 0 1
8	0 0	0	3	1 1	1 1	0 0	0 0 0	3	1 1	0 0	0 0	0 1 0	2	1 1	0 0	0 0	0 0 0
9	0 0	0	3	1 1	1 1	0 0	0 0 0	3	1 1	0 0	0 0	0 0 1	3	1 1	0 0	0 0	0 0 0
10	0 0	0	3	1 1	1 1	0 0	0 0 1	3	1 1	0 0	0 0	0 0 0	3	1 1	0 0	0 0	0 0 0
11	0 0	0	3	1 1	1 1	0 0	0 0 0	3	1 1	0 0	0 0	0 0 0	3	1 1	0 0	0 0	0 0 0

команду машине M_2 начать умножение последовательности (u_2, v_2), (u_3, v_3), ..., а M_2 в конечном итоге дает команду машине M_3 выполнить умножение (u_4, v_4), (u_5, v_5) и т. д. К счастью, этот процесс выполняется без потери времени. Дальнейшие подробности читатель может почерпнуть из приводимого ниже формального описания.

Любой автомат может принимать 2^{11} состояний $(c, x_0, y_0, x_1, y_1, x, y, z_2, z_1, z_0)$, где $0 \le c < 4$ и каждое из x, y и z равно либо 0, либо 1. Первоначально все устройства находятся в состоянии $(0,0,0,0,0,0,0,0,0,0)$. Предположим, что при $j > 1$ в момент t машина M_j находится в состоянии $(c, x_0, y_0, x_1, y_1, x, y, z_2, z_1, z_0)$, ее соседка слева M_{j-1} находится в состоянии $(c^l, x_0^l, y_0^l, x_1^l, y_1^l, x^l, y^l, z_2^l, z_1^l, z_0^l)$, а соседка справа M_{j+1} — в состоянии $(c^r, x_0^r, y_0^r, x_1^r, y_1^r, x^r, y^r, z_2^r, z_1^r, z_0^r)$. Тогда машина M_j перейдет в момент $t+1$ в состояние $(c', x_0', y_0', x_1', y_1', x', y', z_2', z_1', z_0')$, где

$$\begin{aligned}
c' &= \min(c+1, 3), && \text{если } c^l = 3, && \text{иначе } 0;\\
(x_0', y_0') &= (x^l, y^l), && \text{если } c = 0, && \text{иначе } (x_0, y_0);\\
(x_1', y_1') &= (x^l, y^l), && \text{если } c = 1, && \text{иначе } (x_1, y_1);\\
(x', y') &= (x^l, y^l), && \text{если } c \ge 2, && \text{иначе } (x, y);
\end{aligned} \tag{57}$$

$(z_2' z_1' z_0')_2$ — двоичная запись для

$$z_0^r + z_1 + z_2^l + \begin{cases} x^l y^l & \text{при } c = 0;\\ x_0 y^l + x^l y_0 & \text{при } c = 1;\\ x_0 y^l + x_1 y_1 + x^l y_0 & \text{при } c = 2;\\ x_0 y^l + x_1 y + x y_1 + x^l y_0 & \text{при } c = 3. \end{cases} \tag{58}$$

Крайняя слева машина M_1 ведет себя почти так же, как и все остальные. Она функционирует так, как если бы в момент поступления (u, v, q) на ее вход слева от нее находилась машина в состоянии $(3,0,0,0,0,u,v,q,0,0)$. Выход всей конфигурации — компонента z_0 машины M_1.

Пример такой конфигурации, работающей с вводом

$$u = v = (\ldots 00010111)_2, \qquad q = (\ldots 00001011)_2,$$

приведен в табл. 2. Выходная последовательность дается в нижней части состояний машины M_1:

$$0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, \ldots,$$

что представляет собой число $(\ldots 01000011100)_2$, прочитанное справа налево.

В основу описанной конструкции положена похожая конструкция, предложенная А. Дж. Атрубиным (A. J. Atrubin) и описанная в *IEEE Trans.* **EC-14** (1965), 394–399.

Скорее всего, итеративная конфигурация оптимальна только в том случае, когда входные биты появляются последовательно. Если же они появляются одновременно, то следует предпочесть параллельные схемы реализации, которые вычисляют произведение двух n-битовых чисел после $O(\log n)$ задержек. Эффективные схемы такого рода описаны, например, К. С. Уоллесом (C. S. Wallace) в *IEEE Trans.* **EC-13** (1964), 14–17, и Д. Э. Кнутом (D. E. Knuth) в *The Stanford GraphBase* (New York: ACM Press, 1994), 270–279.

Ш. Виноград (S. Winograd) [*JACM* **14** (1967), 793–802] исследовал минимальное время умножения, достижимое в логической цепи, когда n задано и когда входные

данные поступают одновременно в произвольно закодированном виде. Аналогичные вопросы для случая, когда операции умножения и сложения должны поддерживаться одновременно, исследованы в работах A. C. Yao, *STOC* **13** (1981), 308–311; Mansour, Nisan and Tiwari, *STOC* **22** (1990), 235–243.

Умноженье — мое раздраженье,
И деленье — совсем не песня:
Золотое правило вызывает смятенье,
Ну а практика просто бесит!

— ИЗ РУКОПИСНОЙ КОЛЛЕКЦИИ
ДЖ. О. ХЭЛЛИУЭЛЛА (J. O. HALLIWELL) (с. 1570)

УПРАЖНЕНИЯ

1. [*22*] Идея, выраженная в неравенстве (2), при замене основания 2 основанием 10 может быть обобщена для десятичной системы. Используя это обобщение, вычислите произведение чисел 1234 и 2341 (сведите это произведение четырехзначных чисел к трем произведениям двузначных чисел, а каждое из последних — к произведениям однозначных чисел).

2. [*M22*] Докажите, что если на шаге T1 алгоритма T присвоить $R \leftarrow \lfloor\sqrt{Q}\rfloor$, то значение R останется неизменным или увеличится на единицу. (Поэтому, как было отмечено при описании данного шага, нет необходимости вычислять квадратный корень.)

3. [*M22*] Докажите, что последовательности q_k и r_k, определенные в алгоритме T, удовлетворяют неравенству $2^{q_k+1}(2r_k)^{r_k} \le 2^{q_{k-1}+q_k}$ при $k > 0$.

▸ **4.** [*28*] (K. Бейкер (K. Baker).) Покажите, что полином $W(x)$ лучше вычислять в точках $x = -r, \ldots, 0, \ldots, r$, чем в неотрицательных точках $x = 0, 1, \ldots, 2r$, как это делается в алгоритме T. Полином $U(x)$ можно записать в виде

$$U(x) = U_e(x^2) + xU_o(x^2).$$

Полиномы $V(x)$ и $W(x)$ могут быть выражены аналогично. Покажите, как использовать эту идею для повышения скорости вычислений на шагах T7 и T8.

▸ **5.** [*35*] Покажите, что если на шаге T1 алгоритма T вместо $R \leftarrow \lfloor\sqrt{Q}\rfloor$ присвоить $R \leftarrow \lceil\sqrt{2Q}\rceil + 1$ при соответствующих начальных значениях величин q_0, q_1, r_0 и r_1, то оценку (20) можно улучшить до $t_k \le q_{k+1}2^{\sqrt{2\lg q_{k+1}}}(\lg q_{k+1})$.

6. [*M23*] Докажите, что шесть чисел в уравнении (24) попарно просты.

7. [*M23*] Докажите (25).

8. [*M20*] Истинно ли следующее утверждение: можно игнорировать бит, обратный $(s_{k-1}, \ldots, s_0) \rightarrow (s_0, \ldots, s_{k-1})$ в (39), так как обратное преобразование Фурье в любом случае снова обратит биты.

9. [*M15*] Предположим, что в методе преобразования Фурье, изложенном в разделе, во всех случаях параметр ω заменяется на ω^q, где q — некоторое фиксированное целое число. Найдите простую связь между числами $(\tilde{u}_0, \tilde{u}_1, \ldots, \tilde{u}_{K-1})$, вычисленными при помощи обобщенного преобразования Фурье, и числами $(\hat{u}_0, \hat{u}_1, \ldots, \hat{u}_{K-1})$, полученными при $q = 1$.

10. [*M26*] В процессе вычислений по алгоритму умножения Шёнхаге-Штрассена значений $\hat{u}_s$ и $\hat{v}_s$ после масштабирования в (43) становится ясно, что все комплексные числа $A^{[j]}$, вычисленные при выполнении прохода j подпрограммы преобразования, будут по абсолютной величине меньше 2^{j-k}. Покажите, что при выполнении *третьего* преобразования Фурье (вычислении $\hat{\hat{w}}_r$) все значения $A^{[j]}$ будут по абсолютной величине меньше 1.

▸ **11.** [*M26*] Сколько должно быть автоматов в линейной итерационной конфигурации, определяемой (57) и (58) при фиксированном значении n, чтобы можно было вычислить произведение n-битовых чисел? (Заметим, что на автомат M_j влияет только компонента z_0^r машины, расположенной справа, поэтому можно убрать все автоматы, компонента z_0^r которых всегда нулевая для любых n-битовых чисел.)

▸ **12.** [*M41*] (А. Шёнхаге (A. Schönhage).) Назначение данного упражнения — доказать, что простая форма машины с указателем (разделяющей точкой) может выполнить умножение n-битовых чисел за $O(n)$ шагов. В машине отсутствуют встроенные возможности реализации арифметики; все, что она делает, — работает с указателями и узлами. Каждый узел имеет одно и то же конечное число полей связи, и имеется конечное множество связующих регистров. Операции, которые эта машина может выполнять, перечислены ниже:

i) считывание одного входного бита; если этот бит равен 0, то выполнение перехода;
ii) вывод 0 или 1;
iii) загрузка в регистр содержимого другого регистра или содержимого поля связи узла, на который указывает регистр;
iv) сохранение содержимого регистра в полях связи узла, на который указывает регистр;
v) переход в случае равенства двух регистров;
vi) создание нового узла и формирование в регистре указателя на него;
vii) остановка процесса.

Эффективно реализуйте метод преобразования Фурье на такой машине. [*Указание.* Покажите сначала, что для любого положительного N можно создать N узлов, представляющих целые числа $\{0, 1, \ldots, N-1\}$, причем узлы, которые представляют числа p, имеют указатели на узлы, представляющие числа $p+1$, $\lfloor p/2 \rfloor$ и $2p$. Такие узлы могут быть созданы за $O(N)$ шагов. Покажите, что в этом случае арифметика по основанию N моделируется легко. Например, чтобы найти узел для $(p+q) \bmod N$ и определить, являются ли $p+q \geq N$ указателями на p и q, такой арифметике потребуется $O(\log N)$ шагов. Кроме того, операция умножения может быть смоделирована за $O(\log N)^2$ шагов. Рассмотрим теперь алгоритм, приведенный в тексте раздела, при $k = l$, $m = 6k$ и $N = 2^{\lceil m/13 \rceil}$, так что все величины для арифметики в формате с фиксированной точкой представляются 13-разрядными целыми числами по основанию N. Таким образом, покажите, что каждый проход быстрого преобразования Фурье может быть реализован за $O(K + (N \log N)^2) = O(K)$ шагов с использованием следующей идеи. Каждая из K требуемых операций присвоения может быть "скомпилирована" для имитируемого компьютера наподобие `MIX` в виде ограниченного списка команд. При этом размер слова машины равен N, а команды для K машин, выполняющих операции параллельно, можно промоделировать за $O(K + (N \log N)^2)$ шагов при условии, что команды рассортированы таким образом, что все идентичные команды выполняются вместе. (Две команды идентичны, если у них одинаковый код, одинаковое содержимое регистров и операнды расположены в одинаковых ячейках памяти.) Обратите внимание на то, что $N^2 = O(n^{12/13})$, а потому $(N \log N)^2 = O(K)$.]

13. [*M25*] (А. Шёнхаге.) Основываясь на результатах этого раздела для $m = n$, получите хорошую верхнюю оценку для времени, необходимого, чтобы умножить m-битовое число на n-битовое число в случае, если оба числа очень большие, но число n значительно больше числа m.

14. [*M42*] Напишите программу реализации алгоритма T с учетом сделанных в упр. 4 усовершенствований. Сравните ее с программой, разработанной для алгоритма 4.3.1M, и с программой, основанной на использовании (2), чтобы увидеть, насколько большим должно быть число n, чтобы проявилось усовершенствование алгоритма T.

15. [*M49*] (Ш. А. Кук (S. A. Cook).) Алгоритм умножения называется *алгоритмом реального времени*, если ввод $(k+1)$-го бита операнда выполняется только после вычисления k-го выходного бита. Какие самые быстрые алгоритмы умножения в реальном времени можно реализовать на различных автоматах?

▸ **16.** [*25*] Докажите, что для дискретного преобразования Фурье по (35) требуется всего $O(K \log K)$ арифметических операций, даже если K не равно степени 2. [*Указание.* Перепишите выражение (35) в виде

$$\hat{u}_s = \omega^{-s^2/2} \sum_{0 \le t < K} \omega^{(s+t)^2/2} \omega^{-t^2/2} u_t$$

и выразите этот результат в виде свертки.]

17. [*M26*] Схема умножения (2) Карацубы при получении n-разрядного произведения выполняет K_n 1-разрядных операций умножения, где $K_1 = 1$, $K_{2n} = 3K_n$ и $K_{2n+1} = 2K_{n+1} + K_n$ при $n \ge 1$. "Решите" это рекуррентное уравнение путем поиска точной формулы для K_n, когда $n = 2^{e_1} + 2^{e_2} + \cdots + 2^{e_t}$, $e_1 > e_2 > \cdots > e_t \ge 0$.

▸ **18.** [*M30*] Разработайте схему выделения памяти для промежуточных результатов при выполнении операций умножения по рекурсивному алгоритму, основанному на уравнении (2). Заданы два N-разрядных целых числа u и v, каждое из которых занимает N разрядов памяти. Покажите, как ограничить вычисления, чтобы произведение uv оказалось в последних значащих $2N$ разрядах $(3N + O(\log N))$-разрядной области рабочего пространства.

▸ **19.** [*M23*] Покажите, как вычислить $uv \bmod m$ при ограниченном количестве операций, оговоренных в правилах упр. 3.2.1.1–11, если имеется возможность проверить, будет ли один операнд меньше другого. Оба числа u и v переменные, но m постоянно. *Указание.* Рассмотрите вопрос декомпозиции в (2).

4.4. ПРЕОБРАЗОВАНИЕ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ

Если бы наши предки, изобретая арифметику, вели счет при помощи двух рук или восьми пальцев, а не десяти "цифр", у нас никогда бы не было хлопот с разработкой программ двоично-десятичного преобразования чисел. (И, возможно, мы бы никогда столько не узнали о системах счисления.) В этом разделе будут рассмотрены вопросы, связанные с преобразованием чисел из позиционной системы счисления по одному основанию в позиционную систему счисления по другому основанию. Конечно, этот процесс наиболее важен для двоичных компьютеров при преобразовании входных данных, представленных в десятичном формате, в двоичный формат и преобразовании результатов из двоичного формата обратно в десятичный.

А. Четыре основных метода. Преобразование чисел из двоичной системы счисления в десятичную и обратно — одна из наиболее машинно-зависимых операций, поскольку разработчикам компьютеров приходится постоянно изобретать различные способы аппаратной реализации этой операции. В связи с этим далее будут рассматриваться только основные принципы решения данной задачи, на основании которых программисты могут выбирать процедуры, наиболее подходящие для реализации на компьютере конкретной конфигурации.

Будем предполагать, что операции преобразований выполняются только с неотрицательными числами, так как манипулирование знаками легко учесть.

Предположим, что выполняется преобразование из системы счисления по основанию b в систему счисления по основанию B. (Обобщение для систем счисления со смешанным основанием рассматривается в упр. 1 и 2.) Большинство процедур преобразования из одного основания в другое основано на операциях умножения и деления, использующих один из четырех методов, которые описываются ниже. Два первых метода применяются для преобразования целых чисел (разделяющая точка расположена справа), а два других — для дробных частей чисел (разделяющая точка расположена слева). Чаще всего нельзя *точно* выразить конечную дробную часть по основанию b $(0.u_{-1}u_{-2}\ldots u_{-m})_b$ в виде конечной дробной части по основанию B $(0.U_{-1}U_{-2}\ldots U_{-M})_B$. Например, десятичная дробь $\frac{1}{10}$ в двоичном формате представляется бесконечной дробью $(0.0001100110011\ldots)_2$. В связи с этим приходится иногда округлять результат до M разрядов.

Метод 1, а (Деление на B с использованием представления чисел в формате по основанию b). Для заданного целого числа u можно получить представление в формате по основанию B вида $(\ldots U_2U_1U_0)_B$, выполняя

$$U_0 = u \bmod B, \qquad U_1 = \lfloor u/B \rfloor \bmod B, \qquad U_2 = \lfloor \lfloor u/B \rfloor / B \rfloor \bmod B, \qquad \ldots$$

до тех пор, пока не окажется $\lfloor \ldots \lfloor \lfloor u/B \rfloor / B \rfloor \ldots / B \rfloor = 0$.

Метод 1, b (Умножение на B с использованием представления чисел в формате по основанию b). Если представление числа u по основанию b имеет вид $(u_m \ldots u_1u_0)_b$, то можно, воспользовавшись арифметическими операциями с числами, которые представлены в формате по основанию B, получить полином $u_mb^m + \cdots + u_1b + u_0 = u$ в виде

$$\big(\ldots(u_m\,b + u_{m-1})\,b + \cdots\big)\,b + u_1\big)\,b + u_0.$$

Метод 2, a (Умножение на b с использованием представления чисел в формате по основанию B). Для данного дробного числа u можно вычислить значения разрядов $(.U_{-1}U_{-2}\ldots)_B$ его представления по основанию B следующим образом:

$$U_{-1} = \lfloor uB \rfloor, \quad U_{-2} = \lfloor \{uB\}B \rfloor, \quad U_{-3} = \lfloor \{\{uB\}B\}B \rfloor, \quad \ldots,$$

где $\{x\}$ означает $x \bmod 1 = x - \lfloor x \rfloor$. Чтобы округлить результат до M разрядов, вычисления можно прервать после получения U_{-M}, причем если $\{\ldots\{\{uB\}B\}\ldots B\}$ больше $\frac{1}{2}$, то значение U_{-M} следует увеличить на единицу. (Заметим, однако, что эта операция может привести к необходимости выполнения переносов, которые должны быть при помощи арифметических операций по основанию B включены в результат. Было бы проще перед началом вычислений прибавить к исходному числу u константу $\frac{1}{2}B^{-M}$, но это может привести к неправильному результату, если в компьютере число $\frac{1}{2}B^{-M}$ не может быть точно представлено в формате по основанию b. Заметим также, что возможно округление результата до $(1.00\ldots0)_B$, если $b^m \geq 2B^M$.)

В упр. 3 рассматривается обобщение этого метода на случай *переменного* M, достаточно большого для представления исходного числа с заданной точностью. В такой ситуации проблема переносов не возникает.

Метод 2, b (Деление на b с использованием представления чисел в формате по основанию B). Если число u представлено по основанию b в виде $(0.u_{-1}u_{-2}\ldots u_{-m})_b$, то можно, используя арифметические операции по основанию B, вычислить $u_{-1}b^{-1} + u_{-2}b^{-2} + \cdots + u_{-m}b^{-m}$ в виде

$$\big((\ldots(u_{-m}/b + u_{1-m})/b + \cdots + u_{-2})/b + u_{-1}\big)/b.$$

Необходимо внимательно следить за погрешностями, возникающими при усечениях или округлениях во время выполнения операции деления на b; они, как правило, незначительны, но это бывает не всегда.

Подведем итоги. Методы 1, a; 1, b; 2, a и 2, b предоставляют по два возможных способа преобразования целых и дробных чисел. И, конечно, существует возможность выполнять преобразования целых чисел в дробные и наоборот путем умножения или деления на соответствующую степень b или B. Поэтому для выполнения преобразования имеется на выбор по крайней мере четыре метода.

B. Преобразование с однократной точностью. Чтобы проиллюстрировать эти методы, предположим, что MIX — двоичный компьютер и нужно преобразовать неотрицательное целое число u, представленное в двоичном формате, в десятичный формат, т. е. получить $b = 2$ и $B = 10$. Метод 1, a можно запрограммировать следующим образом.

```
    ENT1 0          Присвоить j ← 0.
    LDX  U
    ENTA 0          Присвоить rAX ← u.
1H  DIV  =10=       (rA, rX) ← (⌊rAX/10⌋, rAX mod 10).
    STX  ANSWER,1   U_j ← rX.                              (1)
    INC1 1          j ← j + 1.
    SRAX 5          rAX ← rA.
    JXP  1B         Повторять до тех пор,
                        пока в результате не получим нуль. ∎
```

Для вычисления M разрядов необходимо затратить $18M + 4$ циклов.

В методе 1, а используется операция деления на 10. В методе 2, а используется *умножение* на 10, так что программа, реализующая этот метод, может выполняться немного быстрее. Но, применяя метод 2, а, нужно иметь дело с дробями, а это приводит к интересной ситуации. Пусть w — размер машинного слова, и положим, что $u < 10^n < w$. При помощи одного деления можно найти q и r, где

$$wu = 10^n q + r, \qquad 0 \le r < 10^n. \tag{2}$$

Если же применить метод 2, а к дроби $(q+1)/w$, то за n шагов получим разряды числа u слева направо, так как

$$\left\lfloor 10^n \, \frac{q+1}{w} \right\rfloor = \left\lfloor u + \frac{10^n - r}{w} \right\rfloor = u. \tag{3}$$

(Эта идея предложена П. А. Саметом (P. A. Samet) и опубликована в журнале *Software Practice & Experience* **1** (1971), 93–96.)

Приведем соответствующую MIX-программу.

```
    JOV  OFLO        Удостовериться в отсутствии переполнения.
    LDA  U
    LDX  =10^n=      rAX ← wu + 10^n.
    DIV  =10^n=      rA ← q + 1, rX ← r.
    JOV  ERROR       Переход при u ≥ 10^n.
    ENT1 n-1         Присвоить j ← n − 1.
2H  MUL  =10=        Представим, что разделяющая точка
                         находится слева, rA = x.
    STA  ANSWER,1    Присвоить U_j ← ⌊10x⌋.
    SLAX 5           x ← {10x}.
    DEC1 1           j ← j − 1.
    J1NN 2B          Повторить для n > j ≥ 0.  ∎
```
(4)

Данная программа немного длиннее предыдущей, и на ее выполнение требуется $16n + 19$ циклов, так что при $n = M \ge 8$ она выполняется быстрее программы (1). Однако при наличии ведущих нулей программа (1) будет выполняться быстрее.

Программа (4) в представленном виде при $10^m < w < 10^{m+1}$ не может использоваться для преобразования целых чисел $u \ge 10^m$, так как нужно принять $n = m + 1$. В этом случае ведущий разряд числа u можно получить, вычисляя $\lfloor u/10^m \rfloor$; после этого можно преобразовать число $u \bmod 10^m$ по приведенному выше методу при $n = m$.

То обстоятельство, что разряды результата могут быть получены слева направо, может оказаться полезным в некоторых приложениях (например, при последовательном выводе разрядов результата на печать). Итак, метод, применимый для дробей, можно использовать и для преобразований целых чисел, хотя, применив неточное деление при преобразованиях, придется выполнять численный анализ метода.

В методе 1, а деление на 10 можно заменить двумя умножениями. Это может оказаться существенным, поскольку преобразование оснований часто выполняется в компьютерах-"сателлитах", в которых отсутствует встроенная процедура деления. Если предположить, что x — приближение к $\frac{1}{10}$, так что

$$\frac{1}{10} < x < \frac{1}{10} + \frac{1}{w},$$

то легко доказать (см. упр. 7), что $\lfloor ux \rfloor = \lfloor u/10 \rfloor$ или $\lfloor u/10 \rfloor + 1$, пока $0 \le u < w$. Поэтому при вычислении $u - 10\lfloor ux \rfloor$ можно определить значение $\lfloor u/10 \rfloor$:

$$\lfloor u/10 \rfloor = \lfloor ux \rfloor - \lceil u < 10\lfloor ux \rfloor \rceil. \tag{5}$$

Одновременно можно определить $u \bmod 10$. MIX-программа, выполняющая преобразование с использованием (5), приведена в упр. 8. На вычисление каждого разряда она затрачивает около 33 циклов.

Метод 1, а может быть с успехом применен и в компьютерах, в которых отсутствуют встроенные команды как деления, так и умножения, путем выполнения операций сдвига и сложения, как продемонстрировано в упр. 9.

Другой способ преобразования из двоичного представления в десятичное заключается в использовании метода 1, b, но при этом необходимо имитировать удвоение в *десятичной* системе счисления. Этот способ в общем случае наиболее подходит для аппаратной реализации в компьютере; тем не менее процесс удвоения в десятичной системе счисления можно и запрограммировать при помощи операций двоичного сложения, двоичного сдвига и двоичного извлечения или маскирования (побитовое AND), как показано в табл. 1, составленной Петером Л. Монтгомери (Peter L. Montgomery).

Таблица 1

УДВОЕНИЕ ДЕСЯТИЧНОГО ЧИСЛА, ЗАКОДИРОВАННОГО В ДВОИЧНОЙ СИСТЕМЕ

Операция	*Общий вид*	*Пример*
1. Задать число	$u_{11}\,u_{10}\,u_9\,u_8\;u_7\,u_6\,u_5\,u_4\;u_3\,u_2\,u_1\,u_0$	0011 0110 1001 = 3 6 9
2. Прибавить 3 к каждому разряду	$v_{11}\,v_{10}\,v_9\,v_8\;v_7\,v_6\,v_5\,v_4\;v_3\,v_2\,v_1\,v_0$	0110 1001 1100
3. Извлечь каждый старший бит	$v_{11}\;\;0\;\;0\;\;0\;\;v_7\;\;0\;\;0\;\;0\;\;v_3\;\;0\;\;0\;\;0$	0000 1000 1000
4. Сдвинуть вправо на 2 разряда и вычесть	$0\;\;v_{11}\;\;v_{11}0\;\;\;0\;\;v_7\;\;v_7\;\;0\;\;\;0\;\;v_3\;\;v_3\;\;0$	0000 0110 0110
5. Прибавить исходное число	$w_{11}w_{10}w_9w_8\;w_7w_6w_5w_4\;w_3w_2w_1w_0$	0011 1100 1111
6. Прибавить исходное число	$x_{12}\,x_{11}\,x_{10}\,x_9\,x_8\;x_7\,x_6\,x_5\,x_4\;x_3\,x_2\,x_1\,x_0$	0 0111 0011 1000 = 7 3 8

При помощи этого метода значение каждого разряда d заменяется на $2d$ при $0 \le d \le 4$ и на $6 + 2d = (2d - 10) + 2^4$ при $5 \le d \le 9$, что и требуется для удвоения десятичных чисел, каждый разряд которых закодирован четырьмя битами.

Другая идея состоит в том, чтобы хранить таблицу степеней двоек в десятичном формате и складывать соответствующие степени, имитируя операции десятичного сложения. В разделе 7.1 описываются приемы оперирования битами.

И наконец, для преобразования целых чисел, представленных в двоичном формате, в целые числа, представляемые в десятичном формате, можно использовать даже метод 2, b. Можно найти q, как в (2), а затем имитировать операцию десятичного деления $q+1$ на w, используя процесс "уполовинивания" (упр. 10), аналогичный описанному выше процессу удваивания, сохраняя при этом в результате только первые n разрядов справа от разделяющей точки. Похоже, что в этом случае метод 2, b не имеет каких-либо преимуществ по сравнению с остальными тремя методами, проанализированными выше, тем не менее подтверждается высказанный выше тезис о том, что для преобразования целых чисел из одной системы счисления в другую существует четыре различных метода.

Теперь рассмотрим преобразование из десятичной системы счисления в двоичную (т. е. $b = 10$, $B = 2$). Метод 1, а позволяет имитировать операцию десятичного деления на 2, что допустимо, но предпочтительнее реализовать ее аппаратно, а не программно (см. упр. 10).

В большинстве случаев наиболее удобным методом преобразования из десятичной системы счисления в двоичную является метод 1, b. В приводимой ниже MIX-программе принято, что число $(u_m \ldots u_1 u_0)_{10}$ содержит по крайней мере два разряда, подлежащих преобразованию, и неравенство $10^{m+1} < w$ выполняется так, чтобы не возникало переполнение.

```
    ENT1 M-1        Присвоить j ← m − 1.
    LDA  INPUT+M    Присвоить U ← u_m.
1H  MUL  =10=
    SLAX 5                                          (6)
    ADD  INPUT,1    U ← 10U + u_j.
    DEC1 1
    J1NN 1B         Повторить при m > j ≥ 0.  ▮
```

Операция умножения на 10 может быть заменена операциями сдвига и сложения.

В упр. 19 рассмотрен более быстрый, возможно, способ преобразования, в котором вместо $m - 1$ операций умножения используется примерно $\lg m$ операций умножения, маскирования и сложения.

Для преобразования в двоичный формат десятичных дробей $(0.u_{-1}u_{-2}\ldots u_{-m})_{10}$ можно воспользоваться методом 2, b или, в более общем случае, сначала преобразовать целое число $(u_{-1}u_{-2}\ldots u_{-m})_{10}$ при помощи метода 1, а, а затем разделить результат на 10^m.

C. Вычисления вручную. Иногда в процессе программирования возникает необходимость выполнить преобразование чисел вручную, а поскольку в обычных школах этому пока что не учат, имеет смысл кратко обсудить здесь этот вопрос. Известны простые методы преобразования чисел из десятичного формата в восьмеричный и обратно, выполняемые вручную; этим методам легко научиться, так что они должны стать известными более широко.

Преобразование целых чисел из восьмеричной системы счисления в десятичную. Простейшим является преобразование из восьмеричного формата в десятичный. Этот способ, по-видимому, впервые опубликовал Уолтер Соден (Walter Soden) в *Math. Comp.* **7** (1953), 273–274. Чтобы выполнить преобразование, нужно записать данное восьмеричное число, удвоить на k-м шаге k ведущих разрядов,

используя десятичную арифметику, и вычесть полученный результат из $(k+1)$ ведущих разрядов при помощи опять же десятичной арифметики. Если заданное число содержит $(m+1)$ разрядов, то процесс прекращается через m шагов. Удачной оказалась идея ввода разделяющей точки для того, чтобы выделить удваиваемые разряды, как показано в следующем примере. Это помогает исключить возможные ошибки.

Пример 1. Преобразовать число $(5325121)_8$ в десятичный формат.

```
  5.3 2 5 1 2 1
- 1 0
-------
  4 3.2 5 1 2 1
-   8 6
---------
  3 4 6.5 1 2 1
-   6 9 2
-----------
  2 7 7 3.1 2 1
-   5 5 4 6
-------------
  2 2 1 8 5.2 1
-   4 4 3 7 0
---------------
  1 7 7 4 8 2.1
-   3 5 4 9 6 4
-----------------
  1 4 1 9 8 5 7
```

Результат: $(1419857)_{10}$.

Достаточно надежный способ проверки вычислений состоит в "выбрасывании девяток": сумма разрядов десятичного числа должна быть конгруэнтной по модулю 9 попеременно сумме и разности разрядов числа, представленного в восьмеричном формате, причем крайний справа разряд последнего числа берется со знаком "плюс". В вышеприведенном примере имеем $1+4+1+9+8+5+7=35$ и $1-2+1-5+2-3+5=-1$; разность равна 36 (кратна 9). Если проверка дала отрицательный результат, то она повторяется с $(k+1)$ ведущими разрядами после k-го шага и местоположение ошибки определяется при помощи процедуры двоичного поиска. Другими словами, можно определить, где произошла ошибка, начав с проверки среднего результата, и затем в зависимости от того, верен ли результат, применить ту же процедуру к первой или второй части вычислений.

Ввиду того, что существует один шанс из девяти, что два *случайных* целых числа будут отличаться по модулю девять, надежность процедуры выбрасывания девяток равна только 89%. Более надежный способ проверки — преобразовать результат обратно в восьмеричный формат с использованием обратного метода, который и будет сейчас рассмотрен.

Преобразование целых чисел, представленных в десятичном формате, в восьмеричный формат. Для выполнения обратного преобразования можно использовать аналогичную процедуру. Запишем заданное в десятичном формате число, удвоим на k-м шаге k ведущих разрядов, используя представление в *восьмеричном* формате, и *сложим* полученные $(k+1)$ ведущих разрядов, опять же используя представление в *восьмеричном* формате. Для заданного числа, содержащего $(m+1)$ разрядов, процесс заканчивается через m шагов.

Пример 2. Преобразовать число $(1419857)_{10}$ в восьмеричный формат.

```
  1.4 1 9 8 5 7
+   2
  ---------------
  1 6.1 9 8 5 7
+   3 4
  ---------------
  2 1 5.9 8 5 7
+   4 3 2
  ---------------
  2 6 1 3.8 5 7
+   5 4 2 6
  ---------------
  3 3 5 6 6.5 7
+   6 7 3 5 4
  ---------------
  4 2 5 2 4 1.7
+ 1 0 5 2 5 0 2
  ---------------
  5 3 2 5 1 2 1        Результат: (5325121)8.
```

(Заметим, что при вычислении восьмеричного представления присутствуют невосьмеричные цифры 8 и 9.) Проверка результата может быть выполнена описанным выше способом. Этот метод был опубликован Шарлем П. Розье (Charles P. Rozier), *IEEE Trans.* **CE-11** (1962), 708–709.

Обе описанные процедуры представляют собой, по существу, вариации метода 1, b обобщенного преобразования из одной системы счисления в другую. Операции удваивания и вычитания в десятичной системе счисления подобны умножению на $10 - 2 = 8$. Операции удваивания и вычитания в восьмеричной системе счисления подобны умножению на $8 + 2 = 10$. Аналогичный метод существует и для преобразования из шестнадцатеричной системы счисления в десятичную, но это преобразование выполняется немного сложнее, так как вместо операции умножения на 2 оно включает операцию умножения на 6.

Чтобы запомнить оба этих метода, нужно уяснить, что при переводе числа из восьмеричной системы счисления в десятичную выполняется *вычитание*, так как представление чисел в десятичном формате короче, чем в восьмеричном. Аналогично при переводе числа из представления в десятичном формате в восьмеричный необходимо выполнять *сложение*. Вычисления выполняются в формате представления *результата*, а не в исходном формате представления числа; в противном случае требуемый результат получен не будет.

Преобразование дробей. Аналогичные методы, пригодные для столь же быстрого преобразования вручную дробей, неизвестны. Похоже, что наилучшим является метод 2, a, в котором с целью упрощения операций умножения на 10 или 8 выполняются операции удвоения и сложения. В этом случае критерий выбора сложения и/или вычитания меняется на обратный — при преобразовании чисел в десятичный формат выполняется сложение, а при преобразовании чисел в восьмеричный формат выполняется вычитание. Кроме того, при вычислениях используется формат, в котором представлено исходное число, а *не* формат, в котором подставляется результат (см. приведенные ниже примеры 3 и 4). Реализация этого процесса требует примерно в два раза больше вычислений, чем рассмотренный выше метод преобразования целых чисел.

Пример 3. Преобразование числа $(.14159)_{10}$ в восьмеричный формат.

$$
\begin{array}{l}
1\,4\,1\,5\,9 \\
\quad 2\,8\,3\,1\,8- \\
\hline
1.1\,3\,2\,7\,2 \\
\quad\;\; 2\,6\,5\,4\,4- \\
\hline
1.0\,6\,1\,7\,6 \\
\quad\;\;\; 1\,2\,3\,5\,2- \\
\hline
0.4\,9\,4\,0\,8 \\
\quad\;\;\;\; 9\,8\,8\,1\,6- \\
\hline
3.9\,5\,2\,4\,6 \\
\quad\;\;\;\;\; 1\,9\,0\,5\,2\,8- \\
\hline
7.6\,2\,1\,1\,2 \\
\quad\;\;\;\;\;\; 1\,2\,4\,2\,2\,4- \\
\hline
4.9\,6\,8\,9\,6
\end{array}
\qquad \textit{Результат:}\ (.110374\ldots)_8.
$$

Пример 4. Преобразование числа $(.110374)_8$ в десятичный формат.

$$
\begin{array}{l}
.1\,1\,0\,3\,7\,4 \\
\quad 2\,2\,0\,7\,7\,0+ \\
\hline
1.3\,2\,4\,7\,3\,0 \\
\quad\;\; 6\,5\,1\,6\,6\,0+ \\
\hline
4.1\,2\,1\,1\,6\,0 \\
\quad\;\;\; 2\,4\,2\,3\,4\,0+ \\
\hline
1.4\,5\,4\,1\,4\,0 \\
\quad\;\;\;\; 1\,1\,3\,0\,3\,0\,0+ \\
\hline
5.6\,7\,1\,7\,0\,0 \\
\quad\;\;\;\;\; 1\,5\,6\,3\,6\,0\,0+ \\
\hline
8.5\,0\,2\,6\,0\,0 \\
\quad\;\;\;\;\;\; 1\,2\,0\,5\,4\,0\,0+ \\
\hline
6.2\,3\,3\,4\,0\,0
\end{array}
\qquad \textit{Результат:}\ (.141586\ldots)_{10}.
$$

D. Преобразование чисел с плавающей точкой. При выполнении преобразований чисел с плавающей точкой необходимо одновременно выполнять операции как с целой частью числа, так и с дробной, поскольку преобразование целой части числа оказывает влияние на дробную часть. Для преобразования числа $f \cdot 2^e$ в десятичный формат можно сначала представить 2^e в виде $F \cdot 10^E$ (обычно при помощи вспомогательных таблиц) и затем уже преобразовать Ff в десятичный формат. Аналогично можно умножить e на $\log_{10} 2$ и затем округлить результат до ближайшего целого числа E; после этого разделить $f \cdot 2^e$ на 10^E и преобразовать результат. Обратно, для преобразования числа $F \cdot 10^E$ в двоичный формат можно преобразовать F, а затем умножить его на число 10^E, представленное в формате с плавающей точкой (снова используя вспомогательные таблицы). Для уменьшения максимальных размеров вспомогательных таблиц используется обычная методика, основанная на применении нескольких операций умножения и/или деления, хотя это

может привести к распространению ошибки вследствие округления промежуточных результатов. Вопросы минимизации таких ошибок рассмотрены в упр. 17.

E. Преобразование с многократной точностью. Начинать преобразование очень длинных чисел удобнее всего с преобразования блоков разрядов, операции с которыми можно выполнять с однократной точностью. Затем следует объединить эти блоки, пользуясь простыми способами, которые специфичны для многократной точности. Например, пусть 10^n — наивысшая степень 10, меньшая, чем размер машинного слова. Тогда:

a) чтобы преобразовать *целое число* с многократной точностью из двоичного формата в десятичный, необходимо многократно разделить его на 10^n (выполняя таким образом преобразование из двоичной системы счисления в десятичную с основанием 10^n по методу 1, a); при помощи операций с однократной точностью получим n десятичных разрядов для каждой единицы представления в системе счисления с основанием 10^n;

b) чтобы преобразовать *дробную часть* числа с многократной точностью из двоичного формата в десятичный, поступим подобным образом, умножив его на 10^n (т. е. использовав метод 2, a, где $B = 10^n$);

c) чтобы преобразовать целое число с многократной точностью из десятичной системы счисления в двоичную, преобразуем сначала блоки по n разрядов; затем для перехода из системы счисления с основанием 10^n в двоичный формат используем метод 1, b;

d) для преобразования дробной части с многократной точностью из представления в десятичном формате в двоичный сначала выполним преобразование в систему с основанием 10^n, как и в процедуре (c), а затем используем метод 2, b.

F. История и библиография. Приемы преобразования чисел из одной системы счисления в другую использовались еще в древности в задачах, связанных с мерами, весами и деньгами, когда обычно приходилось иметь дело с системами счисления со смешанными основаниями. Эти преобразования обычно выполнялись с помощью вспомогательных таблиц. В 17 веке, когда шестидесятеричные дроби были вытеснены десятичными, возникла необходимость выполнять преобразование из одной системы счисления в другую с тем, чтобы можно было пользоваться имеющимися книгами астрономических таблиц. В 1667 году в книге под редакцией Вильяма Отреда (William Oughtred) *Clavis Mathematicæ* (см. гл. 6, раздел 18) был дан систематический метод перевода дробей из системы счисления с основанием 60 в систему счисления с основанием 10. (В первом издании книги Отреда 1631 года этот материал отсутствовал.) Правила преобразований из одной системы счисления в другую были сформулированы еще аль-Каши из Самарканда в его работе *Ключ к арифметике* (1427), в которой ясно изложены методы 1, a; 1, b и 2, a [*Историко-математические исследования* **7** (1954), 126–135], но в Европе его работа была неизвестна. В 18 веке американский математик Хью Джонс (Hugh Jones) для описания правил преобразования из восьмеричной системы счисления в десятичную ввел термины "octavation" (октавирование) и "decimation" (децимирование), но его методы оказались столь же неясными, как и его терминология. А. М. Лежандр заметил, что положительные целые числа могут быть легко преобразованы в двоичный формат с помощью повторного деления на 64 [*Théorie des Nombres* (Paris, 1798), 229].

В 1946 году Г. Г. Гольдстайн (H. H. Goldstine) и Дж. фон Нейман (J. von Neumann) в своем классическом сочинении *Planning and Coding Problems for an Electronic Computing Instrument* дали глубокий анализ проблемы преобразований из одной системы счисления в другую в связи с необходимостью обоснования применимости двоичной арифметики. (См. John von Neumann, *Collected Works* **5** (New York: Macmillan, 1963), 127–142.) Другая ранняя работа, посвященная преобразованию чисел из одной системы счисления в другую в двоичных компьютерах, была опубликована Ф. Кунс (F. Koons) и С. Лубкиным (S. Lubkin) в журнале *Math. Comp.* **3** (1949), 427–431, где они предложили не совсем обычный метод преобразований. Несколько позже Ф. Л. Бауэр (F. L. Bauer) и К. Замельсон (K. Samelson) опубликовали первое обсуждение проблемы преобразования чисел с плавающей точкой [*Zeit. für angewandte Math. und Physik* **4** (1953), 312–316].

Исторический интерес представляют также заметки Г. Т. Лэйка (G. T. Lake) [*CACM* **5** (1962), 468–469], в которых описываются методы аппаратной реализации преобразований и приводятся понятные примеры, и статья А. Х. Строуда (A. H. Stroud) и Д. Секреста (D. Secrest) [*Comp. J.* **6** (1963), 62–66], в которой рассмотрено преобразование чисел, заданных с многократной точностью. Преобразования *ненормализованных* чисел с плавающей точкой, сохраняющие соответствующую представлению "значимость", были рассмотрены Г. Кэннером (H. Kanner) в *JACM* **12** (1965), 242–246, и Н. Метрополисом (N. Metropolis) и Р. Л. Эшенхерстом (R. L. Ashenhurst) в *Math. Comp.* **19** (1965), 435–441. (См. также статью K. Sikdar, *Sankhyā* **B30** (1968), 315–334, и приведенный в ней список литературы.) Ф. Дж. Плогер (P. J. Plauger) в книге *The Standard C Library* (Prentice-Hall, 1992), 301–331, приводит подробное описание подпрограмм форматированного ввода-вывода целых чисел и чисел с плавающей точкой, написанных на языке программирования С.

УПРАЖНЕНИЯ

▸ **1.** [*25*] Обобщите метод 1, b таким образом, чтобы он был применим к позиционным системам счисления со смешанным основанием; преобразуйте выражение

$$a_m b_{m-1} \ldots b_1 b_0 + \cdots + a_1 b_0 + a_0 \quad \text{в} \quad A_M B_{M-1} \ldots B_1 B_0 + \cdots + A_1 B_0 + A_0,$$

где $0 \le a_j < b_j$ и $0 \le A_J < B_J$ при $0 \le j < m$ и $0 \le J < M$.

Проиллюстрируйте работу обобщенного таким образом метода на примере, выполнив перевод вручную величины "3 дня, 9 часов, 12 минут и 37 секунд" в длинные тонны, хандредвейты, стоуны, фунты и унции. (Пусть одна секунда равна одной унции. В британской системе весов 1 стоун равен 14 фунтам, 1 хандредвейт равен 8 стоунам, 1 длинная тонна равна 20 хандредвейтам.) Другими словами, пусть $b_0 = 60$, $b_1 = 60$, $b_2 = 24$, $m = 3$, $B_0 = 16$, $B_1 = 14$, $B_2 = 8$, $B_3 = 20$, $M = 4$. Задача заключается в поиске при помощи систематического метода, обобщающего метод 1, b, таких чисел $A_4, \ldots, A_0$, расположенных в надлежащих интервалах, чтобы $3b_2b_1b_0 + 9b_1b_0 + 12b_0 + 37 = A_4B_3B_2B_1B_0 + A_3B_2B_1B_0 + A_2B_1B_0 + A_1B_0 + A_0$. (Все арифметические операции должны выполняться в системе счисления со смешанным основанием.)

2. [*25*] Обобщите метод 1, а так, чтобы он был применим к позиционной системе счисления со смешанным основанием, как в упр. 1, и приведите пример работы полученного обобщения, решив вручную ту же задачу преобразования, что и в упр. 1.

▸ **3.** [*25*] (Д. Таранто (D. Taranto).) При преобразовании дробей остается открытым вопрос о количестве разрядов представления результата. Разработайте простое обобщение

метода 2, a, такое, что для заданных двух положительных дробей u и ϵ, принимающих значения между 0 и 1 и представленных в формате по основанию b, дробь u преобразуется в свой округленный эквивалент по основанию B, который имеет достаточный размер M справа от разделяющей точки, чтобы обеспечить выполнение неравенства $|U-u|<\epsilon$. (В частности, если u кратно b^{-m} и $\epsilon=b^{-m}/2$, значение U будет представлено достаточным количеством разрядов, так что по заданным U и m дробь u может быть восстановлена точно. Заметим, что M может равняться нулю. Например, если $\epsilon\le\frac{1}{2}$ и $u>1-\epsilon$, то правильный ответ — $U=1$.)

4. [*M21*] (a) Докажите, что любое вещественное число с конечным *двоичным* представлением имеет также конечное *десятичное* представление. (b) Найдите простое соотношение между положительными числами b и B, которое дает необходимые и достаточные условия для того, чтобы любое вещественное число, имеющее конечное представление в формате по основанию b, имело также конечное представление в формате по основанию B.

5. [*M20*] Покажите, что программа (4) будет выполняться, если команду `LDX =10^n=` заменить командой `LDX =c=` при определенных значениях константы c.

6. [*30*] Исследуйте методы 1, a; 1, b; 2, a и 2, b для случая, когда b или B равно -2.

7. [*M18*] Известно, что $0<\alpha\le x\le\alpha+1/w$ и $0\le u\le w$, где u — целое число. Докажите, что $\lfloor ux\rfloor$ равно либо $\lfloor\alpha u\rfloor$, либо $\lfloor\alpha u\rfloor+1$. Более того, $\lfloor ux\rfloor=\lfloor\alpha u\rfloor$ точно, если $u<\alpha w$ и α^{-1} — целое число.

8. [*24*] Напишите MIX-программу, аналогичную (1), которая использует соотношение (5) и не содержит команд деления.

▸ **9.** [*M29*] Назначение данного упражнения — вычисление $\lfloor u/10\rfloor$ и $u \bmod 10$ только при помощи операций двоичного сдвига, маскирования и сложения, если u — неотрицательное целое число. Положите k фиксированным целым числом, которое ≥ 2, и рассмотрите процесс вычисления

$$v\leftarrow u+1, v\leftarrow v+\left\lfloor\frac{v}{2}\right\rfloor, v\leftarrow v+\left\lfloor\frac{v}{16}\right\rfloor, v\leftarrow v+\left\lfloor\frac{v}{256}\right\rfloor, v\leftarrow v+\left\lfloor\frac{v}{2^{2^k}}\right\rfloor;$$

$$q\leftarrow\left\lfloor\frac{v}{16}\right\rfloor, r\leftarrow v \bmod 16, r\leftarrow r+\left\lfloor\frac{r}{4}\right\rfloor, r\leftarrow\left\lfloor\frac{r}{2}\right\rfloor.$$

Чему равно наименьшее положительное целое число u, такое, что $q\ne\lfloor u/10\rfloor$ или $r\ne u \bmod 10$?

10. [*22*] В табл. 1 показано, как на двоичном компьютере с использованием различных операций сдвига, маскирования и сложения может быть удвоено десятичное число, закодированное в двоичной системе. Предложите аналогичный метод, который позволял бы вычислять *половину* двоично-кодированного десятичного числа (с отбрасыванием остатка в случае, когда число нечетное).

11. [*16*] Преобразуйте число $(57721)_8$ в десятичное представление.

▸ **12.** [*22*] Придумайте быстрый метод преобразования вручную целых чисел из троичной системы счисления в десятичную и проиллюстрируйте его, преобразовав в десятичный вид число $(1212011210210)_3$. Как перевести число из десятичной системы счисления в троичную?

▸ **13.** [*25*] Предположим, что в ячейках памяти U+1, U+2, ..., U+m содержится заданная с многократной точностью дробь $(.u_{-1}u_{-2}\ldots u_{-m})_b$, где b — размер слова компьютера MIX. Напишите MIX-программу, выполняющую преобразование этой дроби в десятичный формат и усекающую ее до 180 десятичных разрядов. Ответ должен быть напечатан в двух строчках, разряды должны быть сгруппированы в 20 блоков по 9 разрядов в каждом, разделенных пробелами. (Воспользуйтесь командой CHAR.)

▸ **14.** [*M27*] (A. Шёнхаге (A. Schönhage).) При большом n время, необходимое для выполнения преобразования n-разрядного целого числа рассмотренным в разделе методом преобразования целых чисел многократной точности, имеет порядок n^2. Покажите, что n-разрядное целое число можно перевести в двоичный формат за $O(M(n)\log n)$ шагов, где $M(n)$ — количество циклов, необходимых для выполнения операции умножения n-битовых чисел, которые удовлетворяют "условиям гладкости" $M(2n) \geq 2M(n)$.

15. [*M47*] Можно ли существенным образом понизить верхнюю грань времени выполнения преобразования больших целых чисел, данную в упр. 14? (См. упр. 4.3.3–12.)

16. [*41*] Постройте быструю линейную итерационную конфигурацию для преобразования чисел из десятичной системы счисления в двоичную (см. раздел 4.3.3E).

17. [*M40*] Разработайте "идеальные" подпрограммы, выполняющие преобразования чисел с плавающей точкой, которые переводили бы p-разрядные десятичные числа в P-разрядные двоичные числа и наоборот, выдавая в обоих случаях правильный округленный результат в терминах раздела 4.2.2.

18. [*HM34*] (Дэвид В. Матула (David W. Matula).) Пусть $\text{round}_b(u,p)$ — функция b, u и p, которые являются наилучшим приближением p-битового числа u с плавающей точкой, представленного в системе счисления по основанию b в смысле раздела 4.2.2. Предполагая, что $\log_B b$ иррационально и что целая часть принадлежит бесконечному интервалу, докажите, что

$$u = \text{round}_b(\text{round}_B(u,P),p)$$

для всех p-битовых чисел u с плавающей точкой, представленных по основанию b тогда и только тогда, когда $B^{P-1} \geq b^p$. (Другими словами, "идеальное" входное преобразование произвольного числа u в представление по независимому основанию B и выполняемое после него "идеальное" выходное преобразование этого результата всегда снова даст число u тогда и только тогда, когда промежуточная точность P будет достаточно большой, как определено вышеприведенной формулой.)

19. [*M23*] Предположим, что десятичное число $u = (u_7 \ldots u_1 u_0)_{10}$ представлено как двоично-закодированное десятичное число $U = (u_7 \ldots u_1 u_0)_{16}$. Найдите соответствующие константы c_i и маски m_i, такие, что операция $U \leftarrow U - c_i(U \wedge m_i)$, повторенная для $i = 1, 2, 3$, переводит число U в двоичное представление числа u, где "$\wedge$" означает извлечение (побитового **AND**).

4.5. АРИФМЕТИКА РАЦИОНАЛЬНЫХ ЧИСЕЛ

При решении вычислительных задач зачастую важно знать, чем выражается результат: точным числом (например, 1/3) или числом с плавающей точкой (например, "0.333333574"). Если арифметические операции выполняются над дробями, а не над приближениями к ним, то при выполнении большинства вычислений совершенно *не накапливаются ошибки округления*. Это порождает чувство спокойной уверенности (которое часто отсутствует при выполнении операций над числами с плавающей точкой), что точность вычислений больше повышена быть не может.

4.5.1. Дроби

При выполнении арифметических операций над дробями числа могут быть представлены в виде пары целых чисел (u/u'), где u и u' взаимно просты и $u' > 0$. Число нуль представляется как $(0/1)$. В таком представлении $(u/u') = (v/v')$ тогда и только тогда, когда $u = v$ и $u' = v'$.

Перемножать дроби, конечно, просто. Чтобы найти $(u/u') \times (v/v') = (w/w')$, можно просто вычислить uv и $u'v'$. Два произведения uv и $u'v'$ могут не быть взаимно простыми, но если обозначить $d = \gcd(uv, u'v')$*, то искомый результат будет равен $w = uv/d$, $w' = u'v'/d$ (см. упр. 2). В разделе 4.5.2 рассматривается эффективный алгоритм вычисления наибольшего общего делителя.

Дроби можно перемножить и другим способом, который состоит в том, что находятся значения $d_1 = \gcd(u, v')$ и $d_2 = \gcd(u', v)$; тогда результатом будет $w = (u/d_1)(v/d_2)$, $w' = (u'/d_2)(v'/d_1)$ (см. упр. 3). При перемножении дробей этим методом необходимо вычислить два наибольших общих делителя, но в действительности вычисления по такому методу выполняются не дольше, чем по первому. В процессе нахождения наибольшего общего делителя осуществляется ряд итераций, количество которых фактически пропорционально логарифму входных чисел, так что количество итераций, необходимых для получения чисел d_1 и d_2, по существу, равно количеству итераций, необходимых для вычисления одного числа d. Более того, каждая итерация при определении d_1 и d_2, видимо, выполняется быстрее, так как рассматриваются сравнительно малые числа. Если u, u', v и v' —.величины с однократной точностью, то этот метод имеет преимущество по сравнению с первым, поскольку, если только допускается представление чисел w и w' с однократной точностью, в вычислениях совсем не участвуют числа с удвоенной точностью.

Деление может быть выполнено аналогично (см. упр. 4).

Операции сложения и вычитания дробей немного сложнее. Обычная процедура заключается в том, что принимается $(u/u') \pm (v/v') = \big((uv' \pm u'v)/u'v'\big)$, а затем данная дробь приводится к несократимому виду. При этом используется, как и в первом методе, наибольший общий делитель $d = \gcd(uv' \pm u'v, u'v')$. Но, опять же, можно избежать действий со столь большими числами, если начать с вычисления $d_1 = \gcd(u', v')$. При $d_1 = 1$ искомыми делимым и делителем будут $w = uv' \pm u'v$ и $w' = u'v'$. (Этот случай следует выделить особо, поскольку согласно теореме 4.5.2D, если знаменатели u' и v' распределены случайно, то d_1 равно 1 примерно в 61 случае

---

* Здесь и далее *gcd*() означает *наибольший общий делитель* (greatest common devisor). — *Прим. перев.*

из 100. Если $d_1 > 1$, то положим $t = u(v'/d_1) \pm v(u'/d_1)$ и вычислим $d_2 = \gcd(t, d_1)$; окончательный результат равен $w = t/d_2$, $w' = (u'/d_1)(v'/d_2)$. (В упр. 6 доказывается, что эти значения w и w' взаимно просты.) Операции над числами, представленными с однократной точностью, выполняются также с однократной точностью, но t может быть числом с двойной точностью или чуть больше (см. упр. 7); учитывая, что $\gcd(t, d_1) = \gcd(t \bmod d_1,\ d_1)$, вычисление числа d_2 не требует двойной точности.

Например, чтобы вычислить $(7/66) + (17/12)$, находим $d_1 = \gcd(66, 12) = 6$; тогда $t = 7 \cdot 2 + 17 \cdot 11 = 201$, а $d_2 = \gcd(201, 6) = 3$, так что в результате получим

$$\frac{201}{3} \Big/ \left(\frac{66}{6} \cdot \frac{12}{3}\right) = 67/44.$$

Проверку подпрограмм, реализующих арифметические операции над рациональными числами, можно выполнить, используя обращение матриц с известной обратной матрицей (наподобие матриц Коши; упр. 1.2.3–41).

Опыт вычислений с дробями показывает, что в процессе выполнения операций числа во многих случаях становятся очень большими. Поэтому, если предполагается, что u и u' для каждой дроби (u/u') являются числами с однократной точностью, то очень важно, чтобы в каждую подпрограмму сложения, вычитания, умножения и деления включались проверки переполнения. При решении задач с числами, для которых важна высокая точность, очень полезен набор подпрограмм для выполнения арифметических действий над дробями с допустимой *произвольной* точностью.

Методы, рассматриваемые в этом разделе, распространяются также на другие числовые поля, в частности можно было бы выполнять арифметические действия над величинами вида $(u + u'\sqrt{5})/u''$, где u, u', u'' — целые числа, $\gcd(u, u', u'') = 1$ и $u'' > 0$, или над величинами вида $(u + u'\sqrt[3]{2} + u''\sqrt[3]{4})/u'''$ и т. д.

Интересно рассмотреть также (если не упорствовать в применении точных методов) числа в формате с фиксированной дробной чертой и в формате с плавающей дробной чертой, которые являются аналогами чисел с плавающей точкой, но в их основе лежат рациональные дроби, а не дроби, ориентированные на представление в системах счисления с каким-либо основанием b*.

При представлении дроби в двоичном формате с фиксированной дробной чертой числитель и знаменатель дроби содержат не более p бит, где p — заданное целое число. В случае представления дроби в формате с плавающей дробной чертой *сумма* битов для числителя и знаменателя не превышает некоторого данного q; при этом для определения размера числителя как составной части цепочки из q бит используется другое поле представления. Бесконечность может быть представлена как $(1/0)$. Чтобы выполнять арифметические действия над такими числами, введем определение $x \oplus y = \text{round}(x + y)$, $x \ominus y = \text{round}(x - y)$ и т. д., где $\text{round}(x) = x$, если x представимо. В противном случае в качестве x выбирается одно из представимых чисел в окрестности числа x. На первый взгляд может показаться, что для определения $\text{round}(x)$ лучше всего выбрать представимое число, ближайшее к x, по аналогии с выбором округления в арифметике с плавающей точкой. Но практика

* В оригинале используются термины "fixed slash", "floating slash" и "slash-arithmetic", которые в связи с отсутствием в русскоязычной литературе соответствующих устоявшихся терминов мы будем переводить как "формат с фиксированной дробной чертой", "формат с плавающей дробной чертой", "арифметика в формате с дробной чертой" и "числа в формате с дробной чертой". — *Прим. перев.*

показала, что лучше всего стремиться к выбору округления в виде "простых" чисел, поскольку числа с малыми числителями и знаменателями встречаются гораздо чаще, чем сложные дроби. Предпочтительно округлять числа до $\frac{1}{2}$ чаще, чем до $\frac{127}{255}$. В этом случае правило округления, которое оказывается с практической точки зрения наиболее успешным, носит название "медианное округление" и формулируется следующим образом. Если (u/u') и (v/v') — смежные представимые числа, такие, что для любого $u/u' \le x \le v/v'$ round(x) должно равняться (u/u') или (v/v'), то процедура округления имеет вид

$$\text{round}(x) = \frac{u}{u'} \text{ для } x < \frac{u+v}{u'+v'}, \quad \text{round}(x) = \frac{v}{v'} \text{ для } x > \frac{u+v}{u'+v'}. \tag{1}$$

При точном выполнении равенства $x = (u+v)/(u'+v')$ полагаем, что round(x) равно ближайшей дроби с наименьшим знаменателем (или, если $u' = v'$, с наименьшим числителем). В упр. 4.5.3–43 показано, что пользоваться медианным округлением достаточно просто.

Например, предположим, что выполняются арифметические действия с числами, представленными в формате с фиксированной дробной чертой при $p = 8$, так что для представимых чисел (u/u') выполняется $-128 < u < 128$ и $0 \le u' < 256$ и $u \perp u'$. Такое округление не обеспечивает высокой точности, но дает представление о механизме арифметики в формате с дробной чертой. Ближайшими к $0 = (0/1)$ числами будут $(-1/255)$ и $(1/255)$. Согласно правилу медианного округления получаем round$(x) = 0$ тогда и только тогда, когда $|x| \le 1/256$. Предположим, что выполняются вычисления, в которых используется вид $\frac{22}{7} = \frac{314}{159} + \frac{1300}{1113}$, при условии, что вычисления осуществлялись с использованием точной арифметики рациональных чисел. Однако при промежуточных вычислениях выполнялось округление до представимых чисел. В этом случае $\frac{314}{159}$ округлится до $(79/40)$, а $\frac{1300}{1113}$ — до $(7/6)$. Сумма округленных чисел, $\frac{79}{40} + \frac{7}{6} = \frac{377}{120}$, в свою очередь, округляется до $(22/7)$. Таким образом, несмотря на выполнение трех операций округления получен правильный результат. Этот пример не был специально подобран, чтобы получился правильный результат. При получении результата решения задачи в виде простой дроби арифметика в формате с дробной чертой обеспечивает компенсацию ошибок промежуточных округлений.

Впервые в литературе вопросы точного представления дробей в компьютерах обсуждались П. Хенричи (P. Henrici), *JACM* **3** (1956), 6–9. Представление дробей в формате с фиксированной и плавающей дробными чертами было предложено Дэвидом В. Матулой и опубликовано в книге *Applications of Number Theory to Numerical Analysis*, edited by S. K. Zaremba (New York: Academic Press, 1972), 486–489. Дальнейшее развитие этой идеи рассмотрено Матулой и Корнерупом (Kornerup) в статьях, опубликованных в журналах *Proc. IEEE Symp. Computer Arith.* **4** (1978), 29–47; *Lecture Notes in Comp. Sci.* **72** (1979), 383–397; *Computing*, Suppl. **2** (1980), 85–111; *IEEE Trans.* **C-32** (1983), 378–388; *IEEE Trans.* **C-34** (1985), 3–18; *IEEE Trans.* **C-39** (1990), 1106–1115.

УПРАЖНЕНИЯ

1. [*15*] Предложите приемлемый метод сравнения двух дробей с целью проверки выполнения неравенства $(u/u') < (v/v')$.

2. [*M15*] Докажите, что если $d = \gcd(u, v)$, то u/d и v/d взаимно просты.

3. [*M20*] Докажите, что из того, что $u \perp u'$ и $v \perp v'$, следует

$$\gcd(uv, u'v') = \gcd(u, v')\gcd(u', v).$$

4. [*11*] Разработайте алгоритм деления дробей, аналогичный второму способу умножения, который описан в разделе (обратите внимание на то, что должен быть учтен знак числа v).

5. [*10*] С помощью методов, рекомендуемых в разделе, вычислите $(17/120) + (-27/70)$.

▶ **6.** [*M23*] Покажите, что из условий $u \perp u'$ и $v \perp v'$ следует $\gcd(uv' + vu', u'v') = d_1 d_2$, где $d_1 = \gcd(u', v')$ и $d_2 = \gcd(d_1, u(v'/d_1) + v(u'/d_1))$. (Следовательно, если $d_1 = 1$, то $(uv' + vu') \perp u'v'$.)

7. [*M22*] Насколько большим может стать абсолютное значение величины t в методе сложения-вычитания, рекомендованном в разделе, если числители и знаменатели исходных дробей по абсолютной величине меньше N?

▶ **8.** [*22*] Проанализируйте использование величин $(1/0)$ и $(-1/0)$ для представления ∞ и $-\infty$ и/или для представления переполнения.

9. [*M23*] Для $1 \le u', v' < 2^n$ покажите, что из $\lfloor 2^{2n} u/u' \rfloor = \lfloor 2^{2n} v/v' \rfloor$ следует $u/u' = v/v'$.

10. [*41*] Усовершенствуйте подпрограммы, предложенные в упр. 4.3.1–34, таким образом, чтобы они были применимы к "произвольным" рациональным числам.

11. [*M23*] Рассмотрите дроби вида $(u + u'\sqrt{5}\,)/u''$, где u, u', u'' — целые числа, $\gcd(u, u', u'') = 1$ и $u'' > 0$. Объясните, как разделить две такие дроби и как получить частное в таком же виде.

12. [*M16*] Чему равно максимальное число, представленное в формате с плавающей дробной чертой, если длина его числителя ограничена числом q и задана длина знаменателя? Какие числа округляются до $(0/1)$?

13. [*20*] (Матула (Matula) и Корнеруп (Kornerup).) Рассмотрите представление чисел с плавающей дробной чертой для 32-битового слова.

14. [*M23*] Объясните, как вычислить точное количество пар целых чисел (u, u'), таких, что $M_1 < u \le M_2$ и $N_1 < u' \le N_2$ и $u \perp u'$. (Данное объяснение может быть использовано для определения количества чисел, представимых в формате с дробной чертой. Согласно теореме 4.5.2D это число равно примерно $(6/\pi^2)(M_2 - M_1)(N_2 - N_1)$.)

15. [*42*] Модифицируйте на своем компьютере один из компиляторов так, чтобы он заменял все вычисления с числами, представленными в формате с плавающей точкой, вычислениями с числами, представленными в формате с плавающей дробной чертой. Проведите эксперименты с использованием арифметики в формате с дробной чертой, применив выполняющие программы, написанные программистами, которые ориентировались на арифметику чисел, представленных в формате с плавающей точкой. (При обращении к подпрограммам, реализующим алгоритмы наподобие алгоритмов вычисления квадратного корня или логарифма, ваша система должна перед выполнением подпрограмм автоматически преобразовать числа, представленные в формате с дробной чертой, в числа, представленные в формате с плавающей точкой, и после их выполнения снова вернуться к представлению в формате с дробной чертой. При этом должна быть предусмотрена возможность вывода на печать чисел, представленных в формате с дробной чертой. Тем не менее в случае, когда в программы для пользователей не вносилось никаких изменений, должна быть предусмотрена также возможность вывода на печать чисел, представленных в формате с дробной чертой, в виде десятичной дроби.) Станут ли результаты при замене чисел, представленных в формате с плавающей дробной чертой, лучше или хуже?

16. [*40*] Поэкспериментируйте с интервальной арифметикой над числами, представленными в формате с дробной чертой.

4.5.2. Наибольший общий делитель

Если числа u и v целые, такие, что одно из них не равно нулю, то говорят, что их *наибольший общий делитель*, $\gcd(u,v)$, есть наибольшее целое число, на которое числа u и v делятся без остатка. Данное определение имеет смысл, так как если $u \neq 0$, то самым большим числом, на которое число u делится без остатка, является $|u|$. Но числа u и v делятся также на целое число 1, следовательно, должно существовать самое большое число, на которое делятся оба эти числа. Когда оба числа u и v равны нулю, то нуль делится нацело на любое целое число, так что данное выше определение к этому случаю неприменимо. Условимся считать

$$\gcd(0,0) = 0. \tag{1}$$

Из введенных выше определений очевидным образом следует, что

$$\gcd(u,v) = \gcd(v,u), \tag{2}$$
$$\gcd(u,v) = \gcd(-u,v), \tag{3}$$
$$\gcd(u,0) = |u|. \tag{4}$$

В предыдущем разделе задача представления рационального числа с минимальными членами свелась к поиску наибольшего общего делителя числителя и знаменателя этого числа. Другие применения наибольшего общего делителя упоминались, например, в разделах 3.2.1.2, 3.3.3, 4.3.2 и 4.3.3. Таким образом, понятие наибольшего общего делителя $\gcd(u,v)$ имеет важное значение и заслуживает тщательного изучения.

С понятием наибольшего общего делителя тесно связано важное понятие *наименьшего общего кратного* двух целых чисел u и v, обозначаемого $\operatorname{lcm}(u,v)$. Оно определяется как наименьшее положительное целое число, кратное как u, так и v. $\operatorname{lcm}(u,0) = \operatorname{lcm}(0,v) = 0$. Классический метод обучения детей сложению дробей $u/u' + v/v'$ состоит в том, чтобы научить их находить "наименьший общий знаменатель", т. е. $\operatorname{lcm}(u',v')$.

Согласно "фундаментальной теореме арифметики" (доказанной в упр. 1.2.4–21) любое положительное целое число u может быть выражено в виде

$$u = 2^{u_2}3^{u_3}5^{u_5}7^{u_7}11^{u_{11}}\ldots = \prod_{p \text{ простое}} p^{u_p}, \tag{5}$$

где показатели степеней u_2, u_3, ...— однозначно определенные неотрицательные числа, только конечное число которых не равно нулю. Из этого канонического разложения положительного целого числа сразу следует один из способов вычисления наибольшего общего делителя чисел u и v. В силу соотношений (2)–(4) можно считать, что u и v — положительные целые числа, и если оба эти числа разложены на простые множители, то

$$\gcd(u,v) = \prod_{p \text{ простое}} p^{\min(u_p,v_p)}, \tag{6}$$

$$\operatorname{lcm}(u,v) = \prod_{p \text{ простое}} p^{\max(u_p,v_p)}. \tag{7}$$

Отсюда, например, следует, что наибольший общий делитель числа $u = 7000 = 2^3 \cdot 5^3 \cdot 7$, а числа $v = 4400 = 2^4 \cdot 5^2 \cdot 11$ равен $2^{\min(3,4)}\, 5^{\min(3,2)}\, 7^{\min(1,0)}\, 11^{\min(0,1)} = 2^3 \cdot 5^2 = 200$. Наименьшее общее кратное тех же чисел равно $2^4 \cdot 5^3 \cdot 7 \cdot 11 = 154000$.

Из формул (6) и (7) легко получить ряд основных тождеств, относящихся к наибольшему общему делителю и наименьшему общему кратному:

$$\gcd(u,v)w = \gcd(uw, vw) \qquad \text{при } w \geq 0; \tag{8}$$

$$\operatorname{lcm}(u,v)w = \operatorname{lcm}(uw, vw) \qquad \text{при } w \geq 0; \tag{9}$$

$$u \cdot v = \gcd(u,v) \cdot \operatorname{lcm}(u,v) \qquad \text{при } u, v \geq 0; \tag{10}$$

$$\gcd\big(\operatorname{lcm}(u,v), \operatorname{lcm}(u,w)\big) = \operatorname{lcm}\big(u, \gcd(v,w)\big); \tag{11}$$

$$\operatorname{lcm}\big(\gcd(u,v), \gcd(u,w)\big) = \gcd\big(u, \operatorname{lcm}(v,w)\big). \tag{12}$$

Два последних равенства являются аналогами "дистрибутивного закона" $uv + uw = u(v+w)$. Соотношение (10) сводит вычисление $\gcd(u,v)$ к вычислению $\operatorname{lcm}(u,v)$ и наоборот.

Алгоритм Евклида. Хотя соотношение (6) очень интересно в теоретическом аспекте, для практического вычисления наибольшего общего делителя оно бесполезно, так как для этого требуется сначала найти разложение чисел u и v на простые множители. На сегодняшний день неизвестны способы очень быстрого поиска простых множителей для целых чисел (см. раздел 4.5.4). К счастью, наибольший общий делитель двух целых чисел может быть эффективно найден без разложения чисел на простые множители. Такой метод был открыт более 2 250 лет тому назад — это *алгоритм Евклида*, который уже подробно рассматривался в разделах 1.1 и 1.2.1.

Алгоритм Евклида находится в книге 7 его *Начал* (ок. 300 г. до н. э.), предложения 1 и 2, но, вероятно, он не был придуман Евклидом. Некоторые ученые предполагают, что данный метод был известен за 200 лет до этого, по крайней мере в форме, использующей вычитания, и почти наверняка этот алгоритм был известен Евдоксу (ок. 375 г. до н. э.); см. K. von Fritz, *Ann. Math.* (2) **46** (1945), 242–264. Аристотель (ок. 330 г. до н. э.) упомянут в этой книге в связи с рассматриваемой темой, 158b, 29–35. Тем не менее осталось очень мало свидетельств столь ранней истории этого алгоритма [см. W. R. Knorr, *The Evolution of the Euclidean Elements* (Dordrecht, 1975)].

Алгоритм Евклида можно назвать дедушкой всех алгоритмов, так как он самый старый из всех нетривиальных алгоритмов, дошедших до наших дней. (Эту честь мог бы, пожалуй, оспаривать древнеегипетский метод умножения, в основу которого положен метод удваивания и сложения и на котором базируется эффективный метод вычисления n-х степеней, рассматриваемый в разделе 4.6.3. Но в египетских папирусах просто приведены примеры, не носящие законченного систематического характера, и эти примеры во всяком случае не изложены систематически. Поэтому египетский метод не совсем заслуживает названия "алгоритм". Известно также несколько древних вавилонских методов, применяемых для такого рода задач, как решение специальных систем квадратных уравнений с двумя неизвестными. Это настоящие алгоритмы, а не просто частные решения уравнений для определенных входных параметров. Хотя вавилоняне постоянно сопровождали изложение каждого метода примером для частных значений входных параметров, они в сопроводительном тексте регулярно приводили объяснение основной процедуры. [См. D. E. Knuth, *CACM*

15 (1972), 671–677; **19** (1976), 108.] Многие из этих алгоритмов были известны за 1 500 лет до Евклида и являются наиболее ранними образцами записанных алгоритмов. Однако они не выдерживают сравнения с алгоритмом Евклида, поскольку в них отсутствуют итерации. Именно поэтому они были вытеснены современными алгебраическими методами.)

В свете важности алгоритма Евклида как в историческом так и в теоретическом аспектах посмотрим, как его трактовал сам Евклид. Перефразировав Евклида и использовав современную терминологию, мы можем сказать, что он писал примерно следующее.

> **Предложение.** *Для данных двух положительных целых чисел найти их наибольший общий делитель.*
>
> Пусть A и C — два заданных положительных целых числа; требуется найти их наибольший общий делитель. Если число A делится на C, то число C есть общий делитель чисел C и A, поскольку оно делит самое себя. И очевидно, что оно будет и наибольшим делителем, поскольку нет числа, большего, чем число C, которое бы делило C.
>
> Но если C не делит число A, то будем непрерывно вычитать меньшее из чисел A, C из большего до тех пор, пока не получим число, которое нацело делит предыдущее вычитаемое. Это должно рано или поздно произойти, потому что, если разность будет равна единице, то единица будет делить предыдущее вычитаемое.
>
> Теперь положим, что E — положительный остаток от деления числа A на C; пусть F — положительный остаток от деления числа C на число E и пусть F делит E. Так как F делит E, а E делит $C - F$, F также делит $C - F$. Но оно делит и самое себя, поэтому F делит C, а C делит $A - E$; поэтому F также делит $A - E$, но оно делит и E; поэтому F делит A. Следовательно, F является общим делителем чисел A и C.
>
> Теперь я утверждаю, что оно является и наибольшим делителем. Действительно, если F — не наибольший общий делитель чисел A и C, то найдется большее число, которое будет делить оба эти числа. Пусть таким числом будет G.
>
> Так как число G делит число C, а число C делит $A - E$, то G также делит число $A - E$. Число G делит также все число A, поэтому оно делит и остаток E. Но E делит $C - F$, поэтому G также делит $C - F$. А число G также делит все число C, так что оно делит и остаток F; таким образом, большее число делит меньшее, а это невозможно.
>
> Таким образом, нет такого числа, большего, чем F, которое бы делило A и C; значит, число F является наибольшим общим делителем.
>
> **Следствие.** Это рассуждение делает очевидным предложение, что всякое число, делящее два числа, делит и их наибольший общий делитель. *Ч. Т. Д.*

Формулировка Евклида упрощена здесь в одном немаловажном аспекте. Греческие математики не считали единицу "делителем" другого положительного числа. Два положительных целых числа были либо оба равны единице, либо взаимно просты, либо имели наибольший общий делитель. Фактически единица даже не считалась числом, а нуль, конечно, вообще не существовал. Эти довольно нескладные соглашения были причиной того, что Евклид должен был дублировать значительную часть своих рассуждений, и он дал два отдельных предложения, каждое из которых по существу сходно с вышеприведенным.

В своем доказательстве Евклид впервые предлагает повторно вычитать для каждой пары текущих значений меньшее число из большего до тех пор, пока в

результате получатся два числа, одно из которых кратно другому. Но при доказательстве он фактически берет остаток от деления одного числа на другое, а так как понятие нуля отсутствовало, он не может говорить об остатке, когда одно число делит другое. Поэтому резонно заявить, что он рассматривает каждое *деление* (а не каждое вычитание в отдельности) как один шаг алгоритма, и, следовательно, "аутентичное" изложение его алгоритма выглядит следующим.

Алгоритм Е (*Оригинальный алгоритм Евклида*). По двум целым числам A и C, бóльшим единицы, этот алгоритм находит их наибольший общий делитель.

Е1. [A делится на C?] Если C делит A, то алгоритм заканчивается, давая в качестве результата число C.

Е2. [Заменить A остатком.] Если $A \bmod C$ равно единице, то заданные числа взаимно просты, поэтому алгоритм заканчивается. В противном случае заменить пару значений (A, C) парой $(C, A \bmod C)$ и вернуться к шагу Е1. ▮

"Доказательство" Евклида, приведенное выше, представляет особый интерес, так как оно в действительности совсем не является доказательством! Евклид проверяет результат алгоритма только тогда, когда шаг Е1 выполняется либо один раз, либо три раза. Он, безусловно, должен был понимать, что шаг Е1 может выполняться больше трех раз, хотя и не упоминает о такой возможности. Не имея представления о доказательстве при помощи математической индукции, он мог привести доказательство только для конечного числа случаев. (Фактически, желая доказать теорему для общего случая n, он рассматривал только частный случай $n = 3$.) Хотя алгоритм Евклида заслуженно известен своим большим вкладом в искусство логической дедукции, приемы, применяемые в строгих доказательствах по индукции, были открыты только спустя многие столетия, а ключевые идеи, используемые для доказательства справедливости *алгоритмов*, становятся по-настоящему понятными только сейчас. (Полное доказательство алгоритма Евклида, а также краткое обсуждение основной процедуры доказательства алгоритмов изложены в разделе 1.2.1.)

Следует отметить, что этот алгоритм нахождения наибольшего общего делителя был выбран Евклидом в качестве первого шага в его изложении теории чисел. И в наши дни во многих учебниках все еще используется тот же порядок изложения. Евклид дал, кроме того, метод (предложение 34) нахождения наименьшего общего кратного двух целых чисел u и v, а именно: разделить u на $\gcd(u, v)$ и затем умножить результат на v; это равносильно соотношению (10).

Если пренебречь предубеждением Евклида против чисел 0 и 1, то алгоритм Е можно переформулировать следующим образом.

Алгоритм А (*Алгоритм Евклида в современной редакции*). По данным неотрицательным целым числам u и v этот алгоритм находит их наибольший общий делитель. (*Замечание.* Наибольший общий делитель *произвольных* целых чисел u и v можно получить с учетом соотношений (2) и (3), применив алгоритм к $|u|$ и $|v|$.)

А1. [$v = 0$?] Если $v = 0$, то выполнение алгоритма заканчивается, а в качестве результата возвращается число u.

А2. [Взять $u \bmod v$.] Присвоить $r \leftarrow u \bmod v$, $u \leftarrow v$, $v \leftarrow r$ и вернуться к шагу А1. (В результате выполняемых на этом шаге операций значение v уменьшается, значение $\gcd(u, v)$ остается неизменным.) ▮

Например, можно вычислить $\gcd(40902, 24140)$ следующим образом:

$$\begin{aligned}\gcd(40902, 24140) &= \gcd(24140, 16762) = \gcd(16762, 7378)\\ &= \gcd(7378, 2006) = \gcd(2006, 1360) = \gcd(1360, 646)\\ &= \gcd(646, 68) = \gcd(68, 34) = \gcd(34, 0) = 34.\end{aligned}$$

Справедливость алгоритма A легко доказать из соотношения (4) и уравнения

$$\gcd(u, v) = \gcd(v,\ u - qv) \tag{13}$$

для любого целого числа q. Уравнение (13) выполняется потому, что любой общий делитель чисел u и v является делителем как v, так и $u - qv$, и наоборот, любой общий делитель чисел v и $u - qv$ должен делить оба числа u и v.

В следующей MIX-программе показано, что алгоритм A может быть легко реализован на компьютере.

Программа A (*Алгоритм Евклида*). Положим, что u и v — неотрицательные целые числа однократной точности, помещенные в ячейки U и V соответственно; эта программа помещает $\gcd(u, v)$ в ячейку rA.

```
    LDX  U     1     rX ← u.
    JMP  2F    1
1H  STX  V     T     v ← rX.
    SRAX 5     T     rAX ← rA.
    DIV  V     T     rX ← rAX mod v.
2H  LDA  V   1+T     rA ← v.
    JXNZ 1B  1+T     Выполнено, если rX = 0.  ∎
```

Время выполнения программы составляет $19T + 6$ циклов, где T — число выполненных операций деления. Рассуждения, приведенные в разделе 4.5.3, показывают, что в случае, когда u и v независимо и равномерно распределены в интервале $1 \le u, v \le N$, среднее значение T можно приблизительно представить в виде $T = 0.842766 \ln N + 0.06$.

Бинарный метод. После стольких столетий применения почтенного алгоритма Евклида несколько неожиданным оказалось то, что он не всегда является лучшим способом определения наибольшего общего делителя. В 1961 году Джозеф Стейн (Josef Stein) предложил совершенно другой алгоритм нахождения наибольшего общего делителя, ориентированный, прежде всего, на двоичную арифметику [см. *J. Comp. Phys.* **1** (1967), 397–405]. Этому новому алгоритму совершенно не нужны команды, выполняющие операции деления. Основанный исключительно на операциях вычитания, он проверяет, четно ли число, и делит пополам четные числа (что соответствует в двоичной арифметике сдвигу вправо).

Бинарный алгоритм нахождения наибольшего общего делителя основан на четырех простых фактах относительно положительных целых чисел u и v.

a) Если u и v оба четны, то $\gcd(u, v) = 2\gcd(u/2, v/2)$ [см. уравнение (8)].
b) Если u четно, а v нечетно, то $\gcd(u, v) = \gcd(u/2, v)$ [см. уравнение (6)].
c) Как и в алгоритме Евклида, $\gcd(u, v) = \gcd(u - v,\ v)$ [см. уравнения (13) и (2)].
d) Если u и v оба нечетны, то $u - v$ четно и $|u - v| < \max(u, v)$.

Алгоритм B (*Бинарный алгоритм нахождения наибольшего общего делителя*). По данным целым числам u и v этот алгоритм (рис. 9) находит наибольший общий делитель.

B1. [Найти степень 2.] Присвоить $k \leftarrow 0$, затем повторно присваивать $k \leftarrow k+1$, $u \leftarrow u/2$, $v \leftarrow v/2$ нуль или более раз до тех пор, пока оба числа u и v станут нечетными.

B2. [Начальная установка.] (Исходные значения чисел u и v уже разделены на 2^k, и по крайней мере одно из текущих значений нечетно.) Если нечетно u, то присвоить $t \leftarrow -v$ и перейти к шагу B4. В противном случае присвоить $t \leftarrow u$.

B3. [Уменьшить t наполовину.] (Здесь t четно и не нуль.) Присвоить $t \leftarrow t/2$.

B4. [t четно?] Если t четно, то вернуться к шагу B3.

B5. [Установить $\max(u,v)$ заново.] Если $t > 0$, то присвоить $u \leftarrow t$, в противном случае присвоить $v \leftarrow -t$. (Большее из чисел u и v заменяется на $|t|$ за исключением, возможно, первого выполнения этого шага.)

B6. [Вычесть.] Присвоить $t \leftarrow u - v$. Если $t \neq 0$, то вернуться к шагу B3. В противном случае алгоритм останавливает выполнение, а на выходе будет $u \cdot 2^k$. ▮

В качестве примера работы алгоритма B рассмотрим случай с числами $u = 40902$, $v = 24140$, т. е. с теми же числами, которые использовались при проверке работы алгоритма Евклида. На шаге B1 выполняются присвоения $k \leftarrow 1$, $u \leftarrow 20451$, $v \leftarrow 12070$. Затем t присваивается значение -12070, которое заменяется значением -6035; после этого значение v заменяется числом 6035 и вычисления продолжаются следующим образом.

u	v	t
20451	6035	+14416, +7208, +3604, +1802, +901
901	6035	−5134, −2567
901	2567	−1666, −833
901	833	+68, +34, +17
17	833	−816, −408, −204, −102, −51
17	51	−34, −17
17	17	0

Результат равен $17 \cdot 2^1 = 34$. Здесь нам пришлось выполнить немного больше итераций, чем при работе с алгоритмом A, но каждая из них выполнялась проще, так как совершенно отсутствовали операции деления.

Для разработки MIX-программы, реализующей алгоритм B, потребовалось немного больше команд, чем для реализации алгоритма A. Чтобы получить программу, типичную для двоичного компьютерного представления алгоритма B, предположим, что компьютер MIX расширен путем включения следующих операторов.

• SLB (сдвинуть влево как двоичный код в AX). C = 6; F = 6.

Содержимое регистров A и X "сдвигается влево" на M двоичных разрядов, т. е. $|\mathrm{rAX}| \leftarrow |2^M \mathrm{rAX}| \bmod B^{10}$, где B — размер байта. (Как и во всех других командах сдвига в MIX, знаки регистров rA и rX не изменяются.)

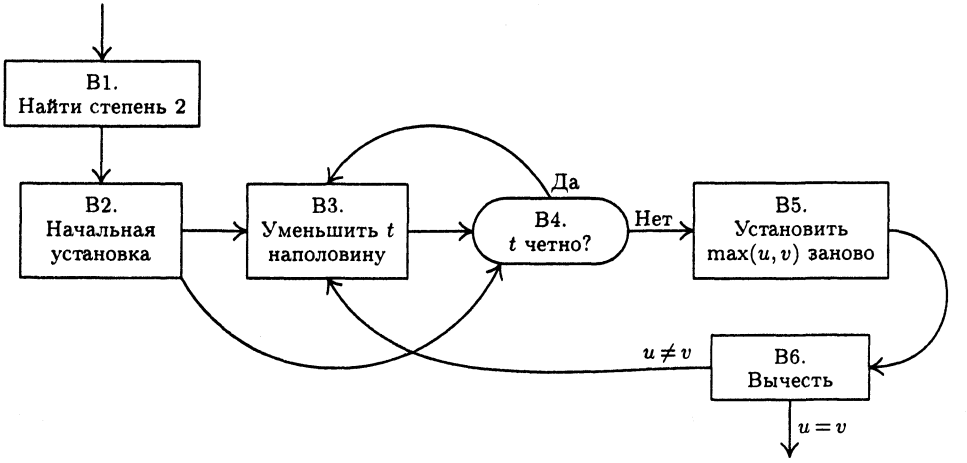


Рис. 9. Бинарный алгоритм нахождения наибольшего общего делителя.

• SRB (сдвинуть вправо двоичный код в AX). C = 6; F = 7.
Содержимое регистров A и X "сдвигается вправо" на M двоичных разрядов, т. е. $|\mathrm{rAX}| \leftarrow \lfloor |\mathrm{rAX}|/2^M \rfloor$.

• JAE, JAO (переход, если в регистре A находится четное число; переход, если в регистре A находится нечетное число). C = 40; F = 6, 7 соответственно.
Переход выполняется, если в регистре rA находится четное или нечетное число соответственно.

• JXE, JXO (переход, если в регистре X находится четное число; переход, если в регистре X находится нечетное число). C = 47; F = 6, 7 соответственно.
Эти операции аналогичны JAE, JAO.

Программа B (*Бинарный алгоритм нахождения наибольшего общего делителя*). Пусть u и v — положительные целые числа с однократной точностью, помещенные соответственно в ячейки памяти U и V. Эта программа, используя алгоритм B, помещает $\gcd(u, v)$ в rA. Содержимое регистров таково: $\mathrm{rA} \equiv t$, $\mathrm{rI1} \equiv k$.

```
01  ABS EQU  1:5
02  B1  ENT1 0          1        B1. Найти степень 2.
03      LDX  U          1        rX ← u.
04      LDAN V          1        rA ← −v.
05      JMP  1F         1
06  2H  SRB  1          A        Разделить пополам rA, rX.
07      INC1 1          A        k ← k + 1.
08      STX  U          A        u ← u/2.
09      STA  V(ABS)     A        v ← v/2.
10  1H  JXO  B4       1 + A      Перейти к шагу B4 с t ← −v, если u нечетно.
11  B2  JAE  2B       B + A      B2. Начальная установка.
12      LDA  U          B        t ← u.
13  B3  SRB  1          D        B3. Разделить пополам t.
14  B4  JAE  B3     1 − B + D    B4. t четно?
```

```
15  B5  JAN   1F        C       B5. Установить заново max(u,v).
16      STA   U         E       Если t > 0, присвоить u ← t.
17      SUB   V         E       t ← u − v.
18      JMP   2F        E
19  1H  STA   V(ABS)    C − E   Если t < 0, присвоить v ← −t.
20  B6  ADD   U         C − E   B6. Вычесть.
21  2H  JANZ  B3        C       Перейти к шагу B3, если t ≠ 0.
22      LDA   U         1       rA ← u.
23      ENTX  0         1       rX ← 0.
24      SLB   0,1       1       rA ← 2^k · rA.  ∎
```

Время выполнения программы равно

$$9A + 2B + 6C + 3D + E + 13$$

машинных циклов. Здесь $A = k$, $B = 1$, если на шаге B2 произошло присвоение $t \leftarrow u$ (иначе $B = 0$), C — число шагов, на которых выполняется вычитание, D — число делений пополам на шаге B3 и E — число, показывающее, сколько раз имеет место неравенство на шаге B5. Из вычислений, выполняемых ниже в этом разделе, следует, что в качестве средних значений этих величин в предположении, что входные величины u и v являются случайными в диапазоне $1 \le u, v < 2^N$, можно взять $A = \frac{1}{3}$, $B = \frac{1}{3}$, $C = 0.71N - 0.5$, $D = 1.41N - 2.7$ и $E = 0.35N - 0.4$. Поэтому общее время выполнения программы составляет примерно $8.8N + 5.2$ циклов; для программы A при тех же предположениях оно равно приблизительно $11.1N + 7.1$ циклов. Наихудшее возможное время выполнения $13N + 8$ циклов будет при $A = 0$, $B = 1$, $C = N$, $D = 2N - 2$, $E = N - 1$, т. е. при условии, что u и v принадлежат одному и тому же диапазону. (Соответствующее время выполнения программы A равно $26.8N + 19$ циклов.)

Таким образом, более высокая скорость выполнения итераций в программе B за счет простоты операций компенсирует бóльшее число итераций, требуемых для выполнения программы. Мы установили, что бинарный алгоритм на компьютере MIX выполняется на 20% быстрее, чем алгоритм Евклида. Безусловно, при реализации алгоритма на других компьютерах ситуация может измениться, но во всяком случае оба алгоритма достаточно эффективны. Тем не менее оказывается, что даже такая освященная веками процедура, как алгоритм Евклида, не может противостоять прогрессу.

История бинарного алгоритма нахождения наибольшего общего делителя поразительна: он был известен еще в Древнем Китае. Так, в разделе 6 главы 1 классического труда *Chiu Chang Suan Shu* ("Девять разделов арифметики", ок. 1 в. н. э.) приведен следующий метод приведения дроби к простейшему виду.

> Если возможно выполнение половинного деления, выполнить его.
>
> В противном случае выписать знаменатель и числитель дроби и вычесть меньшее число из большего.
>
> Повторять эту операцию до тех пор, пока числа не станут равными.
>
> Сократить дробь на это общее значение.

Если повторная операция снова приводит к половинному делению вместо того, чтобы повторять операцию вычитания (этот пункт не совсем понятен), метод фактически совпадает с алгоритмом B. [См. Y. Mikami, *The Development of Mathematics*

in China and Japan (Leipzig, 1913), 11; K. Vogel, *Neun Bücher arithmetischer Technik* (Braunschweig: Vieweg, 1968), 8.]

В. К. Харрис (V. C. Harris) [*Fibonacci Quarterly* **8** (1970), 102–103; см. также V. A. Lebesgue, *J. Math. Pures Appl.* **12** (1847), 497–520] предложил интересный гибрид метода Евклида и бинарного метода. Если числа u и v нечетны и $u \geq v > 0$, то всегда можно написать

$$u = qv \pm r,$$

где $0 \leq r < v$ и r четно. Если $r \neq 0$, то присваиваем $r \leftarrow r/2$ до тех пор, пока значение r не станет нечетным. После этого присваиваем $u \leftarrow v$, $v \leftarrow r$ и повторяем процесс. В последующих итерациях $q \geq 3$.

Обобщения. Методы, используемые для вычисления $\gcd(u, v)$, можно обобщить так, чтобы решать немного более сложные задачи. Например, предположим, нужно вычислить наибольший общий делитель n целых чисел u_1, u_2, ..., u_n.

Один из способов вычисления $\gcd(u_1, u_2, \ldots, u_n)$, если предположить, что все числа u_j неотрицательны, состоит в обобщении алгоритма Евклида следующим образом: если все числа u_j равны нулю, то наибольший общий делитель принимается равным нулю, иначе при наличии только одного ненулевого числа u_j это число и будет наибольшим общим делителем; в противном случае заменяем u_k на $u_k \bmod u_j$ для всех $k \neq j$, где u_j — минимальное из ненулевых чисел u, и повторяем процесс.

Алгоритм, набросок которого здесь приведен, является естественным обобщением алгоритма Евклида. Он может быть обоснован аналогичным образом. Но имеется более простой метод, основанный на легко проверяемом тождестве

$$\gcd(u_1, u_2, \ldots, u_n) = \gcd\bigl(u_1, \gcd(u_2, \ldots, u_n)\bigr). \tag{14}$$

Чтобы вычислить $\gcd(u_1, u_2, \ldots, u_n)$, можно поступить так.

Алгоритм C (*Наибольший общий делитель n целых чисел*). По заданным целым числам u_1, u_2, ..., u_n, где $n \geq 1$, этот алгоритм вычисляет их наибольший общий делитель, используя алгоритм для случая $n = 2$ как подпрограмму.

C1. Присвоить $d \leftarrow u_n$, $k \leftarrow n - 1$.

C2. Если $d \neq 1$ и $k > 0$, то присвоить $d \leftarrow \gcd(u_k, d)$ и $k \leftarrow k - 1$ и повторить этот шаг. В противном случае $d = \gcd(u_1, \ldots, u_n)$. ▮

Данный метод сводит вычисление $\gcd(u_1, \ldots, u_n)$ к повторным вычислениям наибольшего общего делителя двух чисел. В нем используется то обстоятельство, что $\gcd(u_1, \ldots, u_k, 1) = 1$. Оно оказывается полезным, поскольку, как отмечалось выше, в 61 случае из 100, если числа u_{n-1} и u_n рассматривать в качестве случайных, выполняется равенство $\gcd(u_{n-1}, u_n) = 1$. В большинстве случаев значение d на нескольких первых этапах вычисления быстро уменьшается, в результате чего оставшаяся часть вычислений выполняется очень быстро. Здесь алгоритм Евклида имеет преимущество над алгоритмом B ввиду того, что время его выполнения определяется, прежде всего, значением $\min(u, v)$, в то время как время выполнения алгоритма B зависит, главным образом, от значения $\max(u, v)$. Поэтому целесообразно выполнять одну итерацию алгоритма Евклида, заменяя число u на $u \bmod v$, если u значительно больше числа v, а затем продолжать вычисления по алгоритму B.

Утверждение, что значение $\gcd(u_{n-1}, u_n)$ будет равно единице в более чем 60 случаях из 100 для случайных исходных данных, является следствием хорошо известного результата теории чисел.

Теорема D (Г. Люсьен Дирихле (G. Lejeune Dirichlet), *Abhandlungen Königlich Preuß. Akad. Wiss.* (1849), 69–83). *Если u и v — случайно выбираемые целые числа, то вероятность того, что $\gcd(u, v) = 1$, равна $6/\pi^2 \approx .60793$.*

Точная формулировка этой теоремы, в которой четко определяется, что́ имеется в виду под словами "выбирается случайно", а также ее доказательство приводятся в упр. 10. Здесь же ограничимся эвристическим доказательством, показывающим, почему эта теорема правдоподобна.

Если принять без доказательства существование вполне определенной вероятности p того, что $u \perp v$, то можно определить вероятность того, что выполняется равенство $\gcd(u, v) = d$ для любого положительного целого числа d, так как $\gcd(u, v) = d$ тогда и только тогда, когда число u кратно d, число v кратно d и $u/d \perp v/d$. Поэтому вероятность того, что $\gcd(u, v) = d$, равна $1/d$, умноженному на $1/d$, умноженному на p, т. е. p/d^2. Просуммируем теперь эти вероятности по всем возможным значениям d. Должно получиться

$$1 = \sum_{d \geq 1} p/d^2 = p\big(1 + \tfrac{1}{4} + \tfrac{1}{9} + \tfrac{1}{16} + \cdots\big).$$

Так как сумма $1 + \frac{1}{4} + \frac{1}{9} + \cdots = H_\infty^{(2)}$ равна $\pi^2/6$ согласно формуле 1.2.7–(7), для того, чтобы выполнялось предыдущее соотношение, необходимо, чтобы $p = 6/\pi^2$. ▮

Алгоритм Евклида можно обобщить еще одним способом, имеющим большое значение. Можно вычислить целые числа u' и v', такие, что

$$uu' + vv' = \gcd(u, v). \tag{15}$$

Одновременно вычисляется и $\gcd(u, v)$. Это обобщение алгоритма Евклида удобно описать, используя векторные обозначения.

Алгоритм X (*Обобщенный алгоритм Евклида*). Для заданных неотрицательных целых чисел u и v этот алгоритм определяет вектор (u_1, u_2, u_3), такой, что $uu_1 + vu_2 = u_3 = \gcd(u, v)$. В процессе вычисления используются вспомогательные векторы (v_1, v_2, v_3), (t_1, t_2, t_3); действия с векторами производятся таким образом, что в течение всего процесса вычисления выполняются соотношения

$$ut_1 + vt_2 = t_3, \qquad uu_1 + vu_2 = u_3, \qquad uv_1 + vv_2 = v_3. \tag{16}$$

X1. [Начальная установка.] Присвоить $(u_1, u_2, u_3) \leftarrow (1, 0, u)$, $(v_1, v_2, v_3) \leftarrow (0, 1, v)$.

X2. [$v_3 = 0$?] Если $v_3 = 0$, то выполнение алгоритма заканчивается.

X3. [Разделить и вычесть.] Присвоить $q \leftarrow \lfloor u_3/v_3 \rfloor$, затем присвоить

$$(t_1, t_2, t_3) \leftarrow (u_1, u_2, u_3) - (v_1, v_2, v_3)q,$$
$$(u_1, u_2, u_3) \leftarrow (v_1, v_2, v_3), \qquad (v_1, v_2, v_3) \leftarrow (t_1, t_2, t_3).$$

Возвратиться к шагу X2. ▮

Например, пусть $u = 40902$, $v = 24140$. На шаге X2 получаем следующее.

q	u_1	u_2	u_3	v_1	v_2	v_3
—	1	0	40902	0	1	24140
1	0	1	24140	1	−1	16762
1	1	−1	16762	−1	2	7378
2	−1	2	7378	3	−5	2006
3	3	−5	2006	−10	17	1360
1	−10	17	1360	13	−22	646
2	13	−22	646	−36	61	68
9	−36	61	68	337	−571	34
2	337	−571	34	−710	1203	0

Поэтому решение имеет вид $337 \cdot 40902 - 571 \cdot 24140 = 34 = \gcd(40902, 24140)$.

История алгоритма X восходит к трактату *Aryabhatiya* (499 г. н. э.), авторство которого принадлежит Ариабхата (Aryabhata), жившему в северной Индии. Хотя описание здесь было изложено в весьма зашифрованном виде, более поздние комментаторы, например Бхаскара I (Bhâscara I) в 6 веке, сформулировали правило, названное *kuṭṭaka* ("куттака" или "распылитель"). [См. B. Datta, A. N. Singh, *History of Hindu Mathematics* **2** (Lahore: Motilal Banarsi Das, 1938), 89–116.] Справедливость алгоритма X следует из соотношений (16) и того обстоятельства, что алгоритм идентичен алгоритму A в смысле выполнения операций с числами u_3 и v_3. Подробное доказательство алгоритма X приведено в разделе 1.2.1. Гордон Г. Брэдли (Gordon H. Bradley) заметил, что если исключить u_2, v_2 и t_2, то можно значительно сократить процесс вычислений при выполнении алгоритма X; значение u_2 может быть определено впоследствии из соотношения $uu_1 + vu_2 = u_3$.

В упр. 15 показано, что значения $|u_1|$, $|u_2|$, $|v_1|$ и $|v_2|$ остаются в интервале, ограниченном значениями исходных чисел u и v. Подобным образом может быть обобщен и алгоритм B, который вычисляет наибольший общий делитель, используя свойства чисел, представленных в двоичной системе счисления (упр. 39). Некоторые поучительные обобщения алгоритма X представлены в упр. 18 и 19 в разделе 4.6.1.

Идеи, лежащие в основе алгоритма Евклида, можно также применить для нахождения *общего целочисленного решения* произвольной системы линейных уравнений с целочисленными коэффициентами. Например, пусть требуется найти все целые числа w, x, y, z, которые удовлетворяют двум уравнениям:

$$10w + 3x + 3y + 8z = 1, \tag{17}$$

$$6w - 7x \qquad - 5z = 2. \tag{18}$$

Введем новую переменную

$$\lfloor 10/3 \rfloor w + \lfloor 3/3 \rfloor x + \lfloor 3/3 \rfloor y + \lfloor 8/3 \rfloor z = 3w + x + y + 2z = t_1$$

и используем ее для исключения y. Уравнение (17) примет вид

$$(10 \bmod 3)w + (3 \bmod 3)x + 3t_1 + (8 \bmod 3)z = w + 3t_1 + 2z = 1, \tag{19}$$

а уравнение (18) останется неизменным. Используем новое уравнение (19) для исключения w, тогда (18) примет вид

$$6(1 - 3t_1 - 2z) - 7x - 5z = 2$$

т. е.

$$7x + 18t_1 + 17z = 4. \tag{20}$$

Теперь, как и ранее, введем в рассмотрение новую переменную

$$x + 2t_1 + 2z = t_2$$

и исключим x из уравнения (20):

$$7t_2 + 4t_1 + 3z = 4. \tag{21}$$

Таким же способом можно ввести еще одну новую переменную, чтобы исключить переменную z, имеющую наименьший коэффициент:

$$2t_2 + t_1 + z = t_3.$$

Исключение z из уравнения (21) дает

$$t_2 + t_1 + 3t_3 = 4. \tag{22}$$

Наконец, используя данное уравнение, исключим t_2. После всех этих операций остаются две независимые переменные t_1 и t_3. Подставляя их вместо исходных переменных, получаем общее решение:

$$\begin{aligned} w &= 17 - 5t_1 - 14t_3, \\ x &= 20 - 5t_1 - 17t_3, \\ y &= -55 + 19t_1 + 45t_3, \\ z &= -8 + t_1 + 7t_3. \end{aligned} \tag{23}$$

Другими словами, все целочисленные решения (w, x, y, z) исходных уравнений (17) и (18) получаются из уравнений (23) путем независимой прогонки величин t_1 и t_3 через все целые значения.

Общий метод, проиллюстрированный в приведенном примере, основан на следующей процедуре. Найдем в системе уравнений ненулевой коэффициент c с наименьшим абсолютным значением. Предположим, что он появляется в уравнении вида

$$cx_0 + c_1x_1 + \cdots + c_kx_k = d; \tag{24}$$

положим также для простоты, что $c > 0$. Если $c = 1$, то воспользуемся этим уравнением для исключения переменной x_0 из остальных уравнений системы. Затем повторим эту же процедуру по отношению к оставшимся уравнениям системы. (Если уравнений больше не остается, то вычисления прекращаются и, по существу, получаем общее решение, выражаемое через оставшиеся переменные.) Если $c > 1$ и если $c_1 \bmod c = \cdots = c_k \bmod c = 0$, то проверяем выполнение $d \bmod c = 0$; в противном случае целочисленных решений нет. Затем делим обе части уравнения (24) на c и исключаем x_0 так же, как и в случае, когда $c = 1$. В итоге, если $c > 1$ и не все из $c_1 \bmod c, \ldots, c_k \bmod c$ равны нулю, вводим новую переменную

$$\lfloor c/c \rfloor x_0 + \lfloor c_1/c \rfloor x_1 + \cdots + \lfloor c_k/c \rfloor x_k = t, \tag{25}$$

исключаем переменную x_0 из других уравнений при помощи t и заменяем исходное уравнение (24) уравнением

$$ct + (c_1 \bmod c)x_1 + \cdots + (c_k \bmod c)x_k = d. \tag{26}$$

(См. (19) и (21) в рассмотренном выше примере.)

Этот процесс должен завершиться, так как в результате выполнения каждого шага уменьшается либо количество уравнений системы, либо величина наименьшего ненулевого коэффициента системы. Если описанную процедуру применить к уравнению $ux + vy = 1$ для различных целых чисел u и v, то, по существу, будут выполняться шаги алгоритма X.

Рассмотренная только что процедура преобразования переменных является простым и достаточно очевидным средством решения систем линейных уравнений с целочисленными коэффициентами, но отнюдь не самым лучшим методом решения такой задачи. Имеются существенные модификации метода, но они выходят за рамки настоящей книги. [См. Henri Cohen, *A Course in Computational Algebraic Number Theory* (New York: Springer, 1993), Chapter 2.]

Возможно использование алгоритма Евклида с Гауссовыми целыми числами $u + iu'$, а также с некоторыми другими квадратичными числовыми полями. (См., например, A. Hurwitz, *Acta Math.* **11** (1887), 187–200; E. Kaltofen, H. Rolletschek, *Math. Comp.* **53** (1989), 697–720; A. Knopfmacher, J. Knopfmacher, *BIT* **31** (1991), 286–292.)

Вычисление с высокой точностью. Если u и v — очень большие целые числа, требующие представления с многократной точностью, то бинарный метод (алгоритм B) служит простым и достаточно эффективным методом вычисления наибольшего общего делителя этих чисел, так как в нем при вычислениях используются только операции вычитания и сдвига.

Напротив, алгоритм Евклида представляется значительно менее привлекательным, так как на шаге A2 требуется разделить с многократной точностью число u на число v. Но на самом деле это не является препятствием для применения алгоритма Евклида, поскольку, как будет доказано в разделе 4.5.3, частное $\lfloor u/v \rfloor$ почти всегда очень мало́. Например, для случайных входных данных частное $\lfloor u/v \rfloor$ будет меньше 1 000 примерно в 99.856% случаев. Поэтому $\lfloor u/v \rfloor$ и $(u \bmod v)$ почти всегда можно найти при помощи вычислений, выполняемых с однократной точностью в сочетании со сравнительно простой операцией вычисления $u - qv$, где q — число, представленное с однократной точностью. Далее, если окажется, что u намного больше v (к примеру, в таком виде могут задаваться входные данные), трудно представить, что частное q может оказаться большим, так как алгоритм Евклида, если заменить u на $u \bmod v$, в этом случае выполняется достаточно успешно.

Значительного увеличения скорости выполнения алгоритма Евклида при работе с числами высокой точности можно добиться при помощи метода, предложенного Д. Г. Лемером (D. H. Lehmer) [*AMM* **45** (1938), 227–233]. Оперируя только старшими разрядами больших чисел, можно основную часть вычислений выполнять с однократной точностью, значительно сокращая таким образом количество операций, которые необходимо выполнять с многократной точностью. Идея заключается в экономии времени за счет выполнения "виртуальных" вычислений вместо фактических.

Например, рассмотрим два восьмизначных числа $u = 27182818$ и $v = 10000000$, предполагая, что имеется компьютер с четырехразрядными словами. Пусть $u' = 2718$, $v' = 1001$, $u'' = 2719$, $v'' = 1000$; тогда u'/v' и u''/v'' являются приближениями

к u/v, причем

$$u'/v' < u/v < u''/v''. \tag{27}$$

Отношение u/v определяет последовательность частных, полученных в алгоритме Евклида. Если алгоритм Евклида одновременно выполнять над значениями (u', v') и (u'', v'') с однократной точностью до тех пор, пока не получим различные частные, то будет нетрудно заметить, что такая же последовательность частных получится, если выполнять алгоритм над числами (u, v) с многократной точностью. Итак, посмотрим, что произойдет, когда алгоритм Евклида применить к (u', v') и к (u'', v'').

u'	v'	q'	u''	v''	q''
2718	1001	2	2719	1000	2
1001	716	1	1000	719	1
716	285	2	719	281	2
285	146	1	281	157	1
146	139	1	157	124	1
139	7	19	124	33	3

Пять первых частных одинаковы в обоих случаях, так что они должны быть правильными. Но на шестом шаге обнаруживается, что $q' \neq q''$, поэтому вычисления с однократной точностью прекращаются. Тем самым мы выяснили, что если бы вычисления выполнялись с исходными числами как с числами многократной точности, то это происходило бы так.

u	v	q
u_0	v_0	2
v_0	$u_0 - 2v_0$	1
$u_0 - 2v_0$	$-u_0 + 3v_0$	2
$-u_0 + 3v_0$	$3u_0 - 8v_0$	1
$3u_0 - 8v_0$	$-4u_0 + 11v_0$	1
$-4u_0 + 11v_0$	$7u_0 - 19v_0$	?

(28)

(Следующее частное находится между 3 и 19.) Независимо от количества разрядов чисел u и v до тех пор, пока выполняется (27), первые пять шагов алгоритма Евклида будут такими же, как в (28). Поэтому вычисления с однократной точностью можно выполнять на первых пяти шагах, а операции с многократной точностью — только при вычислении значений $-4u_0 + 11v_0$ и $7u_0 - 19v_0$. В этом случае получаем $u = 1268728$, $v = 279726$; дальнейшие вычисления можно продолжить подобным же образом с числами $u' = 1268$, $v' = 280$, $u'' = 1269$, $v'' = 279$ и т. д. При наличии большего накопителя можно было бы увеличить количество шагов, на которых вычисления выполняются с однократной точностью. Из примера видно, что в один сложный шаг объединяются только пять циклов алгоритма Евклида, а если бы, скажем, размер слова равнялся десяти разрядам, можно было бы объединить в один шаг до двенадцати циклов. Из результатов, доказанных в разделе 4.5.3, следует, что на каждой итерации число циклов с многократной точностью, которые можно заменить циклами с однократной точностью, пропорционально размеру слова, используемому в вычислениях с однократной точностью.

Метод Лемера можно сформулировать следующим образом.

Алгоритм L (*Алгоритм Евклида для больших чисел*). Пусть u и v — представляемые с однократной точностью неотрицательные целые числа, такие, что $u \geq v$. Этот алгоритм вычисляет наибольший общий делитель чисел u и v, используя вспомогательные p-разрядные переменные $\hat{u}$, $\hat{v}$, A, B, C, D, T, q, которые представлены с однократной точностью, и вспомогательные переменные t и w, которые представлены с многократной точностью.

L1. [Начальная установка.] Если число v достаточно мало́, чтобы быть представленным в формате с однократной точностью, то $\gcd(u,v)$ вычисляется по алгоритму A, и на этом вычисления заканчиваются. В противном случае обозначим p ведущих разрядов числа u через $\hat{u}$, а соответствующие разряды числа v — через $\hat{v}$. Другими словами, если используется представление чисел в системе счисления по основанию b, то $\hat{u} \leftarrow \lfloor u/b^k \rfloor$ и $\hat{v} \leftarrow \lfloor v/b^k \rfloor$, где k — наименьшее возможное число, удовлетворяющее условию $\hat{u} < b^p$.

Присвоить $A \leftarrow 1$, $B \leftarrow 0$, $C \leftarrow 0$, $D \leftarrow 1$. (Эти переменные представляют коэффициенты в (28), где

$$u = Au_0 + Bv_0 \quad \text{и} \quad v = Cu_0 + Dv_0 \tag{29}$$

в равносильных операциях алгоритма A над числами с многократной точностью. Кроме того,

$$u' = \hat{u} + B, \qquad v' = \hat{v} + D, \qquad u'' = \hat{u} + A, \qquad v'' = \hat{v} + C \tag{30}$$

в обозначениях рассмотренного выше примера.)

L2. [Проверить частное.] Присвоить $q \leftarrow \lfloor (\hat{u}+A)/(\hat{v}+C) \rfloor$. Если $q \neq \lfloor (\hat{u}+B)/(\hat{v}+D) \rfloor$, то перейти к шагу L4. (На этом шаге проверяется выполнение условия $q' \neq q''$ в обозначениях того же примера. В процессе вычислений на этом шаге при некоторых обстоятельствах, когда $\hat{u} = b^p - 1$ и $A = 1$ или когда $\hat{v} = b^p - 1$ и $D = 1$, может возникнуть переполнение при выполнении операций в формате с однократной точностью. В силу равенств (30) всегда будут выполняться условия

$$\begin{aligned} 0 \leq \hat{u} + A \leq b^p, \qquad & 0 \leq \hat{v} + C < b^p, \\ 0 \leq \hat{u} + B < b^p, \qquad & 0 \leq \hat{v} + D \leq b^p. \end{aligned} \tag{31}$$

Может оказаться, что выполнится одно из равенств $\hat{v} + C = 0$ и $\hat{v} + D = 0$, но не оба одновременно. Поэтому попытка деления на нуль на этом шаге означает "Перейти непосредственно к шагу L4".)

L3. [Имитация алгоритма Евклида.] Присвоить $T \leftarrow A - qC$, $A \leftarrow C$, $C \leftarrow T$, $T \leftarrow B - qD$, $B \leftarrow D$, $D \leftarrow T$, $T \leftarrow \hat{u} - q\hat{v}$, $\hat{u} \leftarrow \hat{v}$, $\hat{v} \leftarrow T$ и возвратиться к шагу L2. (Эти вычисления с однократной точностью равносильны операциям с многократной точностью в процедуре (28) с учетом (29).)

L4. [Шаг, на котором выполняются вычисления с многократной точностью.] Если $B = 0$, то, используя деление с многократной точностью, присвоить $t \leftarrow u \bmod v$, $u \leftarrow v$, $v \leftarrow t$. (Это может случиться только тогда, когда с помощью операции с однократной точностью нельзя моделировать операцию с многократной точностью. Отсюда следует, что алгоритму Евклида требуется очень большое частное, что может произойти крайне редко.) В противном случае присвоить $t \leftarrow Au$, $t \leftarrow t + Bv$, $w \leftarrow Cu$, $w \leftarrow w + Dv$, $u \leftarrow t$, $v \leftarrow w$ (выполняя непосредственно операции с многократной точностью). Возвратиться к шагу L1. ∎

С учетом неравенств (31) в процессе вычислений значения величин A, B, C, D представляются с однократной точностью.

Для реализации алгоритма L требуется несколько более сложная программа, чем для алгоритма B, но при оперировании большими числами этот алгоритм на многих компьютерах выполняется быстрее. Подобным образом можно ускорить выполнение бинарного алгоритма B в завершающей стадии (см. упр. 38). Преимущество алгоритма L заключается в следующем: он определяет последовательность частных, получаемых при выполнении алгоритма Евклида, что используется в многочисленных приложениях (см., например, упр. 43, 47, 49, упр. 51 в разделе 4.5.3, а также упр. 4.5.3–46).

***Анализ бинарного алгоритма.** В заключение этого раздела для обоснования установленных ранее формул проанализируем время выполнения алгоритма B.

Выясняется, что точно описать поведение алгоритма B крайне затруднительно, но можно начать анализ этого алгоритма с его приближенной модели. Предположим, что числа u и v нечетны, $u > v$ и

$$\lfloor \lg u \rfloor = m, \qquad \lfloor \lg v \rfloor = n. \tag{32}$$

(Таким образом, u является $(m+1)$-битовым числом, а v — $(n+1)$-битовым числом.) Рассмотрим выполнение в алгоритме B цикла "вычитание и сдвиг", т. е. операцию, которая начинается на шаге B6, а прекращается после окончания выполнения шага B5. Каждый цикл "вычитание и сдвиг" при $u > v$ вычисляет разность $u - v$ и сдвигает эту величину вправо до тех пор, пока не будет получено нечетное число u', которое замещает число u. Если входные числа случайны, можно ожидать, что примерно в половине случаев $u' = (u - v)/2$, примерно в четверти случаев $u' = (u - v)/4$, примерно в одной восьмой случаев $u' = (u - v)/8$ и т. д. Получаем

$$\lfloor \lg u' \rfloor = m - k - r, \tag{33}$$

где k — число разрядов, на которые было сдвинуто вправо число $u - v$, а r есть $\lfloor \lg u \rfloor - \lfloor \lg(u - v) \rfloor$ — количество битов, потерянных слева во время вычитания числа v из числа u. Заметим, что $r \le 1$ при $m \ge n + 2$, а $r \ge 1$ при $m = n$.

Взаимосвязь между k и r довольно беспорядочная (см. упр. 20), однако Ричард Брент (Richard Brent) нашел изящный способ анализа поведения приближенной модели алгоритма, положив u и v достаточно большими, такими, чтобы отношение v/u имело непрерывное распределение при дискретном изменении k. [См. *Algorithms and Complexity*, edited by J. F. Traub (New York: Academic Press, 1976), 321–355.] Предположим, что u и v — большие целые числа, возможно, случайные, но обязательно нечетные и их отношение подчинено определенному закону распределения. Тогда на шаге B6 младшие значащие биты величины $t = u - v$ могут быть случайными, но величина t будет четной. Следовательно, t будет нечетно кратным 2^k с вероятностью 2^{-k}; это приближенная вероятность того, что в цикле "вычитание и сдвиг" потребуется выполнить k сдвигов вправо. Другими словами, получено подходящее приближение, описывающее поведение алгоритма B при сделанных предположениях о том, что переход от шага B4 к шагу B3 всегда будет происходить с вероятностью $1/2$.

Пусть $G_n(x)$ — вероятность того, что $\min(u,v)/\max(u,v)$ будет $\ge x$ после выполнения с учетом этих предположений n циклов "вычитание и сдвиг". Если $u \ge v$

и если выполнено точно k сдвигов вправо, то отношение $X = v/u$ изменится на $X' = \min(2^k v/(u-v), (u-v)/2^k v) = \min\bigl(2^k X/(1-X), (1-X)/2^k X\bigr)$. Таким образом, неравенство $X' \ge x$ будет справедливо тогда и только тогда, когда $2^k X/(1-X) \ge x$ и $(1-X)/2^k X \ge x$, а это то же самое, что и

$$\frac{1}{1+2^k/x} \le X \le \frac{1}{1+2^k x}. \tag{34}$$

Поэтому $G_n(x)$ удовлетворяет интересному рекуррентному соотношению

$$G_{n+1}(x) = \sum_{k\ge 1} 2^{-k}\Bigl(G_n\Bigl(\frac{1}{1+2^k/x}\Bigr) - G_n\Bigl(\frac{1}{1+2^k x}\Bigr)\Bigr), \tag{35}$$

где $G_0(x) = 1 - x$ при $0 \le x \le 1$. Проведенные вычислительные эксперименты показали, что $G_n(x)$ быстро стремится к предельному распределению $G_\infty(x) = G(x)$ несмотря на то, что формальное доказательство сходимости представляется неочевидным. Будем полагать, что существует функция распределения $G(x)$. Тогда она удовлетворяет уравнению

$$G(x) = \sum_{k\ge 1} 2^{-k}\Bigl(G\Bigl(\frac{1}{1+2^k/x}\Bigr) - G\Bigl(\frac{1}{1+2^k x}\Bigr)\Bigr) \quad \text{при } 0 < x \le 1; \tag{36}$$

$$G(0) = 1; \qquad G(1) = 0. \tag{37}$$

Пусть

$$\begin{aligned} S(x) &= \frac{1}{2}G\Bigl(\frac{1}{1+2x}\Bigr) + \frac{1}{4}G\Bigl(\frac{1}{1+4x}\Bigr) + \frac{1}{8}G\Bigl(\frac{1}{1+8x}\Bigr) + \cdots \\ &= \sum_{k\ge 1} 2^{-k} G\Bigl(\frac{1}{1+2^k x}\Bigr); \end{aligned} \tag{38}$$

тогда

$$G(x) = S(1/x) - S(x). \tag{39}$$

Естественно определить

$$G(1/x) = -G(x), \tag{40}$$

так что уравнение (39) справедливо для всех $x > 0$. Поскольку x изменяется от 0 до ∞, $S(x)$ увеличивается от 0 до 1. Следовательно, $G(x)$ уменьшается от $+1$ до -1. Конечно, при $x > 1$ функция $G(x)$ перестает быть вероятностью, тем не менее она имеет смысл (см. упр. 23).

Допустим, что имеются степенные ряды $\alpha(x)$, $\beta(x)$, $\gamma_m(x)$, $\delta_m(x)$, $\lambda(x)$, $\mu(x)$, $\sigma_m(x)$, $\tau_m(x)$ и $\rho(x)$, такие, что

$$G(x) = \alpha(x)\lg x + \beta(x) + \sum_{m=1}^{\infty}\bigl(\gamma_m(x)\cos 2\pi m\lg x + \delta_m(x)\sin 2\pi m\lg x\bigr), \tag{41}$$

$$S(x) = \lambda(x)\lg x + \mu(x) + \sum_{m=1}^{\infty}\bigl(\sigma_m(x)\cos 2\pi m\lg x + \tau_m(x)\sin 2\pi m\lg x\bigr), \tag{42}$$

$$\rho(x) = G(1+x) = \rho_1 x + \rho_2 x^2 + \rho_3 x^3 + \rho_4 x^4 + \rho_5 x^5 + \rho_6 x^6 + \cdots, \tag{43}$$

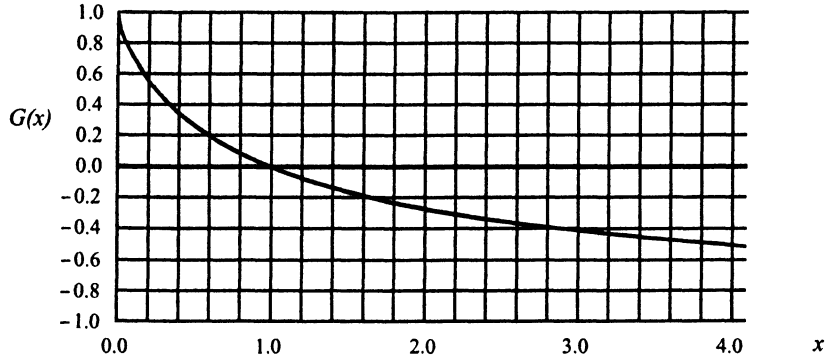


Рис. 10. Предельное распределение отношений в бинарном алгоритме вычисления наибольшего общего делителя.

поскольку можно показать, что этим свойством при $n \geq 1$ обладает решение $G_n(x)$ уравнения (35) (см., например, упр. 30). Эти степенные ряды сходятся при $|x| < 1$.

Какие выводы можно сделать относительно $\alpha(x)$, ..., $\rho(x)$ в уравнениях (36)–(43)? Прежде всего, из уравнений (38), (40) и (43) имеем

$$2S(x) = G\big(1/(1+2x)\big) + S(2x) = S(2x) - \rho(2x). \tag{44}$$

Соответственно равенство (42) выполняется тогда и только тогда, когда

$$2\lambda(x) = \lambda(2x); \tag{45}$$

$$2\mu(x) = \mu(2x) + \lambda(2x) - \rho(2x); \tag{46}$$

$$2\sigma_m(x) = \sigma_m(2x), \qquad 2\tau_m(x) = \tau_m(2x) \qquad \text{при } m \geq 1. \tag{47}$$

Из (45) следует, что $\lambda(x)$ есть просто константа, кратная x; так как она отрицательна, можно записать

$$\lambda(x) = -\lambda x. \tag{48}$$

(Соответствующий коэффициент приобретает вид

$$\lambda = 0.39792\,26811\,88316\,64407\,67071\,61142\,65498\,23098+, \tag{49}$$

но способ, которым его можно вычислить, неизвестен.) Из уравнения (46) следует, что $\rho_1 = -\lambda$ и $2\mu_k = 2^k\mu_k - 2^k\rho_k$ при $k > 1$; другими словами,

$$\mu_k = \rho_k/(1 - 2^{1-k}) \qquad \text{при } k \geq 2. \tag{50}$$

Из уравнения (47) следует также, что оба степенных ряда

$$\sigma_m(x) = \sigma_m x, \qquad \tau_m(x) = \tau_m x \tag{51}$$

являются просто линейными функциями. (Но это не распространяется на функции $\gamma_m(x)$ и $\delta_m(x)$.)

Если в уравнении (44) заменить x на $1/2x$, то получим

$$2S(1/2x) = S(1/x) + G(x/(1+x)), \tag{52}$$

а последнее уравнение при x, близком к 0, при помощи уравнения (39) преобразуется в

$$2G(2x) + 2S(2x) = G(x) + S(x) + G(x/(1+x)). \tag{53}$$

После подстановки в это уравнение степенных рядов вместо функций коэффициенты при $\lg x$ в обеих частях должны быть приравнены. Следовательно,

$$2\alpha(2x) - 4\lambda x = \alpha(x) - \lambda x + \alpha(x/(1+x)). \tag{54}$$

Уравнение (54), определяющее $\alpha(x)$, — рекуррентное. Действительно, предположим, что функция $\psi(z)$ удовлетворяет уравнению

$$\psi(z) = \frac{1}{2}\Big(z + \psi\Big(\frac{z}{2}\Big) + \psi\Big(\frac{z}{2+z}\Big)\Big), \qquad \psi(0) = 0, \quad \psi'(0) = 1. \tag{55}$$

Тогда уравнение (54) означает, что

$$\alpha(x) = \frac{3}{2}\lambda\psi(x). \tag{56}$$

Более того, из итерационного уравнения (55) следует

$$\begin{aligned}\psi(z) &= \frac{z}{2}\Big(\frac{1}{1} + \frac{1}{2}\Big(\frac{1}{2} + \frac{1}{2+z}\Big) + \frac{1}{4}\Big(\frac{1}{4} + \frac{1}{4+z} + \frac{1}{4+2z} + \frac{1}{4+3z}\Big) + \cdots\Big) \\ &= \frac{z}{2}\sum_{k\ge 0}\frac{1}{2^k}\sum_{0\le j<2^k}\frac{1}{2^k + jz}.\end{aligned} \tag{57}$$

Отсюда получаем, что обобщение $\psi(z)$ для степенных рядов имеет вид

$$\psi(z) = \sum_{n\ge 1}(-1)^{n-1}\psi_n z^n, \qquad \psi_n = \frac{1}{2n}\sum_{k=0}^{n-1}\frac{B_k}{2^{k+1}-1}\binom{n}{k} + \frac{\delta_{n1}}{2}; \tag{58}$$

см. упр. 27. Эта формула удивительно похожа на выражение 6.3–(18), полученное в связи с алгоритмами дискретного поиска в дереве, а в упр. 28 приводится доказательство справедливости формулы $\psi_n = \Theta(n^{-2})$.

Теперь нам известно $\alpha(x)$, за исключением случая, когда $\lambda = -\rho_1$ постоянно. Уравнение (50) связывает функции $\mu(x)$ и $\rho(x)$, исключая коэффициент μ_1. Из ответа к упр. 25 видно, что все коэффициенты функции $\rho(x)$ могут быть выражены через $\rho_1, \rho_3, \rho_5, \ldots$. Более того, константы σ_m и τ_m могут быть вычислены при помощи метода, применяемого для решения задачи в упр. 29; при этом между коэффициентами функций $\gamma_m(x)$ и $\delta_m(x)$ сохраняются сложные связи. Тем не менее, похоже, что единственным способом вычисления всех коэффициентов для различных функций, входящих в выражение для $G(x)$, является итеративное решение рекуррентного уравнения (36) численными методами.

После вычисления хорошего приближения к $G(x)$ можно оценить среднее время выполнения алгоритма B следующим образом. Если $u \ge v$ и если выполнить k сдвигов вправо, то величина $Y = uv$ будет заменена на $Y' = (u-v)v/2^k$. Значит, Y/Y' равно $2^k/(1-X)$, где $X = v/u$ равно $\ge x$ с вероятностью $G(x)$. Отсюда следует, что число битов в uv уменьшается в среднем на константу

$$b = \mathrm{E}\lg(Y/Y') = \sum_{k\ge 1}2^{-k}\Big(f_k(0) + \int_0^1 G(x)f_k'(x)\,dx\Big),$$

где $f_k(x) = \lg(2^k/(1-x))$. Получаем

$$b = \sum_{k\ge1} 2^{-k}\Big(k + \int_0^1 \frac{G(x)\,dx}{(1-x)\ln 2}\Big) = 2 + \int_0^1 \frac{G(x)\,dx}{(1-x)\ln 2}. \tag{59}$$

При $u = v$ ожидаемое значение числа $\lg uv$ будет приблизительно равно 0.9779 (см. упр. 14), поэтому общее количество циклов "вычитание и сдвиг" алгоритма B будет приблизительно равно исходному значению числа $\lg uv$, умноженному на $1/b$. Учитывая свойство симметрии, это количество приблизительно равно исходному значению числа $\lg u$, умноженному на $2/b$. В результате выполненных в 1977 году вычислений Ричард Брент (Richard Brent) получил для этой фундаментальной константы значение

$$2/b = 0.70597\,12461\,01916\,39152\,93141\,35852\,88176\,66677+. \tag{60}$$

В результате более глубокого анализа этих функций Бригитта Валле (Brigitte Vallée) высказала предположение, что постоянные λ и b могут быть связаны примечательной формулой

$$\frac{\lambda}{b} = \frac{2\ln 2}{\pi^2}. \tag{61}$$

Значения, вычисленные Брентом, вполне согласуются с этим далеко нетривиальным утверждением. Вызывает большой интерес анализ алгоритма B, успешно выполненный Валле на основе строгих "динамических" методов [см. *Algorithmica* **22** (1998), 660–685].

Вернемся к предположению в (32), состоящему в том, что u и v нечетные и изменяются в интервалах $2^m \le u < 2^{m+1}$ и $2^n \le v < 2^{n+1}$. Эмпирические испытания алгоритма B, проведенные с несколькими миллионами случайных значений на входе и с различными значениями m и n, взятыми из интервала $29 \le m, n \le 37$, показывают, что в действительности усредненное поведение алгоритма определяется соотношениями

$$\begin{aligned} C &\approx \tfrac12 m + 0.203n + 1.9 - 0.4(0.6)^{m-n}, \\ D &\approx \quad m + 0.41n \quad - 0.5 - 0.7(0.6)^{m-n}, \end{aligned} \qquad m \ge n, \tag{62}$$

при довольно небольшом стандартном отклонении от наблюдаемых средних значений. Коэффициенты $\frac12$ и 1 при m в выражениях (62) могут быть строго проверены (см. упр. 21).

Если же предположить, что u и v — *любые* целые числа, независимо и равномерно распределенные в интервалах

$$1 \le u < 2^N, \qquad 1 \le v < 2^N, \tag{63}$$

то можно вычислить средние значения величин C и D по уже имеющимся данным:

$$C \approx 0.70N + O(1), \qquad D \approx 1.41N + O(1). \tag{64}$$

(См. упр. 22.) Это хорошо согласуется с результатами последующих эмпирических экспериментов, выполненных с несколькими миллионами случайных входных данных для $N \le 30$. Эти эксперименты показали, что в качестве подходящих значений

для заданных распределений входных данных u и v можно взять

$$C = 0.70N - 0.5, \qquad D = 1.41N - 2.7. \tag{65}$$

Теоретический анализ непрерывной модели Брента алгоритма B предсказывает, что при предположениях (63) величины C и D асимптотически равны $2N/b$ и $4N/b$, где $2/b \approx 0.70597$ — константа в (60). Согласование с результатами экспериментов настолько хорошее, что константа $2/b$ Брента должна в равенстве (65) принимать в качестве значения число 0.70, поэтому в уравнении (62) число 0.203 необходимо заменить числом 0.206.

На этом анализ средних значений C и D завершается. Анализ остальных трех величин, присутствующих в формуле для времени выполнения алгоритма B, выполняется довольно просто (см. упр. 6–8).

Теперь, когда примерно известно поведение алгоритма B в среднем, рассмотрим сценарий "наихудшего случая". Какие значения u и v являются в некотором смысле наиболее трудно обрабатываемыми? Предположим, как и ранее, что

$$\lfloor \lg u \rfloor = m \qquad \text{и} \qquad \lfloor \lg v \rfloor = n,$$

и попытаемся найти u и v, такие, при которых алгоритм будет выполняться наиболее медленно. Принимая во внимание, что операции вычитания выполняются несколько медленнее, чем операции сдвига, сформулированный вопрос можно перефразировать так: при каких значениях входных данных u и v потребуется выполнить наибольшее число вычитаний. Ответ звучит несколько неожиданно: максимальное значение величины C равно точно

$$\max(m, n) + 1, \tag{66}$$

хотя поверхностный (наивный) анализ позволил бы предположить, что возможны значительно бóльшие значения C (см. упр. 35). Вывод выражения (66), описывающий наихудший случай, довольно интересен, поэтому он предлагается читателям в качестве занимательной задачи (см. упр. 36 и 37).

УПРАЖНЕНИЯ

1. [*M21*] Приведите простой способ вывода соотношений (8)–(12) из соотношений (6) и (7).

2. [*M22*] Пусть известно, что число u делит $v_1 v_2 \ldots v_n$. Докажите, что u делит

$$\gcd(u, v_1)\gcd(u, v_2)\ldots\gcd(u, v_n).$$

3. [*M23*] Покажите, что число упорядоченных пар положительных чисел (u, v), таких, что $\operatorname{lcm}(u, v) = n$, есть число делителей n^2.

4. [*M21*] Покажите, что для любых положительных чисел u и v найдутся такой делитель u' числа u и такой делитель v' числа v, что $u' \perp v'$ и $u'v' = \operatorname{lcm}(u, v)$.

▸ **5.** [*M26*] Разработайте алгоритм для нахождения наибольшего общего делителя двух целых чисел, аналогичный алгоритму B, но базирующийся на представлении этих чисел в *уравновешенной троичной* системе счисления. Продемонстрируйте выполнение алгоритма на примере вычисления $\gcd(40902, 24140)$.

6. [*M22*] Пусть u и v — случайные положительные целые числа. Найдите среднее значение и среднеквадратичное отклонение величины A, входящей в формулу для времени выполнения алгоритма B. (Это количество сдвигов вправо чисел u и v во время подготовительной фазы.)

7. [*M20*] Проанализируйте величину B, входящую в формулу для времени выполнения алгоритма B.

▶ **8.** [*M25*] Покажите, что в программе B среднее значение величины E приблизительно равно $\frac{1}{2}C_{\rm ср}$, где $C_{\rm ср}$ — среднее значение C.

9. [*18*] Найдите $\gcd(31408, 2718)$, выполнив вычисления вручную с использованием алгоритма B. При помощи алгоритма X найдите также такие целые числа m и n, чтобы $31408m + 2718n = \gcd(31408, 2718)$.

▶ **10.** [*HM24*] Пусть q_n — количество упорядоченных пар целых чисел (u, v), принадлежащих интервалу $1 \le u, v \le n$, таких, что $u \perp v$. Назначение данного упражнения — доказать, что $\lim_{n\to\infty} q_n/n^2 = 6/\pi^2$, и тем самым сформулировать теорему D.

a) Используя принцип включения и исключения (раздел 1.3.3), покажите, что

$$q_n = n^2 - \sum_{p_1} \lfloor n/p_1 \rfloor^2 + \sum_{p_1<p_2} \lfloor n/p_1p_2 \rfloor^2 - \cdots,$$

где суммирование выполняется по всем *простым* числам p_i.

b) Функция *Мёбиуса* $\mu(n)$ определяется правилами $\mu(1) = 1$, $\mu(p_1p_2\ldots p_r) = (-1)^r$, если $p_1, p_2, \ldots, p_r$ — различные простые числа, и $\mu(n) = 0$, если n делится на квадрат простого числа. Покажите, что $q_n = \sum_{k\ge1} \mu(k)\lfloor n/k \rfloor^2$.

c) В качестве следствия из (b) докажите, что $\lim_{n\to\infty} q_n/n^2 = \sum_{k\ge1} \mu(k)/k^2$.

d) Докажите, что $(\sum_{k\ge1} \mu(k)/k^2)(\sum_{m\ge1} 1/m^2) = 1$. *Указание.* Если ряды сходятся абсолютно, то

$$\left(\sum_{k\ge1} a_k/k^z\right)\left(\sum_{m\ge1} b_m/m^z\right) = \sum_{n\ge1}\left(\sum_{d\backslash n} a_d b_{n/d}\right)\Big/ n^z.$$

11. [*M22*] Чему равна вероятность того, что $\gcd(u, v) \le 3$? (См. теорему D.) Чему равно *среднее* значение $\gcd(u, v)$?

12. [*M24*] (Э. Чезаро (E. Cesàro).) Пусть u и v — случайные положительные целые числа. Чему равно среднее число их общих (положительных) делителей? [*Указание.* См. тождество в упр. 10, (d) при $a_k = b_m = 1$.]

13. [*HM23*] Пусть u и v — случайные положительные *нечетные* целые числа. Покажите, что с вероятностью $8/\pi^2$ они взаимно просты.

▶ **14.** [*HM25*] Чему равно ожидаемое значение $\ln\gcd(u, v)$, если числа u и v (a) случайные положительные целые числа, (b) случайные положительные нечетные целые числа?

15. [*M21*] Чему равны значения чисел v_1 и v_2 по окончании выполнения алгоритма X?

▶ **16.** [*M22*] Разработайте алгоритм *деления u на v по модулю m* для положительных целых чисел u, v и m при v, взаимно простом с m. Другими словами, ваш алгоритм должен так вычислить w, принадлежащее интервалу $0 \le w < m$, чтобы получить $u \equiv vw$ (по модулю m).

▶ **17.** [*M20*] Пусть даны два целых числа u и v, таких, что выполняется $uv \equiv 1$ (по модулю 2^e). Объясните, как вычислить такое целое число u', чтобы выполнялось $u'v \equiv 1$ (по модулю 2^{2e}). [Это приводит к алгоритму вычисления обратных к нечетным числам по модулю степени 2, так как можно начать вычисления с таблицы всех обратных значений для $e = 8$ или $e = 16$.]

▸ **18.** [*M24*] Покажите, как расширить алгоритм L (как алгоритм A был расширен до алгоритма X) для решения уравнения (15) при больших u и v.

19. [*21*] Примените метод, описанный в разделе, для нахождения общего целочисленного решения следующих систем уравнений.

a) $\begin{aligned} 3x+7y+11z &= 1 \\ 5x+7y-\ 5z &= 3 \end{aligned}$ b) $\begin{aligned} 3x+7y+11z &= \ 1 \\ 5x+7y-\ 5z &= -3 \end{aligned}$

20. [*M37*] Пусть u и v — нечетные целые числа, независимо и равномерно распределенные в интервалах $2^m \le u < 2^{m+1}$, $2^n \le v < 2^{n+1}$. Чему равна *точная* вероятность того, что в алгоритме B за один цикл "вычитание и сдвиг" величины u и v уменьшатся до интервалов $2^{m'} \le u < 2^{m'+1}$, $2^{n'} \le v < 2^{n'+1}$ (выражается как функция m, n, m' и n')?

21. [*HM26*] Пусть C_{mn} и D_{mn} — среднее число шагов вычитания и среднее число сдвигов при выполнении алгоритма B соответственно в случае, если числа u и v нечетны, $\lfloor \lg u \rfloor = m$ и $\lfloor \lg v \rfloor = n$. Покажите, что при фиксированных n и $m \to \infty$ получаем значения $C_{mn} = \frac{1}{2}m + O(1)$ и $D_{mn} = m + O(1)$.

22. [*M28*] Продолжая предыдущее упражнение, покажите, что если для некоторых постоянных α, β и γ выполняется $C_{mn} = \alpha m + \beta n + \gamma$, то

$$\sum_{1\le n<m\le N} (N-m)(N-n)2^{m+n-2}C_{mn} = 2^{2N}\left(\tfrac{11}{27}(\alpha+\beta)N + O(1)\right),$$

$$\sum_{1\le n\le N} (N-n)^2 2^{2n-2}C_{nn} = 2^{2N}\left(\tfrac{5}{27}(\alpha+\beta)N + O(1)\right).$$

▸ **23.** [*M20*] Чему равна вероятность того, что после выполнения алгоритмом B n циклов "вычитание и сдвиг" будет выполняться неравенство $v/u \le x$ при условии, что выполнение алгоритма начинается с больших случайных целых чисел? (Здесь x — произвольное вещественное число ≥ 0; не предполагается, что $u \ge v$.)

24. [*M20*] Предположим, что на шаге B6 $u > v$, и допустим, что отношение v/u имеет ограниченное распределение Брента G. Чему равна вероятность того, что при выполнении шага B6 в следующий раз будет $u < v$?

25. [*M21*] Из уравнения (46) следует, что $\rho_1 = -\lambda$. Докажите, что $\rho_2 = \lambda/2$.

26. [*M22*] Докажите, что если $G(x)$ удовлетворяет уравнениям (36)–(40), то

$$2G(x) - 5G(2x) + 2G(4x) = G(1+2x) - 2G(1+4x) + 2G(1+1/x) - G(1+1/2x).$$

27. [*M22*] Докажите соотношение (58), которое выражает ψ_n в обозначениях чисел Бернулли.

28. [*HM36*] Исследуйте асимптотическое поведение ψ_n. *Указание.* См. упр. 6.3–34.

▸ **29.** [*HM26*] (P. П. Брент (R. P. Brent).) Найдите $G_1(x)$ — распределение величины

$$\min(u,v)/\max(u,v)$$

в результате выполнения первого цикла "вычитание и сдвиг" алгоритма B согласно (35). *Указание.* Положите $S_{n+1}(x) = \sum_{k=1}^{\infty} 2^{-k} G_n(1/(1+2^k x))$ и примените к гармоническим суммам преобразование Меллина [см. P. Flajolet, X. Gourdon, P. Dumas, *Theor. Comp. Sci.* **144** (1995), 3–58].

30. [*HM39*] Продолжая предыдущее упражнение, найдите $G_2(x)$.

31. [*HM46*] Докажите или опровергните справедливость соотношения Валле (Vallée) (61).

32. [*HM47*] Является ли единственной функция $G(x)$, удовлетворяющая уравнениям (36) и (37)?

33. [*M46*] Исследуйте предложенный Харрисом (Harris) "бинарный алгоритм Евклида", сформулированный после программы B.

34. [*HM49*] Докажите строго, что модель Брента описывает асимптотическое поведение алгоритма B.

35. [*M23*] Для любых неотрицательных целых чисел $m, n \geq 0$ рассмотрите ориентированный граф с вершинами (m, n) и дугами от вершин (m, n) к вершинам (m', n') для всех возможных циклов "вычитание и сдвиг" алгоритма B, чтобы преобразовать целые числа u и v, обладающие свойствами $\lfloor \lg u \rfloor = m$ и $\lfloor \lg v \rfloor = n$, в целые числа u' и v', обладающие свойствами $\lfloor \lg u' \rfloor = m'$ и $\lfloor \lg v' \rfloor = n'$. Существует также специальная вершина "Стоп" с дугой, соединяющей вершины (n, n), и "Стоп" для всех $n \geq 0$. Определите длину самого длинного участка пути от вершины (m, n) до вершины "Стоп". (Эта величина дает верхнюю границу максимального времени выполнения алгоритма B.)

▶ **36.** [*M28*] Для $m \geq n \geq 1$ найдите такие значения u и v, обладающих свойствами $\lfloor \lg u \rfloor = m$ и $\lfloor \lg v \rfloor = n$, чтобы для выполнения алгоритма B потребовалось $m + 1$ шагов вычитания.

37. [*M32*] Докажите, что число вычитаний на шаге B6 алгоритма B никогда не превысит величины $1 + \lfloor \lg \max(u, v) \rfloor$.

▶ **38.** [*M32*] (P. B. Госпер (R. W. Gosper).) Покажите, как модифицировать алгоритм B для больших чисел с помощью идеи, аналогичной той, которая используется в алгоритме L.

▶ **39.** [*M28*] (B. P. Пратт (V. R. Pratt).) Распространите алгоритм B на алгоритм Y, который является аналогом алгоритма X.

▶ **40.** [*M25*] (P. П. Брент и X. T. Кунг (H. T. Kung).) Излагаемый ниже вариант бинарного алгоритма лучше алгоритма B с точки зрения аппаратной реализации, так как он не требует проверки знака числа $u - v$. Допустим, что u нечетно; тогда числа u и v могут быть одновременно либо положительны, либо отрицательны.

K1. [Начальная установка.] Присвоить $c \leftarrow 0$. (Этот счетчик оценивает разность между $\lg |u|$ и $\lg |v|$.)

K2. [Выполнено?] Если $v = 0$, то закончить алгоритм с результатом $|u|$.

K3. [Сделать v нечетным.] Присваивать $v \leftarrow v/2$ и $c \leftarrow c + 1$ нуль или более раз до тех пор, пока v не станет нечетным.

K4. [Сделать $c \leq 0$.] Если $c > 0$, то заменить $u \leftrightarrow v$ и присвоить $c \leftarrow -c$.

K5. [Сократить.] Присвоить $w \leftarrow (u + v)/2$. Если w четно, присвоить $v \leftarrow w$, в противном случае присвоить $v \leftarrow w - v$ и вернуться к шагу K2. ▮

Докажите, что шаг K2 выполняется не более чем $2 + 2 \lg \max(|u|, |v|)$ раз.

41. [*M22*] При помощи алгоритма Евклида найдите простую формулу для

$$\gcd(10^m - 1, 10^n - 1),$$

где m и n — неотрицательные целые числа.

42. [*M30*] Вычислите определитель

$$\begin{vmatrix} \gcd(1,1) & \gcd(1,2) & \dots & \gcd(1,n) \\ \gcd(2,1) & \gcd(2,2) & \dots & \gcd(2,n) \\ \vdots & \vdots & & \vdots \\ \gcd(n,1) & \gcd(n,2) & \dots & \gcd(n,n) \end{vmatrix}.$$

*4.5.3. Анализ алгоритма Евклида

Время выполнения алгоритма Евклида зависит от T — количества выполненных шагов деления A2 (см. алгоритм 4.5.2A и программу 4.5.2A). Величина T является также важным фактором, определяющим время выполнения других алгоритмов, например вычисление функций, удовлетворяющих закону обратимости (см. раздел 3.3.3). Мы увидим в этом разделе, что математический анализ величины T интересен и поучителен.

Связь с цепными дробями. Алгоритм Евклида тесно связан с *цепными (непрерывными) дробями*, которые можно выразить в виде

$$\cfrac{b_1}{a_1+\cfrac{b_2}{a_2+\cfrac{b_3}{\cdots\cfrac{}{a_{n-1}+\cfrac{b_n}{a_n}}}}} = b_1/(a_1+b_2/(a_2+b_3/(\cdots/(a_{n-1}+b_n/a_n)\ldots))). \tag{1}$$

Существует красивая теория цепных дробей, которой посвящен ряд классических книг, таких как O. Perron, *Die Lehre von den Kettenbrüchen*, 3rd edition (Stuttgart: Teubner, 1954), 2 volumes; Хинчин А. Я. *Цепные дроби.* — М.-Л.: ГИТТЛ, 1949; H. S. Wall, *Analytic Theory of Continued Fractions* (New York: Van Nostrand, 1948). С ранней историей развития теории цепных дробей можно ознакомиться в книге Claude Brezinski, *History of Continued Fractions and Padé Approximants* (Springer, 1991). Мы ограничимся сравнительно кратким изложением этой теории, рассматривая только те ее аспекты, которые необходимы для более глубокого понимания поведения алгоритма Евклида.

Первостепенный интерес для нас будут представлять только те цепные дроби, в которых все b в выражении (1) равны единице. Для удобства записи введем обозначение

$$//x_1, x_2, \ldots, x_n// = 1/(x_1 + 1/(x_2 + 1/(\cdots(x_{n-1} + 1/x_n)\ldots))). \tag{2}$$

Тогда, к примеру,

$$//x_1// = \frac{1}{x_1}, \qquad //x_1, x_2// = \frac{1}{x_1 + 1/x_2} = \frac{x_2}{x_1x_2 + 1}. \tag{3}$$

Если $n = 0$, символ $//x_1, \ldots, x_n//$ принимается равным нулю. Введем, кроме того, так называемые *континуанты, или K-полиномы* $K_n(x_1, x_2, \ldots, x_n)$, от n переменных при $n \geq 0$, задаваемые правилом

$$K_n(x_1, x_2, \ldots, x_n) = \begin{cases} 1 & \text{при } n = 0; \\ x_1 & \text{при } n = 1; \\ x_1K_{n-1}(x_2, \ldots, x_n) + K_{n-2}(x_3, \ldots, x_n) & \text{при } n > 1. \end{cases} \tag{4}$$

Таким образом, $K_2(x_1, x_2) = x_1x_2+1$, $K_3(x_1, x_2, x_3) = x_1x_2x_3+x_1+x_3$ и т. д. В общем случае, как заметил Л. Эйлер в 18 веке, $K_n(x_1, x_2, \ldots, x_n)$ — сумма всех членов, получаемых, если, начиная с $x_1x_2\ldots x_n$, вычеркивать нуль или более неперекрывающихся пар соседних переменных x_jx_{j+1}; количество таких членов равно F_{n+1}.

Основное свойство K-полиномов состоит в том, что

$$//x_1, x_2, \ldots, x_n// = K_{n-1}(x_2, \ldots, x_n)/K_n(x_1, x_2, \ldots, x_n), \qquad n \geq 1. \tag{5}$$

Это можно доказать по индукции: поскольку из (5) следует, что

$$x_0 + //x_1, \ldots, x_n// = K_{n+1}(x_0, x_1, \ldots, x_n)/K_n(x_1, \ldots, x_n),$$

$//x_0, x_1, \ldots, x_n//$ есть величина, обратная правой части последнего равенства.

K-полиномы симметричны в том смысле, что

$$K_n(x_1, x_2, \ldots, x_n) = K_n(x_n, \ldots, x_2, x_1). \tag{6}$$

Это вытекает из указанного выше свойства, подмеченного Эйлером, и как следствие имеем

$$K_n(x_1, \ldots, x_n) = x_n K_{n-1}(x_1, \ldots, x_{n-1}) + K_{n-2}(x_1, \ldots, x_{n-2}) \tag{7}$$

для $n > 1$. K-полиномы удовлетворяют также важному тождеству

$$K_n(x_1, \ldots, x_n)K_n(x_2, \ldots, x_{n+1}) - K_{n+1}(x_1, \ldots, x_{n+1})K_{n-1}(x_2, \ldots, x_n) = (-1)^n, \qquad n \geq 1. \tag{8}$$

(См. упр. 4.) Из этого тождества и равенства (5) следует, что

$$//x_1, \ldots, x_n// = \frac{1}{q_0 q_1} - \frac{1}{q_1 q_2} + \frac{1}{q_2 q_3} - \cdots + \frac{(-1)^{n-1}}{q_{n-1} q_n}, \quad \text{где } q_k = K_k(x_1, \ldots, x_k). \tag{9}$$

Итак, K-полиномы тесно связаны с цепными дробями.

Всякое вещественное число X в интервале $0 \leq X < 1$ представляется в виде *правильной цепной дроби*, определяемой следующим образом. Положим, что $X_0 = X$. Для всех $n \geq 0$, таких, что $X_n \neq 0$, положим

$$A_{n+1} = \lfloor 1/X_n \rfloor, \qquad X_{n+1} = 1/X_n - A_{n+1}. \tag{10}$$

Если $X_n = 0$, то величины A_{n+1} и X_{n+1} не определены и правильной цепной дробью для X будет $//A_1, \ldots, A_n//$. Если $X_n \neq 0$, данное определение гарантирует, что $0 \leq X_{n+1} < 1$, так что любое A будет положительным целым числом. Из определений (10) следует также, что

$$X = X_0 = \frac{1}{A_1 + X_1} = \frac{1}{A_1 + 1/(A_2 + X_2)} = \cdots,$$

поэтому

$$X = //A_1, \ldots, A_{n-1}, A_n + X_n// \tag{11}$$

для всех $n \geq 1$, для которых определено X_n. В частности, если $X_n = 0$, то $X = //A_1, \ldots, A_n//$. Если $X_n \neq 0$, то число X всегда лежит *между* $//A_1, \ldots, A_n//$ и $//A_1, \ldots, A_n + 1//$, так как согласно (7) величина $q_n = K_n(A_1, \ldots, A_n + X_n)$ монотонно возрастает от $K_n(A_1, \ldots, A_n)$ до $K_n(A_1, \ldots, A_n + 1)$ при возрастании X_n от 0 до 1. Согласно равенству (9) при возрастании q_n цепная дробь будет возрастать или убывать в зависимости от того, будет ли число n четным или нечетным.

Действительно, в силу (5), (7), (8) и (10)

$$\begin{aligned}
|X - /\!/A_1, \ldots, A_n/\!/| &= |/\!/A_1, \ldots, A_n + X_n/\!/ - /\!/A_1, \ldots, A_n/\!/| \\
&= |/\!/A_1, \ldots, A_n, 1/X_n/\!/ - /\!/A_1, \ldots, A_n/\!/| \\
&= \left| \frac{K_n(A_2, \ldots, A_n, 1/X_n)}{K_{n+1}(A_1, \ldots, A_n, 1/X_n)} - \frac{K_{n-1}(A_2, \ldots, A_n)}{K_n(A_1, \ldots, A_n)} \right| \\
&= 1/\big(K_n(A_1, \ldots, A_n) K_{n+1}(A_1, \ldots, A_n, 1/X_n)\big) \\
&\le 1/\big(K_n(A_1, \ldots, A_n) K_{n+1}(A_1, \ldots, A_n, A_{n+1})\big). \qquad (12)
\end{aligned}$$

Поэтому $/\!/A_1, \ldots, A_n/\!/$ — экстремально близкое приближение к X, если n не мало́. Если X_n не равно нулю для всех n, получаем *бесконечную цепную дробь* $/\!/A_1, A_2, A_3, \ldots/\!/$, значение которой определяется как

$$\lim_{n\to\infty} /\!/A_1, A_2, \ldots, A_n/\!/;$$

и из неравенства (12) ясно, что этот предел равен X.

Разложение вещественных чисел в правильную цепную дробь обладает рядом свойств, аналогичных свойствам чисел, представленных в десятичной системе счисления. Если при помощи приведенных выше формул разложить в правильные цепные дроби некоторые известные вещественные числа, получим, например,

$$\begin{aligned}
\tfrac{8}{29} &= /\!/3, 1, 1, 1, 2/\!/; \\
\sqrt{\tfrac{8}{29}} &= /\!/1, 1, 9, 2, 2, 3, 2, 2, 9, 1, 2, 1, 9, 2, 2, 3, 2, 2, 9, 1, 2, 1, 9, 2, 2, 3, 2, 2, 9, 1, \ldots/\!/; \\
\sqrt[3]{2} &= 1 + /\!/3, 1, 5, 1, 1, 4, 1, 1, 8, 1, 14, 1, 10, 2, 1, 4, 12, 2, 3, 2, 1, 3, 4, 1, 1, 2, 14, 3, \ldots/\!/; \\
\pi &= 3 + /\!/7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, 84, 2, 1, 1, 15, 3, 13, \ldots/\!/; \qquad (13) \\
e &= 2 + /\!/1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, 1, 1, 14, 1, 1, 16, 1, 1, 18, 1, \ldots/\!/; \\
\gamma &= /\!/1, 1, 2, 1, 2, 1, 4, 3, 13, 5, 1, 1, 8, 1, 2, 4, 1, 1, 40, 1, 11, 3, 7, 1, 7, 1, 1, 5, 1, 49, \ldots/\!/; \\
\phi &= 1 + /\!/1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, \ldots/\!/.
\end{aligned}$$

Числа $A_1, A_2, \ldots$ называются *частичными отношениями* числа X. Обратите внимание на закономерность поведения частичных отношений для чисел $\sqrt{8/29}$, ϕ и e; причины этой закономерности рассматриваются в упр. 12 и 16. Для чисел же $\sqrt[3]{2}$, π и γ никакой видимой закономерности в поведении частичных отношений не наблюдается.

Интересно отметить, что когда древние греки обнаружили существование иррациональных чисел, то, по существу, первое определение вещественных чисел они дали в терминах бесконечных цепных дробей. (Позже они приняли предложение Евдокса вместо этого определить $x = y$ следующим образом: "$x < r$ для тех же рациональных чисел r, таких, что $y < r$".) (См. O. Becker, *Quellen und Studien zur Geschichte Math., Astron., Physik* **B2** (1933), 311–333.)

Если X — рациональное число, то его разложение в правильную цепную дробь естественным образом соотносится с алгоритмом Евклида. Предположим, что $X = v/u$ для $u > v \ge 0$. Процесс разложения в правильную цепную дробь начинается с $X_0 = X$; обозначим $U_0 = u$, $V_0 = v$. В предположении, что $X_n = V_n/U_n \ne 0$,

уравнение (10) принимает вид

$$A_{n+1} = \lfloor U_n/V_n \rfloor, \qquad X_{n+1} = U_n/V_n - A_{n+1} = (U_n \bmod V_n)/V_n. \tag{14}$$

Поэтому, если принять

$$U_{n+1} = V_n, \qquad V_{n+1} = U_n \bmod V_n, \tag{15}$$

то условие $X_n = V_n/U_n$ будет выполняться в течение всего процесса разложения дроби. Далее, соотношение (15) — это в точности преобразование, выполняемое в алгоритме Евклида над переменными u и v (см. алгоритм 4.5.2A, шаг A2). Например, поскольку $\frac{8}{29} = //3,1,1,1,2//$, известно, что применение алгоритма Евклида к числам $u = 29$ и $v = 8$ потребует выполнения ровно пяти шагов деления и на шаге A2 частными $\lfloor u/v \rfloor$ будут последовательно 3, 1, 1, 1 и 2. Если $X_n = 0$ и $n \geq 1$, то частичное отношение A_n всегда должно равняться 2 или более, поскольку X_{n-1} меньше единицы.

Такое соответствие с алгоритмом Евклида означает, что правильная цепная дробь для числа X заканчивается на некотором шаге со значением $X_n = 0$ тогда и только тогда, когда X — рациональное число, ибо очевидно, что X_n не может равняться нулю, если число X иррационально, и, наоборот, известно, что алгоритм Евклида всегда заканчивается. Если частичные отношения, получаемые в ходе выполнения алгоритма Евклида, равны $A_1, A_2, \ldots, A_n$, то согласно (5)

$$\frac{v}{u} = \frac{K_{n-1}(A_2, \ldots, A_n)}{K_n(A_1, A_2, \ldots, A_n)}. \tag{16}$$

Эта формула справедлива также в случае $u < v$, когда $A_1 = 0$. Более того, согласно (8) $K_{n-1}(A_2, \ldots, A_n)$ и $K_n(A_1, A_2, \ldots, A_n)$ будут взаимно просты и дробь в правой части выражения (16) является несократимой. Поэтому

$$u = K_n(A_1, A_2, \ldots, A_n)d, \qquad v = K_{n-1}(A_2, \ldots, A_n)d, \tag{17}$$

где $d = \gcd(u, v)$.

Наихудший случай. Теперь можно использовать сформулированные выше свойства для того, чтобы выяснить, как себя ведет алгоритм Евклида в наихудшем случае, или, другими словами, чтобы найти верхнюю границу числа шагов деления. Наихудший случай имеет место, когда на входе задаются последовательные числа Фибоначчи.

Теорема F. *Пусть для $n \geq 1$ u и v — целые числа ($u > v > 0$), такие, что при применении к u и v алгоритма Евклида необходимо выполнить ровно n шагов деления, причем число u — наименьшее возможное число, удовлетворяющее этим условиям. Тогда $u = F_{n+2}$ и $v = F_{n+1}$.*

Доказательство. Согласно (17) должно быть $u = K_n(A_1, A_2, \ldots, A_n)d$, где $A_1, A_2, \ldots, A_n$ и d — положительные целые числа, а $A_n \geq 2$. Поскольку K_n — полином с неотрицательными коэффициентами, включающий все переменные, минимальное значение достигается только тогда, когда $A_1 = 1, \ldots, A_{n-1} = 1, A_n = 2, d = 1$. Подставляя эти значения в (17), получаем искомый результат. ∎

Эта теорема явилась первым практическим применением последовательности чисел Фибоначчи. С тех пор числа Фибоначчи нашли широкое применение при выполнении и исследовании алгоритмов. Следующий результат получил Т. Ф. де Ланьи (T. F. de Lagny) [*Mém. Acad. Sci. Paris* **11** (1733), 363–364]. Он составил таблицы нескольких первых K-полиномов и обнаружил, что числа Фибоначчи дают для цепных дробей заданной длины наименьшие числитель и знаменатель. Однако он совсем не имел в виду вычисление наибольшего общего делителя. Первым на связь между числами Фибоначчи и алгоритмом Евклида указал Э. Леже (É. Léger) [*Correspondance Math. et Physique* **9** (1837), 483–485].

Вскоре после этого П. Ж. Е. Финк (P. J. É. Finck) [*Traité Élémentaire d'Arithmétique* (Strasbourg, 1841), 44] другим методом доказал, что если $u > v > 0$, то $\gcd(u, v)$ вычисляется не более чем за $(2\lg v + 1)$ шагов, а Г. Ламе (G. Lamé) [*Comptes Rendus Acad. Sci. Paris* **19** (1844), 867–870] распространил полученный результат на $5\lceil \log_{10}(v+1) \rceil$. Полное изложение этих первых работ по анализу алгоритмов появилось в интересном обзоре Дж. О. Шэллита (J. O. Shallit) *Historia Mathematica* **21** (1994), 401–419. Однако более точная оценка наихудшего случая является прямым следствием теоремы F.

Следствие L. Если $0 \le v < N$, то число шагов деления, необходимых алгоритму 4.5.2A для выполнения операций с числами u и v, не превышает $\lfloor \log_\phi (3-\phi)N \rfloor$.

Доказательство. После выполнения шага A1 имеем $v > u \bmod v$. Поэтому согласно теореме F максимальное число шагов n имеет место в том случае, когда $v = F_{n+1}$ и $u \bmod v = F_n$. Так как $F_{n+1} < N$, то $\phi^{n+1}/\sqrt{5} < N$ (см. 1.2.8–(15)). Следовательно, $\phi^n < (\sqrt{5}/\phi)N = (3-\phi)N$. ∎

Величина $\log_\phi (3-\phi)N$ приближенно равна $2.078 \ln N + .6723 \approx 4.785 \log_{10} N + .6723$. По поводу обобщений теоремы F обратитесь к упр. 31, 36 и 38.

Приближенная модель. Теперь, когда известно максимально возможное количество шагов деления, попытаемся найти их *среднее* число. Пусть $T(m, n)$ — число шагов деления в случае, когда алгоритм Евклида имеет на входе $u = m$ и $v = n$. Тогда

$$T(m,0) = 0; \qquad T(m,n) = 1 + T(n, m \bmod n) \qquad \text{при } n \ge 1. \tag{18}$$

Пусть T_n — среднее число шагов деления, когда $v = n$, а u выбирается случайным образом. Поскольку после выполнения первого шага деления на ход выполнения алгоритма влияет только значение $u \bmod v$, можно записать

$$T_n = \frac{1}{n} \sum_{0 \le k < n} T(k, n). \tag{19}$$

Например, $T(0,5) = 1$, $T(1,5) = 2$, $T(2,5) = 3$, $T(3,5) = 4$, $T(4,5) = 3$, так что

$$T_5 = \tfrac{1}{5}(1+2+3+4+3) = 2\tfrac{3}{5}.$$

Задача состоит в оценке T_n при больших значениях n. Попробуем сначала применить приближение, предложенное Р. У. Флойдом (R. W. Floyd). Можно предположить, что для $0 \le k < n$ значение n, по существу, "случайно" по модулю k, так что можно записать

$$T_n \approx 1 + \frac{1}{n}(T_0 + T_1 + \cdots + T_{n-1}).$$

Тогда $T_n \approx S_n$, где последовательность $\langle S_n \rangle$ есть решение рекуррентного соотношения

$$S_0 = 0, \qquad S_n = 1 + \frac{1}{n}(S_0 + S_1 + \cdots + S_{n-1}), \qquad n \geq 1. \tag{20}$$

Это рекуррентное соотношение легко решить, если учесть, что

$$\begin{aligned} S_{n+1} &= 1 + \frac{1}{n+1}(S_0 + S_1 + \cdots + S_{n-1} + S_n) \\ &= 1 + \frac{1}{n+1}\big(n(S_n - 1) + S_n\big) = S_n + \frac{1}{n+1}. \end{aligned}$$

Следовательно, S_n равно $1 + \frac{1}{2} + \cdots + \frac{1}{n} = H_n$ — гармоническому числу. Теперь приближение $T_n \approx S_n$ дает $T_n \approx \ln n + O(1)$.

Сравнение этого приближения с таблицами истинных значений T_n показывает, однако, что $\ln n$ — это слишком много. (T_n возрастает не так быстро.) Предположение, что число n случайно по модулю k, является по этой причине слишком пессимистическим. И в самом деле, более внимательный анализ показывает, что среднее значение числа $n \bmod k$ меньше среднего значения числа $\frac{1}{2}k$ при $1 \leq k \leq n$:

$$\begin{aligned} \frac{1}{n}\sum_{1\leq k\leq n}(n \bmod k) &= \frac{1}{n}\sum_{1\leq k,q\leq n}(n - qk)\left[\lfloor n/(q+1)\rfloor < k \leq \lfloor n/q\rfloor\right] \\ &= n - \frac{1}{n}\sum_{1\leq q\leq n} q\left(\binom{\lfloor n/q\rfloor + 1}{2} - \binom{\lfloor n/(q+1)\rfloor + 1}{2}\right) \\ &= n - \frac{1}{n}\sum_{1\leq q\leq n}\binom{\lfloor n/q\rfloor + 1}{2} \\ &= \left(1 - \frac{\pi^2}{12}\right)n + O(\log n) \end{aligned} \tag{21}$$

(см. упр. 4.5.2–10(c)). Это равно примерно $.1775n$, а не $.25n$. Поэтому значение $n \bmod k$ меньше значений, предсказываемых моделью Флойда, а алгоритм Евклида выполняется быстрее, чем предсказывает приближенная модель.

Непрерывная модель. При $v = N$ поведение алгоритма Евклида, по существу, определяется поведением правильной цепной дроби для $X = 0/N$, $1/N$ ... ..., $(N-1)/N$. При очень больших N естественно рассматривать разложение числа X в правильную цепную дробь фактически как случайного вещественного числа, равномерно распределенного в интервале $[0\,..\,1)$. Рассмотрим функцию распределения

$$F_n(x) = \Pr(X_n \leq x) \qquad \text{при } 0 \leq x \leq 1 \tag{22}$$

равномерно распределенного числа $X = X_0$. По определению правильных цепных дробей получаем $F_0(x) = x$ и

$$\begin{aligned} F_{n+1}(x) &= \sum_{k\ge 1} \Pr(k \le 1/X_n \le k + x) \\ &= \sum_{k\ge 1} \Pr\bigl(1/(k+x) \le X_n \le 1/k\bigr) \\ &= \sum_{k\ge 1} \bigl(F_n(1/k) - F_n\bigl(1/(k+x)\bigr)\bigr). \end{aligned} \tag{23}$$

Если распределения $F_0(x), F_1(x), \ldots$, определяемые этими формулами, сходятся к предельному распределению $F_\infty(x) = F(x)$, то получаем

$$F(x) = \sum_{k\ge 1} \bigl(F(1/k) - F\bigl(1/(k+x)\bigr)\bigr). \tag{24}$$

(Аналогичное соотношение, 4.5.2–(36), было получено при рассмотрении бинарных алгоритмов определения наибольшего общего делителя.) При любом основании $b > 1$ одна из функций, удовлетворяющих уравнению (24), имеет вид $F(x) = \log_b(1 + x)$ (см. упр. 19). Из дополнительного условия $F(1) = 1$ следует, что $b = 2$. Таким образом, вполне обосновано предположение, что $F(x) = \lg(1 + x)$ и что последовательность $F_n(x)$ сходится к этому пределу.

Можно было бы предположить, например, что $F(\frac{1}{2}) = \lg(\frac{3}{2}) \approx 0.58496$. Посмотрим, насколько близко $F_n(\frac{1}{2})$ к этому значению при малых n. Имеем $F_0(\frac{1}{2}) = 0.50000$ и

$$F_1(x) = \sum_{k\ge 1}\Bigl(\frac{1}{k} - \frac{1}{k+x}\Bigr) = H_x;$$

$$F_1(\tfrac{1}{2}) = H_{1/2} = 2 - 2\ln 2 \approx 0.61371;$$

$$F_2(\tfrac{1}{2}) = H_{2/2} - H_{2/3} + H_{2/4} - H_{2/5} + H_{2/6} - H_{2/7} + \cdots$$

(см. табл. 3 приложения A). Обобщение на степенной ряд

$$H_x = \zeta(2)x - \zeta(3)x^2 + \zeta(4)x^3 - \zeta(5)x^5 + \cdots \tag{25}$$

позволяет определить численное значение

$$F_2(\tfrac{1}{2}) = 0.57655\,93276\,99914\,08418\,82618\,72122\,27055\,92452 - . \tag{26}$$

Получаем значение, близкое к 0.58496. Но совсем не очевидно, как получить хорошую оценку $F_n(\frac{1}{2})$ при $n = 3$, т. е. при значении, меньшем, чем те значения, которые считаются действительно большими.

Впервые распределения $F_n(x)$ были исследованы К. Ф. Гауссом (C. F. Gauss), который начал решать эту проблему 5 февраля 1799 года. В его записях за 1800 год перечислены различные рекуррентные соотношения и приведена краткая таблица значений, включающая (неточные) приближения для $F_2(\frac{1}{2}) \approx 0.5748$. После завершения этих вычислений Гаусс записал: "Tam complicatæ evadunt, ut nulla spes superesse videatur", т. е. "Они получаются такими сложными, что, кажется, нет никакой надежды". Двенадцать лет спустя Гаусс написал письмо Лапласу, в

котором сформулировал эту проблему как таковую, для которой он не смог найти удовлетворяющего его решения. Он писал: "В результате простых рассуждений я обнаружил, что $F_n(x) = \log(1+x)/\log 2$ для бесконечного n. Но все дальнейшие попытки, предпринятые мною для оценки $F_n(x) - \log(1+x)/\log 2$ для очень больших, но конечных значений n, были безуспешны". Гаусс так никогда и не опубликовал свои "очень простые рассуждения", поэтому не совсем ясно, нашел ли он строгое доказательство. [См. Gauss's *Werke*, vol. 10^1, 552–556.] Прошло более ста лет, прежде чем такое доказательство было, наконец, опубликовано Р. О. Кузьминым в *Atti del Congresso Internazionale dei Matematici* **6** (Bologna, 1928), 83–89, который показал, что

$$F_n(x) = \lg(1+x) + O(e^{-A\sqrt{n}})$$

для некоторой положительной константы A. Остаточный член $O(e^{-An})$ был получен вскоре после этого Полем Леви (Paul Lévy) [*Bull. Soc. Math. de France* **57** (1929), 178–194]*. Но проблема, сформулированная Гауссом, которая заключается в поиске асимптотического поведения функции $F_n(x) - \lg(1+x)$, фактически не была решена до 1974 года, пока Эдуард Вирсинг (Eduard Wirsing) не опубликовал ее блестящий анализ [*Acta Arithmetica* **24** (1974), 507–528]. Здесь мы исследуем самые простые аспекты приближения Вирсинга, поскольку его метод основан на использовании линейных операторов.

Если G — произвольная функция x, определенная на интервале $0 \le x \le 1$, то определим функцию SG в виде

$$SG(x) = \sum_{k\ge 1} \left(G\Big(\frac{1}{k}\Big) - G\Big(\frac{1}{k+x}\Big)\right). \tag{27}$$

Таким образом, S есть оператор, переводящий функцию из одного состояния в другое. В частности, из соотношения (23) получаем $F_{n+1}(x) = SF_n(x)$; тогда

$$F_n = S^n F_0. \tag{28}$$

(В данном случае F_n означает функцию распределения, а *не* числа Фибоначчи.) Заметим, что S — "линейный оператор", т. е. $S(cG) = c(SG)$ для всех постоянных c, и $S(G_1 + G_2) = SG_1 + SG_2$.

Теперь, если для G существуют ограниченные первые производные, выражение (27) можно почленно продифференцировать и показать, что

$$(SG)'(x) = \sum_{k\ge 1} \frac{1}{(k+x)^2} G'\Big(\frac{1}{k+x}\Big); \tag{29}$$

так что функция SG также имеет ограниченную первую производную. (Почленное дифференцирование сходящегося ряда оправдано в том случае, когда равномерно сходится ряд, составленный из производных; см., например, K. Knopp, *Theory and Application of Infinite Series* (Glasgow: Blackie, 1951), §47.)

---

* Изложение интересного доказательства Леви приведено в первом издании этой книги.

Пусть $H = SG$ и пусть $g(x) = (1+x)G'(x)$, $h(x) = (1+x)H'(x)$. Отсюда следует, что

$$h(x) = \sum_{k\ge1} \frac{1+x}{(k+x)^2}\Big(1+\frac{1}{k+x}\Big)^{-1} g\Big(\frac{1}{k+x}\Big)$$

$$= \sum_{k\ge1}\Big(\frac{k}{k+1+x} - \frac{k-1}{k+x}\Big) g\Big(\frac{1}{k+x}\Big).$$

Другими словами, $h = Tg$, где T — линейный оператор, такой, что

$$Tg(x) = \sum_{k\ge1}\Big(\frac{k}{k+1+x} - \frac{k-1}{k+x}\Big) g\Big(\frac{1}{k+x}\Big). \tag{30}$$

Продолжая, увидим, что если для функции g существует ограниченная первая производная, то (30) можно почленно продифференцировать, чтобы показать, что это же утверждение справедливо и в отношении Tg:

$$(Tg)'(x) = -\sum_{k\ge1}\Big(\Big(\frac{k}{(k+1+x)^2} - \frac{k-1}{(k+x)^2}\Big) g\Big(\frac{1}{k+x}\Big)$$

$$+ \Big(\frac{k}{k+1+x} - \frac{k-1}{k+x}\Big)\frac{1}{(k+x)^2} g'\Big(\frac{1}{k+x}\Big)\Big)$$

$$= -\sum_{k\ge1}\Big(\frac{k}{(k+1+x)^2}\Big(g\Big(\frac{1}{k+x}\Big) - g\Big(\frac{1}{k+1+x}\Big)\Big)$$

$$+ \frac{1+x}{(k+x)^3(k+1+x)} g'\Big(\frac{1}{k+x}\Big)\Big).$$

Соответственно существует третий линейный оператор U, такой, что $(Tg)' = -U(g')$, т. е.

$$U\varphi(x) = \sum_{k\ge1}\Big(\frac{k}{(k+1+x)^2}\int_{1/(k+1+x)}^{1/(k+x)} \varphi(t)\,dt + \frac{1+x}{(k+x)^3(k+1+x)}\varphi\Big(\frac{1}{k+x}\Big)\Big). \tag{31}$$

Имеет ли все это отношение к сформулированной выше проблеме? Если положить

$$F_n(x) = \lg(1+x) + R_n\big(\lg(1+x)\big), \tag{32}$$

$$f_n(x) = (1+x)\,F_n'(x) = \frac{1}{\ln 2}\big(1 + R_n'\big(\lg(1+x)\big)\big), \tag{33}$$

то получим

$$f_n'(x) = R_n''\big(\lg(1+x)\big)\big/\big((\ln 2)^2(1+x)\big). \tag{34}$$

В результате выполненных преобразований исчезает эффект влияния члена $\lg(1+x)$. Далее, так как $F_n = S^nF_0$, получаем $f_n = T^nf_0$ и $f_n' = (-1)^nU^nf_0'$. По индукции обе функции F_n и f_n имеют ограниченные производные. Таким образом, равенство (34) приобретает вид

$$(-1)^nR_n''\big(\lg(1+x)\big) = (1+x)(\ln 2)^2\,U^nf_0'(x). \tag{35}$$

Тогда $F_0(x) = x$, $f_0(x) = 1+x$ и $f_0'(x)$ есть постоянная функция, равная 1. Покажем, что оператор U^n переводит постоянную функцию в функцию, принимающую очень малые значения, поэтому при $0 \le x \le 1$ значение $|R_n''(x)|$ должно быть очень малым. Покажем также, что функция $R_n(x)$ сама по себе мала. Так как выполняется соотношение $R_n(0) = R_n(1) = 0$, согласно знакомой интерполяционной формуле (см. упр. 4.6.4–15 для случая, когда $x_0 = 0$, $x_1 = x$, $x_2 = 1$) для произвольной функции $\xi_n(x)$, где $0 \le \xi_n(x) \le 1$ при $0 \le x \le 1$, функция $R_n(x)$ имеет вид

$$R_n(x) = -\frac{x(1-x)}{2} R_n''(\xi_n(x)). \tag{36}$$

Таким образом, все сводится к попытке доказать, что U^n порождает функции с малыми значениями, где U — линейный оператор, определяемый уравнением (31). Заметим, что U — *положительный* оператор в том смысле, что если для всех x $\varphi(x) \ge 0$, то и $U\varphi(x) \ge 0$. Отсюда следует, что оператор U сохраняет порядок: если $\varphi_1(x) \le \varphi_2(x)$ для всех x, то $U\varphi_1(x) \le U\varphi_2(x)$ для всех x.

Один из способов использования этого свойства заключается в поиске функции φ, для которой можно вычислить точное значение $U\varphi$ и использовать константу, кратную этому значению функции, для определения верхней границы интересующей нас функции. Сначала рассмотрим такую функцию g, чтобы можно было легко вычислить значение Tg. Если рассмотреть функции, определенные для всех $x \ge 0$, а не только на отрезке $[0\,..\,1]$, то можно исключить в (27) суммирование, заметив, что для непрерывной функции G выполняется соотношение

$$SG(x+1) - SG(x) = G\Big(\frac{1}{1+x}\Big) - \lim_{k\to\infty} G\Big(\frac{1}{k+x}\Big) = G\Big(\frac{1}{1+x}\Big) - G(0). \tag{37}$$

Из того, что $T\big((1+x)G'\big) = (1+x)(SG)'$, следует (см. упр. 20)

$$\frac{Tg(x)}{1+x} - \frac{Tg(1+x)}{2+x} = \Big(\frac{1}{1+x} - \frac{1}{2+x}\Big) g\Big(\frac{1}{1+x}\Big). \tag{38}$$

Если положить, что $Tg(x) = 1/(1+x)$, то находим, что соответствующее значение функции $g(x)$ равно $1 + x - 1/(1+x)$. Положим $\varphi(x) = g'(x) = 1 + 1/(1+x)^2$, так что $U\varphi(x) = -(Tg)'(x) = 1/(1+x)^2$. Функция φ и является искомой функцией.

При таком выборе функции φ получаем $2 \le \varphi(x)/U\varphi(x) = (1+x)^2 + 1 \le 5$ при $0 \le x \le 1$. Следовательно,

$$\tfrac{1}{5}\varphi \le U\varphi \le \tfrac{1}{2}\varphi.$$

Так как функции U и φ положительны, можно снова применить оператор U к этому неравенству и получить $\frac{1}{25}\varphi \le \frac{1}{5}U\varphi \le U^2\varphi \le \frac{1}{2}U\varphi \le \frac{1}{4}\varphi$. После $(n-1)$-го воздействия оператора U на неравенство получаем для конкретного значения φ следующее выражение:

$$5^{-n}\varphi \le U^n\varphi \le 2^{-n}\varphi. \tag{39}$$

Пусть $\chi(x) = f_0'(x) = 1$ — постоянная функция. Тогда для $0 \le x \le 1$ имеем $\frac{5}{4}\chi \le \varphi \le 2\chi$, откуда

$$\tfrac{5}{8}5^{-n}\chi \le \tfrac{1}{2}5^{-n}\varphi \le \tfrac{1}{2}U^n\varphi \le U^n\chi \le \tfrac{4}{5}U^n\varphi \le \tfrac{4}{5}2^{-n}\varphi \le \tfrac{8}{5}2^{-n}\chi.$$

Из равенства (35) следует, что

$$\tfrac{5}{8}(\ln 2)^2 5^{-n} \le (-1)^n R_n''(x) \le \tfrac{16}{5}(\ln 2)^2 2^{-n} \qquad \text{для } 0 \le x \le 1.$$

Таким образом, из (32) и (36) следует, что доказана следующая теорема.

Теорема W. *При $n \to \infty$ распределение $F_n(x)$ равно $\lg(1+x) + O(2^{-n})$. В действительности $F_n(x) - \lg(1+x)$ лежит между $\frac{5}{16}(-1)^{n+1}5^{-n}\big(\ln(1+x)\big)\big(\ln 2/(1+x)\big)$ и $\frac{8}{5}(-1)^{n+1}2^{-n}\big(\ln(1+x)\big)\big(\ln 2/(1+x)\big)$ для $0 \le x \le 1$.* ▮

Если несколько изменить функцию φ, можно получить более сжатые границы (см. упр. 21). На самом деле Вирсинг пошел гораздо дальше, доказав в своей работе, что

$$F_n(x) = \lg(1+x) + (-\lambda)^n \Psi(x) + O\big(x(1-x)(\lambda - 0.031)^n\big), \tag{40}$$

где

$$\begin{aligned} \lambda &= 0.30366\,30028\,98732\,65859\,74481\,21901\,55623\,31109- \\ &= //3,3,2,2,3,13,1,174,1,1,1,2,2,2,1,1,1,2,2,1,\ldots//. \end{aligned} \tag{41}$$

Это фундаментальная постоянная (к счастью, никак не связанная с более известными постоянными), где Ψ — интересная функция, аналитическая в комплексной плоскости за исключением отрицательной вещественной оси от -1 до $-\infty$. Функция Вирсинга удовлетворяет соотношениям $\Psi(0) = \Psi(1) = 0$, $\Psi'(0) < 0$ и $S\Psi = -\lambda\Psi$. Таким образом, с учетом (37) эта функция удовлетворяет тождеству

$$\Psi(z) - \Psi(z+1) = \frac{1}{\lambda}\Psi\Big(\frac{1}{1+z}\Big). \tag{42}$$

Далее Вирсинг показал, что

$$\Psi\Big(-\frac{u}{v} + \frac{i}{N}\Big) = c\lambda^{-n}\log N + O(1) \qquad \text{при } N \to \infty, \tag{43}$$

где c — константа, а $n = T(u,v)$ — число итераций алгоритма Евклида при выполнении операций над целыми числами $u > v > 0$.

Полное решение проблемы Гаусса было найдено несколькими годами позже К. И. Бабенко [*ДАН СССР* **238** (1978), 1021–1024], который использовал мощный аппарат функционального анализа для доказательства следующей формулы:

$$F_n(x) = \lg(1+x) + \sum_{j\ge 2} \lambda_j^n\, \Psi_j(x) \tag{44}$$

для всех $0 \le x \le 1$, $n \ge 1$. В этой формуле $|\lambda_2| > |\lambda_3| \ge |\lambda_4| \ge \cdots$ каждая из функций $\Psi_j(z)$ — аналитическая функция на комплексной плоскости за исключением $[-\infty\,..\,-1]$. Функция Ψ_2 — это функция Вирсинга Ψ, а $\lambda_2 = -\lambda$, в то время как $\lambda_3 \approx 0.10088$, $\lambda_4 \approx -0.03550$, $\lambda_5 \approx 0.01284$, $\lambda_6 \approx -0.00472$, $\lambda_7 \approx 0.00175$. Бабенко установил также дополнительные свойства собственных значений λ_j, доказав, в частности, что они являются экспоненциально убывающими при $j \to \infty$ и что их сумма в (44) при $j \ge k$ ограничена величиной $(\pi^2/6)|\lambda_k|^{n-1}\min(x, 1-x)$. [Дополнительная информация содержится в работах Бабенко и Юрьева, опубликованных в *ДАН СССР* **240** (1978), 1273–1276, в работах Майера (Mayer) и Роепсторфа (Roepstorff), *J. Statistical Physics* **47** (1987), 149–171; **50** (1988), 331–344; Д. Хенсли (D. Hensley), *J. Number Theory* **49** (1994), 142–182; а также в работах Доде (Daudé), Флажоле (Flajolet) и Валле (Vallée), *Combinatorics, Probability and Computing* **6** (1997), 397–433; Flajolet, Vallée, *Theoretical Comp. Sci.* **194** (1998), 1–34.] Джон Хершбергер (John Hershberger) вычислил 40-значное значение λ в (41).

От непрерывного к дискретному. Выше были получены результаты, относящиеся к распределению вероятностей для цепных дробей в случае, когда X — вещественное число, равномерно распределенное в интервале $[0\,..\,1)$. Однако вероятность того, что вещественные числа являются рациональными, равна нулю (почти все числа являются иррациональными), так что эти результаты непосредственно к алгоритму Евклида неприменимы. Прежде чем применять теорему W для решения задачи, необходимо преодолеть некоторые технические затруднения. Рассмотрим следующий результат, основанный на элементарной теории меры.

Лемма M. *Пусть $I_1, I_2, \ldots, J_1, J_2, \ldots$ — попарно непересекающиеся подынтервалы, содержащиеся в интервале $[0\,..\,1)$, и пусть*

$$\mathcal{I} = \bigcup_{k\ge 1} I_k, \qquad \mathcal{J} = \bigcup_{k\ge 1} J_k, \qquad \mathcal{K} = [0\,..\,1] \setminus (\mathcal{I} \cup \mathcal{J}).$$

Предположим, что $\mathcal{K}$ имеет меру нуль. Пусть P_n означает множество $\{0/n, 1/n, \ldots, (n-1)/n\}$. Тогда

$$\lim_{n\to\infty} \frac{|\mathcal{I} \cap P_n|}{n} = \mu(\mathcal{I}). \tag{45}$$

Здесь $\mu(\mathcal{I})$ — мера Лебега множества $\mathcal{I}$, т. е. $\sum_{k\ge 1} \text{length}(I_k)$, а $|\mathcal{I} \cap P_n|$ обозначает число элементов множества $\mathcal{I} \cap P_n$.

Доказательство. Пусть $\mathcal{I}_N = \bigcup_{1\le k\le N} I_k$ и $\mathcal{J}_N = \bigcup_{1\le k\le N} J_k$. Для заданного $\epsilon > 0$ найдем N, достаточно большое, чтобы выполнялось соотношение $\mu(\mathcal{I}_N) + \mu(\mathcal{J}_N) \ge 1 - \epsilon$, и положим

$$\mathcal{K}_N = \mathcal{K} \cup \bigcup_{k>N} I_k \cup \bigcup_{k>N} J_k.$$

Ясно, что если I — любой из интервалов $(a\,..\,b)$, $[a\,..\,b)$, $(a\,..\,b]$ и $[a\,..\,b]$, то $\mu(I) = b - a$ и

$$n\mu(I) - 1 \le |I \cap P_n| \le n\mu(I) + 1.$$

Пусть теперь $r_n = |\mathcal{I}_N \cap P_n|$, $s_n = |\mathcal{J}_N \cap P_n|$, $t_n = |\mathcal{K}_N \cap P_n|$; тогда

$$r_n + s_n + t_n = n;$$
$$n\mu(\mathcal{I}_N) - N \le r_n \le n\mu(\mathcal{I}_N) + N;$$
$$n\mu(\mathcal{J}_N) - N \le s_n \le n\mu(\mathcal{J}_N) + N.$$

Значит,

$$\mu(\mathcal{I}) - \frac{N}{n} - \epsilon \le \mu(\mathcal{I}_N) - \frac{N}{n} \le \frac{r_n}{n} \le \frac{r_n + t_n}{n}$$
$$= 1 - \frac{s_n}{n} \le 1 - \mu(\mathcal{J}_N) + \frac{N}{n} \le \mu(\mathcal{I}) + \frac{N}{n} + \epsilon.$$

Это неравенство справедливо для всех n и для любого ϵ; следовательно,

$$\lim_{n\to\infty} r_n/n = \mu(\mathcal{I}). \quad \blacksquare$$

В упр. 25 показано, что лемма M нетривиальна в том отношении, что для справедливости формулы (45) необходимо принимать какие-то довольно ограничивающие предположения.

Распределение частичных отношений. Объединив теорему W и лемму M, докажем теперь несколько фундаментальных свойств, касающихся алгоритма Евклида.

Теорема E. *Пусть n и k — положительные целые числа; пусть также $p_k(a,n)$ — вероятность того, что $(k+1)$-е частное A_{k+1} в алгоритме Евклида равно a, когда $v=n$ и u выбирается случайно. Тогда*

$$\lim_{n\to\infty} p_k(a,n) = F_k\Big(\frac{1}{a}\Big) - F_k\Big(\frac{1}{a+1}\Big),$$

где $F_k(x)$ — функция распределения (22).

Доказательство. Множество $\mathcal{I}$ всех X в $[0\,..\,1)$, для которых $A_{k+1}=a$, есть объединение непересекающихся интервалов, как и множество $\mathcal{J}$ всех X, для которых $A_{k+1}\neq a$. Поэтому применима лемма M, причем множество $\mathcal{K}$ — множество всех X, для которых A_{k+1} не определено. Далее, $F_k(1/a) - F_k\big(1/(a+1)\big)$ есть вероятность того, что $1/(a+1) < X_k \le 1/a$, а это не что иное, как $\mu(\mathcal{I})$, т. е. вероятность того, что $A_{k+1}=a$. ▮

Из теорем E и W следует, что частное, равное a, встречается с вероятностью

$$\lg(1+1/a) - \lg\big(1+1/(a+1)\big) = \lg\big((a+1)^2/((a+1)^2-1)\big).$$

Так,

частное 1 встречается примерно в $\lg(\frac{4}{3})$ = 41.504% случаев;

частное 2 встречается примерно в $\lg(\frac{9}{8})$ = 16.993% случаев;

частное 3 встречается примерно в $\lg(\frac{16}{15})$ = 9.311% случаев;

частное 4 встречается примерно в $\lg(\frac{25}{24})$ = 5.889% случаев.

В действительности, если алгоритм Евклида формирует частные A_1, A_2, ..., A_t, приведенное доказательство гарантирует такое поведение только для тех A_k, k которых мал по сравнению с t. На значения A_{t-1}, A_{t-2}, ... это доказательство не распространяется. Однако можно показать экспериментально, что распределение последнего ряда A_{t-1}, A_{t-2}, ... частных, по существу, такое же, как и первого ряда.

Рассмотрим, например, разложение в правильную цепную дробь ряд правильных дробей со знаменателем, равным 29.

$\frac{1}{29} = //29//$	$\frac{8}{29} = //3,1,1,1,2//$	$\frac{15}{29} = //1,1,14//$	$\frac{22}{29} = //1,3,7//$
$\frac{2}{29} = //14,2//$	$\frac{9}{29} = //3,4,2//$	$\frac{16}{29} = //1,1,4,3//$	$\frac{23}{29} = //1,3,1,5//$
$\frac{3}{29} = //9,1,2//$	$\frac{10}{29} = //2,1,9//$	$\frac{17}{29} = //1,1,2,2,2//$	$\frac{24}{29} = //1,4,1,4//$
$\frac{4}{29} = //7,4//$	$\frac{11}{29} = //2,1,1,1,3//$	$\frac{18}{29} = //1,1,1,1,1,3//$	$\frac{25}{29} = //1,6,4//$
$\frac{5}{29} = //5,1,4//$	$\frac{12}{29} = //2,2,2,2//$	$\frac{19}{29} = //1,1,1,9//$	$\frac{26}{29} = //1,8,1,2//$
$\frac{6}{29} = //4,1,5//$	$\frac{13}{29} = //2,4,3//$	$\frac{20}{29} = //1,2,4,2//$	$\frac{27}{29} = //1,13,2//$
$\frac{7}{29} = //4,7//$	$\frac{14}{29} = //2,14//$	$\frac{21}{29} = //1,2,1,1,1,2//$	$\frac{28}{29} = //1,28//$

В этой таблице следует обратить внимание на несколько моментов.

а) Как указывалось ранее, последнее частное всегда равно 2 или более. Далее, имеем очевидное тождество

$$//x_1,\ldots,x_{n-1},\,x_n+1// = //x_1,\ldots,x_{n-1},x_n,1//, \tag{46}$$

показывающее, каким образом связаны цепные дроби, в которых последнее частное равно единице, с правильными цепными дробями.

b) Имеется простая связь значений, расположенных в правых столбцах, со значениями, расположенными в левых столбцах. Видит ли читатель эту связь? Она заключается вот в чем:

$$1 - //x_1, x_2, \ldots, x_n// = //1, x_1 - 1, x_2, \ldots, x_n//; \tag{47}$$

см. упр. 9.

c) Между левыми и правыми числами в первых двух столбцах наблюдается симметрия: если встречается $//A_1, A_2, \ldots, A_t//$, то встречается также $//A_t, \ldots, A_2, A_1//$. Так происходит всегда (см. упр. 26).

d) Если исследовать все частные в таблице, то обнаружится, что всего их имеется 96 и из них $\frac{39}{96} = 40.6\%$ равны 1, $\frac{21}{96} = 21.9\%$ равны 2, $\frac{8}{96} = 8.3\%$ равны 3. Эти данные хорошо согласуются с приведенными выше значениями вероятностей.

Число шагов деления. Вернемся теперь к исходной проблеме и исследуем величину T_n — среднее число шагов деления при $v = n$ (см. формулу (19)). Приведем отдельные значения T_n.

$n =$	95	96	97	98	99	100	101	102	103	104	105
$T_n =$	5.0	4.4	5.3	4.8	4.7	4.6	5.3	4.6	5.3	4.7	4.6

$n =$	996	997	998	999	1000	1001	$\cdots$	9999	10000	10001
$T_n =$	6.5	7.3	7.0	6.8	6.4	6.7	$\cdots$	8.6	8.3	9.1

$n =$	49998	49999	50000	50001	$\cdots$	99999	100000	100001
$T_n =$	9.8	10.6	9.7	10.0	$\cdots$	10.7	10.3	11.0

Обратите внимание на некоторую неустойчивость поведения. Если n простое, то соответствующее ему значение T_n больше соседних значений. И это значение соответственно меньше соседних, если n содержит много делителей. (В приведенном списке числа 97, 101, 103, 997 и 49999 — простые числа; $10001 = 73 \cdot 137$; $49998 = 2 \cdot 3 \cdot 13 \cdot 641$; $50001 = 3 \cdot 7 \cdot 2381$; $99999 = 3 \cdot 3 \cdot 41 \cdot 271$ и $100001 = 11 \cdot 9091$.) Нетрудно понять причину такого поведения. Если $\gcd(u, v) = d$, то алгоритм Евклида выполняет операции с числами u и v так же, как и с числами u/d и v/d. Поэтому, когда число $v = n$ имеет несколько делителей, существует много способов выбора такого u, для которого n ведет себя так, как будто его значение меньше значения, которое есть на самом деле.

Соответственно рассмотрим *другую* величину, τ_n, которая является средним числом шагов деления, для случая, когда $v = n$ и u есть число, *взаимно простое* с n. Тогда

$$\tau_n = \frac{1}{\varphi(n)} \sum_{\substack{0 \le m < n \\ m \perp n}} T(m, n). \tag{48}$$

Отсюда следует, что

$$T_n = \frac{1}{n} \sum_{d \backslash n} \varphi(d) \tau_d. \tag{49}$$

Ниже приводится таблица значений τ_n для тех же значений n, которые были рассмотрены выше.

$n =$	95	96	97	98	99	100	101	102	103	104	105
$\tau_n =$	5.4	5.3	5.3	5.6	5.2	5.2	5.4	5.3	5.4	5.3	5.6

$n =$	996	997	998	999	1000	1001	$\cdots$	9999	10000	10001
$\tau_n =$	7.2	7.3	7.3	7.3	7.3	7.4	$\cdots$	9.21	9.21	9.22

$n =$	49998	49999	50000	50001	$\cdots$	99999	100000	100001
$\tau_n =$	10.59	10.58	10.57	10.59	$\cdots$	11.170	11.172	11.172

Очевидно, что τ_n ведет себя значительно лучше, чем T_n, и оно должно значительно легче поддаваться анализу. При внимательном изучении таблицы значений τ_n при малых n обнаруживается ряд серьезных отклонений, например $\tau_{50} = \tau_{100}$ и $\tau_{60} = \tau_{120}$. Но по мере роста числа n значения τ_n становятся вполне нормальными, как это видно в приведенной выше таблице, и не указывают ни на какую особую зависимость от свойств разложимости на простые множители числа n. Если построить график значений τ_n как функций $\ln n$ по приведенным выше данным, то окажется, что эти значения расположены очень близко к прямой линии:

$$\tau_n \approx 0.843 \ln n + 1.47. \tag{50}$$

Это свойство будет учтено чуть позже, при исследовании процесса формирования правильной цепной дроби. Учитывая выражение (15), для алгоритма Евклида получаем

$$\frac{V_0}{U_0}\frac{V_1}{U_1}\cdots\frac{V_{t-1}}{U_{t-1}} = \frac{V_{t-1}}{U_0},$$

поскольку $U_{k+1} = V_k$. Далее, если $U = U_0$ и $V = V_0$ взаимно просты и если имеется t шагов деления, то

$$X_0 X_1 \ldots X_{t-1} = 1/U.$$

Полагая $U = N$ и $V = m < N$, найдем, что

$$\ln X_0 + \ln X_1 + \cdots + \ln X_{t-1} = -\ln N. \tag{51}$$

Так как известно распределение величин X_0, X_1, X_2, ..., это уравнение можно использовать для оценки величины

$$t = T(N; m) = T(m, N) - 1.$$

Возвращаясь к формулам, приведенным перед теоремой W, находим, что среднее значение величины $\ln X_n$ в случае, когда X_0 — вещественное число, равномерно распределенное в интервале $[0\,..\,1)$, равно

$$\int_0^1 \ln x\, F_n'(x)\, dx = \int_0^1 \ln x\, f_n(x)\, dx/(1+x), \tag{52}$$

где $f_n(x)$ определено в уравнении (33). Отсюда, рассуждая так же, как и ранее (см. упр. 23), получаем

$$f_n(x) = \frac{1}{\ln 2} + O(2^{-n}). \tag{53}$$

Следовательно, среднее значение величины $\ln X_n$ очень хорошо аппроксимируется величиной

$$\begin{aligned}
\frac{1}{\ln 2}\int_0^1 \frac{\ln x}{1+x}\,dx &= -\frac{1}{\ln 2}\int_0^\infty \frac{ue^{-u}}{1+e^{-u}}\,du \\
&= -\frac{1}{\ln 2}\sum_{k\ge 1}(-1)^{k+1}\int_0^\infty ue^{-ku}\,du \\
&= -\frac{1}{\ln 2}\Big(1-\frac{1}{4}+\frac{1}{9}-\frac{1}{16}+\frac{1}{25}-\cdots\Big) \\
&= -\frac{1}{\ln 2}\Big(1+\frac{1}{4}+\frac{1}{9}+\cdots-2\Big(\frac{1}{4}+\frac{1}{16}+\frac{1}{36}+\cdots\Big)\Big) \\
&= -\frac{1}{2\ln 2}\Big(1+\frac{1}{4}+\frac{1}{9}+\cdots\Big) \\
&= -\pi^2/(12\ln 2).
\end{aligned}$$

Поэтому в силу (51) следует ожидать, что приближенная формула будет иметь вид

$$-t\pi^2/(12\ln 2) \approx -\ln N,$$

т. е. t должно приближенно равняться величине $\big((12\ln 2)/\pi^2\big)\ln N$. Эта константа $(12\ln 2)/\pi^2 = 0.842765913\ldots$ полностью согласуется с эмпирической формулой (50), полученной ранее, так что есть веские основания полагать, что формула

$$\tau_n \approx \frac{12\ln 2}{\pi^2}\ln n + 1.47 \tag{54}$$

описывает истинное поведение τ_n при $n\to\infty$.

Если предположить, что выполняется приближенное равенство (54), получим следующую формулу:

$$T_n \approx \frac{12\ln 2}{\pi^2}\left(\ln n - \sum_{d\backslash n}\frac{\Lambda(d)}{d}\right) + 1.47, \tag{55}$$

где $\Lambda(d)$ — функция *фон Мангольдта*, определяемая правилами

$$\Lambda(n) = \begin{cases} \ln p, & \text{если } n = p^r, \text{ где } p \text{ — простое число и } r \ge 1; \\ 0 & \text{в противном случае.} \end{cases} \tag{56}$$

(См. упр. 27.) Например,

$$\begin{aligned}
T_{100} &\approx \frac{12\ln 2}{\pi^2}\Big(\ln 100 - \frac{\ln 2}{2} - \frac{\ln 2}{4} - \frac{\ln 5}{5} - \frac{\ln 5}{25}\Big) + 1.47 \\
&\approx (0.843)(4.605 - 0.347 - 0.173 - 0.322 - 0.064) + 1.47 \\
&\approx 4.59;
\end{aligned}$$

точное значение T_{100} равно 4.56.

Можно также оценить среднее количество шагов деления в случае, когда оба числа u и v равномерно распределены в интервале между 1 и N, вычислив

$$\frac{1}{N}\sum_{n=1}^{N} T_n. \tag{57}$$

В предположении справедливости формулы (55) в упр. 29 показано, что эта сумма имеет вид

$$\frac{12\ln 2}{\pi^2}\ln N + O(1), \tag{58}$$

а эмпирические расчеты, выполненные с теми же числами, которые использовались при выводе соотношения 4.5.2–(65), хорошо согласуются с формулой

$$\frac{12\ln 2}{\pi^2}\ln N + 0.06. \tag{59}$$

Конечно, пока мы ничего не *доказали* о поведении T_n и τ_n в общем случае. До сих пор рассматривались только возможные обстоятельства, при которых должны быть верны определенные формулы. К счастью, сейчас уже можно применить методику строгого доказательства, которая основана на тщательном анализе, выполненном рядом математиков.

Впервые главный член $(12\ln 2)/\pi^2$ в формулах, приведенных выше, был получен независимо Джоном Д. Диксоном (John D. Dixon) и Гансом А. Хайльбронном (Hans A. Heilbronn). Диксон [*J. Number Theory* **2** (1970), 414–422] развил теорию распределений $F_n(x)$, чтобы показать, что индивидуальные частичные отношения являются в определенном смысле независимыми одно от другого, и доказал, что для любого положительного ϵ выполняется соотношение $\big|T(m,n) - \big((12\ln 2)/\pi^2\big)\ln n\big| < (\ln n)^{(1/2)+\epsilon}$, но не для значений m и n $\exp\big(-c(\epsilon)(\log N)^{\epsilon/2}\big)N^2$ в интервале $1 \le m < n \le N$, где $c(\epsilon) > 0$. Совсем иной подход к этой проблеме, при котором вместо непрерывных переменных рассматриваются только целые числа, предложил Хайльбронн. В основу его идеи, излагаемой в несколько модифицированном виде в упр. 33 и 34, положен тот факт, что величину τ_n можно определенным образом связать с числом способов представления n. Кроме того, в его работе *Number Theory and Analysis*, edited by Paul Turán (New York: Plenum, 1969), 87–96, показано, что распределение индивидуальных частичных отношений 1, 2, ..., которое рассматривалось выше, в действительности применимо ко всему множеству частичных отношений, принадлежащих дробям с заданным знаменателем. Это более точная форма теоремы E. Через несколько лет Дж. В. Портер (J. W. Porter) [*Mathematika* **22** (1975), 20–28] получил еще более точный результат. Он установил, что

$$\tau_n = \frac{12\ln 2}{\pi^2}\ln n + C + O(n^{-1/6+\epsilon}), \tag{60}$$

где $C \approx 1.46707\,80794$ есть постоянная

$$\frac{6\ln 2}{\pi^2}\big(3\ln 2 + 4\gamma - 24\pi^{-2}\zeta'(2) - 2\big) - \frac{1}{2}; \tag{61}$$

см. D. E. Knuth, *Computers and Math. with Applic.* **2** (1976), 137–139. Таким образом, утверждение (50) полностью доказано. Используя формулу (60), Грэхэм Х. Нортон

(Graham H. Norton) [*J. Symbolic Computation* **10** (1990), 53–58] продолжил вычисления упр. 29, чтобы доказать, что эмпирическая константа 0.06 в формуле (59) в действительности равна

$$\frac{6\ln 2}{\pi^2}\bigl(3\ln 2 + 4\gamma - 12\pi^{-2}\zeta'(2) - 3\bigr) - 1 \approx 0.06535\,14259\ldots. \tag{62}$$

Г. Э. Коллинз (G. E. Collins), применив классические алгоритмы выполнения арифметических операций, показал в *SICOMP* **3** (1974), 1–10, что среднее время выполнения алгоритма Евклида при оперировании числами многократной точности равно

$$\bigl(1 + \log(\max(u,v)/\gcd(u,v))\bigr)\log\min(u,v). \tag{63}$$

Резюме. Мы обнаружили, что наихудший случай алгоритма Евклида имеет место, когда входные числа u и v связаны с числами Фибоначчи (теорема F), а число шагов деления при $0 \le v < N$ никогда не превышает величины $\lceil 4.8\log_{10} N - 0.32\rceil$. Мы определили частоту появления различных частичных отношений, показав, к примеру, что приблизительно в 41% случаев на шаге деления получается $\lfloor u/v\rfloor = 1$ (теорема E). Наконец, в теоремах Хайльбронна и Портера доказывается, что среднее число T_n шагов деления при $v = n$ приблизительно равно

$$\bigl((12\ln 2)/\pi^2\bigr)\ln n \approx 1.9405\log_{10} n,$$

если не учитывать корректирующий член, основанный на делителях числа n, как следует из уравнения (55).

УПРАЖНЕНИЯ

▸ **1.** [*20*] Так как частное $\lfloor u/v\rfloor$ равно единице более чем в 40% времени выполнения алгоритма 4.5.2A, на некоторых компьютерах может оказаться выгодным проверить этот случай и запретить выполнение операции деления, когда частное равно единице. Является ли следующая MIX-программа, реализующая алгоритм Евклида, более эффективной, чем программа 4.5.2A?

```
    LDX  U    rX ← u.                 SRAX 5    rAX ← rA.
    JMP  2F                           JL   2F   u − v < v?
1H  STX  V    v ← rX.                 DIV  V    rX ← rAX mod v.
    SUB  V    rA ← u − v.         2H  LDA  V    rA ← v.
    CMPA V                            JXNZ 1B   Выполнено, если rX = 0.
```

∎

2. [*M21*] Вычислите произведение матриц

$$\begin{pmatrix} x_1 & 1\\ 1 & 0\end{pmatrix}\begin{pmatrix} x_2 & 1\\ 1 & 0\end{pmatrix}\cdots\begin{pmatrix} x_n & 1\\ 1 & 0\end{pmatrix}.$$

3. [*M21*] Чему равен определитель

$$\det\begin{pmatrix} x_1 & 1 & 0 & \ldots & 0\\ -1 & x_2 & 1 & & 0\\ 0 & -1 & x_3 & 1 & \vdots\\ \vdots & & -1 & \ddots & 1\\ 0 & 0 & \ldots & -1 & x_n\end{pmatrix}?$$

4. [*M20*] Докажите тождество (8).

5. [*HM25*] Пусть x_1, x_2, ... — последовательность вещественных чисел и каждое из них больше некоторого положительного числа ϵ. Докажите, что существует бесконечная цепная дробь $//x_1, x_2, \ldots // = \lim_{n\to\infty} //x_1, \ldots, x_n//$. Покажите также, что $//x_1, x_2, \ldots //$ необязательно существует, если предположить только, что $x_j > 0$ для всех j.

6. [*M23*] Докажите, что разложение числа в правильную цепную дробь *единственно* в следующем смысле. Если B_1, B_2, ... — положительные целые числа, то бесконечная цепная дробь $//B_1, B_2, \ldots //$ есть иррациональное число X, расположенное между 0 и 1, разложение которого в правильную цепную дробь имеет элементы $A_n = B_n$ для всех $n \geq 1$. Если же B_1, ..., B_m — положительные целые числа, причем $B_m > 1$, то правильная цепная дробь для числа $X = //B_1, \ldots, B_m//$ имеет элементы $A_n = B_n$ для $1 \leq n \leq m$.

7. [*M26*] Найдите все перестановки $p(1)p(2)\ldots p(n)$ целых чисел $\{1, 2, \ldots, n\}$, таких, что $K_n(x_1, x_2, \ldots, x_n) = K_n(x_{p(1)}, x_{p(2)}, \ldots, x_{p(n)})$ является тождеством для всех x_1, x_2, ..., x_n.

8. [*M20*] Покажите, что если при разложении в правильную цепную дробь числа X определено X_n, то $-1/X_n = //A_n, \ldots, A_1, -X//$.

9. [*M21*] Покажите, что цепные дроби удовлетворяют следующим тождествам:

a) $//x_1, \ldots, x_n// = //x_1, \ldots, x_k + //x_{k+1}, \ldots, x_n////, \qquad 1 \leq k \leq n;$

b) $//0, x_1, x_2, \ldots, x_n// = x_1 + //x_2, \ldots, x_n//, \qquad n \geq 1;$

c) $//x_1, \ldots; x_{k-1}, x_k, 0, x_{k+1}, x_{k+2}, \ldots, x_n// = //x_1, \ldots, x_{k-1}, x_k + x_{k+1}, x_{k+2}, \ldots, x_n//, \; 1 \leq K < n;$

d) $1 - //x_1, x_2, \ldots, x_n// = //1, x_1 - 1, x_2, \ldots, x_n//, \qquad n \geq 1.$

10. [*M28*] Из результата упр. 6 следует, что любое иррациональное число X единственным образом разложимо в правильную цепную дробь вида

$$X = A_0 + //A_1, A_2, A_3, \ldots //,$$

где A_0 — целое число и A_1, A_2, A_3, ... — положительные целые числа. Покажите, что если X имеет такое представление, то разложение числа $1/X$ в правильную цепную дробь имеет вид

$$1/X = B_0 + //B_1, \ldots, B_m, A_5, A_6, \ldots //$$

для соответствующих целых чисел B_0, B_1, ..., B_m. (Наиболее интересен, безусловно, случай, когда $A_0 < 0$.) Объясните, как можно выразить все B через A_0, A_1, A_2, A_3 и A_4.

11. [*M30*] (Ж.-А. Серре (J.-A. Serret), 1850.) Пусть $X = A_0 + //A_1, A_2, A_3, A_4, \ldots //$ и $Y = B_0 + //B_1, B_2, B_3, B_4, \ldots //$ — представления в виде правильных цепных дробей двух вещественных чисел X и Y в смысле упр. 10. Покажите, что эти представления "эвентуально согласованы" в том смысле, что $A_{m+k} = B_{n+k}$ для некоторых m и n и для всех $k \geq 0$ тогда и только тогда, когда для некоторых целых чисел q, r, s, t выполняется $X = (qY + r)/(sY + t)$ при $|qt - rs| = 1$. (Эта теорема является аналогом представлений цепными дробями простого результата, заключающегося в том, что представления целых чисел X и Y в десятичной системе счисления в конечном счете совпадают тогда и только тогда, когда для некоторых целых чисел q, r и s выполняется равенство $X = (10^q Y + r)/10^s$.)

▶ **12.** [*M30*] *Квадратичной иррациональностью* называют число вида $(\sqrt{D} - U)/V$, где D, U и V — целые числа, удовлетворяющие условиям $D > 0$, $V \neq 0$, и D не есть полный квадрат. Без потери общности можно предположить, что V является делителем числа $D - U^2$, в противном случае это число можно переписать в виде $(\sqrt{DV^2} - U|V|)/(V|V|)$.

a) Докажите, что разложение в правильную цепную дробь (в смысле упр. 10) квадратичной иррациональности $X = (\sqrt{D} - U)/V$ получается по следующим формулам:

$$V_0 = V, \qquad A_0 = \lfloor X \rfloor, \qquad U_0 = U + A_0 V;$$
$$V_{n+1} = (D - U_n^2)/V_n, \qquad A_{n+1} = \lfloor (\sqrt{D} + U_n)/V_{n+1} \rfloor, \qquad U_{n+1} = A_{n+1} V_{n+1} - U_n.$$

b) Докажите, что для всех $n > N$, где N — некоторое целое число, зависящее от X, выполняются неравенства $0 < U_n < \sqrt{D}$, $0 < V_n < 2\sqrt{D}$. Следовательно, представление квадратичной иррациональности правильной цепной дробью в конечном счете периодично. [*Указание.* Покажите, что

$$(-\sqrt{D} - U)/V = A_0 + //A_1, \ldots, A_n, -V_n/(\sqrt{D} + U_n)//,$$

и, используя равенство (5), докажите, что при больших значениях n величина $(\sqrt{D} + U_n)/V_n$ положительна.]

c) Положим $p_n = K_{n+1}(A_0, A_1, \ldots, A_n)$ и $q_n = K_n(A_1, \ldots, A_n)$. Докажите тождество $Vp_n^2 + 2Up_nq_n + ((U^2 - D)/V)q_n^2 = (-1)^{n+1}V_{n+1}$.

d) Докажите, что представление иррационального числа X правильной цепной дробью в конечном счете периодично тогда и только тогда, когда число X есть квадратичная иррациональность. (Это утверждение относительно цепной дроби является аналогом утверждения о том, что разложение вещественного числа X в десятичную дробь в конечном счете периодично тогда и только тогда, когда X рационально.)

13. [*M40*] (Ж. Лагранж, 1767.) Пусть $f(x) = a_nx^n + \cdots + a_0$, $a_n > 0$ — полином с целочисленными коэффициентами, у которого нет рациональных корней и имеется точно один вещественный корень $\xi > 1$. Разработайте компьютерную программу для нахождения примерно первой тысячи частичных отношений числа ξ с помощью следующего алгоритма (который включает в себя только сложение).

L1. Присвоить $A \leftarrow 1$.

L2. Для $k = 0, 1, \ldots, n-1$ (в таком порядке) и $j = n-1, \ldots, k$ (в таком порядке) присвоить $a_j \leftarrow a_{j+1} + a_j$. (На этом шаге функция $f(x)$ заменяется функцией $g(x) = f(x+1)$, полиномом, корни которого на единицу меньше корней полинома f.)

L3. Если $a_n + a_{n-1} + \cdots + a_0 < 0$, то присвоить $A \leftarrow A + 1$ и возвратиться к шагу L2.

L4. Вывести A (которое является значением следующего частичного отношения). Заменить коэффициенты $(a_n, a_{n-1}, \ldots, a_0)$ на $(-a_0, -a_1, \ldots, -a_n)$ и возвратиться к шагу L1. (На этом шаге выполняется замена $f(x)$ полиномом, корни которого обратны корням полинома f.)

Например, начав с $f(x) = x^3 - 2$, алгоритм выведет сначала "1" (заменив $f(x)$ на $x^3 - 3x^2 - 3x - 1$), затем — "3" (заменив $f(x)$ на $10x^3 - 6x^2 - 6x - 1$) и т. д.

14. [*M22*] (А. Гурвиц (A. Hurwitz), 1891.) Покажите, что при помощи следующих правил можно найти разложение в цепную дробь числа $2X$, если известны частичные отношения числа X:

$$2// 2a, b, c, \ldots // = // a, 2b + 2//c, \ldots ////;$$
$$2// 2a + 1, b, c, \ldots // = // a, 1, 1 + 2//b - 1, c, \ldots ////.$$

Используйте этот метод для нахождения разложения в цепную дробь числа $\frac{1}{2}e$, если известно разложение в цепную дробь числа e (это разложение дается формулой (13)).

▸ **15.** [*M31*] (Р. У. Госпер (R. W. Gosper).) Обобщая упр. 14, разработайте алгоритм, который вычисляет цепную дробь $X_0 + //X_1, X_2, \ldots //$ для $(ax + b)/(cx + d)$ по заданному разложению числа x в цепную дробь $x_0 + //x_1, x_2, \ldots //$ и заданным целым числам a, b, c, d, таким, что $ad \neq bc$. Сделайте свой алгоритм таким, чтобы он выполнялся как "оперативная сопрограмма", которая перед вводом каждого из x_j выводит как можно больше значений X_k. Продемонстрируйте, как ваш алгоритм вычисляет $(97x + 39)/(-62x - 25)$, когда $x = -1 + //5, 1, 1, 1, 2, 1, 2//$.

16. [*HM30*] (Л. Эйлер, 1731.) Пусть $f_0(z) = (e^z - e^{-z})/(e^z + e^{-z}) = \tanh z$ и $f_{n+1}(z) = 1/f_n(z) - (2n+1)/z$. Докажите, что для всех n функция $f_n(z)$ есть аналитическая функция комплексной переменной z в окрестности начала координат, которая удовлетворяет дифференциальному уравнению $f_n'(z) = 1 - f_n(z)^2 - 2nf_n(z)/z$. Используя этот факт, докажите, что

$$\tanh z = /\!/z^{-1}, 3z^{-1}, 5z^{-1}, 7z^{-1}, \ldots /\!/.$$

Затем докажите, используя правило Гурвица, что

$$e^{-1/n} = /\!/ \overline{1, (2m+1)n - 1, 1} /\!/, \qquad m \geq 0.$$

(Это общепринятое обозначение бесконечной цепной дроби $/\!/\, 1, n-1, 1, 1, 3n-1, 1, 1, 5n-1, 1, \ldots /\!/$.) Найдите также разложение в цепную дробь числа $e^{-2/n}$, где $n > 0$ нечетно.

▶ **17.** [*M23*] (a) Докажите, что $/\!/x_1, -x_2/\!/ = /\!/x_1 - 1, 1, x_2 - 1/\!/$. (b) Обобщите это тождество, получив формулу для $/\!/x_1, -x_2, x_3, -x_4, x_5, -x_6, \ldots, x_{2n-1}, -x_{2n}/\!/$, в которой все частичные отношения являются положительными целыми числами при условии, что все x — большие положительные целые числа. (c) Из результата упр. 16 следует, что $\tan 1 = /\!/1, -3, 5, -7, \ldots /\!/$. Найдите разложение $\tan 1$ в правильную цепную дробь.

18. [*M25*] Покажите, что $/\!/a_1, a_2, \ldots, a_m, x_1, a_1, a_2, \ldots, a_m, x_2, a_1, a_2, \ldots, a_m, x_3, \ldots /\!/ - /\!/a_m, \ldots, a_2, a_1, x_1, a_m, \ldots, a_2, a_1, x_2, a_m, \ldots, a_2, a_1, x_3, \ldots /\!/$ не зависит от $x_1, x_2, x_3, \ldots$. *Указание.* Умножьте обе цепные дроби на $K_m(a_1, a_2, \ldots, a_m)$.

19. [*M20*] Докажите, что $F(x) = \log_b(1+x)$ удовлетворяет уравнению (24).

20. [*HM20*] Выведите (38) из (37).

21. [*HM29*] (Э. Вирсинг (E. Wirsing).) Ограничения (39) были получены для функции φ, соответствующей функции g, с помощью оператора $Tg(x) = 1/(x+1)$. Покажите, что функция, соответствующая $Tg(x) = 1/(x+c)$, при подходящей константе $c > 0$ обеспечивает лучшие оценки.

22. [*HM46*] (К. И. Бабенко.) Разработайте эффективный способ вычисления точных приближений для величин λ_j и $\Psi_j(x)$ в (44) при малых $j \geq 3$ и $0 \leq x \leq 1$.

23. [*HM23*] Докажите (53), используя результаты, полученные при доказательстве теоремы W.

24. [*M22*] Каково среднее значение частичного отношения A_n в разложении в цепную дробь случайного вещественного числа?

25. [*HM25*] Найдите пример множества $\mathcal{I} = I_1 \cup I_2 \cup I_3 \cup \cdots \subseteq [0\,..\,1]$, где все I — непересекающиеся интервалы, для которых (45) не выполняется.

26. [*M23*] Покажите, что если числа $\{1/n, 2/n, \ldots, \lfloor n/2 \rfloor / n\}$ выражены в виде правильных цепных дробей, то полученные результаты обнаруживают лево-правую симметрию в том смысле, что всякий раз одновременно с $/\!/A_t, \ldots, A_2, A_1/\!/$ появляется $/\!/A_1, A_2, \ldots, A_t/\!/$.

27. [*M21*] Выведите (55) из (49) и (54).

28. [*M23*] Докажите следующие тождества, в которые входят три теоретико-числовые функции $\varphi(n)$, $\mu(n)$, $\Lambda(n)$.

a) $\displaystyle\sum_{d \backslash n} \mu(d) = \delta_{n1}$. b) $\displaystyle\ln n = \sum_{d \backslash n} \Lambda(d), \qquad n = \sum_{d \backslash n} \varphi(d)$.

c) $\displaystyle\Lambda(n) = \sum_{d \backslash n} \mu\left(\frac{n}{d}\right) \ln d, \qquad \varphi(n) = \sum_{d \backslash n} \mu\left(\frac{n}{d}\right) d$.

29. [*M23*] Полагая, что T_n задается формулой (55), покажите, что (57) равно (58).

▸ **30.** [*HM32*] Часто предлагается следующий вариант алгоритма Евклида: при выполнении шага деления вместо замены v величиной $u \bmod v$ заменить его величиной $|(u \bmod v) - v|$, если $u \bmod v > \frac{1}{2}v$. Так, например, если $u = 26$ и $v = 7$, то $\gcd(26, 7) = \gcd(-2, 7) = \gcd(7, 2)$; когда кратные 7 вычитаются из 26, *наименьшим по абсолютной величине остатком* будет -2. Сравните эту процедуру с алгоритмом Евклида; оцените число шагов деления, сэкономленных в результате применения этого метода.

▸ **31.** [*M35*] Найдите наихудший случай для модифицированного алгоритма Евклида, предложенного в упр. 30. Каковы наименьшие исходные числа $u > v > 0$, для обработки которых необходимо затратить n шагов деления?

32. [*20*] (a) Словом длиной n в азбуке Морзе называется цепочка из r точек и s тире, для которой $r + 2s = n$. Например, словами длиной 4 в азбуке Морзе являются

$$\cdot\cdot\cdot\cdot, \quad \cdot\cdot-, \quad \cdot-\cdot, \quad -\cdot\cdot, \quad --.$$

Учитывая, что $K_4(x_1, x_2, x_3, x_4)$ — полином, равный $x_1x_2x_3x_4 + x_1x_2 + x_1x_4 + x_3x_4 + 1$, найдите и докажите простую связь между полиномом $K_n(x_1, \ldots, x_n)$ и словами длиной n в азбуке Морзе. (b) (Л. Эйлер, *Novi Comm. Acad. Sci. Pet.* **9** (1762), 53–69.) Докажите, что

$$K_{m+n}(x_1, \ldots, x_{m+n}) = K_m(x_1, \ldots, x_m)K_n(x_{m+1}, \ldots, x_{m+n}) + K_{m-1}(x_1, \ldots, x_{m-1})K_{n-1}(x_{m+2}, \ldots, x_{m+n}).$$

33. [*M32*] Пусть $h(n)$ — количество различных представлений числа n в виде

$$n = xx' + yy', \qquad x > y > 0, \qquad x' > y' > 0, \qquad x \perp y, \qquad \text{где } x, x', y, y' \text{ целые.}$$

a) Покажите, что если ослабить эти условия, допустив выполнение равенства $x' = y'$, то количество возможных представлений числа n будет равняться $h(n) + \lfloor (n-1)/2 \rfloor$.

b) Покажите, что для фиксированных $y > 0$ и $0 < t \le y$, где $t \perp y$, и для любых фиксированных x', принадлежащих интервалу $0 < x' < n/(y+t)$ и таких, что $x't \equiv n$ (по модулю y), существует точно одно представление числа n, удовлетворяющее ограничениям в (a) и условию $x \equiv t$ (по модулю y).

c) Покажите, что $h(n) = \sum \lceil (n/(y+t) - t')/y \rceil - \lfloor (n-1)/2 \rfloor$, где сумма берется по всем положительным целым числам y, t, t', таким, что $t \perp y$, $t \le y$, $t' \le y$, $tt' \equiv n$ (по модулю y).

d) Покажите, что каждое из таких представлений $h(n)$ можно выразить единственным способом в виде

$$\begin{aligned} x &= K_m(x_1, \ldots, x_m), & y &= K_{m-1}(x_1, \ldots, x_{m-1}), \\ x' &= K_k(x_{m+1}, \ldots, x_{m+k})\,d, & y' &= K_{k-1}(x_{m+2}, \ldots, x_{m+k})\,d, \end{aligned}$$

где m, k, d и все x_j — положительные целые числа, для которых $x_1 \ge 2$, $x_{m+k} \ge 2$, а d — делитель числа n. Теперь из тождества, приведенного в упр. 32, следует, что $n/d = K_{m+k}(x_1, \ldots, x_{m+k})$. Обратно, любой заданной последовательности целых чисел $x_1, \ldots, x_{m+k}$, такой, что $x_1 \ge 2$, $x_{m+k} \ge 2$, и для которой $K_{m+k}(x_1, \ldots, x_{m+k})$ делит n, соответствует, таким образом, $m + k - 1$ представлений числа n.

e) Покажите, что $nT_n = \lfloor (5n-3)/2 \rfloor + 2h(n)$.

34. [*HM40*] (Г. Хайльбронн (H. Heilbronn).) Пусть $h_d(n)$ — количество представлений числа n, описанных в упр. 33, для которых $xd < x'$, плюс половина количества представлений, для которых $xd = x'$.

a) Пусть $g(n)$ — количество представлений, на которые не накладывается ограничение $x \perp y$. Докажите, что

$$h(n) = \sum_{d \backslash n} \mu(d) g\Big(\frac{n}{d}\Big), \qquad g(n) = 2 \sum_{d \backslash n} h_d\Big(\frac{n}{d}\Big).$$

b) Обобщите результат упр. 33, (b), показав, что для $d \geq 1$ выполняется равенство $h_d(n) = \sum(n/(y(y+t))) + O(n)$, где сумма берется по всем целым числам y и t, для которых $t \perp y$ и $0 < t \leq y < \sqrt{n/d}$.

c) Покажите, что $\sum_{y=1}^{n}(y/(y+t)) = \varphi(y)\ln 2 + O(\sigma_{-1}(y))$, где сумма берется по t из интервала $0 < t \leq y$, причем $t \perp y$ и $\sigma_{-1}(y) = \sum d\backslash y(1/d)$.

d) Покажите, что $\sum_{y=1}^{n}\varphi(y)/y^2 = \sum_{d=1}^{n}\mu(d)H_{\lfloor n/d\rfloor}/d^2$.

e) Отсюда получаем асимптотическую формулу

$$T_n = ((12\ln 2)/\pi^2)(\ln n - \sum_{d\backslash n}\Lambda(d)/d) + O(\sigma_{-1}(n)^2).$$

35. [*HM41*] (Э. Ч. Яо (A. C. Yao) и Д. Э. Кнут (D. E. Knuth).) Докажите, что при $1 \leq m < n$ сумма по всем частичным частным для дробей m/n равна $2(\sum\lfloor x/y\rfloor + \lfloor n/2\rfloor)$, где суммирование берется по всем представлениям $n = xx' + yy'$, удовлетворяющим условиям упр. 33, (a). Покажите, что $\sum\lfloor x/y\rfloor = 3\pi^{-2}n(\ln n)^2 + O(n\log n\,(\log\log n)^2)$, и примените полученный результат к "античному" виду алгоритма Евклида, в котором вместо операции деления используется только операция вычитания.

36. [*M25*] (Г. Х. Брэдли (G. H. Bradley).) Каково наименьшее значение u_n, при котором для вычисления $\gcd(u_1, \ldots, u_n)$ по алгоритму 4.5.2C требуется N делений, если в процессе вычислений постоянно используется алгоритм Евклида? Предполагается, что $N \geq n \geq 3$.

37. [*M38*] (Т. С. Моцкин (T. S. Motzkin) и Е. Г. Штраус (E. G. Straus).) Пусть $a_1, \ldots, a_n$ — положительные целые числа. Покажите, что найдется $\max K_n(a_{p(1)}, \ldots, a_{p(n)})$ по всем перестановкам $p(1)\ldots p(n)$ для всех $\{1, 2, \ldots, n\}$, когда $a_{p(1)} \geq a_{p(n)} \geq a_{p(2)} \geq a_{p(n-1)} \geq \cdots$, и минимум будет при $a_{p(1)} \leq a_{p(n)} \leq a_{p(3)} \leq a_{p(n-2)} \leq a_{p(5)} \leq \cdots \leq a_{p(6)} \leq a_{p(n-3)} \leq a_{p(4)} \leq a_{p(n-1)} \leq a_{p(2)}$.

38. [*M25*] (Я. Микусинский (J. Mikusiński).) Пусть $L(n) = \max_{m\geq 0} T(m, n)$. Из теоремы F следует, что $L(n) \leq \log_\phi(\sqrt{5}\,n + 1) - 2$. Докажите, что $2L(n) \geq \log_\phi(\sqrt{5}\,n + 1) - 2$.

▶ **39.** [*M25*] (Р. У. Госпер.) Среднее количество ударов, которые выполняет бейсболист, равно .334. Каково минимально возможное число ударов, которое он выполняет? [Напомним читателям, не являющимся приверженцами бейсбола, что среднее количество ударов = (количество бросков)/(число битов), округленное до трех десятичных знаков.]

▶ **40.** [*M28*] (*Дерево Штерна-Броко (Stern-Brocot).*) Рассмотрим бесконечное бинарное дерево, в котором каждый узел связан с дробью $(p_l + p_r)/(q_l + q_r)$, где p_l/q_l — метка узла, ближайшего к левому предшественнику, и p_r/q_r — метка узла, ближайшего к правому предшественнику. (Левый предшественник расположен перед узлом в симметричном порядке, в то время как правый предшественник расположен за узлом. Определение симметричного порядка приведено в разделе 2.3.1.) Если для узла левые предшественники отсутствуют, то $p_l/q_l = 0/1$; при отсутствии правых предшественников $p_r/q_r = 1/0$. Таким образом, метка корневого узла есть 1/1; метками узлов, порожденных корневым узлом, будут 1/2 и 2/1; метками четырех узлов второго уровня слева направо будут 1/3, 2/3, 3/2 и 3/1; метками восьми узлов третьего уровня будут 1/4, 2/5, 3/5, 3/4, 4/3, 5/3, 5/2, 4/1 и т.д.

Докажите, что для каждой метки p/q число p является взаимно простым с q; более того, узлы с метками p/q предшествуют узлам с метками p'/q' в симметричном порядке тогда и только тогда, когда метки удовлетворяют неравенству $p/q < p'/q'$. Найдите связь между цепной дробью для метки узла и пути к узлу, показав таким образом, что любое положительное рациональное число появляется как метка точно одного узла дерева.

41. [*M40*] (Дж. Шэллит (J. Shallit), 1979.) Покажите, что разложение в правильную цепную дробь выражения

$$\frac{1}{2^1}+\frac{1}{2^3}+\frac{1}{2^7}+\cdots=\sum_{n\ge1}\frac{1}{2^{2^n-1}}$$

содержит только единицы и двойки и представляется в исключительно простом виде. Докажите, что в случае, когда l — любое целое число ≥ 2, частичные отношения чисел Лиувилля $\sum_{n\ge1} l^{-n!}$ имеют регулярный вид. [Эти числа, введенные Ж. Лиувиллем (J. Liouville) в *J. de Math. Pures et Appl.* **16** (1851), 133–142, впервые были строго определены как *трансцендентные*. Первым доказал трансцендентность такой формы числа и подобных констант О. Дж. Кемпнер (A. J. Kempner) в *Trans. Amer. Math. Soc.* **17** (1916), 476–482.]

42. [*M30*] (Ж. Лагранж, 1798.) Предположим, что разложение числа X в правильную цепную дробь имеет вид $//A_1, A_2, \ldots //$, и пусть $q_n = K_n(A_1, \ldots, A_n)$. Обозначим через $\|x\|$ расстояние от x до ближайшего целого числа $\min_p |x-p|$. Покажите, что $\|qX\| \ge \|q_{n-1}X\|$ для $1 \le q < q_n$. (Таким образом, знаменатели q_n так называемых сходящихся дробей $p_n/q_n = //A_1, \ldots, A_n//$ представляют собой целые числа, "обрывающие ряд", что приводит к приобретению $\|qX\|$ новых свойств.)

43. [*M30*] (Д. В. Матула (D. W. Matula).) Покажите, что описываемое уравнением 4.5.1–(1) правило "медианного округления" для чисел, представленных в формате с фиксированной и плавающей дробными чертами, в случае, если число $x > 0$ не представимо, может быть введено следующим простым образом. Пусть разложение числа x в правильную цепную дробь имеет вид $a_0 + //a_1, a_2, \ldots //$ и пусть $p_n = K_{n+1}(a_0, \ldots, a_n)$, $q_n = K_n(a_1, \ldots, a_n)$. Тогда $\operatorname{round}(x) = (p_i/q_i)$, где дробь (p_i/q_i) представима, а дробь (p_{i+1}/q_{i+1}) — нет. [*Указание.* См. упр. 40.]

44. [*M25*] Предположим, что выполняются арифметические операции в формате с фиксированной дробной чертой с медианным округлением, в которых дроби (u/u') представимы тогда и только тогда, когда $|u| < M$, $0 \le u' < N$ и $u \perp u'$. Докажите или опровергните тождество $((u/u') \oplus (v/v')) \ominus (v/v') = (u/u')$ для всех представимых дробей (u/u') и (v/v'), обеспечивающих выполнение условия $u' < \sqrt{N}$ и отсутствие переполнения.

45. [*M25*] Покажите, что алгоритм Евклида (алгоритм 4.5.2A) в случае применения к двум двоичным числам при $n \to \infty$ требует для выполнения $O(n^2)$ единиц времени. (Такая же верхняя оценка справедлива и для выполнения алгоритма 4.5.2B.)

46. [*M43*] Можно ли уменьшить верхнюю границу $O(n^2)$ в упр. 45, если для вычисления наибольшего общего делителя использовать другой алгоритм?

47. [*M40*] Разработайте компьютерную программу нахождения как можно большего числа частичных отношений для вещественного числа x, задаваемого с высокой точностью. Примените ее для вычисления первых нескольких тысяч частичных отношений для постоянной Эйлера γ, которые можно вычислить по алгоритму, описанному Д. В. Суини (D. W. Sweeney) в *Math. Comp.* **17** (1963), 170–178. (Если γ есть рациональное число, то можно найти числитель и знаменатель этой константы, решив таким образом знаменитую математическую проблему. Согласно теории, изложенной в разделе, если рассматривать исходное число как случайное, следует ожидать получения около 0.97 частичных отношений на каждый десятичный разряд. При этом не возникает необходимости в операциях деления с многократной точностью. (См. алгоритм 4.5.2L и статью Дж. У. Ренча (J. W. Wrench) и Д. Шэнкса (D. Shanks), *Math. Comp.* **20** (1966), 444–447.)

48. [*M21*] Пусть $T_0 = (1, 0, u)$, $T_1 = (0, 1, v)$, $\ldots$, $T_{n+1} = ((-1)^{n+1}v/d, (-1)^n u/d, 0)$ представляют последовательности векторов, вычисляемых по алгоритму 4.5.2X (расширенный алгоритм Евклида), и пусть $//a_1, \ldots, a_n//$ — правильная цепная дробь для v/u. Выразите T_j через континуанты, включающие a_1, $\ldots$, a_n, где $1 < j \le n$.

49. [*M33*] Откорректируйте последнюю итерацию алгоритма 4.5.2X так, чтобы можно было заменить элемент a_n двумя частичными отношениями $(a_n - 1, 1)$. Подразумевается, что число итераций n подчиняется заданной закономерности. Продолжая предыдущее упражнение, положим, что λ и μ — произвольные положительные вещественные числа, и пусть $\theta = \sqrt{\lambda\mu v/d}$, где $d = \gcd(u, v)$. Докажите, что если число n четно и если $T_j = (x_j, y_j, z_j)$, то $\min_{j=1}^{n+1} |\lambda x_j + \mu z_j - [j \text{ even}]\,\theta| \le \theta$.

▸ **50.** [*M25*] Для данного иррационального числа $\alpha \in (0\,..\,1)$ и вещественных чисел β и γ, таких, что $0 \le \beta < \gamma < 1$, положим, что $f(\alpha, \beta, \gamma)$ — наименьшее неотрицательное целое число n, такое, что $\beta \le \alpha n \bmod 1 < \gamma$. (Такое целое число всегда существует в силу теоремы Вейля (Weyl), упр. 3.5–22.) Разработайте алгоритм для вычисления $f(\alpha, \beta, \gamma)$.

▸ **51.** [*M30*] (*Рациональная реконструкция.*) Число 28481 превращается в число 41/316 (по модулю 199999) в том смысле, что $316 \cdot 28481 \equiv 41$. Как это можно обнаружить? Объясните, как для заданных целых чисел a и m при $m > a > 1$ найти целые числа x и y, такие, что $ax \equiv y$ (по модулю m), $x \perp y$, $0 < x \le \sqrt{m/2}$ и $|y| \le \sqrt{m/2}$, или определить, что таких чисел x и y нет. Может ли существовать более одного решения?

4.5.4. Разложение на простые множители

В основу ряда вычислительных методов, которые рассматривались в этой книге, положен тот факт, что любое положительное целое число n можно однозначно выразить в виде

$$n = p_1 p_2 \ldots p_t, \qquad p_1 \le p_2 \le \cdots \le p_t, \tag{1}$$

где каждое p_k — простое число. (В случае, когда $n = 1$, это равенство выполняется при $t = 0$.) К сожалению, довольно сложно найти это разложение на простые множители или определить, является ли число n простым. Общеизвестно, что разложить на простые множители большое число значительно труднее, чем найти наибольший общий делитель двух больших чисел m и n. Поэтому там, где это возможно, следует избегать разложения больших чисел на простые множители. Но, учитывая, что разработан целый ряд оригинальных методов, позволяющих ускорить процесс разложения чисел на простые множители, проанализируем некоторые из этих методов. [Всесторонний исторический обзор методов разложения чисел на простые множители, известных до 1950 года, выполнен Х. К. Уильямсом (H. C. Williams) и Дж. О. Шэллитом (J. O. Shallit), *Proc. Symp. Applied Math.* **48** (1993), 481–531.]

Деление и разложение на множители. Прежде всего рассмотрим самый очевидный алгоритм разложения на простые множители. Если число $n > 1$, то его можно делить на последовательные простые числа $p = 2, 3, 5, \ldots$ до тех пор, пока не будет обнаружено наименьшее число p, для которого $n \bmod p = 0$. Тогда p и будет наименьшим простым множителем числа n. Тот же процесс можно применить к числу $n \leftarrow n/p$ и попытаться разделить полученное значение числа n на p и на бóльшие простые числа. Если на некотором этапе обнаружится, что $n \bmod p \ne 0$, но $\lfloor n/p \rfloor \le p$, можно сделать следующий вывод: число n — простое, так как если n не является простым числом, то в силу равенства (1) должно быть $n \ge p_1^2$. Но из условия $p_1 > p$ следует, что $p_1^2 \ge (p+1)^2 > p(p+1) > p^2 + (n \bmod p) \ge \lfloor n/p \rfloor p + (n \bmod p) = n$. В результате получаем следующую процедуру.

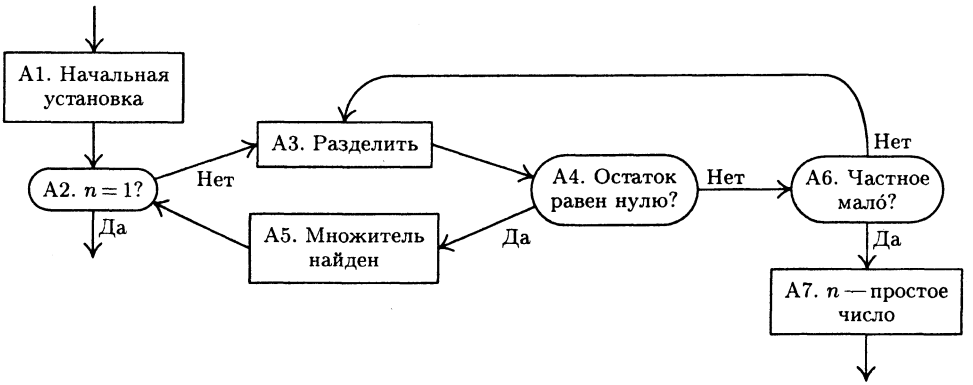


Рис. 11. Простой алгоритм разложения на множители.

Алгоритм А (*Разложение на простые множители путем деления*). По данному положительному целому числу N этот алгоритм (рис. 11) находит простые множители $p_1 \le p_2 \le \cdots \le p_t$ числа N в соответствии с равенством (1). В этом методе используется вспомогательная последовательность пробных делителей

$$2 = d_0 < d_1 < d_2 < d_3 < \cdots, \tag{2}$$

которая включает в себя все простые числа $\le \sqrt{N}$ (и, если это удобно, может содержать числа, *не* являющиеся простыми). Последовательность чисел d_i должна также содержать по крайней мере одно значение, такое, что $d_k \ge \sqrt{N}$.

A1. [Начальная установка.] Присвоить $t \leftarrow 0$, $k \leftarrow 0$, $n \leftarrow N$. (В ходе выполнения алгоритма переменные t, k, n подчинены следующим условиям: "$n = N/p_1 \ldots p_t$ и n не имеет простых множителей, меньших d_k".)

A2. [$n = 1$?] Если $n = 1$, алгоритм заканчивается.

A3. [Разделить.] Присвоить $q \leftarrow \lfloor n/d_k \rfloor$, $r \leftarrow n \bmod d_k$. (Здесь q и r — соответственно частное и остаток от деления числа n на d_k.)

A4. [Остаток равен нулю?] Если $r \ne 0$, то перейти к шагу A6.

A5. [Множитель найден.] Увеличить t на 1 и присвоить $p_t \leftarrow d_k$, $n \leftarrow q$. Возвратиться к шагу A2.

A6. [Частное мало́?] Если $q > d_k$, увеличить k на 1 и возвратиться к шагу A3.

A7. [n — простое число.] Увеличить t на 1, присвоить $p_t \leftarrow n$ и завершить выполнение алгоритма. ▮

В качестве примера алгоритма А рассмотрим разложение на простые множители числа $N = 25\,852$. Сразу же находим, что $N = 2 \cdot 12\,926$; следовательно, $p_1 = 2$. Далее, $12\,926 = 2 \cdot 6\,463$, так что $p_2 = 2$. Но теперь число $n = 6\,463$ не делится на числа 2, 3, 5, ..., 19; находим, что $n = 23 \cdot 281$; значит, $p_3 = 23$. В итоге имеем $281 = 12 \cdot 23 + 5$ и $12 \le 23$, т. е. $p_4 = 281$. В рассмотренном примере для определения простых множителей числа 25 852 нужно было выполнить 12 операций деления. С другой стороны, при разложении на простые множители чуть меньшего числа 25 849 (которое оказывается простым) пришлось бы затратить не менее 38 операций

деления. Это показывает, что время выполнения алгоритма A приблизительно пропорционально $\max(p_{t-1}, \sqrt{p_t})$. (При $t = 1$ эта формула справедлива, если положить, что $p_0 = 1$.)

Последовательность d_0, d_1, d_2, ... пробных делителей, используемая в алгоритме A, можно просто считать последовательностью чисел 2, 3, 5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, ..., члены которой, кроме первых трех, получаются посредством попеременного увеличения предыдущих на 2 и 4. Данная последовательность содержит все числа, не являющиеся кратными 2 или 3; при этом она содержит и такие числа, как 25, 35, 49 и т. д., не являющиеся простыми. Тем не менее алгоритм будет давать правильный результат. Если из этой последовательности убрать числа $30m \pm 5$ при $m \geq 1$, исключая тем самым как ложные простые все числа, кратные пяти, то можно сэкономить до 20% времени выполнения алгоритма. Исключив из рассмотрения числа, кратные 7, можно сократить список еще на 14% и т. д. Для управления выбором пробных делителей можно воспользоваться какой-нибудь компактной таблицей.

Если известно, что N мало́, резонно иметь таблицу всех необходимых простых чисел как часть программы. Например, если N меньше миллиона, то нужно включить 168 простых чисел, меньших 1000 (к этому списку нужно еще добавить значение $d_{168} = 1000$ как заключительного члена для случая, когда число N окажется простым, бо́льшим 997^2). Такую таблицу можно получить при помощи короткой вспомогательной программы (см., например, алгоритм 1.3.2P или упр. 8).

Сколько пробных делителей необходимо для алгоритма A? Пусть $\pi(x)$ — количество простых чисел $\leq x$, так что $\pi(2) = 1$, $\pi(10) = 4$. Асимптотическое поведение этой функции интенсивно изучалось многими величайшими математиками, начиная с Лежандра (Legendre), который исследовал эту проблему в 1798 году. Кульминацией многочисленных достижений в решении этой проблемы в течение 19 века явилось доказательство в 1899 году Шарлем де Ла Валле Пуссеном (Charles de La Vallée Poussin) того факта, что для некоторого $A > 0$

$$\pi(x) = \int_2^x \frac{dt}{\ln t} + O\bigl(xe^{-A\sqrt{\log x}}\bigr). \tag{3}$$

[*Mém. Couronnés Acad. Roy. Belgique* **59** (1899), 1–74; см. также J. Hadamard, *Bull. Soc. Math. France* **24** (1896), 199–220.] Интегрирование по частям дает

$$\pi(x) = \frac{x}{\ln x} + \frac{x}{(\ln x)^2} + \frac{2!\,x}{(\ln x)^3} + \cdots + \frac{r!\,x}{(\ln x)^{r+1}} + O\left(\frac{x}{(\log x)^{r+2}}\right) \tag{4}$$

для всех фиксированных $r \geq 0$. В дальнейшем остаточный член в формуле (3) последовательно уточнялся. Например, его можно заменить следующим выражением: $O\bigl(x\exp\bigl(-A(\log x)^{3/5}/(\log\log x)^{1/5}\bigr)\bigr)$. [См. A. Walfisz, *Weyl'sche Exponentialsummen in der neueren Zahlentheorie* (Berlin, 1963), Chapter 5.] В 1859 году Бернард Риман (Bernhard Riemann) предположил, что

$$\pi(x) = \sum_{k=1}^{\lg x} \frac{\mu(k)}{k} L\bigl(\sqrt[k]{x}\bigr) + O(1) = L(x) - \frac{1}{2}L\bigl(\sqrt{x}\bigr) - \frac{1}{3}L\bigl(\sqrt[3]{x}\bigr) + \cdots + O(1), \tag{5}$$

где $L(x) = \int_2^x dt/\ln t$. Эта формула хорошо согласуется с выполненными расчетами при выборе x в подходящем диапазоне.

x	$\pi(x)$	$L(x)$	Формула Римана
10^3	168	176.6	168.3
10^6	78498	78626.5	78527.4
10^9	50847534	50849233.9	50847455.4
10^{12}	37607912018	37607950279.8	37607910542.2
10^{15}	29844570422669	29844571475286.5	29844570495886.9
10^{18}	24739954287740860	24739954309690414.0	24739954284239494.4

(См. упр. 41.) Однако проблема распределения простых чисел не так проста, и в 1914 году Дж. Э. Литтлвуд (J. E. Littlewood) показал, что существует такая положительная константа C, что неравенство

$$\pi(x) > L(x) + C\sqrt{x}\log\log\log x/\log x$$

справедливо для бесконечно многих значений x, чем опроверг предположение Римана (5). (См. Hardy, Littlewood, *Acta Math.* **41** (1918), 119–196.) Результат Литтлвуда показал, что в простых числах заложено что-то мистическое и что для действительного понимания законов их распределения необходимо разработать глубокую математическую теорию. Риман высказал намного более конструктивное предположение, известное как "гипотеза Римана", согласно которому комплексная функция $\zeta(z)$ равна нулю только тогда, когда ее действительная часть равна $1/2$, за исключением тривиальных случаев, когда z есть отрицательное целое число. Если эта гипотеза верна, то из нее следует $\pi(x) = L(x) + O\big(\sqrt{x}\log x\big)$; см. упр. 25. Ричард Брент (Richard Brent) выполнил численную проверку гипотезы Римана для всех "малых" значений z, использовав метод Д. Г. Лемера (D. H. Lehmer) и показав, что функция $\zeta(z)$, мнимая часть которой принадлежит интервалу $0 < \Im z < 32585736.4$, имеет точно 75 000 000 нулей; для всех этих нулей $\Re z = \frac{1}{2}$ и $\zeta'(z) \neq 0$. [*Math. Comp.* **33** (1979), 1361–1372.]

Для анализа поведения алгоритма A в среднем необходимо выяснить, насколько большим может оказаться наибольший простой множитель p_t. Этот вопрос был впервые исследован Карлом Дикманом (Karl Dickman) [*Arkiv för Mat., Astron. och Fys.* **22A**, 10 (1930), 1–14], который проанализировал вероятность того, что случайное целое число, принимающее значения между 1 и x, будет иметь наибольший простой множитель $\le x^\alpha$. Дикман показал, что эта вероятность при $x \to \infty$ стремится к предельному значению $F(\alpha)$, где F может быть вычислено в результате решения функционального уравнения

$$F(\alpha) = \int_0^\alpha F\Big(\frac{t}{1-t}\Big)\frac{dt}{t} \quad \text{при } 0 \le \alpha \le 1; \qquad F(\alpha) = 1 \quad \text{при } \alpha \ge 1. \tag{6}$$

Ход его рассуждений был примерно следующим. Для заданного $0 < t < 1$ количество целых чисел, меньших x, наибольшие простые множители которых расположены между x^t и x^{t+dt}, равно $xF'(t)\,dt$. Количество простых чисел p в этом интервале равно $\pi(x^{t+dt}) - \pi(x^t) = \pi\big(x^t + (\ln x)x^t\,dt\big) - \pi(x^t) = x^t\,dt/t$. Для каждого такого p количество целых чисел n, таких, что "$np \le x$ и наибольший простой множитель числа n не превышает p", равно количеству чисел, для которых $n \le x^{1-t}$

и наибольший простой множитель не превышает $(x^{1-t})^{t/(1-t)}$, т. е. $x^{1-t}F\big(t/(1-t)\big)$. Следовательно, $xF'(t)\,dt = (x^t\,dt/t)\big(x^{1-t}F\big(t/(1-t)\big)\big)$ и уравнение (6) получается посредством интегрирования последнего уравнения. Этому эвристическому доводу может быть придана строгость. В. Рамасвами (V. Ramaswami) [*Bull. Amer. Math. Soc.* **55** (1949), 1122–1127] показал, что в случае фиксированных α при $x \to \infty$ интересующая нас вероятность асимптотически равна $F(\alpha) + O(1/\log x)$; анализом этой проблемы занимались и многие другие авторы [см. обзор Карла К. Нортона (Karl K. Norton), *Memoirs Amer. Math. Soc.* **106** (1971), 9–27].

В случае, если $\frac{1}{2} \le \alpha \le 1$, формула (6) упрощается:

$$F(\alpha) = 1 - \int_\alpha^1 F\Big(\frac{t}{1-t}\Big)\frac{dt}{t} = 1 - \int_\alpha^1 \frac{dt}{t} = 1 + \ln\alpha.$$

Таким образом, например, вероятность того, что для случайного целого числа $\le x$ существует простой множитель $> \sqrt{x}$, равна $1 - F(\frac{1}{2}) = \ln 2$, что составляет около 69%. Во всех этих случаях выполнение алгоритма А сопряжено с определенными трудностями.

Обсудим теперь вопрос о том, насколько быстро алгоритм А выдаст результат в случае, когда множитель ограничен шестиразрядным числом. Заметим, что для больших чисел N, если нам не повезет, общее время выполнения процедуры разложения на множители с использованием пробных делителей очень быстро превысит практические возможности алгоритма.

Ниже в этом разделе будет показано, что существуют достаточно хорошие способы, позволяющие определить, является ли относительно большое n простым числом, не проверяя все пробные делители до $\sqrt{n}$. Поэтому алгоритм А будет выполняться быстрее, если между шагами А2 и А3 включить проверку принадлежности числа к простым числам. Для алгоритма, усовершенствованного таким образом, время выполнения можно в первом приближении считать пропорциональным p_{t-1}, т. е. *второму по величине* простому множителю числа N вместо $\max(p_{t-1}, \sqrt{p_t}\,)$. Используя аргументы, аналогичные аргументам Дикмана (см. упр. 18), можно показать, что с приближенной вероятностью $G(\beta)$ второй по величине простой множитель случайного целого числа $\le x$ будет $\le x^\beta$, где

$$G(\beta) = \int_0^\beta \left(G\Big(\frac{t}{1-t}\Big) - F\Big(\frac{t}{1-t}\Big)\right)\frac{dt}{t} \quad \text{при } 0 \le \beta \le \tfrac{1}{2}. \tag{7}$$

Очевидно, что для $\beta \ge \frac{1}{2}$ эта функция $G(\beta) = 1$ (см. рис. 12). Численное решение уравнений (6) и (7) дает следующие "процентные отношения значений" этих функций.

$F(\alpha), G(\beta) =$	.01	.05	.10	.20	.35	.50	.65	.80	.90	.95	.99
$\alpha =$	.2697	.3348	.3785	.4430	.5220	.6065	.7047	.8187	.9048	.9512	.9900
$\beta =$	.0056	.0273	.0531	.1003	.1611	.2117	.2582	.3104	.3590	.3967	.4517

Таким образом, примерно в половине случаев второй по величине простой множитель будет $\le x^{.2117}$ и т. д.

Можно проанализировать также величину t — *общего числа простых множителей.* Обычно $1 \le t \le \lg N$, но эти нижние и верхние границы достигаются редко. Можно доказать, что если N выбирается как случайное число в интервале между 1

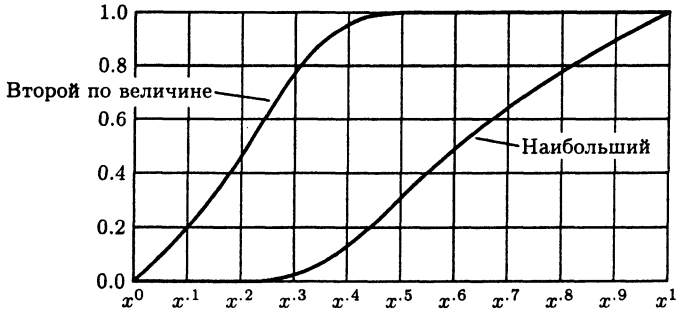


Рис. 12. Функция распределения вероятностей для двух наибольших простых множителей случайного целого числа $\le x$.

и x, то вероятность того, что $t \le \ln\ln x + c\sqrt{\ln\ln x}$ при $x \to \infty$, стремится к

$$\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{c} e^{-u^2/2}\, du \tag{8}$$

для любых фиксированных c. Другими словами, распределение величины t является, по существу, нормальным со средним значением и дисперсией, равными $\ln\ln x$; примерно для 99.73% всех больших чисел $\le x$ выполняется $|t - \ln\ln x| \le 3\sqrt{\ln\ln x}$. Далее, известно, что среднее значение для $t - \ln\ln x$ при $1 \le N \le x$ достигает величины

$$\gamma + \sum_{p \text{ простое}} \big(\ln(1 - 1/p) + 1/(p-1)\big) = \gamma + \sum_{n\ge 2} \frac{\varphi(n) \ln \zeta(n)}{n}$$
$$= 1.03465\,38818\,97437\,91161\,97942\,98464\,63825\,46703+. \tag{9}$$

[См. G. H. Hardy, E. M. Wright, *An Introduction to the Theory of Numbers*, 5th edition (Oxford, 1979), §22.11; см. также P. Erdős, M. Kac, *Amer. J. Math.* **26** (1940), 738–742.]

Размер простых множителей имеет примечательную связь с перестановками. Среднее число битов в k-м наибольшем простом множителе случайного n-битового числа при $n \to \infty$ асимптотически стремится к средней длине k-го наибольшего цикла перестановок случайного n-элемента. [См. D. E. Knuth and L. Trabb Pardo, *Theoretical Comp. Sci.* **3** (1976), 321–348; А. М. Вершик, *Soviet Math. Doklady* **34** (1987), 57–61.] Отсюда следует, что алгоритм A сначала находит несколько малых множителей, а затем начинает долгий поиск остальных бóльших множителей.

Прекрасное описание распределения вероятностей простых множителей для случайных целых чисел приведено в статье Патрика Биллингслея (Patrick Billingsley), опубликованной в журнале *AMM* **80** (1973), 1099–1115 (см. также его статью в *Annals of Probability* **2** (1974), 749–791).

Разложение на простые множители с использованием псевдослучайных циклов. В начале главы 3 указывалось, что выбранный генератор случайных чисел дает числа, которые оказываются не совсем случайными. Этот тезис, работавший в той главе против нас, здесь оборачивается благом и ведет к поразительно эффективному методу разложения на простые множители, открытому Дж. М. Поллардом (J. M. Pollard) [*BIT* **15** (1975), 331–334].

Число шагов вычислений в методе Полларда имеет порядок $\sqrt{p_{t-1}}$, поэтому при больших N он значительно быстрее алгоритма А. Как следует из уравнения (7) и рис. 12, время выполнения алгоритма, реализующего метод Полларда, обычно меньше, чем $N^{1/4}$.

Пусть $f(x)$ — полином с целочисленными коэффициентами. Рассмотрим две последовательности

$$x_0 = y_0 = A; \qquad x_{m+1} = f(x_m) \bmod N, \qquad y_{m+1} = f(y_m) \bmod p, \tag{10}$$

где p — любой простой множитель числа N. Тогда

$$y_m = x_m \bmod p \qquad \text{при } m \geq 1. \tag{11}$$

Из упр. 3.1–7 видно, что для некоторого $m \geq 1$ выполняется равенство $y_m = y_{\ell(m)-1}$, где $\ell(m)$ — наибольшая степень 2, не превышающая m. Таким образом, разность $x_m - x_{\ell(m)-1}$ будет кратна p. Далее, если $f(y) \bmod p$ ведет себя как случайное отображение множества $\{0, 1, \ldots, p-1\}$ на самое себя, то, как показано в упр. 3.1–12, среднее значение наименьших из таких m будет порядка $\sqrt{p}$. Фактически это среднее значение для случайных выборок меньше $1.625\,Q(p)$, как показано в упр. 4 ниже; функция $Q(p) \approx \sqrt{\pi p/2}$ была определена в разделе 1.2.11.3.

Если различные простые делители числа N соответствуют различным значениям m (что для больших N почти всегда верно), их можно найти, вычисляя $\gcd(x_m - x_{\ell(m)-1}, N)$ для $m = 1, 2, 3, \ldots$ до тех пор, пока остаток от деления на множитель не станет простым. Поллард назвал этот метод *ро*-методом, так как периодическая последовательность вида $y_0, y_1, \ldots$ напоминает греческую букву ρ.

Из главы 3 известно, что линейный полином $f(x) = ax+c$ не обеспечивает достаточной случайности последовательности. Следующим простейшим видом полинома является квадратичный полином $f(x) = x^2 + 1$. Нам *неизвестно*, обеспечивает ли данная функция случайность, но недостаток знаний по этому вопросу склоняет нас, скорее всего, в пользу гипотезы о том, что функция обеспечивает достаточную случайность, а эмпирическая проверка показала, что она ведет себя так, как и предполагалось. В действительности же функция f, вероятно, даже дает немного *больше*, чем случайность, так как $x^2 + 1$ содержит только $\frac{1}{2}(p + 1)$ различных значений по модулю p (см. Arney, Bender, *Pacific J. Math.* **103** (1982), 269–294). В связи со сказанным уместна следующая процедура.

Алгоритм В (*Разложение на простые множители при помощи ро-метода*). Этот алгоритм с высокой вероятностью вычисляет простые множители данного целого числа $N \geq 2$, хотя не исключен и отрицательный результат (т. е. множитель найден не будет. — *Прим. перев.*).

В1. [Начальная установка.] Присвоить $x \leftarrow 5$, $x' \leftarrow 2$, $k \leftarrow 1$, $l \leftarrow 1$, $n \leftarrow N$. (Во время выполнения этого алгоритма число n не является множителем числа N, а переменные x и x' представляют величины $x_m \bmod n$ и $x_{\ell(m)-1} \bmod n$ в выражении (10), где также $f(x) = x^2 + 1$, $A = 2$, $l = \ell(m)$ и $k = 2l - m$.)

В2. [Проверить, будет ли число простым.] Если n — простое число (рассматривается ниже), вывести n в качестве результата; на этом выполнение алгоритма завершается.

B3. [Множитель найден?] Присвоить $g \leftarrow \gcd(x' - x, n)$. Если $g = 1$, то перейти к шагу B4; в противном случае вывести g. Теперь, если $g = n$, алгоритм завершается (его выполнение прерывается, ибо нам известно, что n не является простым числом). В противном случае присвоить $n \leftarrow n/g$, $x \leftarrow x \bmod n$, $x' \leftarrow x' \bmod n$ и возвратиться к шагу B2. (Заметим, что g может и не быть простым числом — это подлежит проверке. В каждом случае, когда g — не простое число, его простые множители не могут быть определены при помощи этого алгоритма.)

B4. [Продвинуться.] Присвоить $k \leftarrow k - 1$. Если $k = 0$, то присвоить $x' \leftarrow x$, $l \leftarrow 2l$, $k \leftarrow l$. Присвоить $x \leftarrow (x^2 + 1) \bmod n$ и возвратиться к шагу B3. ▮

В качестве примера алгоритма B попробуем снова разложить на простые множители число $N = 25\,852$. В результате третьей итерации шага B3 будет получен следующий результат: $g = 4$ (не является простым). Еще после шести итераций алгоритм находит множитель $g = 23$. В этом примере алгоритм не различил сам себя, но он, конечно же, был разработан для поиска *больших* простых множителей. Алгоритм A на поиск больших простых множителей затрачивает гораздо больше времени, но в том, что касается определения малых простых множителей, к нему нет никаких претензий. На практике сначала некоторое время выполняется алгоритм A, а затем запускается алгоритм B.

Рассмотрев десять наибольших простых шестиразрядных чисел, можно получить лучший способ использования алгоритма B. Количество итераций $m(p)$, требуемое алгоритму B для нахождения множителя p, приведено в следующей таблице.

$p =$	999863	999883	999907	999917	999931	999953	999959	999 961	999979	999983
$m(p) =$	276	409	2106	1561	1593	1091	474	1819	395	814

Экспериментальным путем установлено, что среднее значение $m(p)$ примерно равно $2\sqrt{p}$ и никогда не превышает $12\sqrt{p}$, когда $p < 1\,000\,000$. При $p < 10^6$ максимум $m(p)$ равен $m(874\,771) = 7\,685$. Максимум функции $m(p)/\sqrt{p}$ достигается при $p = 290\,047$ и $m(p) = 6\,251$. На основании этих результатов почти все 12-разрядные числа можно разложить на простые множители быстрее чем за 2 000 итераций алгоритма B (сравните с 75 000 операций деления, требуемых для выполнения алгоритма A).

Время на каждой итерации алгоритма B, в основном, затрачивается на выполнение умножения и деления с многократной точностью на шаге B4 и на вычисление наибольшего общего делителя на шаге B3. Выполнение этих операций может быть ускорено за счет применения методики "умножения Монтгомери" (см. упр. 4.3.1–41). Более того, в случае, когда операция нахождения наибольшего общего делителя выполняется медленно, Поллард предложил ускорить процесс путем накапливания произведения по модулю n, скажем, десяти последовательных значений $(x' - x)$ перед тем, как искать наибольший общий делитель. Таким образом, 90% операций нахождения наибольшего общего делителя заменяется одним умножением по модулю N ценой некоторого увеличения вероятности того, что решение при этом не будет найдено. Поллард также предложил начинать вычисления на шаге B1 с $m = q$, а не с $m = 1$, где q равно одной десятой от количества итераций, которые планируется реализовать.

В тех редких случаях, когда для больших N результат не был найден, можно применить функцию $f(x) = x^2 + c$ при некотором $c \neq 0$ или 1. Следует избегать

также значения $c = -2$, поскольку рекуррентное уравнение $x_{m+1} = x_m^2 - 2$ имеет решение в виде $x_m = r^{2^m} + r^{-2^m}$. Похоже, что другие значения параметра c не приводят к возникновению простых связей по модулю p и все они должны быть удовлетворительными при подходящих начальных значениях.

Ричард Брент применил эту модификацию алгоритма B для нахождения простого множителя 1 238 926 361 552 897 числа $2^{256} + 1$. [См. *Math. Comp.* **36** (1981), 627–630; **38** (1982), 253–255.]

Метод Ферма. Другой подход к решению проблемы разложения чисел на простые множители, который в 1643 году предложил Пьер де Ферма (Pierre de Fermat), более подходит для поиска больших множителей, нежели малых. [Описание метода, данное самим Ферма и переведенное на английский язык, можно найти в книге Л. Е. Диксона (L. E. Dickson) *History of the Theory of Numbers* **1** (Carnegie Inst. of Washington, 1919), 357.]

Допустим, что $N = uv$, где $u \le v$. Для практических целей можно положить, что N — нечетное число. Это означает, что u и v тоже нечетны. Можно также положить, что

$$x = (u + v)/2, \qquad y = (v - u)/2, \tag{12}$$

$$N = x^2 - y^2, \qquad 0 \le y < x \le N. \tag{13}$$

Суть метода Ферма заключается в том, что ищутся такие значения x и y, которые удовлетворяют уравнению (13). В следующем алгоритме показано, как таким путем можно разложить число на простые множители, *не выполняя операций деления.*

Алгоритм C (*Разложение на простые множители при помощи операций сложения и вычитания*). По данному нечетному числу N этот алгоритм определяет наибольший множитель числа N, меньший или равный $\sqrt{N}$.

C1. [Начальная установка.] Присвоить $x \leftarrow 2\lfloor\sqrt{N}\rfloor + 1$, $y \leftarrow 1$, $r \leftarrow \lfloor\sqrt{N}\rfloor^2 - N$. (Во время выполнения этого алгоритма величины x, y, r отвечают соответственно величинам $2x + 1$, $2y + 1$, $x^2 - y^2 - N$ в уравнении (13). Должно соблюдаться условие $|r| < x$ и $y < x$.)

C2. [Выполнено?] Если $r = 0$, то выполнение алгоритма завершается. Имеем

$$N = \big((x - y)/2\big)\big((x + y - 2)/2\big),$$

а $(x - y)/2$ — наибольший множитель для N, меньший или равный $\sqrt{N}$.

C3. [Шаг по x.] Присвоить $r \leftarrow r + x$ и $x \leftarrow x + 2$.

C4. [Шаг по y.] Присвоить $r \leftarrow r - y$ и $y \leftarrow y + 2$.

C5. [Проверить r.] Если $r > 0$, то возвратиться к шагу C4, иначе возвратиться к шагу C2. ▮

Возможно, читатель сочтет небезынтересным поиск вручную простых множителей числа 377 при помощи этого алгоритма. Число шагов, необходимых для нахождения множителей u и v числа $N = uv$, по существу, пропорционально

$$(x + y - 2)/2 - \lfloor\sqrt{N}\rfloor = v - \lfloor\sqrt{N}\rfloor;$$

это, конечно, может оказаться очень большой величиной, хотя каждый шаг на большинстве компьютеров может выполняться очень быстро. Р. Ш. Леман

(R. S. Lehman) [*Math. Comp.* **28** (1974), 637–646] усовершенствовал алгоритм таким образом, что в худшем случае для его выполнения требуется только $O(N^{1/3})$ операций.

Называть алгоритм C методом Ферма не совсем правильно, поскольку Ферма использовал несколько более обтекаемый подход. В компьютерах основной цикл алгоритма C выполняется довольно быстро, но для вычислений вручную он мало пригоден. На самом деле Ферма не сохранял текущие значения y; он рассматривал величину $x^2 - N$ и, исходя из наименее значимых разрядов, делал вывод о том, является ли эта величина полным квадратом. (Последние два разряда полного квадрата должны быть 00, $e1$, $e4$, 25, $o6$ или $e9$, где e — четная, а o — нечетная цифра.) По этой причине он избегал операций, которые выполнялись на шагах C4 и C5, заменяя их при помощи специальных приемов операцией определения числа, не являющегося полным квадратом.

Предложенный Ферма способ просмотра правых крайних разрядов можно, конечно, обобщить, используя другие модули. Предположим для ясности, что $N = 8\,616\,460\,799$ (историческое значение этого числа описывается ниже), и рассмотрим следующую таблицу.

m	Если $x \bmod m$ равно	, то $x^2 \bmod m$ равно	и $(x^2 - N) \bmod m$ равно
3	0, 1, 2	0, 1, 1	1, 2, 2
5	0, 1, 2, 3, 4	0, 1, 4, 4, 1	1, 2, 0, 0, 2
7	0, 1, 2, 3, 4, 5, 6	0, 1, 4, 2, 2, 4, 1	5, 6, 2, 0, 0, 2, 6
8	0, 1, 2, 3, 4, 5, 6, 7	0, 1, 4, 1, 0, 1, 4, 1	1, 2, 5, 2, 1, 2, 5, 2
11	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10	0, 1, 4, 9, 5, 3, 3, 5, 9, 4, 1	10, 0, 3, 8, 4, 2, 2, 4, 8, 3, 0

Если величина $x^2 - N$ есть полный квадрат y^2, то она должна иметь остаток по модулю m, соответствующий этому факту. Например, если $N = 8\,616\,460\,799$ и $x \bmod 3 \neq 0$, то $(x^2 - N) \bmod 3 = 2$, т. е. величина $x^2 - N$ не может быть полным квадратом. Поэтому x должно быть кратным 3 для всех значений $N = x^2 - y^2$. Из таблицы видно, что

$$\begin{aligned} x \bmod 3 &= 0; \\ x \bmod 5 &= 0, 2 \text{ или } 3; \\ x \bmod 7 &= 2, 3, 4 \text{ или } 5; \\ x \bmod 8 &= 0 \text{ либо } 4 \text{ (отсюда } x \bmod 4 = 0); \\ x \bmod 11 &= 1, 2, 4, 7, 9 \text{ или } 10. \end{aligned} \tag{14}$$

Это значительно сокращает процесс поиска x. Пусть, например, x должно быть кратно 12. Тогда должно быть $x \geq \lceil \sqrt{N} \rceil = 92\,825$ и наименьшим числом, кратным 12, является число 92 832. Это значение дает в остатке $(2, 5, 3)$ по модулю $(5, 7, 11)$ соответственно, поэтому оно не удовлетворяет уравнению (14) по отношению к модулю 11. Увеличив x на 12, можно заменить остаток по модулю 5 на 2, по модулю 7 на 5 и по модулю 11 на 1. Поэтому легко увидеть, что первое значение величины $x \geq 92\,825$, удовлетворяющее всем условиям в уравнении (14), есть $x = 92\,880$. Теперь $92\,880^2 - N = 10\,233\,601$, и, вычислив вручную квадратный корень, получаем, что $10\,233\,601 = 3\,199^2$ действительно является полным квадратом. Таким образом получено искомое решение $x = 92\,880$, $y = 3\,199$, а разложение на простые

множители имеет вид

$$8\,616\,460\,799 = (x-y)(x+y) = 89\,681 \cdot 96\,079.$$

Это значение N интересно тем, что английским экономистом и логиком У. С. Дживонсом (W. S. Jevons) в хорошо известной книге оно было введено следующим образом: "По данным двум числам можно найти их произведение простым и надежным способом, но совсем другое дело, когда для большого числа необходимо найти его множители. Можно ли сказать, какие два числа были перемножены, чтобы получилось число 8 616 460 799? Я думаю, что вряд ли кто-либо, кроме меня, знает это". [*The Principles of Science* (Macmillan, 1874), Chapter 7.] Однако, как следует из вышесказанного, Ферма смог разложить это число на простые множители на обратной стороне конверта меньше чем за десять минут! Основная мысль Дживонса о сложности разложения чисел на простые множители по сравнению с их перемножением справедлива, но только в случае, когда мы имеем дело с произведением чисел, не настолько близких друг к другу.

Вместо модулей, рассматриваемых в уравнении (14), можно использовать любые степени различных простых чисел. Например, если взять 25 вместо 5, то возможные значения величины $x \bmod 25$ будут равняться 0, 5, 7, 10, 15, 18 и 20. Это дает больше информации, чем (14). В общем случае можно получить больше информации, выполняя операции по модулю p^2, чем по модулю p, при нечетных p, если $x^2 - N \equiv 0$ (по модулю p) имеет решение x.

Рассмотренный модулярный метод называется *методом решета (сита)* (sieve procedure), так как можно представить, что все целые числа проходят через "решето", пропускающее только те значения, для которых $x \bmod 3 = 0$; затем эти числа просеиваются через другое сито, которое пропускает только числа, для которых $x \bmod 5 = 0$, 2 или 3, и т. д. Каждое сито в отдельности отсеивает примерно половину оставшихся значений (см. упр. 6). Когда же просеивание ведется при помощи попарно взаимно простых модулей, то на основании китайской теоремы об остатках (теорема 4.3.2C) каждое сито работает независимо от остальных. Поэтому, если выполнять просеивание относительно, скажем, 30 различных простых чисел, то для того, чтобы определить, будет ли величина $x^2 - N$ полным квадратом для y^2, достаточно из каждых 2^{30} величин проверить только одну.

Алгоритм D (*Разложение на простые множители методом решета*). По данным нечетным числам N этот алгоритм определяет наибольший множитель числа N, меньший или равный $\sqrt{N}$. В процедуре используются попарно взаимно простые модули $m_1, m_2, \ldots, m_r$, которые также взаимно просты с N. Предположим, что доступны r *таблиц просеивания* $S[i,j]$, $0 \le j < m_i$, $1 \le i \le r$, где

$$S[i,j] = \big[j^2 - N \equiv y^2 \text{ (по модулю } m_i) \text{ имеет решение } y\big].$$

D1. [Начальная установка.] Присвоить $x \leftarrow \lceil \sqrt{N} \rceil$ и $k_i \leftarrow (-x) \bmod m_i$ при $1 \le i \le r$. (Во время выполнения этого алгоритма индексные переменные $k_1, k_2, \ldots, k_r$ будут установлены таким образом, что $k_i = (-x) \bmod m_i$.)

D2. [Просеять.] Если $S[i,k_i] = 1$ при $1 \le i \le r$, то перейти к шагу D4.

D3. [Найти x.] Присвоить $x \leftarrow x+1$ и $k_i \leftarrow (k_i - 1) \bmod m_i$ при $1 \le i \le r$. Возвратиться к шагу D2.

D4. [Проверить $x^2 - N$.] Присвоить $y \leftarrow \lfloor\sqrt{x^2-N}\rfloor$ или $\lceil\sqrt{x^2-N}\rceil$. Если $y^2 = x^2 - N$, то $(x - y)$ — искомый множитель. Завершить выполнение алгоритма; в противном случае возвратиться к шагу D3. ▮

Ускорить выполнение этой процедуры можно различными способами. Например, выше было отмечено, что для случая $N \bmod 3 = 2$ значение x должно быть кратным 3; можно положить, что $x = 3x'$, и использовать другое сито, соответствующее x', повысив скорость выполнения операций в три раза. Если $N \bmod 9 = 1$, 4 или 7, то x должно быть сравнимо с ±1, ±2 или ±4 (по модулю 9); так что можно, пропустив числа через два сита (одно для x', а другое — для x'', где $x = 9x' + a$ и $x = 9x'' - a$), повысить скорость в $4\frac{1}{2}$ раза. Если $N \bmod 4 = 3$, то $x \bmod 4$ известно и скорость повысится еще в 4 раза; в другом случае, когда $N \bmod 4 = 1$, x должно быть нечетным, что повысит скорость в два раза. Еще один способ удвоения скорости (ценой расширения объема применяемой памяти) заключается в объединении пары модулей путем использования $m_{r-k}\, m_k$ вместо m_k для $1 \le k < \frac{1}{2}r$.

Еще более важный способ повышения скорости выполнения алгоритма D состоит в использовании булевых операций, которые реализованы в большинстве двоичных компьютеров. Будем считать, что MIX представляет собой двоичный компьютер с длиной слова 30 бит. Таблицы $S[i, k_i]$ можно хранить в памяти так, чтобы на каждую позицию приходился один бит; таким образом, в одном слове можно хранить 30 значений. Операцию AND, которая заменяет k-й бит накопителя нулем, если k-й бит заданного слова в памяти есть нуль для $1 \le k \le 30$, можно использовать для одновременной обработки 30 значений x! Для удобства можно сделать несколько копий таблиц $S[i, j]$ с тем, чтобы элементы таблицы для m_i занимали $\operatorname{lcm}(m_i, 30)$ бит. Тогда таблицы просеивания для каждого модуля заполнят некоторое целое число слов. При таких предположениях выполнение основного цикла алгоритма D 30 раз эквивалентно такой последовательности команд.

```
D2 LD1  K1     rI1 ← k'_1.
   LDA  S1,1   rA ← S'[1, rI1].
   DEC1 1      rI1 ← rI1 − 1.
   J1NN *+2
   INC1 M1     Если rI1 < 0, то присвоить rI1 ← rI1 + lcm(m_1, 30).
   ST1  K1     k'_1 ← rI1.
   LD1  K2     rI1 ← k'_2.
   AND  S2,1   rA ← rA ∧ S'[2, rI1].
   DEC1 1      rI1 ← rI1 − 1.
   J1NN *+2
   INC1 M2     Если rI1 < 0, то присвоить rI1 ← rI1 + lcm(m_2, 30).
   ST1  K2     k'_2 ← rI1.
   LD1  K3     rI1 ← k'_3.
   ...         (От m_3 до m_r так же, как m_2.)
   ST1  Kr     k'_r ← rI1.
   INCX 30     x ← x + 30.
   JAZ  D2     Повторить, если все просеяно. ▮
```

По существу, количество циклов, необходимых для выполнения 30 итераций, равно $2 + 8r$; в случае, если $r = 11$, это означает, что на выполнение одной итерации

затрачивается три цикла, как и в алгоритме C, но в алгоритме C, кроме того, выполняется еще $y = \frac{1}{2}(v - u)$ итераций.

Если бы элементы в таблице для m_i занимали не целое число слов, то на каждой итерации необходимо было бы выполнять сдвиг элементов таблицы, чтобы биты были расположены должным образом. Это привело бы к добавлению в основной цикл множества дополнительных команд и, вероятно, сделало бы выполнение программы слишком медленным для всех значений $v/u \geq 100$ по сравнению с алгоритмом C (упр. 7).

Процедуры просеивания можно применять к множеству других задач, не обязательно связанных с выполнением арифметических действий. Обзор этих методов выполнен Марвином Ч. Вундерлихом (Marvin C. Wunderlich) и приведен в *JACM* **14** (1967), 10–19.

В 19 веке для разложения чисел на простые множители Ф. У. Лоуренс (F. W. Lawrence) предложил конструкцию специальных просеивающих машин [*Quart. J. of Pure and Applied Math.* **28** (1896), 285–311], а в 1919 году Э. О. Карисан (E. O. Carissan) дополнил такое устройство еще 14-ю модулями. [С интересной историей того, как были заново открыты и сохранены для потомства давно забытые сита Карисана, можно ознакомиться в работе Shallit, Williams, Morain, *Math. Intelligencer* **17**, 3 (1995), 41–47.] Много различных просеивающих машин было разработано и использовалось в течение 1926–1989 годов Д. Г. Лемером и его сотрудниками, которые начали с велосипедных цепей, а позже использовали фотоэлектронные элементы и другие технологии (см., например, *AMM* **40** (1933), 401–406). Электронное решето Лемера, использующее линию задержки, которое было запущено в эксплуатацию в 1965 году, обрабатывает один миллион чисел в секунду. К 1995 году стало возможным сконструировать машину, которая просеивает 6 144 млн чисел в секунду, выполняя 256 итераций на шагах D2 и D3 за почти 5.2 нс. [См. Lukes, Patterson, Williams, *Nieuw Archief voor Wiskunde* (4) **13** (1995), 113–139.] Д. Г. Лемер и Эмма Лемер (D. H. and Emma Lehmer) описали в *Math. Comp.* **28** (1974), 625–635, другой способ разложения на простые множители с использованием решета.

Проверка принадлежности чисел к простым. Из всех рассмотренных до сих пор алгоритмов ни один не может эффективно определить, является ли большое число n простым. К счастью, существуют другие способы решения этой задачи. Эффективные способы были разработаны Э. Люка (É. Lucas) и др., в частности Д. Г. Лемером [см. *Bull. Amer. Math. Soc.* **33** (1927), 327–340].

Согласно теореме Ферма (теорема 1.2.4F)

$$x^{p-1} \bmod p = 1,$$

когда p — простое число и x не кратно p. При этом имеются эффективные методы вычисления $x^{n-1} \bmod n$, требующие только $O(\log n)$ операций умножения по модулю n. (Они будут исследоваться в разделе 4.6.3.) Поэтому зачастую можно определить, что n *не* является простым, убедившись, что данное условие не выполняется.

Например, однажды Ферма установил, что числа $2^1 + 1$, $2^2 + 1$, $2^4 + 1$, $2^8 + 1$ и $2^{16} + 1$ являются простыми. В письме Мерсенну (Mersenne), написанному в 1640 году, Ферма предположил, что $2^{2^n} + 1$ — всегда простое число, и сообщил, что он не в состоянии определить, является ли простым число $4\,294\,967\,297 = 2^{32} + 1$. Ни Ферма,

ни Мерсенн так и не решили этой задачи, хотя могли сделать это следующим образом: можно вычислить число $3^{2^{32}} \bmod (2^{32}+1)$, выполнив 32 операции возведения в квадрат по модулю $2^{32}+1$, и получить результат, равный 3 029 026 160; поэтому (по теореме, открытой Ферма в том же 1640 году!) $2^{32}+1$ — *не* простое число. Данный аргумент не дает никакого представления о том, чему равны множители, но является ответом на поставленный Ферма вопрос.

Теорема Ферма представляет собой мощное средство анализа, которое дает возможность определить, что данное число не является простым. Если число n не простое, то всегда можно найти такое значение $x < n$, что $x^{n-1} \bmod n \neq 1$. Опыт показывает, что такое значение почти всегда находится очень быстро. Существует несколько редких значений числа n, для которых $x^{n-1} \bmod n$ часто равно единице, но тогда n имеет множитель, меньший $\sqrt[3]{n}$ (упр. 9).

Этот метод может быть расширен для доказательства того, что большое число n действительно *является* простым, если использовать следующую идею. *Если имеется число x, для которого порядок x по модулю n равен $n-1$, то n — простое число.* (Порядок числа x по модулю n — это наименьшее положительно целое число k, такое, что $x^k \bmod n = 1$; см. раздел 3.2.1.2.) Из этого условия следует, что числа $x^1 \bmod n$, ..., $x^{n-1} \bmod n$ различны и взаимно просты с n, а следовательно, это должны быть числа 1, 2, ..., $n-1$, расположенные в некотором порядке. Таким образом, n не имеет ни одного собственного делителя. Если n — простое число, то такое число x (называемое *первообразным корнем* числа n) всегда существует (см. упр. 3.2.1.2–16). В действительности таких первообразных корней довольно много. Существует $\varphi(n-1)$ таких чисел, и это достаточно большое число, так как $n/\varphi(n-1) = O(\log\log n)$.

Чтобы определить, будет ли порядок x равен $n-1$, совсем необязательно вычислять $x^k \bmod n$ для всех $k \leq n-1$. Порядок x будет равен $n-1$ тогда и только тогда, когда выполняются условия

i) $x^{n-1} \bmod n = 1$;

ii) $x^{(n-1)/p} \bmod n \neq 1$ для всех простых чисел p, которые делят $n-1$.

Следовательно, $x^s \bmod n = 1$ тогда и только тогда, когда s кратно порядку числа x по модулю n. Поэтому, если оба условия выполняются и если k есть порядок x по модулю n, k является делителем $n-1$, но не является делителем числа $(n-1)/p$ ни для одного простого множителя p числа $n-1$. Значит, остается единственная возможность — $k = n-1$. Этим завершается доказательство того, что условий (i) и (ii) достаточно, чтобы установить, является ли число n простым.

В упр. 10 показано, что для каждого из простых p можно использовать различные значения x, а n все еще будет оставаться простым числом. Можно ограничиться этими соображениями относительно простых чисел для x, поскольку порядок произведения uv по модулю n делит наименьшее общее кратное порядков u и v согласно результатам упр. 3.2.1.2–15. Соблюдение условий (i) и (ii) можно эффективно проверить при помощи быстрых методов вычисления степеней чисел, которые рассматриваются в разделе 4.6.3. Но необходимо знать простые множители числа $n-1$, поэтому возникает интересная ситуация, когда разложение на простые множители числа n зависит от разложения на простые множители числа $n-1$.

Пример. Разложение на простые множители достаточно большого числа помогает уяснить идеи, рассмотренные до сих пор. Попробуем найти простые множители 65-разрядного числа $2^{214}+1$. Проявив некоторую сообразительность, процесс разложения на простые множители можно начать, приняв во внимание интересное свойство исходного числа

$$2^{214}+1=(2^{107}-2^{54}+1)(2^{107}+2^{54}+1); \tag{15}$$

это частный случай разложения $4x^4+1=(2x^2+2x+1)(2x^2-2x+1)$, о котором Эйлер сообщал Гольдбаху (Goldbach) в 1742 году [P. H. Fuss, *Correspondance Math. et Physique* **1** (1843), 145]. Задача заключается в исследовании каждого из 33-разрядных множителей в соотношении (15).

Компьютерная программа легко обнаруживает, что $2^{107}-2^{54}+1=5\cdot 857\cdot n_0$, где

$$n_0=37\,866\,809\,061\,660\,057\,264\,219\,253\,397 \tag{16}$$

есть 29-разрядное число, не имеющее ни одного простого множителя, меньшего 1 000. Вычисления с многократной точностью, выполняемые при помощи алгоритма 4.6.3A, показывают, что

$$3^{n_0-1} \bmod n_0 = 1,$$

так что есть основание предполагать, что n_0 — простое число. Конечно, не может быть и речи о том, чтобы для проверки, является ли n_0 простым числом, проанализировать 10 миллионов миллионов или около того возможных делителей, но рассмотренный выше метод вполне пригоден для такой проверки. Следующая задача — разложение на простые множители числа n_0-1. Преодолев некоторые трудности, компьютер сообщит, что

$$n_0-1=2\cdot 2\cdot 19\cdot 107\cdot 353\cdot n_1, \qquad n_1=13\,191\,270\,754\,108\,226\,049\,301.$$

Здесь $3^{n_1-1} \bmod n_1 \neq 1$, поэтому n_1 — не простое число. Продолжая выполнение алгоритма A или B, можно получить следующие множители:

$$n_1=91\,813\cdot n_2, \qquad n_2=143\,675\,413\,657\,196\,977.$$

Теперь $3^{n_2-1} \bmod n_2 = 1$, поэтому можно попытаться доказать, что n_2 — простое число. Приняв во внимание, что множители $<1\,000$, получаем

$$n_2-1=2\cdot 2\cdot 2\cdot 2\cdot 3\cdot 3\cdot 547\cdot n_3,$$

где $n_3=1\,824\,032\,775\,457$. Так как $3^{n_3-1} \bmod n_3 \neq 1$, приходим к заключению, что n_3 — составное число, а при помощи алгоритма A находим, что $n_3=1\,103\cdot n_4$, где $n_4=1\,653\,701\,519$. Число n_4 ведет себя, как простое (т. е. $3^{n_4-1} \bmod n_4 = 1$), поэтому

$$n_4-1=2\cdot 7\cdot 19\cdot 23\cdot 137\cdot 1\,973.$$

Итак, выполнено первое полное разложение на простые множители. Теперь мы готовы вернуться к предыдущей подзадаче, а именно — к доказательству того, что n_4

есть простое число. Воспользовавшись процедурой, предложенной в упр. 10, вычислим следующие значения.

x	p	$x^{(n_4-1)/p} \bmod n_4$	$x^{n_4-1} \bmod n_4$
2	2	1	(1)
2	7	766 408 626	(1)
2	19	332 952 683	(1)
2	23	1 154 237 810	(1)
2	137	373 782 186	(1)
2	1 973	490 790 919	(1)
3	2	1	(1)
5	2	1	(1)
7	2	1 653 701 518	1

(17)

(Здесь "(1)" означает результат, равный 1, который не требуется вычислять, так как он может быть выведен из предыдущих вычислений.) Таким образом, n_4 — простое число, а число $n_2 - 1$ полностью разложено на простые множители. Аналогичные вычисления показывают, что и n_2 является простым числом; такое же полное разложение числа $n_0 - 1$ на простые множители показывает наконец, после вычислений еще и значений из табл. (17), что n_0 — простое число.

Последние три строки в табл. (17) представляют процесс поиска целого числа x, удовлетворяющего соотношению $x^{(n_4-1)/2} \not\equiv x^{n_4-1} \equiv 1$ (по модулю n_4). Если n_4 — простое число, то шансы на успех равны только 50/50, так что случай, когда $p = 2$, является, как правило, наиболее трудным для проверки. Можно обойти эту часть вычислений, используя закон квадратичной взаимности (упр. 23), который гласит, например, что $5^{(q-1)/2} \equiv 1$ (по модулю q), когда q есть простое число ± 1 (по модулю 5). Выбрав значение $x = 5$, нельзя убедиться в том, что число n_4 простое. Это сразу показывает вычисление $n_4 \bmod 5$. Тем не менее из результата упр. 26 действительно следует, что при проверке, является ли число n простым, вовсе нет необходимости рассматривать случай, когда $p = 2$, несмотря на то что $n-1$ делится на большие степени 2. Таким образом, три последние строки в табл. (17) необязательны.

Следующая величина, подлежащая разложению на простые множители, — другая часть соотношения (15), т. е.

$$n_5 = 2^{107} + 2^{54} + 1.$$

Поскольку $3^{n_5-1} \bmod n_5 \neq 1$, известно, что n_5 не является простым числом, и выполнение алгоритма B показывает, что

$$n_5 = 843\,589 \cdot n_6, \quad \text{где } n_6 = 92\,343\,993\,140\,277\,293\,096\,491\,917.$$

К сожалению, $3^{n_6-1} \bmod n_6 \neq 1$, поэтому остается 27-разрядное непростое число. Повторное обращение к алгоритму B могло бы истощить наше терпение (но не бюджет — мы чаще тратим свободное время в выходные дни, чем "рабочее время"). Пришло время ввести в действие метод решета. Алгоритм D, реализующий этот метод, может разбить число n_6 на два множителя:

$$n_6 = 8\,174\,912\,477\,117 \cdot 23\,528\,569\,104\,401.$$

(Применение алгоритма B тоже привело бы к положительному результату, но после выполнения 6 432 966 итераций.) При помощи алгоритма A *не удалось бы* получить этот результат за приемлемое время.

Теперь вычисления завершены: разложение числа $2^{214}+1$ на простые множители имеет вид

$$5 \cdot 857 \cdot 843\,589 \cdot 8\,174\,912\,477\,117 \cdot 23\,528\,569\,104\,401 \cdot n_0,$$

где n_0 представляет собой 29-разрядное простое число, приведенное в (16). В этих вычислениях нам сопутствовала определенная доля удачи, поскольку, если бы вычисления начались не с известного разложения на простые множители по уравнению (15), вполне возможно, что мы сначала получили бы малые множители, сведя n к $n_6 n_0$. А это 55-разрядное число намного сложнее разложить на простые множители — применять алгоритм D бесполезно, а алгоритм B должен был бы работать бесконечно долго из-за необходимости выполнять операции с высокой точностью.

Десятки дополнительных примеров разложения чисел на простые множители можно найти в статье Джона Бриллхарта (John Brillhart) и Дж. Л. Селфриджа (J. L. Selfridge) в журнале *Math. Comp.* **21** (1967), 87–96.

Усовершенствованные методики проверки принадлежности чисел к простым. Описанная выше процедура требует полного разложения числа $n-1$ на простые множители, прежде чем можно будет доказать, что число n — простое. Поэтому при работе с большими числами можно просто увязнуть в вычислениях. В упр. 15 описана другая процедура, использующая разложение на простые множители числа $n+1$. Если окажется, что разложение числа $n-1$ слишком затруднительно, то может статься, проще будет разложить на простые множители число $n+1$.

При работе с большими числами можно добиться существенного усовершенствования способов проверки. Например, нетрудно доказать более строгое обратное утверждение теоремы Ферма, которое требует только частичного разложения числа $n-1$. Из упр. 26 следует, что можно избежать большей части вычислений в (17); для доказательства того, что число n_4 является простым, достаточно выполнения трех условий:

$$2^{n_4-1} \bmod n_4 = \gcd(2^{(n_4-1)/23}-1,\, n_4) = \gcd(2^{(n_4-1)/1973}-1,\, n_4) = 1.$$

Бриллхарт, Лемер и Селфридж разработали метод, работающий с числами $n-1$ и $n+1$, которые имеют только частичные разложения на простые множители [*Math. Comp.* **29** (1975), 620–647, Corollary 11]. Предположим, что $n-1 = f^- r^-$ и $n+1 = f^+ r^+$, где известно полное разложение на простые множители чисел f^- и f^+; известно также, что все множители чисел r^- и $r^+ \geq b$. Если произведение $\big(b^3 f^- f^+ \max(f^-, f^+)\big)$ больше $2n$, то небольшого объема дополнительных вычислений, описанных в этой работе, достаточно для ответа на вопрос, является ли число n простым. Поэтому принадлежность чисел длиной вплоть до 35 разрядов может быть в доли секунды проверена путем выделения из $n \pm 1$ всех простых множителей $< 30\,030$ [см. J. L. Selfridge, M. C. Wunderlich, *Congressus Numerantium* **12** (1974), 109–120]. Для дальнейшего усовершенствования этого метода может быть использовано частичное разложение на простые множители других величин вида $n^2 \pm n + 1$ и n^2+1 [см. H. C. Williams, J. S. Judd, *Math. Comp.* **30** (1976), 157–172, 867–886].

На практике, когда n не содержит малых простых множителей и $3^{n-1} \bmod n = 1$, последующие вычисления почти всегда показывают, что n — простое число. (Одним из редких исключений в практике автора является число $n = \frac{1}{7}(2^{28} - 9) = 2\,341 \cdot 16\,381$.) С другой стороны, некоторые значения числа n, не являющиеся простыми, представляют собой весьма тяжелый случай для рассмотренных способов проверки, так как может случиться, что $x^{n-1} \bmod n = 1$ для всех x, взаимно простых с n (упр. 9). Наименьшим таким числом является $n = 3 \cdot 11 \cdot 17 = 561$; здесь $\lambda(n) = \operatorname{lcm}(2, 10, 16) = 80$ в обозначениях уравнения 3.2.1.2–(9), так что $x^{80} \bmod 561 = 1 = x^{560} \bmod 561$, когда x есть взаимно простое с 561. При попытке показать, что такое число n простое, наша процедура будет терпеть неудачу всякий раз, когда мы будем сталкиваться с одним из делителей этого числа. Метод можно усовершенствовать, если найти способ быстрого определения "непростоты" непростого числа n даже в таких патологических случаях.

Гарантируется, что следующая удивительно простая процедура выполняет анализ с высокой вероятностью.

Алгоритм P (*Вероятностная проверка, является ли число простым*). Этот алгоритм пытается определить, является ли данное целое нечетное число n простым. Несколько повторных попыток выполнения алгоритма, как объяснено в примечаниях ниже, позволяют с весьма большой вероятностью считать число n простым, хотя строгим это доказательство не является. Пусть $n = 1 + 2^k q$, где q — нечетное число.

P1. [Генерировать x.] Пусть x — случайное целое число в интервале $1 < x < n$.

P2. [Возвести в степень.] Присвоить $j \leftarrow 0$ и $y \leftarrow x^q \bmod n$. (Как и в предыдущем примере проверки, будет ли число простым, $x^q \bmod n$ должно быть вычислено за $O(\log q)$ шагов; см. раздел 4.6.3.)

P3. [Выполнено?] (Теперь $y = x^{2^j q} \bmod n$.) Если $j = 0$ и $y = 1$ или если $y = n - 1$, то выполнение алгоритма завершается и выдается сообщение "n, вероятно, простое". Если $j > 0$ и $y = 1$, перейти к шагу P5.

P4. [Увеличить j.] Увеличить j на 1. Если $j < k$, то присвоить $y \leftarrow y^2 \bmod n$ и возвратиться к шагу P3.

P5. [Не простое.] Завершить выполнение алгоритма сообщением "n, определенно, не простое". ▮

Отличительным свойством алгоритма P является то, что, если $x^q \bmod n \neq 1$ и $n = 1 + 2^k q$ — простое число, последовательность значений

$$x^q \bmod n, \quad x^{2q} \bmod n, \quad x^{4q} \bmod n, \quad \ldots, \quad x^{2^k q} \bmod n$$

будет заканчиваться 1 и значение, предшествующее первому появлению 1, будет равно $n - 1$. (Единственными решениями уравнения $y^2 \equiv 1$ (по модулю p) будут $y \equiv \pm 1$, когда p — простое, поскольку $(y - 1)(y + 1)$ должно быть кратным p.)

В упр. 22 доказывается основополагающее свойство алгоритма P, состоящее в том, что для любого n он будет давать неправильный результат не более чем в одном случае из четырех. На практике алгоритм P очень редко выдает неправильный результат для большинства значений n. Однако критическим является тот факт, что вероятность отказа ограничена *независимо* от значений n.

Допустим, что мы обращаемся к алгоритму P несколько раз, выбирая x независимо и случайно, когда выполняется шаг P1. Если алгоритм сообщает, что число n непростое, можно быть уверенным, что так оно и есть. Но если алгоритм сообщает 25 раз подряд, что n "возможно, простое", значит, n — "почти наверное простое", так как вероятность того, что процедура, выполняющаяся 25 раз подряд, дает неправильную информацию об исходном числе, меньше $(1/4)^{25}$. Эта величина меньше, чем один шанс на квадрильон; даже после проверки при помощи такой процедуры миллиарда различных чисел ожидаемое количество ошибок будет меньше $\frac{1}{1000000}$. Скорее всего, можно предположить, что компьютер потерял бит при вычислениях из-за аппаратных неполадок или влияния космической радиации, чем то, что алгоритм P непрерывно ошибается!

Вероятностные алгоритмы, подобные описанному, приводят нас к традиционному вопросу о достоверности. *Нуждаемся* ли мы в строгом доказательстве принадлежности числа к простым? Для тех, кто не склонен игнорировать традиционные понятия доказательства, Гэри Л. Миллер (Gary L. Miller) продемонстрировал (в немного ослабленном виде), что если определенное, хорошо известное предположение в теории чисел, называемое обобщенной гипотезой Римана, может быть доказано, то либо n — простое число, либо существует число $x < 2(\ln n)^2$, такое, что алгоритм P обнаружит, что n — не простое число. [Обзор различных обобщений гипотезы Римана приведен в *J. Comp. System Sci.* **13** (1976), 300–317. Присутствующая в этой оценке постоянная 2 получена Эриком Бахом (Eric Bach), *Math. Comp.* **55** (1990), 355–380; см. гл. 8 в *Algorithmic Number Theory* **1** by E. Bach, J. O. Shallit (MIT Press, 1996).]

Таким образом, можно найти строгий способ проверки, будет ли число простым, выполняемый за $O(\log n)^5$ элементарных операций, в противовес вероятностному методу, время выполнения которого при условии, что доказаны обобщенные гипотезы Римана, равно $O(\log n)^3$. Вполне резонно задать вопрос: дают ли доказанные гипотезы такой же надежный результат, какой дает неоднократное применение к случайным числам алгоритма P.

Вероятностная проверка принадлежности числа к простым была впервые предложена в 1974 году Р. Соловеем (R. Solovay) и В. Штрассеном (V. Strassen), разработавшими интересный, но более сложный алгоритм проверки, описанный в упр. 23, (b). [См. *SICOMP* **6** (1977), 84–85; **7** (1978), 118.] Алгоритм P является упрощенным вариантом процедуры, разработанной М. О. Рабином (M. O. Rabin), в основу которого частично положены идеи Гэри Л. Миллера [см. *Algorithms and Complexity*, edited by J. F. Traub (Academic Press, 1976), 35–36]. Исследования, выполненные Б. Арази (B. Arazi) [*Comp. J.* **37** (1994), 219–222], показали, что можно существенно повысить скорость выполнения алгоритма P для больших n за счет применения быстрого метода вычисления остатков Монтгомери (см. упр. 4.3.1–41).

В 1980 году Леонард М. Эдлеман (Leonard M. Adleman) предложил совершенно другой подход к решению этой задачи. Его очень интересный метод основан на теории алгебраических целых чисел, но это выходит за рамки данной книги. Метод Эдлемана приводит к невероятностной процедуре, которая определяет, будет ли любое число длиной, скажем, 250 знаков простым не более чем за несколько часов. [В общем случае время, необходимое для решения задачи по этому методу, равно $(\log n)^{O(\log\log\log n)}$; см. L. M. Adleman, C. Pomerance, R. S. Rumely, *Annals of*

Math. **117** (1983), 173–206.] Х. В. Ленстра (мл.) (H. W. Lenstra, Jr.) усовершенствовал этот метод, благодаря чему он стал выполняться еще быстрее, что подтвердила практика применения метода Г. Кохеном (H. Cohen) и А. К. Ленстрой (A. K. Lenstra) [*Math. Comp.* **42** (1984), 297–330; **48** (1987), 103–121].

Позже Эдлеман и Минг-Дех А. Хуанг (Ming-Deh A. Huang) предложили способ строгого доказательства принадлежности числа к простому для всех простых n. С высокой вероятностью время выполнения этого алгоритма пропорционально $\log n$ [*Lecture Notes in Math.* **1512** (1992)]. Однако их метод, кажется, представляет только чисто теоретический интерес.

Разложение на простые множители при помощи цепных дробей. Камнем преткновения для методов разложения на простые множители, рассмотренных выше, является разложение чисел, содержащих 30 и более разрядов, поэтому для их разложения нужна другая идея, которая позволила бы идти дальше в этом направлении. К счастью, такая идея существует; точнее, есть две такие идеи, предложенные А. М. Лежандром (A. M. Legendre) и М. Крайчиком (M. Kraitchik). Основываясь на них, Д. Г. Лемер и Р. Е. Пауэрс (R. E. Powers) много лет назад разработали новый подход к решению этой задачи [*Bull. Amer. Math. Soc.* **37** (1931), 770–776]. Однако некоторое время данный метод не применялся из-за трудностей, связанных с его реализацией на настольных арифмометрах. Негативное мнение об этом методе было преодолено к концу 60-х годов, когда Джон Бриллхарт (John Brillhart) обнаружил, что приближение Лемера-Пауэрса можно "воскресить" благодаря тому, что метод очень хорошо подходит для компьютерного программирования. Действительно, чуть позже он вместе с Майклом А. Моррисоном (Michael A. Morrison) разработал алгоритм, ставший в 70-е годы непревзойденным средством разложения на простые множители с использованием формата многократной точности. Их программа обрабатывала на компьютере IBM 360/91 произвольные 25-разрядные числа за 30 с, а 40-разрядные числа — за 50 мин [см. *Math. Comp.* **29** (1975), 183–205]. Первый триумфальный успех был достигнут в 1970 году, когда был продемонстрирован результат: $2^{128} + 1 = 59\,649\,589\,127\,497\,217 \cdot 5\,704\,689\,200\,685\,129\,054\,721$.

Основная идея заключается в том, чтобы найти числа x и y , такие, что

$$x^2 \equiv y^2 \pmod{N}, \quad 0 < x, y < N, \quad x \neq y, \quad x + y \neq N. \tag{18}$$

Метод Ферма накладывает более строгое условие $x^2 - y^2 = N$, но в действительности соответствия (18) достаточно для того, чтобы разложить число N на множители. Из (18) следует, что число N есть делитель числа $x^2 - y^2 = (x - y)(x + y)$, но число N не делит ни $x - y$, ни $x + y$; следовательно, $\gcd(N, x - y)$ и $\gcd(N, x + y)$ являются делителями числа N, которые могут быть найдены при помощи эффективных алгоритмов, рассмотренных в разделе 4.5.2.

Один из способов нахождения решения уравнения (18) — найти такие значения x, при которых $x^2 \equiv a \pmod{N}$ для малых значений $|a|$. В дальнейшем будет показано, что зачастую, чтобы получить решения (18), достаточно просто объединить решения приведенного выше уравнения. Далее, если $x^2 = a + kNd^2$ для некоторых k и d при малом значении $|a|$, дробь x/d является хорошим приближением к $\sqrt{kN}$; и обратно, если x/d — очень хорошее приближение к $\sqrt{kN}$, то $|x^2 - kNd^2|$ будет мало́. Сказанное наводит на мысль о необходимости рассмотреть разложение

в цепную дробь числа $\sqrt{kN}$, поскольку, как видно из соотношения 4.5.3–(12) и упр. 4.5.3–42, цепные дроби дают хорошие рациональные приближения.

Для квадратичных иррациональностей цепные дроби обладают многими приятными свойствами, как было доказано в упр. 4.5.3–12. В приводимом ниже алгоритме эти свойства используются для получения решений уравнения

$$x^2 \equiv (-1)^{e_0} p_1^{e_1} p_2^{e_2} \dots p_m^{e_m} \pmod{N}. \tag{19}$$

Здесь используется фиксированное множество простых чисел $p_1 = 2$, $p_2 = 3$, ... вплоть до p_m. Из этого множества выбираются только такие простые числа p, для которых либо $p = 2$, либо $(kN)^{(p-1)/2} \bmod p \le 1$, поскольку другие простые числа никогда не будут множителями чисел, порождаемых этим алгоритмом (упр. 14). Если $(x_1, e_{01}, e_{11}, \dots, e_{m1})$, ..., $(x_r, e_{0r}, e_{1r}, \dots, e_{mr})$ — решения (19), такие, что в векторной сумме

$$(e_{01}, e_{11}, \dots, e_{m1}) + \cdots + (e_{0r}, e_{1r}, \dots, e_{mr}) = (2e'_0, 2e'_1, \dots, 2e'_m) \tag{20}$$

каждый компонент *четный*, то

$$x = (x_1 \dots x_r) \bmod N, \qquad y = \big((-1)^{e'_0} p_1^{e'_1} \dots p_m^{e'_m}\big) \bmod N \tag{21}$$

дает решение уравнения (18) за исключением случая, когда $x \equiv \pm y$. Условие (20) равнозначно утверждению, что рассматриваемые векторы линейно зависимы по модулю 2, поэтому, если найдено по меньшей мере $m + 2$ решений (19), должно существовать хотя бы одно решение, соответствующее (20).

Алгоритм Е (*Разложение на простые множители при помощи цепных дробей*). По данным положительному целому числу N и положительному целому числу k, такому, что число kN не является точным квадратом, этот алгоритм предпринимает попытку найти решения уравнения (19) для данной последовательности простых чисел p_1, ..., p_m, исследуя сходимость цепной дроби для $\sqrt{kN}$. (Другой алгоритм, использующий для поиска множителей числа N выходные данные (алгоритма Е. — *Прим. перев.*), является предметом рассмотрения в упр. 12.)

Е1. [Начальная установка.] Присвоить $D \leftarrow kN$, $R \leftarrow \lfloor\sqrt{D}\rfloor$, $R' \leftarrow 2R$, $U \leftarrow U' \leftarrow R'$, $V \leftarrow 1$, $V' \leftarrow D - R^2$, $P \leftarrow R$, $P' \leftarrow 1$, $A \leftarrow 0$, $S \leftarrow 0$. (Следуя основной процедуре упр. 4.5.3–12, алгоритм находит разложение в цепную дробь числа $\sqrt{kN}$. Переменные U, U', V, V', P, P', A и S представляют $R + U_n$, $R + U_{n-1}$, V_n, V_{n-1}, $p_n \bmod N$, $p_{n-1} \bmod N$, A_n и $n \bmod 2$ соответственно. Всегда будет выполняться неравенство $0 < V \le U \le R'$, поэтому самая высокая точность потребуется только для нахождения чисел P и P'.)

Е2. [Продвинуть U, V, S.] Присвоить $T \leftarrow V$, $V \leftarrow A(U' - U) + V'$, $V' \leftarrow T$, $A \leftarrow \lfloor U/V \rfloor$, $U' \leftarrow U$, $U \leftarrow R' - (U \bmod V)$, $S \leftarrow 1 - S$.

Е3. [Разложить на множители V.] (В соответствии с результатами упр. 4.5.3–12(c) здесь имеем $P^2 - kNQ^2 = (-1)^S V$.) Присвоить $(e_0, e_1, \dots, e_m) \leftarrow (S, 0, \dots, 0)$, $T \leftarrow V$. Теперь нужно выполнить следующее. Для $1 \le j \le m$, если $T \bmod p_j = 0$, присвоить $T \leftarrow T/p_j$ и $e_j \leftarrow e_j + 1$ и повторять этот процесс до тех пор, пока не получится $T \bmod p_j \neq 0$.

Таблица 1

ПРИМЕР ВЫПОЛНЕНИЯ АЛГОРИТМА Е
$N = 197\,209$, $k = 1$, $m = 3$, $p_1 = 2$, $p_2 = 3$, $p_3 = 5$

	U	V	A	P	S	T	Выход
После E1:	888	1	0	444	0	—	
После E4:	876	73	12	444	1	73	
После E4:	882	145	6	5 329	0	29	
После E4:	857	37	23	32 418	1	37	
После E4:	751	720	1	159 316	0	1	$159\,316^2 \equiv +2^4 \cdot 3^2 \cdot 5^1$
После E4:	852	143	5	191 734	1	143	
После E4:	681	215	3	131 941	0	43	
После E4:	863	656	1	193 139	1	41	
После E4:	883	33	26	127 871	0	11	
После E4:	821	136	6	165 232	1	17	
После E4:	877	405	2	133 218	0	1	$133\,218^2 \equiv +2^0 \cdot 3^4 \cdot 5^1$
После E4:	875	24	36	37 250	1	1	$37\,250^2 \equiv -2^3 \cdot 3^1 \cdot 5^0$
После E4:	490	477	1	93 755	0	53	

E4. [Решение?] Если $T = 1$, вывести значения $(P, e_0, e_1, \ldots, e_m)$, которые составляют решение уравнения (19). (Если получено достаточное число решений, то на этом шаге алгоритм может быть завершен.)

E5. [Продвинуть P, P'.] Если $V \neq 1$, присвоить $T \leftarrow P$, $P \leftarrow (AP + P') \bmod N$, $P' \leftarrow T$ и возвратиться к шагу E2. В противном случае процесс разложения в цепную дробь начинает повторять цикл сначала, за возможным исключением S, и поэтому выполнение алгоритма на этом завершается. (Обычно данный цикл настолько продолжителен, что завершение не происходит.) ▮

В качестве примера применения алгоритма Е к сравнительно малым числам рассмотрим случай, когда $N = 197\,209$, $k = 1$, $m = 3$, $p_1 = 2$, $p_2 = 3$, $p_3 = 5$. Начальная часть вычислений представлена в табл. 1.

Продолжение вычислений приводит к получению за первые 100 итераций 25 выходных данных; другими словами, алгоритм находит решения очень быстро, но некоторые из них тривиальны. Например, если продолжить вычисления еще на 14 шагов, можно получить $197\,197^2 \equiv 2^4 \cdot 3^2 \cdot 5^0$ выходных данных, которые не представляют интереса, поскольку $197\,197 \equiv -12$. Для завершения процесса разложения на простые множители достаточно первых двух решений, полученных выше. Так как получено

$$(159\,316 \cdot 133\,218)^2 \equiv (2^2 \cdot 3^3 \cdot 5^1)^2 \text{ (по модулю 197209)},$$

соотношение (18) выполняется при $x = (159\,316 \cdot 133\,218) \bmod 197\,209 = 126\,308$, $y = 540$. В соответствии с алгоритмом Евклида $\gcd(126\,308 - 540, 197\,209) = 199$, и получаем изящное разложение

$$197\,209 = 199 \cdot 991.$$

Чтобы понять причины, по которым алгоритм Е столь успешно выполняет разложение больших чисел на простые множители, рассмотрим эвристический анализ времени выполнения алгоритма Е, следуя неопубликованной идее, высказанной

автору в 1975 году Р. Шруппелем (R. Schroeppel). Положим для определенности, что $k = 1$. Число выходных данных, необходимых для получения разложения числа N на простые множители, пропорционально m — числу выделенных в процессе вычислений малых простых чисел. Каждый раз на выполнение шага E3 затрачивается порядка $m \log N$ единиц времени, так что общее время выполнения в первом приближении пропорционально величине $m^2 \log N/P$, где P — вероятность получения очередного результата на каждой итерации. Предположим, что в течение всего времени выполнения алгоритма величина V равномерно распределена в интервале от 0 до $2\sqrt{N}$. Тогда вероятность P равна значению $(2\sqrt{N})^{-1}$, умноженному на количество целых чисел $< 2\sqrt{N}$, для которых все простые множители принадлежат множеству $\{p_1, \ldots, p_m\}$. В упр. 29 приведена нижняя граница для P, из которой можно заключить, что наибольшее время выполнения алгоритма имеет порядок

$$\frac{2\sqrt{N}\, m^2 \log N}{m^r/r!}, \qquad \text{где } r = \left\lfloor \frac{\log 2\sqrt{N}}{\log_m} \right\rfloor. \tag{22}$$

Если положить $\ln m$ равным приблизительно $\frac{1}{2}\sqrt{\ln N \ln\ln N}$, то предположив, что $p_m = O(m \log m)$, получим $r \approx \sqrt{\ln N/\ln\ln N} - 1$, вследствие чего формула (22) упрощается и принимает вид

$$\exp\bigl(2\sqrt{(\ln N)(\ln\ln N)} \;+\; O\bigl((\log N)^{1/2}(\log\log N)^{-1/2}(\log\log\log N)\bigr)\bigr).$$

Следовательно, при надлежащих предположениях ожидаемое время выполнения алгоритма E не должно превышать величины $N^{\epsilon(N)}$, где $\epsilon(N) \approx 2\sqrt{\ln\ln N/\ln N}$ стремится к 0 при $N \to \infty$.

Когда число N находится в интервале, чаще всего используемом на практике, нужно быть осторожным и не принимать всерьез такую асимптотическую оценку. Например, если $N = 10^{50}$, получим $N^{1/\alpha} = (\lg N)^\alpha$ при $\alpha \approx 4.75$, но то же самое соотношение справедливо и для $\alpha \approx 8.42$, когда $N = 10^{200}$. Функция $N^{\epsilon(N)}$ имеет порядок роста, который представляет собой некоторого рода пересечение $N^{1/\alpha}$ и $(\lg N)^\alpha$, но все три равенства практически одинаковы для чисел N, не являющихся недопустимо большими. Многочисленные вычислительные эксперименты, выполненные М. Ч. Вундерлихом (M. C. Wunderlich), показали, что хорошо настроенный вариант алгоритма E значительно лучше выполняет разложение, чем предусматривается этой оценкой [см. *Lecture Notes in Math.* **751** (1979), 328–342]; несмотря на то что для $N = 10^{50}$ имеет место $2\sqrt{\ln\ln N/\ln N} \approx .41$, при разложении на простые множители тысяч чисел в интервале $10^{13} \le N \le 10^{42}$ он получил выражение для времени выполнения, равное приблизительно $N^{0.15}$.

Попытка выполнить разложение числа N с помощью алгоритма E начинается, по существу, с замены N на kN, что выглядит довольно любопытно (если не откровенно глупо). "Простите, как вы смотрите на то, что перед попыткой разложения числа я умножу его на 3?" Тем не менее идея выглядит привлекательной, поскольку некоторые значения k делают числа V потенциально делимыми на меньшие простые числа, и поэтому на шаге E3 полное разложение этих чисел на простые множители может быть выполнено проще. С другой стороны, большое значение числа k сделает числа V больше, и тогда их полное разложение на простые множители будет затруднено. Нужно сбалансировать эти тенденции путем соответствующего подбора числа k. Рассмотрим, например, делимость чисел V на числа, равные степеням 5.

На шаге E3 получаем $P^2 - kNQ^2 = (-1)^S V$, так что, если 5 делит v, имеем $P^2 \equiv kNQ^2$ (по модулю 5). В данном соответствии число Q не может быть кратным 5, поэтому оно взаимно просто с P и можно записать $(P/Q)^2 \equiv kN$ (по модулю 5). Если предположить, что P и Q — случайные взаимно простые числа, такие, что 24 возможные пары чисел $(P \bmod 5, Q \bmod 5) \neq (0,0)$ равновероятны, вероятность того, что 5 делит V, равна, таким образом, $\frac{4}{24}$, $\frac{8}{24}$, 0, 0 или $\frac{8}{24}$ в зависимости от того, какие значения принимает число $kN \bmod 5$: 0, 1, 2, 3 или 4. Аналогично вероятность того, что 25 делит V, равна 0, $\frac{40}{600}$, 0, 0, $\frac{40}{600}$ соответственно, если только kN не кратно 25. В общем случае для данного нечетного простого числа p, для которого $(kN)^{(p-1)/2} \bmod p = 1$, получаем, что V кратно p^e с вероятностью $2/\big(p^{e-1}(p+1)\big)$; среднее число p делителей числа V приближается к $2p/(p^2-1)$. Из этого анализа, выполненного Р. Шруппелем (R. Schroeppel), следует, что лучший выбор числа k — это значение, которое обеспечивает максимум функции

$$\sum_{j=1}^{m} f(p_j, kN) \log p_j - \frac{1}{2} \log k, \tag{23}$$

где f определена в упр. 28, поскольку, по существу, это ожидаемое значение величины $\ln(\sqrt{N}/T)$ при переходе к шагу E4.

Если числа k и m хорошо подобраны, выполнение разложения по алгоритму E дает еще лучшие результаты. Выбор подходящего числа m может быть выполнен экспериментально, так как проведенный выше анализ асимптотического поведения носит незавершенный характер и нельзя быть уверенным в точности полученной информации. Поэтому внесение в алгоритм многочисленных уточнений, связанных с таким выбором m, может привести к непредсказуемым последствиям. Например, сравнив выполнение шага E3 с выполнением алгоритма A, можно получить следующее важное усовершенствование. Суть его в том, что процесс разложения на простые множители можно остановить после нахождения $T \bmod p_j \neq 0$ и $\lfloor T/p_j \rfloor \leq p_j$, поскольку в этом случае T будет либо равно 1, либо простым числом. Если T — простое число, большее p_m (в таком случае оно будет не больше $p_m^2 + p_m - 1$), то, учитывая, что получено полное разложение на простые множители, можно вывести в качестве результата $(P, e_0, \ldots, e_m, T)$. На второй фазе алгоритма будут использованы только те выходные данные, для которых все простые числа T встретятся не менее двух раз. Эта модификация приводит к образованию намного большего списка простых чисел без дополнительных затрат времени. Эксперименты Вундерлиха показали, что если значения N близки к 10^{40} при $m \approx 150$, то алгоритм работает хорошо только с учетом этого уточнения.

Так как на выполнение шага E3 приходится бóльшая часть времени выполнения алгоритма, Моррисон, Бриллхарт и Шруппель предложили несколько способов прекращения выполнения этого шага, если результат становится правдоподобным: (a) когда значение T изменяется таким образом, что его можно представить с однократной точностью, выполнение этого шага продолжается только в том случае, когда $\lfloor T/p_j \rfloor > p_j$ и $3^{T-1} \bmod T \neq 1$; (b) если T оказывается все еще $> p_m^2$ после того, как выделены все множители $< \frac{1}{10} p_m$; (c) выделены только все множители, бóльшие p_5, для групп, скажем, из 100 или около того последовательных значений V; процесс разложения на простые множители может быть продолжен позже, но только для значений V из каждой группы, для которой получен наименьший оста-

ток T. (Прежде чем выделять множители, бóльшие p_5, полезно вычислить $V \bmod p_1^{f_1} p_2^{f_2} p_3^{f_3} p_4^{f_4} p_5^{f_5}$, где все f достаточно малы для того, чтобы модули $p_1^{f_1} p_2^{f_2} p_3^{f_3} p_4^{f_4} p_5^{f_5}$ можно было представлять с однократной точностью, но достаточно велики, чтобы выполнялись равенства $V \bmod p_i^{f_i+1} = 0$. Поэтому один остаток с однократной точностью будет характеризовать значение V по модулю пяти малых простых чисел.)

По поводу оценки длины цикла выходных данных алгоритма E обратитесь к работе H. C. Williams, *Math. Comp.* **36** (1981), 593–601.

***Теоретическая верхняя граница.** С точки зрения вычислительной сложности алгоритма желательно знать, существует ли метод разложения на простые множители, ожидаемое время выполнения которого может быть ограничено оценкой $O(N^{\epsilon(N)})$, где $\epsilon(N) \to 0$ при $N \to \infty$.

Выше было показано, что поведение алгоритма E, вероятно, удовлетворяет такой оценке, однако желательно было бы найти строгое доказательство этого факта, так как цепные дроби не достаточно хорошо упорядочены. Первое доказательство того, что существует такой алгоритм разложения на простые множители, нашел в 1978 году Джон Диксон (John Dixon). Он показал, что в действительности достаточно рассматривать упрощенный вариант алгоритма E, из которого убирается аппарат цепных дробей, но основная идея соотношений (18) остается.

Метод Диксона [*Math. Comp.* **36** (1981), 255–260] состоит в следующем. Предположим, что для числа N существует по меньшей мере два различных простых множителя и это число N не делится на первые m простых чисел p_1, p_2, ..., p_m. Выберем случайное целое число X из интервала $0 < X < N$ и положим $V = X^2 \bmod N$. Если $V = 0$, то число $\gcd(X, N)$ является надлежащим множителем числа N. В противном случае, как и на шаге E3, выделяем все малые простые множители числа V. Другими словами, выражаем число V в виде

$$V = p_1^{e_1} \ldots p_m^{e_m} T, \tag{24}$$

где T не делится ни на одно из первых m простых чисел. Если $T = 1$, то выполнение алгоритма продолжается, как на шаге E4, и выводятся данные $(X, e_1, \ldots, e_m)$, которые представляют собой решение уравнения (19) при $e_0 = 0$. Этот процесс продолжается с новыми случайными значениями величины X до тех пор, пока не наберется достаточно много выходных данных для того, чтобы по методу упр. 12 обнаружить множитель для числа N.

При исследовании этого алгоритма нужно найти границы для (a) вероятности того, что случайное значение X приводит к выводу результата, и (b) вероятности того, что для нахождения множителя потребуется большое количество выходных данных. Пусть $P(m, N)$ — вероятность (a), т. е. вероятность того, что $T = 1$, если величина X выбирается случайной. После опробования M значений величины X получим в среднем $MP(m, N)$ выходных значений, и число выходных значений имеет биномиальное распределение, для которого среднеквадратичное отклонение меньше, чем квадратный корень из среднего значения. Вероятность (b) легко найти, воспользовавшись результатом упр. 13: при нахождении одного множителя алгоритму потребуется более $m + k$ выходных значений с вероятностью $\leq 2^{-k}$.

В упр. 30 доказывается, что

$$P(m, N) \geq m^r/(r!N)$$

в случае, когда $r = 2\lfloor \log N/(2\log p_m)\rfloor$, поэтому время выполнения алгоритма можно оценить почти так же, как в (22), но при этом величина $2\sqrt{N}$ должна быть заменена величиной N. На этот раз выбираем

$$r = \sqrt{2\ln N/\ln\ln N} + \theta,$$

где $|\theta| \le 1$ и r — четное число. Теперь значение m выбираем так, чтобы

$$r = \ln N/\ln p_m + O(1/\log\log N);$$

это означает, что

$$\begin{aligned}
\ln p_m &= \sqrt{\frac{\ln N\ln\ln N}{2}} - \frac{\theta}{2}\ln\ln N + O(1),\\
\ln m &= \ln\pi(p_m) = \ln p_m - \ln\ln p_m + O(1/\log p_m)\\
&= \sqrt{\frac{\ln N\ln\ln N}{2}} - \frac{\theta+1}{2}\ln\ln N + O(\log\log\log N),\\
\frac{m^r}{r!\,N} &= \exp\bigl(-\sqrt{2\ln N\ln\ln N} + O(r\log\log\log N)\bigr).
\end{aligned}$$

Выберем M таким, что $Mm^r/(r!N) \ge 4m$. Следовательно, ожидаемое количество выходных значений $MP(m,N)$ будет не меньше $4m$. Время выполнения алгоритма — порядка $Mm\log N$ плюс $O(m^3)$ шагов, что следует из результатов упр. 12; отсюда получаем, что $O(m^3)$ оказывается меньше $Mm\log N$, что равно

$$\exp\bigl(\sqrt{8(\ln N)(\ln\ln N)} \;+\; O\bigl((\log N)^{1/2}(\log\log N)^{-1/2}(\log\log\log N)\bigr)\bigr).$$

Вероятность того, что с помощью данного метода не удастся получить множитель, ничтожно мала, так как вероятность того, что вычислено меньше, чем $2m$ выходных значений (упр. 31), не превышает $e^{-m/2}$, в то время как вероятность того, что из числа первых $2m$ выходных значений не будет найдено ни одного множителя, не превышает 2^{-m} и $m \gg \ln N$. Доказана следующая несколько усиленная теорема Диксона.

Теорема D. *Существует алгоритм, время выполнения которого равно* $O(N^{\epsilon(N)})$, *где* $\epsilon(N) = c\sqrt{\ln\ln N/\ln N}$ *и* c — *произвольная константа, бóльшая, чем* $\sqrt{8}$, *и который находит нетривиальный множитель числа* N *с вероятностью* $1 - O(1/N)$ *в случае, когда для числа* N *существует по меньшей мере два простых делителя.* ▮

Другие подходы. Джон М. Поллард (John M. Pollard) предложил другой способ разложения на простые множители [*Proc. Cambridge Phil. Soc.* **76** (1974), 521–528], в котором дается практический способ нахождения простых множителей p числа N для случая, когда число $p-1$ не имеет больших простых множителей. Этот алгоритм (см. упр. 19) был, вероятно, первой попыткой решения поставленной задачи после того, как выяснилось, что алгоритмы A и B для больших чисел N выполняются слишком долго.

В обзорной работе, написанной Р. К. Ги (R. K. Guy) в соавторстве с Дж. Х. Конвеем (J. H. Conway) и опубликованной в *Congressus Numerantium* **16** (1976), 49–89, проведен анализ состояния проблемы на то время и перспектив разработки новых методов ее решения. Ги утверждал: "Я буду удивлен, если кто-либо в этом веке разложит на простые множители числа длиной 10^{80}, не оговаривая специальных

случаев"; ему действительно пришлось много раз удивляться в течение следующих 20-ти лет.

Значительные успехи в разработке способов разложения на простые множители были достигнуты в 80-е годы, начиная с *метода квадратичного просеивания* Карла Померанса (Carl Pomerance), разработанного им в 1981 году [см. *Lecture Notes in Comp. Sci.* **209** (1985), 169–182]. Затем Хендрик Ленстра (Hendrik Lenstra) разработал *метод эллиптических кривых* [*Annals of Math.* **126** (1987), 649–673]. Он эвристически предположил, что для нахождения простого множителя p необходимо выполнить около $\exp\bigl(\sqrt{(2+\epsilon)(\ln p)(\ln\ln p)}\,\bigr)$ операций умножения. Эта величина — не что иное, как асимптотический квадратный корень из оценки времени выполнения алгоритма E, когда $p \approx \sqrt{N}$, и она становится даже лучше, если число N имеет относительно малые простые множители. Прекрасное описание этого метода дано Джозефом Х. Силверманом (Joseph H. Silverman) и Джоном Тейтом (John Tate) в *Rational Points on Elliptic Curves* (New York: Springer, 1992), Chapter 4.

В 1988 году к решению этой задачи вернулся Джон Поллард. Он предложил новый подход, который стал позже известен как *решето числового поля*. С рядом работ, посвященных этому методу, который является в настоящее время чемпионом по разложению на простые множители чрезвычайно больших чисел, можно ознакомиться в *Lecture Notes in Math.* **1554** (1993). При использовании этого метода прогнозируемый порядок времени выполнения равен

$$\exp\bigl((64/9+\epsilon)^{1/3}(\ln N)^{1/3}(\ln\ln N)^{2/3}\bigr) \tag{25}$$

при $N \to \infty$. Согласно анализу, выполненному А. К. Ленстрой (A. K. Lenstra), порогом, после которого хорошо настроенный метод решета числового поля начинает превосходить хорошо настроенный метод квадратичного просеивания, является число $N \approx 10^{112}$.

Подробности этих методов выходят за рамки данной книги, но представление об их эффективности можно получить, обратив внимание на некоторые уже известные случаи неудачных попыток разложения на простые множители чисел Ферма вида $2^{2^k}+1$. Например, разложение

$$2^{512}+1 = 2\,424\,833\cdot 745\,560\,282\,564\,788\,4208\,337\,395\,736\,200\,454\,918\,783\,366\,342\,657\cdot p_{99}$$

было получено при помощи метода решета числового поля после вычислений в течение четырех месяцев, что заняло все свободное время почти 700 рабочих станций [Lenstra, Manasse, Pollard, *Math. Comp.* **61** (1993), 319–349; **64** (1995), 1357]; здесь p_{99} обозначает 99-разрядное простое число. Следующее число Ферма было в два раза длиннее предыдущего, но при помощи метода эллиптических кривых 20 октября 1995 года был получен результат:

$$2^{1\,024}+1 = 45\,592\,577\cdot 6487\,031\,809\cdot 4\,659\,775\,785\,220\,018\,543\,264\,560\,743\,076\,778\,192\,897\cdot p_{252}.$$

[Richard Brent, *Math. Comp.* **68** (1999), 429–451.] На самом деле Брент уже применял метод эллиптических кривых в 1988 году для решения следующей задачи:

$$2^{2\,048}+1 = 319\,489\cdot 974\,849\cdot 167\,988\,556\,341\,760\,475\,137\cdot 3\,560\,841\,906\,445\,833\,920\,513\cdot p_{564}.$$

Все простые множители, кроме одного, оказались, хотя и с некоторой долей везения, меньшими $< 10^{22}$, так что метод эллиптических кривых стал победителем.

А как насчет числа $2^{4\,096} + 1$? К настоящему моменту эта задача кажется слишком сложной для решения. Данное число содержит пять множителей $< 10^{16}$, но остаток, который не удается разложить на множители, содержит 1 187 десятичных разрядов. Еще один случай: число $2^{8\,192} + 1$ содержит четыре известных множителя $< 10^{27}$ и очень большой неразложенный остаток. [Crandall, Fagin, *Math. Comp.* **62** (1994), 321; Brent, Crandall, Dilcher, van Halewyn, **68** (1999), 429–451.]

Секретные множители. Всеобщий интерес к проблеме разложения на простые множители резко возрос в 1977 году, когда Р. Л. Ривест (R. L. Rivest), А. Шамир (A. Shamir) и Л. Эдлеман (L. Adleman) нашли способ кодирования сообщений, которые могут быть декодированы в явном виде только по известным множителям, на которые разлагается большое число N, даже в том случае, когда метод кодирования известен всем и каждому. Так как большинство величайших математиков мира не могут до сих пор найти эффективный метод разложения чисел на простые множители, этот метод [*CACM* **21** (1978), 120–126] позволяет почти гарантированно защитить засекреченные данные и сообщения в компьютерной сети.

Представим себе маленькое электронное устройство (назовем его *RSA-блоком*), в памяти которого хранятся два больших простых числа p и q. Будем считать, что числа $p - 1$ и $q - 1$ не делятся на 3. RSA-блок, подсоединенный каким-то образом к компьютеру, передал последнему произведение $N = pq$, поэтому ни один человек не может обнаружить значения чисел p и q, не разложив число N на простые множители. При попытке взлома RSA-блок запрограммирован на самоуничтожение. Другими словами, блок сотрет информацию в своей памяти, если будет предпринята попытка доступа к нему или он подвергнется воздействию внешнего излучения, которое может изменить или считать данные, хранящиеся в памяти. Кроме того, RSA-блок достаточно надежен с точки зрения эксплуатации; в случае его выхода из строя вследствие неполадок в системе питания или износа нужно выбросить имеющееся устройство и купить новое. Простые множители p и q генерируются самим RSA-блоком с помощью схемы, основанной на явлениях, которые случайны по самой своей природе (например, космические лучи). Важным является тот факт, что *никто не* знает простых чисел p и q, даже то лицо (или организация), которому принадлежит устройство или которое имеет доступ к нему. Не может быть и речи о подкупе, шантаже или захвате заложников с целью получения информации о простых числах числа N.

Чтобы отослать секретное сообщение владельцу RSA-блока, для которого произведение двух чисел равно N, нужно преобразовать сообщение в последовательность чисел $(x_1, \ldots, x_k)$, в которой каждое из чисел x_i принимает значения в интервале между 0 и N, а затем передать числа

$$(x_1^3 \bmod N,\ \ldots,\ x_k^3 \bmod N).$$

RSA-блок, зная p и q, может декодировать сообщение, поскольку в нем уже имеется вычисленное заранее число $d < N$, такое, что $3d \equiv 1$ (по модулю $(p-1)(q-1)$). Теперь RSA-блок может, используя рассмотренный в разделе 4.6.3 метод, за приемлемое время вычислить $(x^3)^d \bmod N = x$. Естественно, RSA-блок хранит это

магическое число d внутри себя; фактически можно было бы запоминать в RSA-блоке только число d вместо чисел p и q, поскольку в обязанности блока после вычисления числа N входит только защита своих секретов и вычисление кубических корней по модулю N.

Подобная схема кодирования не дает эффекта, если $x < \sqrt[3]{N}$, так как $x^3 \bmod N = x^3$, а кубический корень легко можно найти. Из описанного в разделе 4.2.4 логарифмического закона ведущих разрядов следует, что старшая позиция x_1 k-размерного сообщения $(x_1, \ldots, x_k)$ будет меньше $\sqrt[3]{N}$ примерно в $\frac{1}{3}$ случаев, поэтому данная проблема требует своего решения. В упр. 32 рассмотрен один из способов, позволяющих избежать этих сложностей.

Надежность схемы кодирования при помощи RSA базируется на том факте, что никто не может узнать, как вычислять кубический корень по модулю N, не зная множителей числа N. Похоже, что не существует методов, позволяющих это сделать, однако абсолютной уверенности нет. Все, что можно сказать по данному поводу, — это то, что обычные методы обнаружения кубических корней задачу не решают. Например, бесполезно пытаться вычислять число d как функцию числа N; причина в том, что если известно d или фактически если известно любое число m разумной длины, такое, что равенство $x^m \bmod N = 1$ справедливо для значительного количества величин x, то потребуется намного больше шагов для поиска множителей числа N (упр. 34). Таким образом, любой из методов, основанных на явной или скрытой попытке нахождения такого числа m, не может оказаться лучше, чем метод разложения на простые множители.

Тем не менее нужно соблюдать осторожность. Если одно и то же сообщение по компьютерной сети отослано трем лицам, то некто, кому известно x^3 по модулю N_1, N_2 и N_3, может восстановить $x^3 \bmod N_1N_2N_3 = x^3$, используя китайскую теорему об остатках, после чего число x уже не будет секретом. Фактически даже в случае, когда одно и то же сообщение отослано семерым разным лицам с отметками времени $(2^{\lceil \lg t_i \rceil}x + t_i)^3 \bmod N_i$, в которых времена t_i известны или могут быть с достаточной достоверностью предугаданы, значение величины x может быть найдено (упр. 44). В связи с этим некоторые криптографы рекомендуют кодировать информацию с использованием числа $2^{16} + 1 = 65\,537$ вместо 3. Данный показатель степени является простым, поэтому на вычисление числа $x^{65\,537} \bmod N$ затрачивается примерно в 8.5 раз больше времени, чем на вычисление числа $x^3 \bmod N$. [*CCITT Recommendations Blue Book* (Geneva: International Telecommunication Union, 1989), Fascicle VIII.8, Recommendation X.509, Annex C, 74–76.]

В оригинальном описании методики шифрования Р. Л. Ривест, А. Шамир и Л. Эдлеман предлагали кодировать число x числом $x^a \bmod N$, где a — любой простой показатель функции $\varphi(N)$, а не только $a = 3$. Тем не менее на практике предпочтение отдается показателю, для которого кодирование выполняется быстрее, чем декодирование.

Чтобы схема, реализуемая в RSA, была эффективной, числа p и q не должны оказаться "случайно" близкими простыми. Как подчеркивалось выше, чтобы обеспечить существование единственных кубических корней по модулю N, числа $p - 1$ и $q - 1$ не должны делиться на 3. Другое условие сводится к тому, что для числа $p - 1$ должен существовать по крайней мере один очень большой простой множитель. То же самое относится и к числу $q - 1$; в противном случае число N может быть раз-

ложено на простые множители по алгоритму из упр. 19. Фактически этот алгоритм сводится к поиску малого числа m, характеризуемого тем, что $x^m \bmod N$ быстро становится равным 1, а мы уже знаем, что использовать такое число небезопасно. Если для чисел $p-1$ и $q-1$ существуют большие простые множители p_1 и q_1, то согласно теории, изложенной в упр. 34, число m будет либо кратным p_1q_1 (тогда его будет трудно найти), либо вероятность того, что $x^m \equiv 1$, будет меньше, чем $1/p_1q_1$ (поскольку число $x^m \bmod N$ почти никогда не равно 1). К тому же нежелательно, чтобы числа p и q были близки одно к другому; иначе их можно будет довольно просто найти, используя алгоритм D. Нежелательно также, чтобы отношение этих чисел p/q было близко к простой дроби, в противном случае эти числа можно будет получить, используя обобщение Лемана из алгоритма C.

Предлагаемая процедура генерирования чисел p и q почти всегда выполняется безошибочно. Начинаем с правдоподобного случайного числа p_0, принадлежащего интервалу, скажем, между 10^{80} и 10^{81}. Находим первое простое число p_1, большее, чем p_0; для этого потребуется проверить $\frac{1}{2}\ln p_0 \approx 90$ нечетных чисел. Будет вполне достаточно, если после 50 пробных делений, выполняемых алгоритмом P, число p_1 окажется "вероятно, простым" с вероятностью $> 1 - 2^{-100}$. После этого выбираем другое правдоподобное случайное число p_2 между, скажем, 10^{39} и 10^{40}. Находим первое простое число p вида $kp_1 + 1$, где $k \geq p_2$, k — четное и $k \equiv p_1$ (по модулю 3). Для этого, прежде чем простое число p будет найдено, потребуется проверить $\frac{1}{3}\ln p_1p_2 \approx 90$ чисел. Простое число p будет длиной около 120 разрядов; подобная конструкция может быть применена и для поиска простого числа q длиной около 130 разрядов. Чтобы обеспечить супернадежность, желательно убедиться в том, что числа $p+1$ и $q+1$ не содержат малых простых множителей (упр. 20). Теперь произведение $N = pq$, величина которого имеет порядок 10^{250}, удовлетворяет всем требованиям, и в настоящее время такое число не может быть разложено на простые множители.

Например, предположим, что известен метод разложения 250-разрядного числа N, время выполнения которого имеет порядок $N^{0.1}$ мс. Следовательно, для разложения такого числа на простые множители потребуется 10^{25} мс, но в году имеется только 31 556 952 000 000 мкс, так что для завершения процесса разложения потребуется более 3×10^{11} лет машинного времени. Если предположить, что закуплено 10 млрд компьютеров и все они задействованы в решении этой проблемы, пройдет более 31 года, прежде чем один из них разложит число N на простые множители. Но пока правительство будет закупать так много специализированных компьютеров, люди начнут применять 300-разрядные числа N.

Поскольку метод кодирования $x \mapsto x^3 \bmod N$ известен, он обладает еще и дополнительными достоинствами, помимо того факта, что расшифровать код можно только при помощи RSA-блока. Такие системы "общедоступных ключей" были впервые рассмотрены В. Диффи (W. Diffie) и М. Э. Хеллманом (M. E. Hellman) в журнале *IEEE Trans.* **IT-22** (1976), 644–654. Как пример того, что можно сделать, когда метод кодирования широко известен, предположим, что Алиса желает связаться с Бобом по электронной почте и при этом они оба хотят, чтобы их письма были *подписаны* так, чтобы получатели были уверены, что никто не подделывает сообщение. Пусть $E_A(M)$ — кодирующая функция для сообщений M, направляемых Алисе, а $D_A(M)$ — декодирующая функция RSA-блока, установленного у Алисы. Пусть так-

же $E_B(M)$, $D_B(M)$ — соответственно кодирующая и декодирующая функции RSA-блока, принадлежащего Бобу. Тогда Алиса может отослать подписанное сообщение, поставив свою подпись и дату, а затем переслать Бобу сообщение $E_B\big(D_A(M)\big)$, использовав для вычисления $D_A(M)$ свой компьютер. Когда сообщение получает Боб, его RSA-блок преобразует это сообщение в сообщение $D_A(M)$; Бобу известно E_A, так что он может вычислить $M = E_A\big(D_A(M)\big)$. Это должно убедить его, что сообщение поступило именно от Алисы; никто другой не мог отослать сообщение с кодом $D_A(M)$. (Ну хорошо, теперь Бобу известен и код $D_A(M)$, поэтому он, посылая сообщение $E_X\big(D_A(M)\big)$ Ксавьеру, может представлять Алису. Чтобы защититься от таких попыток подделки, в содержимом кода M должно быть четко указано, что оно предназначено только для Боба.)

Естественно задать вопрос, каким образом Алиса и Боб узнают кодирующие функции E_A и E_B друг друга? Простого хранения этих функций в общедоступном файле явно недостаточно, поскольку некто Чарли может проникнуть в этот файл, подставив вычисленный им код N. Затем Чарли получит возможность тайно перехватывать и декодировать чужие сообщения до того, как Алиса или Боб что-нибудь заподозрят. Выход из этой ситуации заключается в том, чтобы хранить такие произведения N_A и N_B в специальном общедоступном каталоге, который имеет собственный RSA-блок и широко известное произведение N_D. Когда Алиса пожелает узнать, как связаться с Бобом, она запросит в этом каталоге произведение для Боба; компьютер, управляющий каталогом, отправит ей *подписанное* сообщение, выдав значение N_B. Такое сообщение должно быть легитимным, поскольку подделать его никто не сможет.

Интересную альтернативу RSA-схеме предложил Майкл Рабин (Michael Rabin). В работе, опубликованной в MIT Lab. for Comp. Sci., report TR-212 (1979), он описал способ кодирования функцией $x^2 \bmod N$ вместо функции $x^3 \bmod N$. В этом случае механизм декодирования, который можно было бы назвать SQRT-блоком, возвращает четыре различных сообщения; суть в том, что четыре различных сообщения содержат один и тот же квадрат по модулю N, а именно — x, $-x$, $fx \bmod N$ и $(-fx) \bmod N$, где

$$f = (p^{q-1} - q^{p-1}) \bmod N.$$

Если условиться на будущее, что x — четное число или что $x < \frac{1}{2}N$, то двусмысленность коснется двух сообщений, из которых предположительно только одно имеет смысл. Эта двусмысленность может быть легко устранена, как показано в упр. 35. Схема Рабина обладает важным свойством — найти квадратные корни по модулю N, вероятно, так же трудно, как найти разложение на простые множители числа $N = pq$. Если число x выбрать случайно, шансы на то, что при вычислении квадратного корня из $x^2 \bmod N$ будет найдено значение y, такое, что выполняются условия

$$x^2 \equiv y^2 \qquad \text{и} \qquad x \not\equiv \pm y,$$

причем $\gcd(x - y, N) = p$ или q, равны 50/50. Однако эта система имеет существенный изъян, который, похоже, отсутствует в RSA-схеме (упр. 33). Каждый, кто обращается к SQRT-блоку, может легко определить простые множители числа N. Это не только допускает возможность жульничества со стороны нечестных служащих или угрозы вымогательства, но также позволяет выдать секрет чисел p и q. После этого

нечестивцы могут утверждать, что их подпись на каком-то переданном документе была подделана. Таким образом, ясно, что цель безопасной связи заключается в том, чтобы учесть тонкости проблем, совсем не похожих на те проблемы, с которыми приходится обычно сталкиваться при разработке и анализе алгоритмов.

Историческая справка. В 1988 году появилось сообщение, что Клиффорд Кокс (Clifford Cocks) рассмотрел возможность кодирования сообщения посредством преобразования $x^{pq} \bmod pq$ еще в 1973 году, но его работа засекречена.

Самые большие известные простые числа. В различных разделах этой книги было рассмотрено несколько вычислительных методов, использующих в процессе работы большие простые числа. Описанные в данном разделе методы позволяют сравнительно легко находить простые числа, имеющие, скажем, не более 25 разрядов. В табл. 2 приведены 10 наибольших простых чисел, меньших, чем длина машинного слова типового компьютера. (Другие полезные простые числа приводятся в упр. 3.2.1.2–22 и 4.6.4–57.)

В действительности известны значительно бóльшие простые числа специального вида, а иногда бывает важно найти простые числа максимально возможной величины. Поэтому давайте завершим настоящий раздел исследованием интересного способа, которым были найдены наибольшие из известных простых чисел. Такие простые числа имеют вид $2^n - 1$ для различных частных значений n, и поэтому они особенно подходят для некоторых приложений на двоичных компьютерах.

Число вида $2^n - 1$ не может быть простым, если не является простым число n, поскольку $2^{uv} - 1$ делится на $2^u - 1$. В 1644 году Марен Мерсенн (Marin Mersenne) удивил своих современников, заявив, что числа $2^p - 1$ являются простыми при $p = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127, 257$ и не являются таковыми ни при каких других p, меньших 257. (Это утверждение появилось в связи с рассуждениями Мерсенна о совершенных числах в предисловии к его *Cogitata Physico-Mathematica*. Любопытно, что он также сделал следующее примечание: "Какими бы известными на сегодня методами мы ни пользовались, для того чтобы определить, будет ли заданное 15- или 20-значное число простым, не хватит целой жизни". Мерсенн, который в предыдущие годы часто переписывался с Ферма, Декартом и другими учеными по сходным вопросам, не привел никаких доказательств своих утверждений и в течение последующих 200 лет никто не знал, был ли он прав. В 1772 году Эйлер после многолетних безуспешных попыток наконец доказал, что $2^{31} - 1$ — простое число. Почти через 100 лет Э. Люка (É. Lucas) установил, что $2^{127} - 1$ — тоже простое число, а судьба числа $2^{67} - 1$ осталась под вопросом. Следовательно, Мерсенн был не совсем точен. Затем в 1883 году И. М. Первушин доказал, что $2^{61} - 1$ — простое число [см. *Историко-мат. исследования* **6** (1953), 559], и это послужило поводом для подозрений, что Мерсенн допустил описку, записав 67 вместо 61. В конце концов были обнаружены и другие ошибки в утверждениях Мерсенна; Р. Е. Пауэрс (R. E. Powers) [*AMM* **18** (1911), 195] показал, что простым является число $2^{89} - 1$, как уже предполагали до него некоторые авторы; три года спустя он доказал, что число $2^{107} - 1$ также является простым. В 1922 году М. Крайчик (M. Kraitchik) обнаружил, что число $2^{257} - 1$ *не является* простым [см. его *Recherches sur la Théorie des Nombres* (Paris, 1924), 21]; в его вычисления, возможно, вкрались ошибки, но вывод оказался верным.

Таблица 2

ПОЛЕЗНЫЕ ПРОСТЫЕ ЧИСЛА

N	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
2^{15}	19	49	51	55	61	75	81	115	121	135
2^{16}	15	17	39	57	87	89	99	113	117	123
2^{17}	1	9	13	31	49	61	63	85	91	99
2^{18}	5	11	17	23	33	35	41	65	75	93
2^{19}	1	19	27	31	45	57	67	69	85	87
2^{20}	3	5	17	27	59	69	129	143	153	185
2^{21}	9	19	21	55	61	69	105	111	121	129
2^{22}	3	17	27	33	57	87	105	113	117	123
2^{23}	15	21	27	37	61	69	135	147	157	159
2^{24}	3	17	33	63	75	77	89	95	117	167
2^{25}	39	49	61	85	91	115	141	159	165	183
2^{26}	5	27	45	87	101	107	111	117	125	135
2^{27}	39	79	111	115	135	187	199	219	231	235
2^{28}	57	89	95	119	125	143	165	183	213	273
2^{29}	3	33	43	63	73	75	93	99	121	133
2^{30}	35	41	83	101	105	107	135	153	161	173
2^{31}	1	19	61	69	85	99	105	151	159	171
2^{32}	5	17	65	99	107	135	153	185	209	267
2^{33}	9	25	49	79	105	285	301	303	321	355
2^{34}	41	77	113	131	143	165	185	207	227	281
2^{35}	31	49	61	69	79	121	141	247	309	325
2^{36}	5	17	23	65	117	137	159	173	189	233
2^{37}	25	31	45	69	123	141	199	201	351	375
2^{38}	45	87	107	131	153	185	191	227	231	257
2^{39}	7	19	67	91	135	165	219	231	241	301
2^{40}	87	167	195	203	213	285	293	299	389	437
2^{41}	21	31	55	63	73	75	91	111	133	139
2^{42}	11	17	33	53	65	143	161	165	215	227
2^{43}	57	67	117	175	255	267	291	309	319	369
2^{44}	17	117	119	129	143	149	287	327	359	377
2^{45}	55	69	81	93	121	133	139	159	193	229
2^{46}	21	57	63	77	167	197	237	287	305	311
2^{47}	115	127	147	279	297	339	435	541	619	649
2^{48}	59	65	89	93	147	165	189	233	243	257
2^{59}	55	99	225	427	517	607	649	687	861	871
2^{60}	93	107	173	179	257	279	369	395	399	453
2^{63}	25	165	259	301	375	387	391	409	457	471
2^{64}	59	83	95	179	189	257	279	323	353	363
10^6	17	21	39	41	47	69	83	93	117	137
10^7	9	27	29	57	63	69	71	93	99	111
10^8	11	29	41	59	69	153	161	173	179	213
10^9	63	71	107	117	203	239	243	249	261	267
10^{10}	33	57	71	119	149	167	183	213	219	231
10^{11}	23	53	57	93	129	149	167	171	179	231
10^{12}	11	39	41	63	101	123	137	143	153	233
10^{16}	63	83	113	149	183	191	329	357	359	369

Десять наибольших простых чисел, меньших, чем N, суть $N - a_1, \ldots, N - a_{10}$.

В настоящее время числа вида $2^p - 1$ называются *числами Мерсенна* и известно, что простые числа Мерсенна вычислены для следующего ряда значений p:

$$2,\ 3,\ 5,\ 7,\ 13,\ 17,\ 19,\ 31,\ 61,\ 89,\ 107,\ 127,\ 521,\ 607,\ 1\,279,\ 2\,203,\ 2\,281,$$
$$3\,217,\ 4\,253,\ 4\,423,\ 9\,689,\ 9\,941,\ 11\,213,\ 19\,937,\ 21\,701,\ 23\,209,\ 44\,497,\ 86\,243,$$
$$110\,503,\ 132\,049,\ 216\,091,\ 756\,839,\ 859\,433,\ 1\,257\,787,\ 1\,398\,269,\ 2\,976\,221. \quad (26)$$

Большая часть из значений в нижней строке получена Дэвидом Словинским (David Slowinski) и используется для тестирования новых суперкомпьютеров [см. *J. Recreational Math.* **11** (1979), 258–261]; числа 756 839, 859 433 и 1 257 787 он получил вместе с Полем Гейджем (Paul Gage) в 90-х годах. Тем не менее две наибольшие степени, 1 398 269 и 2 976 221, были получены Жоэлем Арменгодом (Joel Armengaud) и Гордоном Спенсом (Gordon Spence) на персональных компьютерах; для решения задачи они использовали программу, разработанную в 1996 году Георгом Вольтманом (George Woltman), руководителем проекта Great Internet Mersenne Prime Search (GIMPS). Заметим, что простое число $8\,191 = 2^{13} - 1$ в списке (26) отсутствует; Мерсенн утверждал, что $2^{8\,191} - 1$ — простое число, и многие высказывали предположение, что любое простое число Мерсенна может быть использовано в качестве показателя степени.

К проекту GIMPS присоединились тысячи людей, и Скотт Куровский (Scott Kurowski) с целью более систематизированного использования показателей степени разработал программное обеспечение для Internet под названием PrimeNet. Уже к февралю 1998 года, за год с небольшим после внедрения этого продукта в Internet, активность, связанная с поисками простых чисел Мерсенна, постоянно возрастала.

Таким образом, следует ожидать существенного прогресса в поиске этих исторических чисел.

После 1997 года найдены такие простые числа $2^p - 1$.

p	Автор	Дата
3021377	Роланд Кларксон (Roland Clarkson)	27 января 1998 года
6972593	Найан Хьяратвала (Nayan Hajratwala)	1 июня 1999 года

Процесс поиска больших простых чисел до сих пор не систематизирован, потому что людям свойственно стремление устанавливать мировые рекорды, которые трудно побить, вместо того чтобы тратить время на поиск меньших значений показателей степеней. Например, в 1983 году было доказано, что $2^{132\,049} - 1$ — простое число, в 1984 году, что $2^{216\,091} - 1$ — также простое число, а число $2^{110\,503} - 1$ — нет. Поэтому до сих пор может существовать одно или несколько неизвестных чисел Мерсенна, меньших $2^{2\,976\,221} - 1$. (Под руководством Вольтмана к 26 мая 1997 года были проверены все показатели степени до 1 000 000, и его добровольцами были последовательно заполнены все ранее незаполненные позиции.)

Поскольку число $2^{2\,976\,221} - 1$ содержит около 900 000 десятичных разрядов, ясно, что для доказательства того факта, что числа подобного вида простые, был применен какой-то специальный метод. (Действительно, предварительная проверка 12 апреля 1996 года числа $2^{1\,257\,787} - 1$ отняла менее 29 845 с машинного времени (это примерно 8.3 ч) на компьютере Cray T94. Предварительная проверка числа $2^{2\,976\,221} - 1$, выполненная в августе 1997 года на компьютере Pentium PC 100 MHz, заняла 15 суток.) Впервые эффективный способ проверки, является ли число Мерсенна $2^p - 1$ простым, был предложен Э. Люка [*Amer. J. Math.* **1** (1878), 184–239,

289–321, особенно с. 316] и усовершенствован Д. Г. Лемером [*Annals of Math.* **31** (1930), 419–448, особенно с. 443].

Этот способ проверки является частным случаем метода, применяемого в настоящее время для проверки принадлежности числа n к простым числам, когда известны множители числа $n+1$, и состоит в следующем.

Теорема L. *Пусть q — нечетное простое число и последовательность $\langle L_n \rangle$ задается правилом*

$$L_0 = 4, \qquad L_{n+1} = (L_n^2 - 2) \bmod (2^q - 1). \tag{27}$$

Тогда число $2^q - 1$ будет простым тогда и только тогда, когда $L_{q-2} = 0$.

Например, $2^3 - 1$ — простое число, поскольку $L_1 = (4^2 - 2) \bmod 7 = 0$. Этот способ проверки особенно подходит для реализации на двоичных компьютерах из-за простоты вычислений по модулю $(2^q - 1)$ (см. раздел 4.3.2). В упр. 4.3.2–14 изложен способ, позволяющий сэкономить время выполнения операций в случае очень больших чисел q.

Доказательство. Докажем теорему L, используя лишь самые простые принципы теории чисел, путем исследования некоторых свойств рекуррентных последовательностей, представляющих и самостоятельный интерес. Рассмотрим последовательности $\langle U_n \rangle$ и $\langle V_n \rangle$, определяемые правилами

$$\begin{aligned} U_0 &= 0, \qquad U_1 = 1, \qquad U_{n+1} = 4U_n - U_{n-1}; \\ V_0 &= 2, \qquad V_1 = 4, \qquad V_{n+1} = 4V_n - V_{n-1}. \end{aligned} \tag{28}$$

По индукции легко доказать справедливость следующих соотношений:

$$V_n = U_{n+1} - U_{n-1}; \tag{29}$$

$$U_n = \big((2+\sqrt{3}\,)^n - (2-\sqrt{3}\,)^n\big)/\sqrt{12}; \tag{30}$$

$$V_n = (2+\sqrt{3}\,)^n + (2-\sqrt{3}\,)^n; \tag{31}$$

$$U_{m+n} = U_m U_{n+1} - U_{m-1} U_n. \tag{32}$$

Докажем сейчас один вспомогательный результат, когда p — простое число и $e \geq 1$.

$$\text{Если} \quad U_n \equiv 0 \ (\text{по модулю } p^e), \quad \text{то} \quad U_{np} \equiv 0 \ (\text{по модулю } p^{e+1}). \tag{33}$$

Этот результат следует из более общих соображений, изложенных в упр. 3.2.2–11, но для соотношений (28) можно привести и непосредственное доказательство. Предположим, что $U_n = bp^e$, $U_{n+1} = a$. Согласно уравнению (32) и (28) имеем $U_{2n} = bp^e(2a - 4bp^e) \equiv 2aU_n$ (по модулю p^{e+1}), а $U_{2n+1} = U_{n+1}^2 - U_n^2 \equiv a^2$. Аналогично $U_{3n} = U_{2n+1}U_n - U_{2n}U_{n-1} \equiv 3a^2U_n$ и $U_{3n+1} = U_{2n+1}U_{n+1} - U_{2n}U_n \equiv a^3$. В общем случае

$$U_{kn} \equiv ka^{k-1}U_n \qquad \text{и} \qquad U_{kn+1} \equiv a^k \ (\text{по модулю } p^{e+1}).$$

Следовательно, взяв $k = p$, получим (33).

Из формул (30) и (31), раскрывая $(2 \pm \sqrt{3}\,)^n$ по биномиальной теореме, можно получить остальные выражения для U_n и V_n:

$$U_n = \sum_k \binom{n}{2k+1} 2^{n-2k-1} 3^k, \qquad V_n = \sum_k \binom{n}{2k} 2^{n-2k+1} 3^k. \tag{34}$$

Положив теперь $n = p$, где p — нечетное простое число, и воспользовавшись тем фактом, что $\binom{p}{k}$ кратно p, кроме случая, когда $k = 0$ или $k = p$, приходим к выводу, что

$$U_p \equiv 3^{(p-1)/2}, \qquad V_p \equiv 4 \qquad \text{(по модулю } p\text{)}. \tag{35}$$

При $p \neq 3$ по теореме Ферма имеем $3^{p-1} \equiv 1$; следовательно, $(3^{(p-1)/2} - 1) \times (3^{(p-1)/2} + 1) \equiv 0$ и $3^{(p-1)/2} \equiv \pm 1$. Если $U_p \equiv -1$, то $U_{p+1} = 4U_p - U_{p-1} = 4U_p + V_p - U_{p+1} \equiv -U_{p+1}$; отсюда $U_{p+1} \bmod p = 0$. Если $U_p \equiv +1$, то $U_{p-1} = 4U_p - U_{p+1} = 4U_p - V_p - U_{p-1} \equiv -U_{p-1}$; значит, $U_{p-1} \bmod p = 0$. Таким образом доказано, что для каждого простого p существует целое число $\epsilon(p)$, такое, что

$$U_{p+\epsilon(p)} \bmod p = 0, \qquad |\epsilon(p)| \leq 1. \tag{36}$$

Далее, если N — произвольное положительное целое число и если $m = m(N)$ — такое наименьшее положительное целое число, что $U_{m(N)} \bmod N = 0$, то

$$U_n \bmod N = 0 \qquad \text{тогда и только тогда, когда} \qquad n \text{ кратно } m(N). \tag{37}$$

(Это число $m(N)$ называется *рангом появления* в последовательности числа N.) Для доказательства (37) учтем, что последовательность $U_m, U_{m+1}, U_{m+2}, \ldots$ конгруэнтна (по модулю N) последовательности $aU_0, aU_1, aU_2, \ldots$, где число $a = U_{m+1} \bmod N$ взаимно просто с N, так как $\gcd(U_n, U_{n+1}) = 1$.

После всех этих приготовлений докажем теорему L. В силу соотношения (27) по индукции имеем

$$L_n = V_{2^n} \bmod (2^q - 1). \tag{38}$$

Далее, из тождества $2U_{n+1} = 4U_n + V_n$ следует, что $\gcd(U_n, V_n) \leq 2$, поскольку всякий общий делитель чисел U_n и V_n должен делить U_n и $2U_{n+1}$, в то время как $U_n \perp U_{n+1}$. Поэтому U_n и V_n не имеют общих нечетных множителей, и, если $L_{q-2} = 0$, должно выполняться

$$U_{2^{q-1}} = U_{2^{q-2}} V_{2^{q-2}} \equiv 0 \text{ (по модулю } 2^q - 1),$$
$$U_{2^{q-2}} \not\equiv 0 \text{ (по модулю } 2^q - 1).$$

Затем, если $m = m(2^q - 1)$ — ранг появления числа $2^q - 1$, то оно должно быть делителем числа 2^{q-1}, но не числа 2^{q-2}; таким образом, $m = 2^{q-1}$. Докажем теперь, учитывая сказанное выше, что число $n = 2^q - 1$ должно быть простым. Предположим, что разложение числа n на простые множители имеет вид $p_1^{e_1} \ldots p_r^{e_r}$. Все простые числа p_j больше 3, так что число n нечетно и конгруэнтно $(-1)^q - 1 = -2$ (по модулю 3). Из (33), (36) и (37) следует, что $U_t \equiv 0$ (по модулю $2^q - 1$), где

$$t = \operatorname{lcm}\big(p_1^{e_1 - 1}(p_1 + \epsilon_1), \ldots, p_r^{e_r - 1}(p_r + \epsilon_r)\big),$$

и каждое ϵ_j равно ± 1. Отсюда получаем, что t кратно $m = 2^{q-1}$. Пусть $n_0 = \prod_{j=1}^r p_j^{e_j - 1}(p_j + \epsilon_j)$; тогда $n_0 \leq \prod_{j=1}^r p_j^{e_j - 1}(p_j + \frac{1}{5}p_j) = (\frac{6}{5})^r n$. Кроме того, поскольку $p_j + \epsilon_j$ — четное число, то $t \leq n_0/2^{r-1}$, ибо всякий раз, когда берется наименьшее общее кратное двух четных чисел, множитель 2 теряется. Объединяя эти результаты, получаем $m \leq t \leq 2(\frac{3}{5})^r n < 4(\frac{3}{5})^r m < 3m$; отсюда, $r \leq 2$ и $t = m$ или $t = 2m$, т. е. t равно степени 2. Поэтому $e_1 = 1$, $e_r = 1$ и, если n — не простое число, должно быть $n = 2^q - 1 = (2^k + 1)(2^l - 1)$, где $2^k + 1$ и $2^l - 1$ — простые числа. Последнее разложение при нечетном q невозможно, поэтому n — простое.

Обратно, предположим, что $n = 2^q - 1$ — простое число. Необходимо показать, что $V_{2^{q-2}} \equiv 0$ (по модулю n). Для этого достаточно доказать, что $V_{2^{q-1}} \equiv -2$ (по модулю n), поскольку $V_{2^{q-1}} = (V_{2^{q-2}})^2 - 2$. Но

$$V_{2^{q-1}} = \big((\sqrt{2}+\sqrt{6})/2\big)^{n+1} + \big((\sqrt{2}-\sqrt{6})/2\big)^{n+1}$$

$$= 2^{-n}\sum_k \binom{n+1}{2k}\sqrt{2}^{\,n+1-2k}\sqrt{6}^{\,2k} = 2^{(1-n)/2}\sum_k \binom{n+1}{2k}3^k.$$

Так как n — нечетное простое число, биномиальный коэффициент

$$\binom{n+1}{2k} = \binom{n}{2k} + \binom{n}{2k-1}$$

делится на n за исключением случая, когда $2k = 0$ и $2k = n+1$. Следовательно,

$$2^{(n-1)/2}V_{2^{q-1}} \equiv 1 + 3^{(n+1)/2} \text{ (по модулю } n).$$

Здесь $2 \equiv (2^{(q+1)/2})^2$, поэтому согласно теореме Ферма

$$2^{(n-1)/2} \equiv (2^{(q+1)/2})^{(n-1)} \equiv 1.$$

В итоге в силу одного простого случая закона взаимности квадратичных вычетов (упр. 23) $3^{(n-1)/2} \equiv -1$, поскольку $n \bmod 3 = 1$ и $n \bmod 4 = 3$. Это означает, что $V_{2^{q-1}} \equiv -2$ и, следовательно, должно быть $V_{2^{q-2}} \equiv 0$, что и требуется. ▮

В 1460 году анонимный автор, работы которого в настоящее время хранятся в итальянских библиотеках, открыл, что числа $2^{17} - 1$ и $2^{19} - 1$ являются простыми [см. E. Picutti, *Historia Math.* **16** (1989), 123–136]. С тех пор наибольшими из известных простых чисел почти всегда оказываются числа Мерсенна. Но положение может измениться, поскольку находить простые числа Мерсенна становится все труднее, и поэтому в упр. 27 представлен эффективный способ проверки чисел другого вида.

УПРАЖНЕНИЯ

1. [*10*] Если последовательность пробных делителей в алгоритме A $d_0, d_1, d_2, \ldots$ содержит число, не являющееся простым, то почему оно никогда не появится на выходе?

2. [*15*] Можно ли исключить из алгоритма A шаг A2, если известно, что исходное число N для алгоритма A не меньше 3?

3. [*M20*] Покажите, что существует число P со следующим свойством: если $1\,000 \le n \le 1\,000\,000$, то число n будет простым тогда и только тогда, когда $\gcd(n, P) = 1$.

4. [*M29*] Используя обозначения упр. 3.1–7 и раздела 1.2.11.3, докажите, что среднее значение наименьших n, таких, что $X_n = X_{\ell(n)-1}$, находится в интервале между $1.5Q(m) - 0.5$ и $1.625Q(m) - 0.5$.

5. [*21*] При помощи метода Ферма (алгоритм D) найдите вручную множители числа 11 111 по модулям 3, 5, 7, 8 и 11.

6. [*M24*] Докажите, что если p — нечетное простое число и N не кратно числу p, то количество целых чисел x, принадлежащих интервалу $0 \le x < p$, для которых существует решение y уравнения $x^2 - N \equiv y^2$ (по модулю p), равно $(p \pm 1)/2$.

7. [*25*] Проанализируйте задачи программирования метода решета — алгоритм D — для двоичного компьютера в случае, когда табличные значения для модулей m_i заполняют не точно целое число слов памяти.

▶ **8.** [*23*] (*Решето Эратосфена*, 3 в. до н. э.) Следующая процедура позволяет найти все нечетные простые числа, меньшие данного целого числа N, поскольку она удаляет все непростые числа. Выполнение процедуры начинается с того, что для всех нечетных целых чисел от 1 до N последовательно вычеркиваются числа p_k^2, $p_k(p_k+2)$, $p_k(p_k+4)$, ..., кратные k-му простому числу p_k при $k = 2, 3, 4, \ldots$, пока не будет найдено такое простое число p_k, для которого $p_k^2 > N$.

Покажите, как превратить описанную процедуру в алгоритм, пригодный для прямой реализации в компьютере, без использования операций умножения.

9. [*M25*] Пусть n — нечетное целое число и $n \geq 3$. Покажите, что если число $\lambda(n)$ из теоремы 3.2.1.2B является делителем числа $n-1$, но не равно $n-1$, то n должно иметь вид $p_1 p_2 \ldots p_t$, где все p_i — различные простые числа и $t \geq 3$.

▶ **10.** [*M26*] (Джон Селфридж (John Selfridge).) Докажите, что если для любого простого делителя p числа $n-1$ существует такое x_p, что $x_p^{(n-1)/p} \bmod n \neq 1$, а $x_p^{n-1} \bmod n = 1$, то n — простое число.

11. [*M20*] Что выведет алгоритм E, если $N = 197\,209$, $k = 5$, $m = 1$?
[*Указание.* $\sqrt{5 \cdot 197\,209} = 992 + //\overline{1, 495, 2, 495, 1, 1984}//$.]

▶ **12.** [*M28*] Разработайте алгоритм, который, используя выходные данные алгоритма E, находит подходящий простой множитель N, который обеспечивает возможность алгоритму E формировать достаточно данных для вывода решения уравнения (18).

13. [*HM25*] (Дж. Д. Диксон (J. D. Dixon).) Докажите, что как только алгоритм из упр. 12 представляется решением $(x, e_0, \ldots, e_m)$, показатели степени в котором линейно зависят по модулю 2 от показателей степени в предыдущих решениях, когда число N имеет различные простые множители, а величина x выбирается случайно, вероятность того, что разложение на простые множители не будет найдено, равна 2^{1-d}.

14. [*M20*] Докажите, что при выполнении шага E3 алгоритма E число T никогда не кратно нечетному простому множителю p, для которого выполняется неравенство

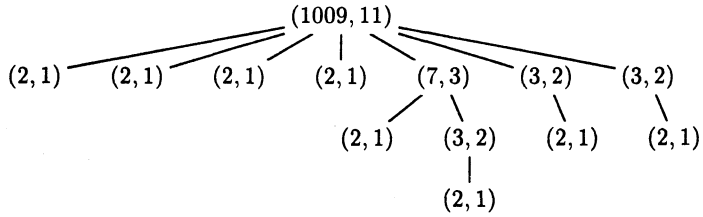
$$(kN)^{(p-1)/2} \bmod p > 1.$$

▶ **15.** [*M34*] (Люка (Lucas) и Лемер (Lehmer).) Положим, что P и Q — взаимно простые целые числа, и пусть $U_0 = 0$, $U_1 = 1$, $U_{n+1} = PU_n - QU_{n-1}$ при $n \geq 1$. Докажите, что если N — положительное целое число, взаимно простое с числом $2P^2 - 8Q$, и если $U_{N+1} \bmod N = 0$, а $U_{(N+1)/p} \bmod N \neq 0$ для каждого простого числа p, делящего $N+1$, то N — простое число. (В результате получаем метод проверки принадлежности чисел к простым числам в случае, когда известны делители числа $N+1$, а не делители числа $N-1$. Значение U_m можно вычислить за $O(\log m)$ шагов, как в упр. 4.6.3–26.) [*Указание.* См. доказательство теоремы L.]

16. [*M50*] Бесконечно ли множество простых чисел Мерсенна?

17. [*M25*] (В. Р. Пратт (V. R. Pratt).) Полное доказательство принадлежности чисел к простым при помощи обратной теоремы Ферма принимает вид дерева с узлами (q, x), где q и x — положительные целые числа, удовлетворяющие следующим условиям. (i) Если узлы (q_1, x_1), ..., (q_t, x_t) порождены узлом (q, x), то $q = q_1 \ldots q_t + 1$. [В частности, если (q, x) — младший потомок, то $q = 2$.] (ii) Если узел (r, y) порожден узлом (q, x), то $x^{(q-1)/r} \bmod q \neq 1$. (iii) Для каждого узла (q, x) выполняется условие $x^{q-1} \bmod q = 1$.

Из этих условий следует, что число q простое и x есть простой корень по модулю q для всех узлов (q, x). [Например, дерево



показывает, что 1009 — простое число.] Докажите, что подобное дерево с корнем (q, x) содержит не более $f(q)$ узлов, где функция f довольно медленно возрастает.

▸ **18.** [*HM23*] Приведите эвристическое доказательство соотношения (7) по аналогии с выводом соотношений (6), рассмотренных в этом разделе. Чему равна приближенная вероятность того, что $p_{t-1} \le \sqrt{p_t}$?

▸ **19.** [*M25*] (Дж. М. Поллард (J. M. Pollard).) Покажите, как вычислить число M, которое делится на все нечетные простые множители p, такие, что $p - 1$ является делителем некоторого заданного числа D. [*Указание.* Рассмотрите числа вида $a^n - 1$.] Такое число M полезно при разложении чисел на простые множители, поскольку множитель числа N может быть получен в результате вычисления $\gcd(M, N)$. Постарайтесь развить эту идею и сформулировать эффективный метод нахождения с высокой вероятностью простых множителей p данного числа N для случая, когда все простые степенные множители для чисел $p - 1$ меньше 10^3, кроме, может быть, одного, меньшего 10^5. [Например, при помощи такого метода будет обнаружен второй по величине простой множитель, который делит (15), так как он равен $1 + 2^4 \cdot 5^2 \cdot 67 \cdot 107 \cdot 199 \cdot 41231$.]

20. [*M40*] Рассмотрите упр. 19, подставив в условие $p + 1$ вместо $p - 1$.

21. [*M49*] (Р. К. Ги (R. K. Guy).) Пусть $m(p)$ — число итераций, необходимых алгоритму B для выделения простого множителя p. Будет ли выполняться равенство $m(p) = O(\sqrt{p \log p})$ при $p \to \infty$?

▸ **22.** [*M30*] (М. О. Рабин (M. O. Rabin).) Пусть для данного числа n p_n — вероятность того, что алгоритм P дает ошибочный результат. Покажите, что $p_n < \frac{1}{4}$ для всех n.

23. [*M35*] *Символ Якоби* $\left(\frac{p}{q}\right)$ по определению равен -1, 0 или $+1$ для всех целых чисел $p \ge 0$ и всех нечетных целых чисел $q > 1$ в соответствии со следующим правилом: $\left(\frac{p}{q}\right) \equiv p^{(q-1)/2}$ (по модулю q), когда q — простое число, $\left(\frac{p}{q}\right) = \left(\frac{p}{q_1}\right) \ldots \left(\frac{p}{q_t}\right)$, когда число q равно произведению $q_1 \ldots q_t$ и t простых чисел (необязательно различных). Таким образом, символ Якоби является обобщением символа Лежандра (см. упр. 1.2.4–47).

a) Докажите, что $\left(\frac{p}{q}\right)$ удовлетворяет следующим зависимостям, которые позволяют эффективно его вычислять: $\left(\frac{0}{q}\right) = 0$; $\left(\frac{1}{q}\right) = 1$; $\left(\frac{p}{q}\right) = \left(\frac{p \bmod q}{q}\right)$; $\left(\frac{2}{q}\right) = (-1)^{(q^2-1)/8}$; $\left(\frac{pp'}{q}\right) = \left(\frac{p}{q}\right)\left(\frac{p'}{q}\right)$; $\left(\frac{p}{q}\right) = (-1)^{(p-1)(q-1)/4}\left(\frac{q}{p}\right)$, если оба числа p и q нечетны. [Последняя закономерность, которая представляет собой обратную зависимость и сводит вычисление $\left(\frac{p}{q}\right)$ к вычислению $\left(\frac{q}{p}\right)$, доказана в упр. 1.2.4–47(d) для p и q, являющихся простыми числами; поэтому в таком особом случае можно считать данную закономерность справедливой.]

b) (Соловей (Solovay) и Штрассен (Strassen).) Докажите, что если n — нечетное число, но не простое, то количество целых чисел x, таких, что $1 \le x < n$ и $0 \ne \left(\frac{x}{n}\right) \equiv x^{(n-1)/2}$ (по модулю n), не превышает величины $\frac{1}{2}\varphi(n)$. (Значит, следующая процедура с вероятностью, равной как минимум $1/2$, для всех фиксированных чисел n корректно

определяет, является ли данное число n простым. "Сгенерировать случайное число x в интервале $1 \le x < n$. Если $0 \ne (\frac{x}{n}) \equiv x^{(n-1)/2}$ (по модулю n), сказать, что n является, вероятно, простым; в противном случае сказать, что число n, определенно, не является простым.")

c) (Л. Моньер (L. Monier).) Докажите, что если n и x — числа, для которых алгоритм P делает вывод, что "n, вероятно, простое", то $0 \ne (\frac{x}{n}) \equiv x^{(n-1)/2}$ (по модулю n). [Следовательно, алгоритм P является основным при выполнении проверки в случае (b).]

▶ **24.** [*M25*] (Л. Эдлеман (L. Adleman).) Если $n > 1$ и нечетно, а $x > 1$ — целое число, будем говорить, что число n "проходит проверку алгоритмом P посредством x", если либо $x \bmod n = 0$, либо выполнение шагов P2–P5 приводит к заключению, что число n, вероятно, простое. Докажите, что для любого N существует множество положительных целых нечетных чисел $x_1, \ldots, x_m \le N$ для $m \le \lfloor \lg N \rfloor$, такое, что положительное нечетное целое число в интервале $1 < n \le N$ будет простым тогда и только тогда, когда оно проходит проверку алгоритмом P посредством чисел x для $x = x_1 \bmod n, \ldots, x = x_m \bmod n$. Таким образом, процесс вероятностной проверки "простоты", в принципе, может быть превращен в эффективный инструмент проверки, устраняющий всякие сомнения. (Сейчас не требуется приводить эффективный способ вычисления величин x_j; нужно только доказать, что такие величины существуют.)

25. [*HM41*] (Б. Риман (B. Riemann).) Докажите, что

$$\pi(x) + \frac{\pi(x^{1/2})}{2} + \frac{\pi(x^{1/3})}{3} + \cdots = \int_2^x \frac{dt}{\ln t} - 2\sum \int_{-\infty}^{\sigma} \frac{e^{(t+i\tau)\ln x}\,dt}{t+i\tau} + O(1),$$

где суммирование выполняется по всем комплексным числам $\sigma + i\tau$, таким, что $\tau > 0$ и $\zeta(\sigma + i\tau) = 0$.

▶ **26.** [*M25*] (Г. К. Поклингтон (H. C. Pocklington), 1914.) Пусть $N = fr+1$, где $0 < r \le f+1$. Докажите, что число N будет простым тогда, когда для каждого простого делителя p числа f существует такое целое число x_p, что $x_p^{N-1} \bmod N = \gcd(x_p^{(N-1)/p} - 1,\ N) = 1$.

▶ **27.** [*M30*] Покажите, что существует способ проверки принадлежности к простым числам чисел вида $N = 5 \cdot 2^n + 1$, использующий приблизительно столько же операций вычисления квадратов по модулю N, сколько применялось в способе Люка-Лемера (Lucas-Lehmer) проверки чисел Мерсенна в теореме L. [*Указание.* См. предыдущее упражнение.]

28. [*M27*] .Для данных простого числа p и положительного целого числа d найдите значение функции $f(p,d)$, среднее число случаев, когда число p делит $A^2 - dB^2$ (учитывая кратность), если A и B — независимые случайные целые числа за исключением условия $A \perp B$.

29. [*M25*] Докажите, что количество положительных целых чисел $\le n$, простые множители которых принадлежат множеству простых чисел $\{p_1, \ldots, p_m\}$, не меньше $m^r/r!$, если $r = \lfloor \log n / \log p_m \rfloor$ и $p_1 < \cdots < p_m$.

30. [*HM35*] (Дж. Д. Диксон (J. D. Dixon) и Клаус-Петер Шнорр (Claus-Peter Schnorr).) Пусть $p_1 < \cdots < p_m$ — простые числа, не делящие нечетное число N, и пусть r — четное целое число, не превышающее величины $\log N / \log p_m$. Докажите, что количество целых чисел X, принадлежащих интервалу $0 \le X < N$ и таких, что $X^2 \bmod N = p_1^{e_1} \ldots p_m^{e_m}$, не меньше $m^r/r!$. *Указание.* Положите, что разложение числа N на простые множители имеет вид $q_1^{f_1} \ldots q_d^{f_d}$. Покажите, что последовательность показателей $(e_1, \ldots, e_m)$ приводит к 2^d решениям X в случае выполнения неравенства $e_1 + \cdots + e_m \le r$ и что $p_1^{e_1} \ldots p_m^{e_m}$ есть квадратичный остаток по модулю q_i при $1 \le i \le d$. Такие последовательности показателей могут быть вычислены как упорядоченные пары $(e'_1, \ldots, e'_m; e''_1, \ldots, e''_m)$, где $e'_1 + \cdots + e'_m \le$

$\frac{1}{2}r$, $e_1'' + \cdots + e_m'' \le \frac{1}{2}r$ и

$$(p_1^{e_1'} \ldots p_m^{e_m'})^{(q_i-1)/2} \equiv (p_1^{e_1''} \ldots p_m^{e_m''})^{(q_i-1)/2} \text{ (по модулю } q_i) \qquad \text{при } 1 \le i \le d.$$

31. [*M20*] Используя результаты упр. 1.2.10–21, оцените вероятность того, что алгоритм разложения на простые множители Диксона (описанный перед изложением теоремы D) вычисляет менее чем $2m$ выходных значений.

▶ **32.** [*M21*] Покажите, как улучшить RSA-схему кодирования так, чтобы не было проблем с сообщениями $< \sqrt[3]{N}$ в том смысле, что длина сообщений не должна существенно увеличиться.

33. [*M50*] Докажите или опровергните следующее утверждение: существует достаточно эффективный алгоритм, такой, что если для заданного числа $N = pq$, простые множители которого удовлетворяют условию $p \equiv q \equiv 2$ (по модулю 3), и заданного значения $x^3 \bmod N$ он с вероятностью, не являющейся пренебрежимо малой, может найти значение $x \bmod N$, то существует достаточно эффективный алгоритм, способный с подобной вероятностью найти множители числа N. [Если данное утверждение удастся доказать, это будет означать не только то, что проблема кубического корня так же сложна, как и проблема разложения на простые множители, но и то, что как RSA-схема, так и SQRT-схема обладают одним и тем же неустранимым недостатком.]

34. [*M30*] (Петер Уайнбергер (Peter Weinberger).) Предположим, что в RSA-схеме $N = pq$ и известно число m, такое, что по меньшей мере для 10^{-12} всех положительных целых чисел x выполняется равенство $x^m \bmod N = 1$. Поясните, как без больших трудностей решить задачу разложения на простые множители числа N, если число m не слишком велико (скажем, $m < N^{10}$).

▶ **35.** [*M25*] (Х. К. Уильямс (H. C. Williams), 1979.) Пусть N — произведение двух простых чисел p и q, где $p \bmod 8 = 3$ и $q \bmod 8 = 7$. Докажите, что символ Якоби удовлетворяет равенству $(\frac{-x}{N}) = (\frac{x}{N}) = -(\frac{2x}{N})$, и используйте его для построения схемы кодирования и/или декодирования, которая аналогична SQRT-блоку Рабина, исключив при этом двусмысленность сообщений.

36. [*HM24*] Асимптотическая оценка времени выполнения алгоритма E в виде уравнения (22) является слишком грубой для того, чтобы использовать ее для получения осмысленных значений числа N, если только оно не является очень большим, поскольку при значениях числа N, с которыми обычно приходится иметь дело, величина $\ln \ln N$ всегда сравнительно мала. Выполните более тонкий анализ, позволяющий уточнить поведение уравнения (22) для приемлемых значений числа N. Поясните также, как выбирать значение $\ln m$, минимизирующее (22), за исключением случаев, когда множитель достигает величины $\exp(O(\log \log N))$.

37. [*M27*] Докажите, что квадратный корень из любого положительного целого числа D, если только оно не является точным квадратом, имеет периодическую цепную дробь вида

$$\sqrt{D} = R + //a_1, \ldots, a_n, 2R, a_1, \ldots, a_n, 2R, a_1, \ldots, a_n, 2R, \ldots //,$$

где $R = \lfloor \sqrt{D} \rfloor$ и $(a_1, \ldots, a_n)$ — *палиндром* (т. е. симметричная последовательность $a_i = a_{n+1-i}$ при $1 \le i \le n$).

38. [*25*] (*Интересные простые числа.*) Для $0 \le d \le 9$ найдите наибольшее 50-разрядное простое число P_d, которое содержит максимально возможное количество десятичных разрядов, равных d. (Сначала найдите максимум по d, а затем — наибольшее такое число.)

39. [*40*] Для многих простых чисел p характерно свойство, состоящее в том, что числа $2p + 1$ также являются простыми, например $5 \to 11 \to 23 \to 47$. Обобщая, можно сказать, что число q — *преемник* числа p, если p и q — оба простые числа и $q = 2^k p + 1$ для некоторого

$k \geq 0$. Например, $2 \to 3 \to 7 \to 29 \to 59 \to 1\,889 \to 3779 \to 7\,559 \to 4\,058\,207\,223\,809 \to 32\,465\,657\,790\,473 \to 4\,462\,046\,030\,502\,692\,971\,872\,257 \to 95\langle 30 \text{ разрядов пропущено} \rangle 37 \to \cdots$; наименьший преемник для числа $95 \ldots 37$ содержит 103 цифры.

Найдите максимально длинную цепочку преемников простых чисел.

▶ **40.** [*M36*] (A. Шамир (A. Shamir).) Рассмотрим абстрактный автомат, который может в течение одного цикла выполнять операции $x + y$, $x - y$, $x \cdot y$ и $\lfloor x/y \rfloor$ над целыми числами x и y произвольной длины; длина таких чисел не имеет значения. Автомат хранит их в памяти с произвольным доступом и может выбирать для выполнения операций различные шаги программы в зависимости от того, будет ли для заданных чисел x и y выполняться равенство $x = y$. Назначение данного упражнения — показать, что в таком автомате можно применить изумительно быстрый способ разложения чисел на простые множители. (Поэтому, вероятно, будет достаточно трудно показать, что на *реальных* компьютерах разложение на простые множители выполнить исключительно сложно, хотя мы и подозреваем, что это так.)

a) Для заданного целого числа $n \geq 2$ найдите способ вычисления на таком гипотетическом автомате величины $n!$ за $O(\log n)$ циклов. [*Указание.* Если A — достаточно большое целое число, то биномиальные коэффициенты $\binom{m}{k} = m!/(m-k)!\,k!$ можно легко вычислить по значению числа $(A+1)^m$.]

b) Покажите, как на таком автомате вычислить число $f(n)$ за $O(\log n)$ циклов для заданного целого значения $n \geq 2$ со следующими свойствами: $f(n) = n$, если n — простое число, в противном случае $f(n)$ — собственный делитель (необязательно простой) числа n. [*Указание.* Если $n \neq 4$, то одной из таких функций $f(n)$ является $\gcd(m(n), n)$, где $m(n) = \min\{m \mid m! \bmod n = 0\}$.]

(Как следствие (b), полное разложение на простые множители произвольно большого числа n можно найти, выполнив только $O(\log n)^2$ арифметических операций. Для заданного частичного разложения $n = n_1 \ldots n_r$ каждое из чисел n_i, не являющихся простыми, можно заменить функциями $f(n_i) \cdot (n_i/f(n_i))$ за $\sum O(\log n_i) = O(\log n)$ циклов. Этот процесс можно повторять до тех пор, пока все числа n_i не станут простыми.)

▶ **41.** [*M28*] (Лагариас (Lagarias), Миллер (Miller) и Одлыжко (Odlyzko).) Назначение этого упражнения — показать, что может быть вычислено количество простых чисел, меньших N^3, если принимать во внимание только простые числа, меньшие N^2, и, таким образом, вычислять величину $\pi(N^3)$ за $O(N^{2+\epsilon})$ шагов.

Положительное целое число, для которого все простые множители превышают число m, назовем m-долгожителем; так что один m-долгожитель останется в решете Эратосфена (упр. 8) после просеивания всех чисел, кратных простым числам, не превышающим m. Пусть $f(x, m)$ — количество m-долгожителей, которое не превышает x, и пусть $f_k(x, m)$ — количество таких долгожителей, для которых имеется ровно k простых множителей (учитывая кратность).

a) Докажите, что $\pi(N^3) = \pi(N) + f(N^3, N) - 1 - f_2(N^3, N)$.

b) Поясните, как вычислить $f_2(N^3, N)$ по значениям $\pi(x)$ для $x \leq N^2$. Используйте метод вычисления значения $f_2(1000, 10)$ вручную.

c) Та же задача, что и в (b), но вместо $f_2(N^3, N)$ вычислите $f_2(N^3, N)$. [*Указание.* Используйте тождество $f(x, p_j) = f(x, p_{j-1}) - f(x/p_j, p_{j-1})$, где p_j есть j-е простое и $p_0 = 1$.]

d) Проанализируйте структуры данных, необходимые для эффективного вычисления величин при решении задач (b) и (c).

42. [*M35*] (Х. В. Ленстра (мл.) (H. W. Lenstra, Jr).) Для заданного интервала $0 < r < s < N$, в котором $r \perp s$ и $N \perp s$, покажите, что можно найти все делители числа N, равные

$\equiv r$ (по модулю s), выполнив $O(\lceil N/s^3\rceil^{1/2}\log s)$ хорошо подобранные арифметические операции над $(\lg N)$-битовыми числами. [*Указание.* Примените результаты упр. 4.5.3–49.]

▸ **43.** [*M43*] Пусть $m = pq$ — r-битовое целое число Блюма, как в теореме 3.5P, и пусть $Q_m = \{y \mid y = x^2 \bmod m \text{ для некоторого } x\}$. Тогда Q_m имеет $(p+1)(q+1)/4$ элементов и каждый из этих элементов $y \in S_m$ имеет единственный квадратный корень $x = \sqrt{y}$, такой, что $x \in Q_m$. Предположим, что $G(y)$ — это алгоритм, который правильно предсказывает значение $\sqrt{y} \bmod 2$ с вероятностью $\geq \frac{1}{2} + \epsilon$, где y — случайный элемент Q_m. Цель этого упражнения — доказать, что задача, решаемая при помощи G, почти так же трудна, как и задача разложения на простые множители числа m.

a) Разработайте алгоритм $A(G, m, \epsilon, y, \delta)$, который использует случайные числа, и алгоритм G для того, чтобы предсказать без обязательного вычисления $\sqrt{y}$, будет ли данное целое число y принадлежать Q_m. Результат выполнения алгоритма должен быть корректным с вероятностью $\geq 1 - \delta$, а время $T(A)$ его выполнения не должно превышать величины $O(\epsilon^{-2}(\log \delta^{-1})T(G))$ в предположении, что $T(G) \geq r^2$. (Если $T(G) < r^2$, в этой формуле замените $T(G)$ величиной $(T(G) + r^2)$.)

b) Разработайте алгоритм $F(G, m, \epsilon)$, который находит множители числа m и ожидаемое время выполнения которого равно $T(F) = O(r^2(\epsilon^{-6} + \epsilon^{-4}(\log \epsilon^{-1})T(G)))$.

Указание. Для $0 \leq v < m$ и фиксированного $y \in Q_m$ положим $\tau v = v\sqrt{y} \bmod m$ и $\lambda v = \tau v \bmod 2$. Учтем, что $\lambda(-v) + \lambda v = 1$ и $\lambda(v_1 + \cdots + v_n) = (\lambda v_1 + \cdots + \lambda v_n + \lfloor(\tau v_1 + \cdots + \tau v_m)/m\rfloor) \bmod 2$. Имеем далее $\tau(\frac{1}{2}v) = \frac{1}{2}(\tau v + m\lambda v)$; здесь величина $\frac{1}{2}v$ заменяет $(\frac{m+1}{2}v) \bmod m$. Если $\pm v \in Q_m$, имеем $\tau(\pm v) = \sqrt{v^2 y}$, поэтому алгоритм G обеспечивает способ предсказания величины λv для примерно половины всех v.

44. [*M35*] (Й. Хаастад (J. Håstad).) Покажите, что нетрудно найти x в случае, когда $a_{i0} + a_{i1}x + a_{i2}x^2 + a_{i3}x^3 \equiv 0$ (по модулю m_i), $0 < x < m_i$, $\gcd(a_{i0}, a_{i1}, a_{i2}, a_{i3}, m_i) = 1$ и $m_i > 10^{27}$ при $1 \leq i \leq 7$, если $m_i \perp m_j$ при $1 \leq i < j \leq 7$. (Все переменные — целые числа, и все они, кроме x, известны.) *Указание.* Когда L — произвольная неособая матрица вещественных элементов, алгоритм Ленстра (Lenstra) и Ловача (Lovász) [*Mathematische Annalen* **261** (1982), 515–534] эффективно находит ненулевой целочисленный вектор $v = (v_1, \ldots, v_n)$, такой, что длина $(vL) \leq \sqrt{n2^n}\,|\det L|^{1/n}$.

▸ **45.** [*M41*] (Дж. М. Поллард (J. M. Pollard) и Клаус-Петер Шнорр (Claus-Peter Schnorr).) Покажите, что для целых чисел x и y, заданных целых чисел a, b и n, для которых $ab \perp n$ и n нечетно, существует эффективный способ решения уравнения

$$x^2 - ay^2 \equiv b \quad (\text{по модулю } n)$$

даже в том случае, когда неизвестно разложение на простые множители числа n. [*Указание.* Используйте тождество $(x_1^2 - ay_1^2)(x_2^2 - ay_2^2) = x^2 - ay^2$, где $x = x_1x_2 - ay_1y_2$ и $y = x_1y_2 + x_2y_1$.]

46. [*HM30*] (Л. Эдлеман (L. Adleman).) Пусть p — достаточно большое простое число и a — простой корень по модулю p; таким образом, все целые числа b в интервале $1 \leq b < p$ могут быть записаны в виде $b = a^n \bmod p$ для некоторого единственного числа n из интервала $1 \leq n < p$.

Используя идеи, аналогичные идеям из алгоритма Диксона разложения на простые множители, разработайте алгоритм, который почти всегда по заданному b находит число n за $O(p^\epsilon)$ шагов для всех $\epsilon > 0$. [*Указание.* Начните с формирования набора чисел n_i, таких, что число $a^{n_i} \bmod p$ имеет только малые простые множители.]

47. [*M50*] Некоторая цитата из литературного источника $x = x_1x_2$, представленная в ASCII-кодах, в зашифрованном виде выглядит как $(x_1^3 \bmod N,\ x_2^3 \bmod N) =$

(14E97EF5C531D92591B89CDBAB48444A04612C01AA29C2A8FA10FA804EF7AC3CE03D7D3667C4D3E132A24A68
E6797FE28650DC3ADF327474B86B0CBD5387A49872CE012269A59B3E4B3BD83B74681A78AD7B6D1772A7451B,
15B025E2AEE095A9542590184CF62F72B2E8E8DD794AEF8511F2591E6BC2C8B8A8E48AF1FE04FF2FD933E730
9205A3418DBB9BB8C6A7665DA309531735FE86C741D1261B34CB2668FA34D0C0C28575A2454E3DB00E408AC7)

в шестнадцатеричных обозначениях, где N равно

17B2353B9595ECA69FEF80940160C4084286D1255FFE49D114F2E633F82C88D5224FC4AA6F9104CED2BCA810
BEA76157FFDC78F9656A0ED9B3F6CCAB99001B8B2571F4EBD095925F07F9BEE5111E8375DFD71593628AD8D1.

Что собой представляет x?

Проблема распознавания простых чисел из составных
и разложения составных чисел на простые множители
является одной из самых важных и полезных среди всех арифметических задач.
... Высокое призвание науки в том, кажется, и состоит,
чтобы любой вклад в решение такой элегантной и знаменитой проблемы
усердно культивировался.

— К. Ф. ГАУСС (C. F. GAUSS), *Disquisitiones Arithmeticæ*, Article 329 (1801)

4.6. ПОЛИНОМИАЛЬНАЯ АРИФМЕТИКА

Изученные нами технологии естественным образом применимы не только к числам, но и к различным математическим величинам. В этом разделе речь пойдет о полиномах, что представляет шаг вперед по сравнению с числами. Формально говоря, *полином над* S представляет собой выражение вида

$$u(x) = u_n x^n + \cdots + u_1 x + u_0, \tag{1}$$

где *коэффициенты* $u_n, \ldots, u_1, u_0$ — элементы некоторой алгебраической системы S, а *переменная* x может рассматриваться как формальный символ без определенного значения. Будем полагать, что алгебраическая система S представляет собой *коммутативное кольцо с единицей*. Это означает, что S допускает операции сложения, вычитания и умножения, удовлетворяющие обычным свойствам: сложение и умножение являются ассоциативными и коммутативными бинарными операциями, определенными на S, причем умножение дистрибутивно по отношению к сложению. Существует также единичный элемент по сложению 0 и единичный элемент по умножению 1, такие, что $a + 0 = a$ и $a \cdot 1 = a$ для всех a из S. Вычитание является обратной по отношению к сложению операцией, но о возможности деления как об операции, обратной по отношению к умножению, ничего не предполагается. Полином $0x^{n+m} + \cdots + 0x^{n+1} + u_n x^n + \cdots + u_1 x + u_0$ рассматривается как идентичный полиному (1), хотя формально он отличается от него.

Мы говорим, что (1) является полиномом *степени* n со *старшим коэффициентом* u_n, если $u_n \neq 0$; в этом случае запишем*

$$\deg(u) = n, \qquad \ell(u) = u_n. \tag{2}$$

Кроме того, по определению

$$\deg(0) = -\infty, \qquad \ell(0) = 0, \tag{3}$$

где 0 означает нулевой полином, т. е. полином, все коэффициенты которого равны нулю. Мы говорим также, что $u(x)$ — *нормированный полином*, если его старший коэффициент $\ell(u)$ равен 1.

Арифметика полиномов состоит, в первую очередь, из сложения, вычитания и умножения; иногда к этим операциям добавляются другие, например, деление, возведение в степень, разложение на множители и поиск наибольшего общего делителя. Сложение, вычитание и умножение определяются естественным образом, как если бы переменная x была элементом S: мы складываем или вычитаем полиномы посредством сложения или вычитания коэффициентов при одинаковых степенях x. Умножение выполняется согласно правилу

$$(u_r x^r + \cdots + u_0)(v_s x^s + \cdots + v_0) = w_{r+s} x^{r+s} + \cdots + w_0,$$

где

$$w_k = u_0 v_k + u_1 v_{k-1} + \cdots + u_{k-1} v_1 + u_k v_0. \tag{4}$$

В последней формуле u_i и v_j рассматриваются как равные нулю при $i > r$ и $j > s$ соответственно.

* Здесь символ ℓ означает *leading* (*ведущий*; в русскоязычной математической литературе — *старший*). — *Прим. перев.*

Алгебраическая система S обычно представляет собой множество целых или рациональных чисел. Она может быть и множеством полиномов (с другими, отличными от x переменными); тогда (1) — полином от нескольких переменных. В частности, алгебраическая система S может состоять из целых чисел 0, 1, ..., $m-1$ со сложением, вычитанием и умножением, выполняемыми по модулю m (см. формулу 4.3.2–(11)); этот важный случай называется *полиномиальной арифметикой по модулю* m. Особенно важна полиномиальная арифметика по модулю 2, когда каждый коэффициент равен 0 или 1.

Читатель должен обратить внимание на сходство между полиномиальной арифметикой и арифметикой многократной точности (раздел 4.3.1), где основание счисления b заменено на x. Основное отличие состоит в том, что коэффициент u_k при x^k в полиномиальной арифметике, вообще говоря, никак не связан с соседними коэффициентами $u_{k\pm1}$, так что понятие "перенос" из одного места в другое в полиномиальной арифметике отсутствует. В действительности полиномиальная арифметика по модулю b, по существу, идентична арифметике с многократной точностью по основанию b за исключением отсутствия переносов. Например, сравним умножение $(1101)_2$ на $(1011)_2$ в двоичной системе счисления с аналогичным умножением x^3+x^2+1 на x^3+x+1 по модулю 2.

Двоичные числа	Полиномы по модулю 2
1101	1101
× 1011	× 1011
1101	1101
1101	1101
1101	1101
10001111	1111111

Произведение этих полиномов по модулю 2 получено путем отказа от всех переносов и составляет $x^6+x^5+x^4+x^3+x^2+x+1$. Если умножать полиномы так же, как целые числа, без получения остатков по модулю 2, результат будет равен $x^6+x^5+x^4+3x^3+x^2+x+1$; переносы в данном случае также не используются, однако коэффициенты в произведении могут оказаться произвольно большими.

В связи с сильным сходством полиномиальной арифметики и арифметики с многократной точностью нет необходимости подробно обсуждать операции сложения, вычитания и умножения в этом разделе. Однако нужно отметить некоторые аспекты, отличающие практическое использование полиномиальной арифметики от арифметики многократной точности. Очень часто при работе с полиномами наблюдается тенденция к появлению большого количества нулевых коэффициентов и полиномов огромных степеней, а потому желательно использовать специальные формы представления полиномов (см. раздел 2.2.4). Кроме того, арифметика полиномов от нескольких переменных приводит к программам, которые легче всего понять в рамках рекурсии (эта ситуация будет обсуждаться в главе 8). Хотя методы сложения, вычитания и умножения полиномов сравнительно просты и понятны, несколько важных аспектов полиномиальной арифметики достойны специального рассмотрения. В следующих разделах будут обсуждаться *деление* полиномов и связанные с ним методы, такие как поиск наибольшего общего делителя и разложение на множители. Мы рассмотрим также проблему эффективного *вычисления*

полиномов, т. е. задачу поиска значения $u(x)$ при данном $x \in S$ с использованием минимально возможного числа операций. Частный случай вычисления x^n при больших n достаточно важен и разбирается отдельно в разделе 4.6.3.

Первым большим набором компьютерных подпрограмм для работы с полиномиальной арифметикой стала система ALPAK [W. S. Brown, J. P. Hyde, and B. A. Tague, *Bell System Tech. J.* **42** (1963), 2081–2119; **43** (1964), 785–804, 1547–1562]. Другой ранней вехой в этой области стала PM system Джорджа Коллинза (George Collins) [*CACM* **9** (1966), 578–589; см. также C. L. Hamblin, *Comp. J.* **10** (1967), 168–171].

УПРАЖНЕНИЯ

1. [*10*] При работе с полиномиальной арифметикой по модулю 10 чему будет равна разность $7x + 2$ и $x^2 + 5$? Чему будет равно произведение $6x^2 + x + 3$ и $5x^2 + 2$?

2. [*17*] Истинны ли следующие утверждения? (a) Произведение нормированных полиномов нормировано. (b) Произведение полиномов степени m и n имеет степень $m + n$. (c) Сумма полиномов степени m и n имеет степень $\max(m, n)$.

3. [*M20*] Если каждый из коэффициентов $u_r, \ldots, u_0, v_s, \ldots, v_0$ в (4) представляет собой целое число, удовлетворяющее условиям $|u_i| \le m_1$, $|v_j| \le m_2$, то чему равно максимальное абсолютное значение произведения коэффициентов w_k?

▶ **4.** [*21*] Можно ли умножение полиномов по модулю 2 упростить с помощью обычных арифметических операций на двоичном компьютере, если коэффициенты упакованы в машинные слова?

▶ **5.** [*M21*] Покажите, как можно умножить два полинома степени $\le n$ по модулю 2 со временем умножения, пропорциональным $O(n^{\lg 3})$ при больших n, адаптируя метод Карацубы (см. раздел 4.3.3).

4.6.1. Деление полиномов

Разделить один полином на другой можно так же, как одно целое число с многократной точностью на другое, при выполнении арифметических операций с полиномами над *полем*. Поле S представляет собой коммутативное кольцо с единицей, в котором точное деление возможно так же, как и операции сложения, вычитания и умножения. Как обычно, это означает, что для любых $u, v \in S$ и $v \neq 0$ в S всегда имеется элемент w, такой, что $u = vw$. Наиболее важными полями коэффициентов, появляющимися в приложениях, являются

a) рациональные числа (представленные в виде дробей; см. раздел 4.5.1);
b) действительные или комплексные числа (представленные в компьютере как приближения с плавающей точкой; см. раздел 4.2);
c) целые по модулю p, где p — простое число (деление может быть реализовано так, так предложено в упр. 4.5.2–16);
d) *рациональные функции* над полем, т. е. частное двух полиномов, коэффициенты которых находятся в этом поле, а знаменатель нормирован.

Особо важный случай представляет собой поле целых по модулю 2, в котором элементы могут принимать значения 0 и 1. Полиномы над этим полем (т. е. полиномы по модулю 2) имеют много общего с целыми числами в двоичной записи; рациональные функции над данным полем поразительно схожи с рациональными числами, числители и знаменатели которых представлены в двоичной записи.

Если даны два полинома, $u(x)$ и $v(x)$, над полем и $v(x) \neq 0$, можно разделить $u(x)$ на $v(x)$ и получить полином-частное $q(x)$ и полином-остаток $r(x)$, удовлетворяющие условиям

$$u(x) = q(x) \cdot v(x) + r(x), \qquad \deg(r) < \deg(v). \tag{1}$$

Легко увидеть, что существует не более одной пары полиномов $\big(q(x), r(x)\big)$, удовлетворяющих этим соотношениям; в самом деле, если условиям (1) при одних и тех же полиномах $u(x)$ и $v(x)$ удовлетворяют две пары — $\big(q_1(x), r_1(x)\big)$ и $\big(q_2(x), r_2(x)\big)$, — то $q_1(x)v(x) + r_1(x) = q_2(x)v(x) + r_2(x)$, т. е. $\big(q_1(x) - q_2(x)\big)v(x) = r_2(x) - r_1(x)$. Теперь, если $q_1(x) - q_2(x) \neq 0$, имеем $\deg\big((q_1 - q_2) \cdot v\big) = \deg(q_1 - q_2) + \deg(v) \geq \deg(v) > \deg(r_2 - r_1)$, т. е. получаем противоречие (так как из равенства $\big(q_1(x) - q_2(x)\big)v(x) = r_2(x) - r_1(x)$ следует $\deg\big((q_1 - q_2) \cdot v\big) = \deg(r_1 - r_2)$ — *Прим. перев.*). Значит, $q_1(x) - q_2(x) = 0$ и $r_1(x) = r_2(x)$.

Чтобы определить $q(x)$ и $r(x)$, можно использовать алгоритм 4.3.1D для деления чисел с многократной точностью, только без переносов.

Алгоритм D (*Деление полиномов над полем*). Даны полиномы

$$u(x) = u_m x^m + \cdots + u_1 x + u_0, \qquad v(x) = v_n x^n + \cdots + v_1 x + v_0$$

над полем S, где $v_n \neq 0$ и $m \geq n \geq 0$. Алгоритм находит полиномы

$$q(x) = q_{m-n} x^{m-n} + \cdots + q_0, \qquad r(x) = r_{n-1} x^{n-1} + \cdots + r_0$$

над полем S, удовлетворяющие условиям (1).

D1. [Итерация по k.] Выполнять шаг D2 для $k = m - n, m - n - 1, \ldots, 0$; затем прекратить выполнение алгоритма с $(r_{n-1}, \ldots, r_0) = (u_{n-1}, \ldots, u_0)$.

D2. [Цикл деления.] Установить сначала $q_k \leftarrow u_{n+k}/v_n$, а затем — $u_j \leftarrow u_j - q_k v_{j-k}$ для $j = n + k - 1, n + k - 2, \ldots, k$. (Действие последней операции состоит в замещении $u(x)$ на $u(x) - q_k x^k v(x)$, полином степени $< n + k$.) ▮

Пример использования алгоритма D приводится ниже, в (5). Количество выполняемых арифметических операций, в сущности, пропорционально $n(m - n + 1)$. Обратите внимание, что явное деление коэффициентов выполняется только в начале шага D2 и знаменателем всегда является v_n. Таким образом, если $v(x)$ — нормированный полином (с $v_n = 1$), реальное деление не производится вовсе. Если умножение более предпочтительно, чем деление, имеет смысл предварительно вычислить значение $1/v_n$ и на шаге D2 выполнить умножение на эту величину.

Зачастую остаток $r(x)$ из (1) будет записываться как $u(x) \bmod v(x)$.

Области единственного разложения на множители. Если мы ограничимся рассмотрением полиномов над полем, то не охватим полиномы над множеством целых чисел и полиномы от нескольких переменных. Поэтому будем рассматривать более общую ситуацию, когда алгебраическая система коэффициентов S представляет собой *область единственного разложения на множители* и не обязана быть полем. Это означает, что S — коммутативное кольцо с единицей и что

i) $uv \neq 0$, если u и v — ненулевые элементы S;

ii) каждый ненулевой элемент $u \in S$ либо *обратим*, либо имеет "единственное" представление в виде произведения *простых* $p_1, \ldots, p_t$:

$$u = p_1 \ldots p_t, \qquad t \geq 1. \tag{2}$$

Обратимый элемент в данном случае представляет собой элемент, который имеет обратную величину, а именно — элемент u, такой, что $uv = 1$ для некоторого $v \in S$. *Простой* элемент — это не являющийся обратимым элемент p, такой, что уравнение $p = qr$ истинно только тогда, когда либо q, либо r представляет собой обратимый элемент. Представление (2) единственно в том смысле, что если $p_1 \ldots p_t = q_1 \ldots q_s$, где все p и q просты, то $s = t$ и имеется перестановка $\pi_1 \ldots \pi_t$ множества $\{1, \ldots, t\}$, такая, что $p_1 = a_1 q_{\pi_1}, \ldots, p_t = a_t q_{\pi_t}$ для некоторых обратимых $a_1, \ldots, a_t$. Другими словами, разложение на простые множители единственно с точностью до порядка множителей и до умножения на обратимые.

Любое поле представляет собой область единственного разложения на множители, в которой каждый ненулевой элемент обратим и в которой не существует простых элементов. Целые числа образуют область единственного разложения, в которой обратимые элементы — $+1$ и -1, а простые — ± 2, ± 3, ± 5, ± 7, ± 11 и т. д. Случай, когда S содержит все целые числа, принципиален, поскольку зачастую предпочтительнее работать с целыми коэффициентами, а не с произвольными рациональными.

Одним из ключевых фактов, связанных с полиномами (см. упр. 10), является то, что *полиномы над областью единственного разложения образуют область единственного разложения.* Полином, который является простым в этой области, обычно называется *неприводимым.* Многократно используя теорему о единственности разложения, можно доказать, что полиномы от нескольких переменных над множеством целых чисел или над любым полем с любым числом переменных могут быть единственным образом разложены на неприводимые полиномы. Например, полином от трех переменных $90x^3 - 120x^2y + 18x^2yz - 24xy^2z$ над целыми числами представляет собой произведение пяти неприводимых полиномов $2 \cdot 3 \cdot x \cdot (3x - 4y) \cdot (5x + yz)$. Тот же полином над рациональными числами является произведением трех неприводимых полиномов $(6x) \cdot (3x - 4y) \cdot (5x + yz)$. Это разложение может быть записано как $x \cdot (90x - 120y) \cdot (x + \frac{1}{5}yz)$ и еще бесконечным числом способов, хотя разложение, по существу, единственно.

Как обычно, мы говорим, что $u(x)$ *кратен* полиному $v(x)$, а $v(x)$ является *делителем* $u(x)$, если $u(x) = v(x)q(x)$ для некоторого полинома $q(x)$. Если есть алгоритм, который определяет, является ли u кратным v для произвольных ненулевых элементов u и v некоторой области единственного разложения S, и позволяет определить w, если $u = v \cdot w$, то алгоритм D дает метод для выяснения, является ли $u(x)$ кратным $v(x)$ для произвольных ненулевых полиномов $u(x)$ и $v(x)$ над S. Если $u(x)$ кратно $v(x)$, легко увидеть, что u_{n+k} должно быть кратно v_n при переходе к шагу D2; отсюда будет найдено частное $u(x)/v(x)$. Применяя это наблюдение рекурсивно, получим алгоритм, который отвечает на вопрос, является ли данный полином над S от любого числа переменных кратным другому полиному над S, а также алгоритм, который будет находить частное, если таковое существует.

Множество элементов области единственного разложения называется *взаимно простым*, если нет простого элемента этой области единственного разложения, который делит все остальные элементы. Полином над областью единственного

разложения называется *примитивным*, если его коэффициенты взаимно просты (не путайте это понятие с совершенно другим понятием *примитивных полиномов по модулю p*, которое обсуждалось в разделе 3.2.2). Следующий факт, представленный для полиномов над целыми числами К. Ф. Гауссом (C. F. Gauss) в 42 разделе его знаменитой книги *Disquisitiones Arithmeticæ* (Leipzig, 1801), имеет первостепенную важность.

Лемма G (Лемма Гаусса). *Произведение примитивных полиномов над областью единственного разложения примитивно.*

Доказательство. Пусть $u(x) = u_m x^m + \cdots + u_0$ и $v(x) = v_n x^n + \cdots + v_0$ — примитивные полиномы. Пусть p — любой простой элемент области; тогда следует показать, что p не делит все коэффициенты $u(x)v(x)$. По условию имеется такой индекс j, что u_j не делится на p, и такой индекс k, что v_k не делится на p. Пусть j и k — наименьшие такие индексы; тогда коэффициент при x^{j+k} в $u(x)v(x)$ равен

$$u_j v_k + u_{j+1} v_{k-1} + \cdots + u_{j+k} v_0 + u_{j-1} v_{k+1} + \cdots + u_0 v_{k+j}.$$

Так как первый член этой суммы не делится на p, а все остальные делятся, на p не делится и вся сумма. ▮

Если ненулевой полином $u(x)$ над областью единственного разложения S не примитивен, можно записать $u(x) = p_1 \cdot u_1(x)$, где p_1 — простой элемент из S, делящий все коэффициенты $u(x)$, а $u_1(x)$ — другой ненулевой полином над S. Все коэффициенты $u_1(x)$ имеют на один простой множитель меньше соответствующих коэффициентов $u(x)$. Теперь, если $u_1(x)$ не примитивен, можно записать $u_1(x) = p_2 \cdot u_2(x)$ и т. д. Этот процесс безусловно прекратится при представлении $u(x) = c \cdot u_k(x)$, где $c \in S$ и $u_k(x)$ — примитивен. Фактически имеется соответствующая лемме G лемма H.

Лемма H. *Любой ненулевой полином $u(x)$ над областью единственного разложения S можно разложить на множители в виде $u(x) = c \cdot v(x)$, где c представляет собой элемент S, а $v(x)$ — примитивный полином. Кроме того, это представление единственно в том смысле, что если $u = c_1 \cdot v_1(x) = c_2 \cdot v_2(x)$, то $c_1 = ac_2$ и $v_2(x) = av_1(x)$, где a — обратимый элемент S.*

Доказательство. Мы уже показали, что такое разложение существует, так что доказательства требует только единственность разложения. Положим, что $c_1 \cdot v_1(x) = c_2 \cdot v_2(x)$, где $v_1(x)$ и $v_2(x)$ — примитивные полиномы. Пусть p — некоторое простое из S. Если p^k делит c_1, то оно делит и c_2; в противном случае p^k должно делить все коэффициенты $c_2 \cdot v_2(x)$ и p должно делить все коэффициенты $v_2(x)$. Таким образом, получается противоречие с посылкой примитивности $v_2(x)$. Точно так же p^k делит c_2 только в том случае, если p^k делит c_1. Следовательно, в силу единственности разложения $c_1 = ac_2$, где a обратимо; поскольку $0 = ac_2 \cdot v_1(x) - c_2 \cdot v_2(x) = c_2 \cdot \big(av_1(x) - v_2(x)\big)$, получаем $av_1(x) - v_2(x) = 0$. ▮

Таким образом, можно записать любой ненулевой полином $u(x)$ в виде

$$u(x) = \operatorname{cont}(u) \cdot \operatorname{pp}\big(u(x)\big), \tag{3}$$

где $\operatorname{cont}(u)$ — *содержание* (*content*) u, представляющее собой элемент S, а $\operatorname{pp}\big(u(x)\big)$ — *примитивная часть* (*primitive part*) $u(x)$, являющаяся примитивным полиномом

над S. В случае, когда $u(x) = 0$, удобно определить $\operatorname{cont}(u) = \operatorname{pp}\big(u(x)\big) = 0$. Комбинация лемм G и H приводит к соотношениям

$$\begin{aligned} \operatorname{cont}(u \cdot v) &= a \operatorname{cont}(u) \operatorname{cont}(v), \\ \operatorname{pp}\big(u(x) \cdot v(x)\big) &= b \operatorname{pp}\big(u(x)\big) \operatorname{pp}\big(v(x)\big), \end{aligned} \tag{4}$$

где a и b — обратимые элементы, зависящие от пути вычисления содержимого, причем $ab = 1$. При работе с полиномами над целыми числами обратимыми элементами являются только $+1$ и -1, и поэтому удобно определить $\operatorname{pp}\big(u(x)\big)$ так, чтобы ее старший коэффициент был положителен; тогда (4) истинно при $a = b = 1$. При работе с полиномами над полем можно принять $\operatorname{cont}(u) = \ell(u)$, так что $\operatorname{pp}\big(u(x)\big)$ будет нормированным полиномом; в этом случае (4) вновь выполняется при $a = b = 1$ для всех $u(x)$ и $v(x)$.

Например, пусть мы работаем с полиномами над целыми числами и пусть $u(x) = -26x^2 + 39$ и $v(x) = 21x + 14$. Тогда

$$\begin{aligned} \operatorname{cont}(u) &= -13, & \operatorname{pp}\big(u(x)\big) &= 2x^2 - 3, \\ \operatorname{cont}(v) &= +7, & \operatorname{pp}\big(v(x)\big) &= 3x + 2, \\ \operatorname{cont}(u \cdot v) &= -91, & \operatorname{pp}\big(u(x) \cdot v(x)\big) &= 6x^3 + 4x^2 - 9x - 6. \end{aligned}$$

Наибольшие общие делители. В случае единственного разложения можно говорить о *наибольшем общем делителе* двух элементов; это общий делитель, который делится на наибольшее количество простых элементов (см. формулу 4.5.2–(6)). Однако, поскольку область единственного разложения может иметь много обратимых элементов, в определении наибольшего общего делителя присутствует некоторая неоднозначность. Если w — наибольший общий делитель u и v, то $a \cdot w$, где a — обратимый элемент, тоже является наибольшим общим делителем. И обратно, предположение о единственности разложения на множители влечет за собой вывод о том, что если и w_1, и w_2 представляют собой наибольшие общие делители u и v, то $w_1 = a \cdot w_2$, где a — некоторый обратимый элемент. Другими словами, не имеет смысла говорить о конкретном наибольшем общем делителе (в оригинале — "to speak of *the* greatest common divisor". — *Прим. перев.*) u и v. Существует целое множество наибольших общих делителей, каждый из которых отличается от других на обратимый множитель.

Рассмотрим теперь задачу поиска наибольшего общего делителя двух данных полиномов над алгебраической системой S, первоначально поставленную Пабло Нуньесом (Pablo Nuñez) в работе *Libro de Algebra* (Antwerp, 1567). Если S — поле, то проблема относительно проста; наш алгоритм деления, алгоритм D, может быть расширен до алгоритма вычисления наибольшего общего делителя так же, как алгоритм Евклида (алгоритм 4.5.2A), находящий наибольший общий делитель двух целых чисел, получается на основе алгоритма деления целых чисел:

если $v(x) = 0$, то $\gcd\big(u(x), v(x)\big) = u(x)$;
в противном случае $\gcd\big(u(x), v(x)\big) = \gcd\big(v(x), r(x)\big)$,

где $r(x)$ определяется (1). Эта процедура называется алгоритмом Евклида для полиномов над полем. Впервые она была использована Симоном Стевином (Simon Stevin) в *L'Arithmetique* (Leiden, 1585); см. A. Girard, *Les Œuvres Mathématiques de Simon Stevin* **1** (Leiden, 1634), 56.

Например, необходимо найти gcd $x^8 + x^6 + 10x^4 + 10x^3 + 8x^2 + 2x + 8$ и $3x^6 + 5x^4 + 9x^2 + 4x + 8$, mod 13, используя алгоритм Евклида для полиномов над полем целых чисел по модулю 13. Сначала запишем только коэффициенты для того, чтобы показать шаги алгоритма D.

$$\begin{array}{rl} & \qquad\qquad\qquad\quad 9\ 0\ 7 \\ 3\ 0\ 5\ 0\ 9\ 4\ 8 \;)\; & 1\ 0\ 1\ 0\ 10\ 10\ \ 8\ 2\ 8 \\ & \quad 1\ 0\ 6\ 0\ \ 3\ 10\ \ 7 \\ \hline & \qquad 0\ 8\ 0\ \ 7\ \ 0\ \ 1\ 2\ 8 \\ & \qquad\ 8\ 0\ \ 9\ \ 0\ 11\ 2\ 4 \\ \hline & \qquad\qquad 0\ 11\ \ 0\ \ 3\ 0\ 4 \end{array} \tag{5}$$

Так что $x^8 + x^6 + 10x^4 + 10x^3 + 8x^2 + 2x + 8$ равно

$$(9x^2 + 7)(3x^6 + 5x^4 + 9x^2 + 4x + 8) \;+\; (11x^4 + 3x^2 + 4).$$

Аналогично

$$\begin{aligned} 3x^6 + 5x^4 + 9x^2 + 4x + 8 &= (5x^2 + 5)(11x^4 + 3x^2 + 4) \;+\; (4x + 1); \\ 11x^4 + 3x^2 + 4 &= (6x^3 + 5x^2 + 6x + 5)(4x + 1) \;+\; 12;. \\ 4x + 1 &= (9x + 12) \cdot 12 \;+\; 0. \end{aligned} \tag{6}$$

(Знак равенства здесь означает равенство по модулю 13, так как все арифметические действия над коэффициентами выполняются по модулю 13.) Проведенное вычисление показывает, что 12 является наибольшим общим делителем двух исходных полиномов. Так как любой ненулевой элемент поля представляет собой обратимый элемент области полиномов над полем, для полей принято делить результат выполнения алгоритма на его старший коэффициент, получая *нормированный* полином, и именно его имеют в виду, говоря о наибольшем общем делителе (в оригинале — "is called *the* greatest common divisor". — *Прим. перев.*) двух данных полиномов. gcd, вычисленный в (6), таким образом, оказывается равным 1, а не 12. Последний шаг в (6) может быть опущен, так как если $\deg(v) = 0$, то $\gcd\big(u(x), v(x)\big) = 1$ безотносительно к выбору полинома $u(x)$. В упр. 4 определяется среднее время работы алгоритма Евклида над случайными полиномами по модулю p.

Вернемся к более общей ситуации, в которой полиномы даны над областью единственного разложения, не являющейся полем. Из (4) можно вывести важные соотношения

$$\begin{aligned} \operatorname{cont}\big(\gcd(u, v)\big) &= a \cdot \gcd\big(\operatorname{cont}(u), \operatorname{cont}(v)\big), \\ \operatorname{pp}\big(\gcd(u(x), v(x))\big) &= b \cdot \gcd\big(\operatorname{pp}\big(u(x)\big), \operatorname{pp}\big(v(x)\big)\big), \end{aligned} \tag{7}$$

где a и b — обратимые элементы. Здесь $\gcd\big(u(x), v(x)\big)$ означает любой полином от x, являющийся наибольшим общим делителем $u(x)$ и $v(x)$. Формулы (7) сводят проблему поиска наибольшего общего делителя произвольных полиномов к поиску наибольшего общего делителя *примитивных* полиномов.

Алгоритм D деления полиномов над полем может быть обобщен для *псевдоделения* полиномов над любой алгебраической системой, представляющей собой коммутативное кольцо с единицей. Можно заметить, что алгоритм D требует явного деления только на $\ell(v)$, старший коэффициент — $v(x)$ и шаг D2 выполняется в точности $m - n + 1$ раз. Таким образом, если $u(x)$ и $v(x)$ начинаются с целых коэффициентов

и если мы работаем над полем рациональных чисел, то единственными делителями $\ell(v)^{m-n+1}$ будут знаменатели $q(x)$ и $r(x)$. Это наблюдение приводит к мысли, что всегда можно найти полиномы $q(x)$ и $r(x)$, такие, что

$$\ell(v)^{m-n+1}u(x) = q(x)v(x) + r(x), \qquad \deg(r) < n, \tag{8}$$

где $m = \deg(u)$ и $n = \deg(v)$, для любых полиномов $u(x)$ и $v(x) \neq 0$ при условии, что $m \geq n$.

Алгоритм R (*Псевдоделение полиномов*). Даны полиномы

$$u(x) = u_m x^m + \cdots + u_1 x + u_0, \qquad v(x) = v_n x^n + \cdots + v_1 x + v_0,$$

где $v_n \neq 0$ и $m \geq n \geq 0$. Этот алгоритм находит полиномы $q(x) = q_{m-n}x^{m-n} + \cdots + q_0$ и $r(x) = r_{n-1}x^{n-1} + \cdots + r_0$, удовлетворяющие (8).

R1. [Итерация по k.] Выполнить шаг R2 для $k = m-n, m-n-1, \ldots, 0$; после этого алгоритм завершается, выдав ответ $(r_{n-1}, \ldots, r_0) = (u_{n-1}, \ldots, u_0)$.

R2. [Цикл умножения.] Установить сначала $q_k \leftarrow u_{n+k}v_n^k$, а затем — $u_j \leftarrow v_n u_j - u_{n+k}v_{j-k}$ для $j = n+k-1, n+k-2, \ldots, 0$. (При $j < k$ это означает, что $u_j \leftarrow v_n u_j$, поскольку мы рассматриваем v_{-1}, v_{-2}, ... как нули. Этих умножений можно избежать, если начать алгоритм с замены u_t на $v_n^{m-n-t}u_t$ для $0 \leq t < m-n$.) ▮

Пример вычислений приведен ниже, в (10). Правильность алгоритма R легко доказать индукцией по $m-n$, поскольку при каждом выполнении шага R2, по сути, $u(x)$ заменяется на $\ell(v)u(x) - \ell(u)x^k v(x)$, где $k = \deg(u) - \deg(v)$. Заметьте, что в алгоритме не использовано деление; коэффициенты $q(x)$ и $r(x)$ сами по себе представляют некоторые полиномиальные функции от коэффициентов $u(x)$ и $v(x)$. Если $v_n = 1$, то алгоритм идентичен алгоритму D. Если $u(x)$ и $v(x)$ — полиномы над областью единственного разложения, можно, как и ранее, доказать, что полиномы $q(x)$ и $r(x)$ — единственны, поэтому другой способ псевдоделения над областью единственного разложения состоит в умножении $u(x)$ на v_n^{m-n+1} и применении алгоритма D, если известно, что все частные на шаге D2 будут существовать.

Алгоритм R может быть расширен до "обобщенного алгоритма Евклида" для примитивных полиномов над областью единственного разложения следующим образом. Пусть $u(x)$ и $v(x)$ — примитивные полиномы с $\deg(u) \geq \deg(v)$. Определим при помощи алгоритма R полином $r(x)$, удовлетворяющий (8). Теперь можно доказать, что $\gcd\big(u(x), v(x)\big) = \gcd\big(v(x), r(x)\big)$: любой общий делитель $u(x)$ и $v(x)$ делит $v(x)$ и $r(x)$; и обратно, любой общий делитель $v(x)$ и $r(x)$ делит $\ell(v)^{m-n+1}u(x)$ и должен быть примитивным (поскольку примитивен $v(x)$), так что он делит $u(x)$. Если $r(x) = 0$, имеем $\gcd\big(u(x), v(x)\big) = v(x)$; с другой стороны, если $r(x) \neq 0$, из-за примитивности $v(x)$ имеем $\gcd\big(v(x), r(x)\big) = \gcd\big(v(x), \mathrm{pp}(r(x))\big)$, так что процесс может быть итерирован.

Алгоритм E (*Обобщенный алгоритм Евклида*). Даны ненулевые полиномы $u(x)$ и $v(x)$ над областью единственного разложения S. Алгоритм вычисляет наибольший общий делитель $u(x)$ и $v(x)$. Предполагается существование вспомогательных алгоритмов для вычисления наибольшего общего делителя элементов S, а также для деления a на b в S при $b \neq 0$ и a, кратном b.

E1. [Сведение к примитивным полиномам.] Установить $d \leftarrow \gcd(\mathrm{cont}(u), \mathrm{cont}(v))$, используя вспомогательный алгоритм для вычисления наибольшего общего делителя в S. (По определению $\mathrm{cont}(u)$ представляет собой наибольший общий делитель коэффициентов $u(x)$.) Заменить $u(x)$ полиномом $u(x)/\mathrm{cont}(u) = \mathrm{pp}(u(x))$; точно так же заменить $v(x)$ на $\mathrm{pp}(v(x))$.

E2. [Псевдоделение.] Вычислить $r(x)$ с использованием алгоритма R. (Нет необходимости вычислять полином-частное $q(x)$.) Если $r(x) = 0$, перейти к шагу E4. Если $\deg(r) = 0$, заменить $v(x)$ постоянным полиномом "1" и перейти к шагу E4.

E3. [Сделать остаток примитивным.] Заменить $u(x)$ на $v(x)$ и заменить $v(x)$ на $\mathrm{pp}(r(x))$. Вернуться к шагу E2. (Это — "евклидов шаг", аналогичный другим рассмотренным нами алгоритмам Евклида.)

E4. [Присоединение содержания.] Алгоритм завершается, выдав ответ $d \cdot v(x)$. ▮

В качестве примера работы алгоритма E вычислим gcd полиномов

$$\begin{aligned} u(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5, \\ v(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21 \end{aligned} \tag{9}$$

над целыми. Эти полиномы примитивны, так что на шаге E1 устанавливается $d \leftarrow 1$. На шаге E2 выполняем псевдоделение.

$$\begin{array}{rrrrrrrrr|rrrrrrrr}
 & & & & & & & & & & & & 1 & 0 & -6 \\
\multicolumn{9}{r|}{3\ 0\ 5\ 0\ -4\ -9\ 21} & 1 & 0 & & 1 & 0 & -3 & -3 & 8 & 2 & -5 \\
 & & & & & & & & & 3 & 0 & & 3 & 0 & -9 & -9 & 24 & 6 & -15 \\
 & & & & & & & & & 3 & 0 & & 5 & 0 & -4 & -9 & 21 & & \\
\hline
 & & & & & & & & & & 0 & -2 & 0 & -5 & 0 & 3 & 6 & -15 \\
 & & & & & & & & & & 0 & -6 & 0 & -15 & 0 & 9 & 18 & -45 \\
 & & & & & & & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
 & & & & & & & & & & & -6 & 0 & -15 & 0 & 9 & 18 & -45 \\
 & & & & & & & & & & & -18 & 0 & -45 & 0 & 27 & 54 & -135 \\
 & & & & & & & & & & & -18 & 0 & -30 & 0 & 24 & 54 & -126 \\
\hline
 & & & & & & & & & & & & & -15 & 0 & 3 & 0 & -9
\end{array} \tag{10}$$

Здесь частное $q(x)$ равно $1 \cdot 3^2 x^2 + 0 \cdot 3^1 x + -6 \cdot 3^0$. Имеем

$$27u(x) = v(x)(9x^2 - 6) + (-15x^4 + 3x^2 - 9). \tag{11}$$

Теперь шаг E3 замещает $u(x)$ на $v(x)$ и $v(x)$ на $\mathrm{pp}(r(x)) = 5x^4 - x^2 + 3$. Дальнейшие вычисления приведены в следующей таблице, в которой показаны только коэффициенты.

$u(x)$	$v(x)$	$r(x)$
$1, 0, 1, 0, -3, -3, 8, 2, -5$	$3, 0, 5, 0, -4, -9, 21$	$-15, 0, 3, 0, -9$
$3, 0, 5, 0, -4, -9, 21$	$5, 0, -1, 0, 3$	$-585, -1125, 2205$
$5, 0, -1, 0, 3$	$13, 25, -49$	$-233150, 307500$
$13, 25, -49$	$4663, -6150$	143193869

(12)

Поучительно сравнить данные вычисления с вычислением того же наибольшего общего делителя над *рациональными* числами, а не над целыми, с использованием

алгоритма Евклида для полиномов над полем, описанного ранее в этом разделе. Получается неожиданно сложная последовательность.

$$\begin{array}{rr} u(x) & v(x) \\ 1,0,1,0,-3,-3,8,2,-5 & 3,0,5,0,-4,-9,21 \\ 3,0,5,0,-4,-9,21 & -\frac{5}{9},0,\frac{1}{9},0,-\frac{1}{3} \\ -\frac{5}{9},0,\frac{1}{9},0,-\frac{1}{3} & -\frac{117}{25},-9,\frac{441}{25} \\ -\frac{117}{25},-9,\frac{441}{25} & \frac{233150}{19773},-\frac{102500}{6591} \\ \frac{233150}{19773},-\frac{102500}{6591} & -\frac{1288744821}{543589225} \end{array} \tag{13}$$

Для улучшения этого алгоритма можно свести $u(x)$ и $v(x)$ к нормированным полиномам на каждом шаге, чтобы удалить обратимые множители, слишком усложняющие коэффициенты. В действительности получится алгоритм E над рациональными числами:

$$\begin{array}{rr} u(x) & v(x) \\ 1,0,1,0,-3,-3,8,2,-5 & 1,0,\frac{5}{3},0,-\frac{4}{3},-3,7 \\ 1,0,\frac{5}{3},0,-\frac{4}{3},-3,7 & 1,0,-\frac{1}{5},0,\frac{3}{5} \\ 1,0,-\frac{1}{5},0,\frac{3}{5} & 1,\frac{25}{13},-\frac{49}{13} \\ 1,\frac{25}{13},-\frac{49}{13} & 1,-\frac{6150}{4663} \\ 1,-\frac{6150}{4663} & 1 \end{array} \tag{14}$$

И в (13), и в (14) последовательность полиномов, полученная при помощи алгоритма E над целыми числами, по сути, та же, что и в (12). Единственное отличие состоит в том, что полиномы умножаются на некоторые рациональные числа. Каким бы ни был полином, будь то $5x^4 - x^2 + 3$, $-\frac{5}{9}x^4 + \frac{1}{9}x^2 - \frac{1}{3}$ или $x^4 - \frac{1}{5}x^2 + \frac{3}{5}$, вычисления остаются теми же. Но любой алгоритм, использующий рациональную арифметику, имеет тенденцию к более медленной работе, чем "всецело целый" алгоритм E, поскольку рациональная арифметика обычно требует большего количества вычислений целых gcd на каждом шаге при полиномах больших степеней.

Поучительно сравнить (12), (13) и (14) с (6), где мы определяли gcd тех же полиномов $u(x)$ и $v(x)$ по модулю 13 со значительно меньшими усилиями. Поскольку $\ell(u)$ и $\ell(v)$ не кратны 13, того факта, что $\gcd\big(u(x),v(x)\big) = 1 \bmod 13$, достаточно для доказательства, что $u(x)$ и $v(x)$ взаимно просты над кольцом целых (и, следовательно, над полем рациональных чисел). Мы вернемся к этому сохраняющему время наблюдению в конце раздела 4.6.2.

Алгоритм Коллинза. Остроумный алгоритм, в целом превосходящий алгоритм E и, кроме того, предоставляющий информацию о поведении алгоритма E, был разработан Джорджем Э. Коллинзом (George E. Collins) [*JACM* **14** (1967), 128–142] и впоследствии улучшен В. С. Брауном (W. S. Brown) и Дж. Ф. Траубом (J. F. Traub) [*JACM* **18** (1971), 505–514; см. также W. S. Brown, *ACM Trans. Math. Software* **4** (1978), 237–249]. Он позволяет избежать вычислений примитивных частей на шаге E3, вместо чего производится деление на элемент S, о котором известно, что он является множителем $r(x)$.

Алгоритм C (*Поиск наибольшего общего делителя над областью единственного разложения*). В этом алгоритме используются те же предположения о входных

и выходных данных, что и в алгоритме E. Он имеет то достоинство, что при поиске наибольших общих делителей коэффициентов требуется выполнять меньше вычислений.

C1. [Сведение к примитивным полиномам.] Как и на шаге E1 алгоритма E, установить $d \leftarrow \gcd(\operatorname{cont}(u), \operatorname{cont}(v))$ и заменить $\big(u(x), v(x)\big)$ на $\big(\operatorname{pp}(u(x)), \operatorname{pp}(v(x))\big)$. Установить $g \leftarrow h \leftarrow 1$.

C2. [Псевдоделение.] Установить $\delta \leftarrow \deg(u) - \deg(v)$. Вычислить $r(x)$ с использованием алгоритма R. Если $r(x) = 0$, перейти к шагу C4. Если $\deg(r) = 0$, заменить $v(x)$ постоянным полиномом "1" и перейти к шагу C4.

C3. [Подгонка остатка.] Заменить полином $u(x)$ на $v(x)$ и $v(x)$ на $r(x)/gh^\delta$. (В этот момент все коэффициенты $r(x)$ кратны gh^δ.) Затем установить $g \leftarrow \ell(u)$, $h \leftarrow h^{1-\delta}g^\delta$ и вернуться к шагу C2. (Новое значение h будет в области S, даже если $\delta > 1$.)

C4. [Присоединение содержания.] Вернуть в качестве ответа $d \cdot \operatorname{pp}(v(x))$. ∎

Если применить этот алгоритм к рассмотренным ранее полиномам (9), то получится такая последовательность результатов в начале шага C2.

$u(x)$	$v(x)$	g	h	
$1, 0, 1, 0, -3, -3, 8, 2, -5$	$3, 0, 5, 0, -4, -9, 21$	1	1	
$3, 0, 5, 0, -4, -9, 21$	$-15, 0, 3, 0, -9$	3	9	(15)
$-15, 0, 3, 0, -9$	$65, 125, -245$	-15	25	
$65, 125, -245$	$-9326, 12300$	65	169	

В конце алгоритма $r(x)/gh^\delta = 260708$.

Эта последовательность полиномов состоит из целых кратных полиномов в последовательности, полученной с помощью алгоритма E. Вопреки тому факту, что полиномы не сведены к примитивному виду, коэффициенты остаются в разумных пределах благодаря приводящему множителю на шаге C3.

Для анализа алгоритма C и доказательства его корректности рассмотрим полученную с его помощью последовательность полиномов $u_1(x)$, $u_2(x)$, $u_3(x)$, ..., где $u_1(x) = u(x)$ и $u_2(x) = v(x)$. Пусть $\delta_j = n_j - n_{j+1}$ для $j \geq 1$, где $n_j = \deg(u_j)$, и пусть $g_1 = h_1 = 1$, $g_j = \ell(u_j)$, $h_j = h_{j-1}^{1-\delta_{j-1}} g_j^{\delta_{j-1}}$ для $j \geq 2$. Тогда

$$\begin{aligned}
g_2^{\delta_1+1} u_1(x) &= u_2(x)q_1(x) + g_1 h_1^{\delta_1} u_3(x), && n_3 < n_2; \\
g_3^{\delta_2+1} u_2(x) &= u_3(x)q_2(x) + g_2 h_2^{\delta_2} u_4(x), && n_4 < n_3; \\
g_4^{\delta_3+1} u_3(x) &= u_4(x)q_3(x) + g_3 h_3^{\delta_3} u_5(x), && n_5 < n_4
\end{aligned} \tag{16}$$

и т. д. Процесс прекращается, когда $n_{k+1} = \deg(u_{k+1}) \leq 0$. Покажем, что полиномы $u_3(x)$, $u_4(x)$, ... имеют коэффициенты из S, а именно — что множитель $g_j h_j^{\delta_j}$ делит все коэффициенты остатков. Кроме того, покажем, что все значения h_j принадлежат S. Доказательство весьма запутанно, и его легче понять, рассмотрев конкретный пример.

Предположим, как и в (15), что $n_1 = 8$, $n_2 = 6$, $n_3 = 4$, $n_4 = 2$, $n_5 = 1$, $n_6 = 0$, так что $\delta_1 = \delta_2 = \delta_3 = 2$, $\delta_4 = \delta_5 = 1$. Запишем $u_1(x) = a_8x^8 + a_7x^7 + \cdots + a_0$,

$u_2(x) = b_6x^6 + b_5x^5 + \cdots + b_0, \ldots, u_5(x) = e_1x + e_0, u_6(x) = f_0$, так что $h_1 = 1$, $h_2 = b_6^2$, $h_3 = c_4^2/b_6^2$, $h_4 = d_2^2b_6^2/c_4^2$. Рассмотрим табл. 1 с учетом принятых обозначений. Для определенности будем считать, что коэффициенты полиномов целые. Имеем $b_6^3u_1(x) = u_2(x)q_1(x) + u_3(x)$; так что, если умножить строку A_5 на b_6^3 и вычесть подходящие кратные строк B_7, B_6 и B_5 (соответствующие коэффициентам $q_1(x)$), можно получить строку C_5. Если также умножить строку A_4 на b_6^3 и вычесть кратные строк B_6, B_5 и B_4, можно получить строку C_4. Аналогично получим $c_4^3u_2(x) = u_3(x)q_2(x) + b_6^5u_4(x)$. Умножив же строку B_3 на c_4^3, вычтя целые кратные строк C_5, C_4 и C_3 и разделив на b_6^5, получим строку D_3.

Для доказательства того, что $u_4(x)$ имеет целые коэффициенты, рассмотрим матрицу

$$\begin{array}{c} A_2 \\ A_1 \\ A_0 \\ B_4 \\ B_3 \\ B_2 \\ B_1 \\ B_0 \end{array} \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{pmatrix} = M. \tag{17}$$

Указанные операции со строками и перестановки строк приводят к преобразованию матрицы M в

$$\begin{array}{c} B_4 \\ B_3 \\ B_2 \\ B_1 \\ C_2 \\ C_1 \\ C_0 \\ D_0 \end{array} \begin{pmatrix} b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & d_2 & d_1 & d_0 \end{pmatrix} = M'. \tag{18}$$

В соответствии с тем, каким образом была получена матрица M' из M, имеем

$$b_6^3 \cdot b_6^3 \cdot b_6^3 \cdot (c_4^3/b_6^5) \cdot \det M_0 = \pm \det M_0',$$

если M_0 и M_0' — любые квадратные матрицы, полученные в результате выбора восьми соответствующих столбцов из M и M'. Например, выберем семь первых столбцов и столбец, содержащий d_1; тогда

$$b_6^3 \cdot b_6^3 \cdot b_6^3 \cdot (c_4^3/b_6^5) \cdot \det \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_0 \\ 0 & 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_1 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_0 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_1 \end{pmatrix} = \pm b_6^4 \cdot c_4^3 \cdot d_1.$$

Поскольку $b_6c_4 \neq 0$, это доказывает, что d_1 — целое. Аналогично целыми являются d_2 и d_0.

Таблица 1

КОЭФФИЦИЕНТЫ, ПОЯВЛЯЮЩИЕСЯ В АЛГОРИТМЕ С

Имя строки	Строка														Умножить на	Заменить строкой
A_5	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	0	0	0	0	0	b_6^3	C_5
A_4	0	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	0	0	0	0	b_6^3	C_4
A_3	0	0	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	0	0	0	b_6^3	C_3
A_2	0	0	0	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	0	0	b_6^3	C_2
A_1	0	0	0	0	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	0	b_6^3	C_1
A_0	0	0	0	0	0	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	b_6^3	C_0
B_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	0	0	0	0	0		
B_6	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	0	0	0	0		
B_5	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	0	0	0		
B_4	0	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	0	0		
B_3	0	0	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	0	c_4^3/b_6^5	D_3
B_2	0	0	0	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	0	c_4^3/b_6^5	D_2
B_1	0	0	0	0	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	0	c_4^3/b_6^5	D_1
B_0	0	0	0	0	0	0	0	b_6	b_5	b_4	b_3	b_2	b_1	b_0	c_4^3/b_6^5	D_0
C_5	0	0	0	0	c_4	c_3	c_2	c_1	c_0	0	0	0	0	0		
C_4	0	0	0	0	0	c_4	c_3	c_2	c_1	c_0	0	0	0	0		
C_3	0	0	0	0	0	0	c_4	c_3	c_2	c_1	c_0	0	0	0		
C_2	0	0	0	0	0	0	0	c_4	c_3	c_2	c_1	c_0	0	0		
C_1	0	0	0	0	0	0	0	0	c_4	c_3	c_2	c_1	c_0	0	$d_2^2 b_6^4/c_4^5$	E_1
C_0	0	0	0	0	0	0	0	0	0	c_4	c_3	c_2	c_1	c_0	$d_2^2 b_6^4/c_4^5$	E_0
D_3	0	0	0	0	0	0	0	0	d_2	d_1	d_0	0	0	0		
D_2	0	0	0	0	0	0	0	0	0	d_2	d_1	d_0	0	0		
D_1	0	0	0	0	0	0	0	0	0	0	d_2	d_1	d_0	0		
D_0	0	0	0	0	0	0	0	0	0	0	0	d_2	d_1	d_0	$e_2^2 c_4^2/d_2^3 b_6^2$	F_0
E_1	0	0	0	0	0	0	0	0	0	0	0	e_1	e_0	0		
E_0	0	0	0	0	0	0	0	0	0	0	0	0	e_1	e_0		
F_0	0	0	0	0	0	0	0	0	0	0	0	0	0	f_0		

В общем, так же можно показать, что $u_{j+1}(x)$ имеет целые коэффициенты. Если начать с матрицы M, состоящей из строк с $A_{n_2-n_j}$ по A_0 и с $B_{n_1-n_j}$ по B_0, и выполнить указанные в табл. 1 операции над строками, можно получить матрицу M', состоящую из расположенных в некотором порядке строк от $B_{n_1-n_j}$ по $B_{n_3-n_j+1}$, затем — от $C_{n_2-n_j}$ по $C_{n_4-n_j+1}$, ..., от $P_{n_{j-2}-n_j}$ по P_1, от $Q_{n_{j-1}-n_j}$ по Q_0 и, наконец, R_0 (строки, содержащей коэффициенты $u_{j+1}(x)$). Извлекая подходящие столбцы, покажем, что

$$(g_2^{\delta_1+1}/g_1 h_1^{\delta_1})^{n_2-n_j+1}(g_3^{\delta_2+1}/g_2 h_2^{\delta_2})^{n_3-n_j+1}\ldots(g_j^{\delta_{j-1}+1}/g_{j-1}h_{j-1}^{\delta_{j-1}})^{n_j-n_j+1} \times \det M_0 = \pm g_2^{n_1-n_3} g_3^{n_2-n_4}\ldots g_{j-1}^{n_{j-2}-n_j} g_j^{n_{j-1}-n_j+1} r_t, \quad (19)$$

где r_t — данный коэффициент $u_{j+1}(x)$, а M_0 — подматрица M. h выбираются очень хитрым способом (см. упр. 24) — так, чтобы это уравнение упростилось до

$$\det M_0 = \pm r_t. \quad (20)$$

Следовательно, *каждый коэффициент* $u_{j+1}(x)$ *может быть выражен как определитель матрицы размера* $(n_1 + n_2 - 2n_j + 2) \times (n_1 + n_2 - 2n_j + 2)$, *элементы которой представляют собой коэффициенты* $u(x)$ *и* $v(x)$.

Остается показать, что выбранные подобным образом h также являются целыми числами. Применима следующая методика: рассмотрим, например, матрицу

$$\begin{matrix} A_1 \\ A_0 \\ B_3 \\ B_2 \\ B_1 \\ B_0 \end{matrix} \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{pmatrix} = M. \tag{21}$$

Операции над строками, определенные в табл. 1, и перестановка строк приведут к матрице

$$\begin{matrix} B_3 \\ B_2 \\ B_1 \\ B_0 \\ C_1 \\ C_0 \end{matrix} \begin{pmatrix} b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\ 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 \end{pmatrix} = M'; \tag{22}$$

следовательно, если рассмотреть любые подматрицы M_0 и M_0', полученные путем выбора шести соответствующих столбцов M и M', можно получить $b_6^3 \cdot b_6^3 \cdot \det M_0 = \pm \det M_0'$. Когда M_0 выбирается таким образом, что является первыми шестью столбцами M, получаем, что $\det M_0 = \pm c_4^2 / b_6^2 = \pm h_3$, так что h_3 является целым числом.

В целом, чтобы показать, что h_j целое при $j \geq 3$, начнем с матрицы M, состоящей из строк с $A_{n_2-n_j-1}$ по A_0 и с $B_{n_1-n_j-1}$ по B_0; затем будем выполнять соответствующие операции над строками до тех пор, пока не получим матрицу M', состоящую из строк с $B_{n_1-n_j-1}$ по $B_{n_3-n_j}$, затем — с $C_{n_2+n_j-1}$ по $C_{n_4-n_j}$, ..., с $P_{n_{j-2}-n_j-1}$ по P_0 и с $Q_{n_{j-1}-n_j-1}$ по Q_0. Рассмотрев M_0, представляющую собой первые $n_1 + n_2 - 2n_j$ столбцов матрицы M, получаем

$$(g_2^{\delta_1+1}/g_1 h_1^{\delta_1})^{n_2-n_j} (g_3^{\delta_2+1}/g_2 h_2^{\delta_2})^{n_3-n_j} \dots (g_j^{\delta_{j-1}+1}/g_{j-1} h_{j-1}^{\delta_{j-1}})^{n_j-n_j} \det M_0 = \pm g_2^{n_1-n_3} g_3^{n_2-n_4} \dots g_{j-1}^{n_{j-2}-n_j} g_j^{n_{j-1}-n_j}, \tag{23}$$

и уравнение изящно упрощается до

$$\det M_0 = \pm h_j. \tag{24}$$

(Хотя это доказательство и приводится для областей целых чисел, оно очевидным образом применимо к любой области единственного разложения.)

В процессе проверки алгоритма C мы также получили, что каждый элемент S, с которым мы имели дело в алгоритме, может быть выражен как детерминант с элементами, являющимися коэффициентами примитивных частей исходных полиномов. Хорошо известная теорема Адамара (см. упр. 15) гласит, что

$$|\det(a_{ij})| \leq \prod_{1 \leq i \leq n} \left(\sum_{1 \leq j \leq n} a_{ij}^2 \right)^{1/2}, \tag{25}$$

а потому каждый коэффициент, появляющийся в полиномах, которые вычислены согласно алгоритму C, не превышает

$$N^{m+n}(m+1)^{n/2}(n+1)^{m/2}, \tag{26}$$

если все коэффициенты данных полиномов $u(x)$ и $v(x)$ ограничены по абсолютному значению величиной N. Та же верхняя грань применима к коэффициентам всех полиномов $u(x)$ и $v(x)$, вычисленных при выполнении алгоритма E, поскольку полиномы, получаемые в алгоритме E, всегда являются делителями полиномов, получаемых в алгоритме C.

Эта оценка верхней грани коэффициентов очень хороша, поскольку она гораздо лучше той, которую можно было бы ожидать. Например, рассмотрим, что случится, если не корректировать шаги E3 и C3, а просто заменить $v(x)$ на $r(x)$. Это простейший алгоритм поиска gcd, традиционно приводимый в учебниках алгебры (в теоретических целях, не для практических вычислений). Если предположить, что $\delta_1 = \delta_2 = \cdots = 1$, можно найти, что коэффициенты $u_3(x)$ ограничены величиной N^3, коэффициенты $u_4(x)$ — величиной N^7, $u_5(x)$ — N^{17}, ... и коэффициенты $u_k(x)$ — величиной N^{a_k}, где $a_k = 2a_{k-1} + a_{k-2}$. Таким образом, верхняя грань при $m = n+1$ вместо (26) приблизительно составляет

$$N^{0.5(2.414)^n}. \tag{27}$$

Эксперименты показывают, что простой алгоритм действительно ведет себя именно так; количество цифр в коэффициентах растет с каждым шагом экспоненциально! В алгоритме E, напротив, рост количества цифр лишь немного превосходит линейный.

Еще одним побочным результатом доказательства корректности алгоритма C является тот факт, что степени полиномов будут почти всегда увеличиваться на 1 на каждом шаге, так что число итераций шага C2 (или E2) обычно будет составлять $\deg(v)$, если данные полиномы "случайны". Для того чтобы увидеть, почему это происходит, заметим, например, что можно выбрать первые восемь столбцов M и M' в (17) и (18). В таком случае можно найти, что $u_4(x)$ имеет степень меньше 3 тогда и только тогда, когда $d_3 = 0$, т. е. тогда и только тогда, когда

$$\det \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 \\ 0 & 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 \end{pmatrix} = 0.$$

Вообще, δ_j будет больше единицы при $j > 1$ тогда и только тогда, когда подобный детерминант, составленный из коэффициентов $u(x)$ и $v(x)$, равен нулю. Поскольку такой детерминант представляет собой ненулевой полином от многих переменных-коэффициентов, он будет ненулевым "почти всегда" или "с вероятностью 1" (см. в упр. 16 более точную формулировку этого утверждения и связанное с ним доказательство в упр. 4). В примерах полиномов в (15) δ_2 и δ_3 равны 2, так что эти полиномы, скорее всего, — исключение, а не правило.

Все вышесказанное может использоваться для доказательства хорошо известного факта, что два полинома взаимно просты тогда и только тогда, когда их *результант* ненулевой; результант представляет собой определитель, имеющий вид строк с A_5 по A_0 и с B_7 по B_0 в табл. 1* (Это так называемый "детерминант Сильвестра" (см. упр. 12). Свойства результанта рассматриваются в книге B. L. van der Waerden, *Modern Algebra* (имеется перевод на английский язык Фреда Блюма (Fred Blum) (New York: Ungar, 1949)), разделы 27–28.) На основании приведенного выше материала можно сказать, что gcd "почти всегда" имеет нулевую степень, поскольку детерминант Сильвестра почти никогда не равен нулю. Однако во многих вычислениях, представляющих практический интерес, нельзя быть уверенным в том, что gcd не будет являться полиномом положительной степени.

Что происходит при работе алгоритмов E и C при $\gcd \neq 1$, можно в точности увидеть, рассмотрев $u(x) = w(x)u_1(x)$ и $v(x) = w(x)u_2(x)$, где $u_1(x)$ и $u_2(x)$ — взаимно простые полиномы, а $w(x)$ — примитивный полином. Тогда, если $u_1(x)$, $u_2(x)$, $u_3(x)$, ... — полиномы, получаемые при работе алгоритма E с $u(x) = u_1(x)$ и $v(x) = u_2(x)$, легко увидеть, что последовательность, получаемая для $u(x) = w(x)u_1(x)$ и $v(x) = w(x)u_2(x)$, представляет собой просто $w(x)u_1(x)$, $w(x)u_2(x)$, $w(x)u_3(x)$, $w(x)u_4(x)$ и т. д. Поведение алгоритма C несколько отличается: если полиномы $u_1(x)$, $u_2(x)$, $u_3(x)$, ... получены при работе алгоритма C с $u(x) = u_1(x)$ и $v(x) = u_2(x)$ и если $\deg(u_{j+1}) = \deg(u_j) - 1$ (что почти всегда истинно при $j > 1$), то в результате применения алгоритма C к $u(x) = w(x)u_1(x)$ и $v(x) = w(x)u_2(x)$ получается последовательность

$$w(x)u_1(x),\ w(x)u_2(x),\ \ell^2 w(x)u_3(x),\ \ell^4 w(x)u_4(x),\ \ell^6 w(x)u_5(x),\ \ldots, \qquad (28)$$

где $\ell = \ell(w)$ (см. упр. 13). Несмотря на наличие этих дополнительных ℓ-множителей алгоритм C будет превосходить алгоритм E, так как работать с несколько бóльшими полиномами проще, чем постоянно вычислять примитивные части.

Последовательности остатков, такие как получаемые в алгоритмах C и E, полезны не только для поиска наибольших общих делителей и результантов. Важным применением является перечисление действительных корней данного полинома в определенном интервале согласно знаменитой теореме Я. Штурма (J. Sturm) [*Mém. Présentés par Divers Savants* **6** (Paris, 1835), 271–318]. Пусть $u(x)$ — полином над полем действительных чисел, имеющий различные комплексные корни. Из следующего раздела вы узнаете, что корни различны тогда и только тогда, когда $\gcd\big(u(x), u'(x)\big) = 1$, где $u'(x)$ — производная $u(x)$; значит, имеется последователь-

---

* Для большей ясности приведем полный вид результанта двух полиномов $a_0x^m + a_1x^{m-1} + \ldots + a_m$ и $b_0x^n + b_1x^{n-1} + \ldots + b_n$:

$$\det \begin{pmatrix}
a_0 & a_1 & a_2 & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot & \cdot & a_{m-1} & a_m & 0 & \cdot & \cdot & \cdot & \cdot & 0 \\
0 & a_0 & a_1 & a_2 & \ldots & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{m-1} & a_m & 0 & \cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & \cdot & \cdot & \cdot & \ldots & \cdot & 0 & a_0 & a_1 & a_2 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & a_{m-1} & a_m \\
b_0 & b_1 & b_2 & \cdot & \ldots & b_{n-1} & b_n & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\
0 & b_0 & b_1 & b_2 & \ldots & \cdot & b_{n-1} & b_n & 0 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \\
\cdot & \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & \cdot & \cdot & \cdot & \ldots & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 & b_0 & b_1 & b_2 & \cdot & b_{n-1} & b_n
\end{pmatrix}$$

(см., например, Корн Г., Корн Т. *Справочник по математике (для научных работников и инженеров).* — М.: Наука, 1978, раздел 1.7.4). — *Прим. перев.*

ность остатков, доказывающая, что полином $u(x)$ взаимно прост с $u'(x)$. Считая $u_0(x) = u(x)$, $u_1(x) = u'(x)$ (и следуя Штурму), изменим знак всех остатков. Получим

$$\begin{aligned} c_1 u_0(x) &= u_1(x)q_1(x) - d_1 u_2(x), \\ c_2 u_1(x) &= u_2(x)q_2(x) - d_2 u_3(x), \\ &\vdots \\ c_k u_{k-1}(x) &= u_k(x)q_k(x) - d_k u_{k+1}(x) \end{aligned} \tag{29}$$

для некоторых положительных констант c_j и d_j, где $\deg(u_{k+1}) = 0$. Будем говорить, что *отклонение* $V(u,a)$ полинома $u(x)$ в a равно количеству изменений знака в последовательности $u_0(a)$, $u_1(a)$, ..., $u_{k+1}(a)$, не считая нулей. Например, если последовательность знаков представляет собой 0, +, −, −, 0, +, +, −, получим $V(u,a) = 3$. Теорема Штурма утверждает, что *количество корней* $u(x)$ *в интервале* $a < x \le b$ *равно* $V(u,a) - V(u,b)$. Доказательство этого факта на удивление коротко (см. упр. 22).

Хотя алгоритмы C и E интересны, история на них не заканчивается. Важные альтернативные пути вычисления gcd полиномов над целыми числами будут рассмотрены в конце раздела 4.6.2. Имеется также общий алгоритм для вычисления детерминанта, о котором можно сказать, что он включает в себя алгоритм C в качестве частного случая [см. E. H. Bareiss, *Math. Comp.* **22** (1968), 565–578].

В четвертом издании этой книги я намерен переделать материал настоящего раздела, уделив достойное внимание как исследованиям детерминантов в 19 веке, так и работе W. Habicht, Comm. Math. Helvetici **21** *(1948), 99–116. Превосходное обсуждение последней дано в работе R. Loos in Computing, Supplement 4 (1982), 115–137. К этим же методам относится и интересный метод вычисления детерминантов, который выведен Ч. Л. Доджсоном (C. L. Dodgson) (известным также под именем Льюис Кэррол (Lewis Carroll)) из теоремы Якоби. Обзор ранней истории исследований тождественности детерминантов подматриц можно найти в работе D. E. Knuth, Electronic J. Combinatorics* **3**, *2 (1996), статья R5, §3.*

УПРАЖНЕНИЯ

1. [*10*] Вычислите псевдочастное $q(x)$ и псевдоостаток $r(x)$, т. е. полиномы, удовлетворяющие (8), для $u(x) = x^6 + x^5 - x^4 + 2x^3 + 3x^2 - x + 2$ и $v(x) = 2x^3 + 2x^2 - x + 3$ над целыми числами.

2. [*15*] Чему равен наибольший общий делитель полинома $3x^6 + x^5 + 4x^4 + 4x^3 + 3x^2 + 4x + 2$ и "обратного" ему полинома $2x^6 + 4x^5 + 3x^4 + 4x^3 + 4x^2 + x + 3$ по модулю 7? (В данном случае под "обратным" подразумевается полином с обращенным порядком коэффициентов. — *Прим. перев.*)

▸ **3.** [*M25*] Покажите, что алгоритм Евклида для полиномов над полем S может быть расширен для поиска полиномов $U(x)$ и $V(x)$ над S, таких, что

$$u(x)V(x) + U(x)v(x) = \gcd(u(x), v(x))$$

(см. алгоритм 4.5.2X). Чему равны степени полиномов $U(x)$ и $V(x)$, вычисленных при помощи такого расширенного алгоритма? Докажите, что если S — поле рациональных чисел и если $u(x) = x^m - 1$ и $v(x) = x^n - 1$, то расширенный алгоритм дает полиномы $U(x)$

и $V(x)$, имеющие *целые* коэффициенты. Найдите $U(x)$ и $V(x)$ для $u(x) = x^{21} - 1$ и $v(x) = x^{13} - 1$.

▶ **4.** [*M30*] Пусть p — простое число, и предположим, что алгоритм Евклида, примененный к полиномам $u(x)$ и $v(x)$ по модулю p, приводит к последовательности полиномов, имеющих степени соответственно m, n, n_1, ..., n_t, $-\infty$, где $m = \deg(u)$, $n = \deg(v)$ и $n_t \geq 0$. Положим, что $m \geq n$. Если $u(x)$ и $v(x)$ являются нормированными полиномами, независимо и равномерно распределенными среди всех p^{m+n} пар нормированных полиномов степеней соответственно m и n, то чему равны средние значения трех величин t, $n_1 + \cdots + n_t$ и $(n - n_1)n_1 + \cdots + (n_{t-1} - n_t)n_t$, выраженные как функции от m, n и p? (Эти три величины являются основными факторами, влияющими на время работы алгоритма Евклида с полиномами по модулю p, если считать, что деление выполняется при помощи алгоритма D.) [*Указание.* Покажите, что $u(x) \bmod v(x)$ равномерно распределен и независим от $v(x)$.]

5. [*M22*] Чему равна вероятность того, что $u(x)$ и $v(x)$ взаимно просты по модулю p, если $u(x)$ и $v(x)$ — независимые равномерно распределенные нормированные полиномы степени n?

6. [*M23*] Мы видели, что алгоритм Евклида 4.5.2A для целых может быть преобразован в алгоритм для поиска наибольшего общего делителя полиномов. Можно ли аналогичным путем преобразовать бинарный алгоритм поиска gcd 4.5.2B в алгоритм, работающий с полиномами?

7. [*M10*] Чему равны обратимые элементы в области всех полиномов над областью единственного разложения S?

▶ **8.** [*M22*] Покажите, что, если полином с целыми коэффициентами неприводим над областью целых чисел, он неприводим и при рассмотрении его над полем рациональных чисел.

9. [*M25*] Пусть $u(x)$ и $v(x)$ — примитивные полиномы над областью единственного разложения S. Докажите, что $u(x)$ и $v(x)$ взаимно просты тогда и только тогда, когда полиномы $U(x)$ и $V(x)$ над S таковы, что $u(x)V(x) + U(x)v(x)$ — полином нулевой степени. [*Указание.* Расширьте алгоритм E так, как алгоритм 4.5.2A был расширен в упр. 3.]

10. [*M28*] Докажите, что полиномы над областью единственного разложения образуют область единственного разложения. [*Указание.* Используйте результат упр. 9 для того, чтобы показать, что имеется не более одного возможного разложения.]

11. [*M22*] Какие строки появились бы в табл. 1, если бы последовательность степеней была 9, 6, 5, 2, $-\infty$, а не 8, 6, 4, 2, 1, 0?

▶ **12.** [*M24*] Пусть $u_1(x)$, $u_2(x)$, $u_3(x)$, ... — последовательность полиномов, полученная во время работы алгоритма C. Матрица Сильвестра представляет собой квадратную матрицу, построенную из строк с A_{n_2-1} по A_0 и с B_{n_1-1} по B_0 (в обозначениях, аналогичных обозначениям для табл. 1). Покажите, что если $u_1(x)$ и $u_2(x)$ имеют общий множитель положительной степени, то детерминант матрицы Сильвестра равен нулю, и обратно: покажите, что если для некоторого k $\deg(u_k) = 0$, то детерминант матрицы Сильвестра ненулевой (покажите это, выведя формулу для его абсолютного значения через $\ell(u_j)$ и $\deg(u_j)$, $1 \leq j \leq k$).

13. [*M22*] Покажите, что при $\delta_1 = \delta_2 = \cdots = \delta_{k-1} = 1$ старший коэффициент ℓ примитивной части $\gcd(u(x), v(x))$ входит в показанную в (28) последовательность полиномов, которая получается с помощью алгоритма C. Каково поведение для общего случая δ_j?

14. [*M29*] Пусть $r(x)$ — псевдоостаток от псевдоделения $u(x)$ на $v(x)$. Покажите, что если $\deg(u) \geq \deg(v) + 2$ и $\deg(v) \geq \deg(r) + 2$, то $r(x)$ кратен $\ell(v)$.

15. [*M26*] Докажите неравенство Адамара (25). [*Указание.* Рассмотрите матрицу AA^T.]

▸ **16.** [*M22*] Пусть $f(x_1,\ldots,x_n)$ — полином от многих переменных, не равный тождественно нулю, и пусть $r(S_1,\ldots,S_n)$ — множество корней $(x_1,\ldots,x_n)$ уравнения $f(x_1,\ldots,x_n)=0$, таких, что $x_1\in S_1,\ldots,x_n\in S_n$. Если степень f не превышает $d_j\le|S_j|$ для переменной x_j, докажите, что

$$|r(S_1,\ldots,S_n)| \le |S_1|\ldots|S_n| - (|S_1|-d_1)\ldots(|S_n|-d_n).$$

Следовательно, вероятность нахождения корня случайным образом,

$$|r(S_1,\ldots,S_n)|/|S_1|\ldots|S_n|,$$

приближается к нулю при увеличении множеств S_j. [Это неравенство имеет множество приложений при разработке алгоритмов рандомизации, так как предоставляет хороший способ проверки, является ли сложная сумма произведений сумм равной нулю без раскрытия всех сомножителей.]

17. [*M32*] (*Алгоритм П. М. Кона (P. M. Cohn) деления строковых полиномов.*) Пусть A является *алфавитом*, т. е. множеством символов. *Строка* α над алфавитом A представляет собой последовательность из $n\ge 0$ символов $\alpha=a_1\ldots a_n$, где каждое $a_j\in A$. *Длина* α, записываемая как $|\alpha|$, равна количеству символов n. *Строковым полиномом* над алфавитом A является конечная сумма $U=\sum_k r_k\alpha_k$, где каждое r_k является ненулевым рациональным числом, а каждое α_k — строкой над алфавитом A; мы полагаем, что $\alpha_j\ne\alpha_k$ при $j\ne k$. *Степень* U, $\deg(U)$, определяется как $-\infty$, если $U=0$ (т. е. если сумма пуста); в противном случае $\deg(u)=\max|\alpha_k|$. Сумма и произведение строковых полиномов определяются очевидным образом. Так, $(\sum_j r_j\alpha_j)(\sum_k s_k\beta_k)=\sum_{j,k} r_j s_k\alpha_j\beta_k$, где произведение двух строк получается путем их простого соединения; после этого они становятся членами суммы. Например, если $A=\{a,b\}$, $U=ab+ba-2a-2b$ и $V=a+b-1$, то $\deg(U)=2$, $\deg(V)=1$, $V^2=aa+ab+ba+bb-2a-2b+1$ и $V^2-U=aa+bb+1$. Ясно, что $\deg(UV)=\deg(U)+\deg(V)$ и $\deg(U+V)\le\max(\deg(U),\deg(V))$ с равенством в последней формуле при $\deg(U)\ne\deg(V)$. (Строковые полиномы могут рассматриваться как обычные полиномы от многих переменных над полем рациональных чисел, но переменные *не коммутативны* относительно умножения. На математическом языке множество строковых полиномов с определенными здесь операциями представляет собой "свободную ассоциативную алгебру", порожденную A над полем рациональных чисел.)

a) Пусть Q_1, Q_2, U и V — такие строковые полиномы с $\deg(U)\ge\deg(V)$, что $\deg(Q_1U-Q_2V)<\deg(Q_1U)$. Разработайте алгоритм поиска строкового полинома Q, такого, что $\deg(U-QV)<\deg(U)$. (Следовательно, если даны U и V, такие, что $Q_1U=Q_2V+R$ и $\deg(R)<\deg(Q_1U)$ для некоторых Q_1 и Q_2, то существует решение для этих условий при $Q_1=1$.)

b) Даны строковые полиномы U и V с $\deg(V)>\deg(Q_1U-Q_2V)$ для некоторых Q_1 и Q_2. Покажите, что результат п. (a) может быть улучшен для поиска частного Q, такого, что $U=QV+R$, $\deg(R)<\deg(V)$. (Это аналог (1) для строковых полиномов; в п. (a) показано, что можно достичь $\deg(R)<\deg(U)$ при более слабой гипотезе.)

c) *Однородный полином* — это полином, все члены которого имеют одну и ту же степень (длину). Пусть U_1, U_2, V_1, V_2 являются однородными строковыми полиномами с $U_1V_1=U_2V_2$ и $\deg(V_1)\ge\deg(V_2)$. Покажите, что существует однородный строковый полином U, такой, что $U_2=U_1U$ и $V_1=UV_2$.

d) Даны однородные строковые полиномы U и V с $UV=VU$. Докажите, что имеется однородный строковый полином W, такой, что $U=rW^m$, $V=sW^n$ для некоторых целых m, n и рациональных чисел r, s. Разработайте алгоритм для вычисления W, имеющего наибольшую возможную степень. (Этот алгоритм интересен, например, когда $U=\alpha$ и $V=\beta$ — строки, удовлетворяющие соотношению $\alpha\beta=\beta\alpha$; тогда W — просто некоторая строка γ. Когда $U=x^m$ и $V=x^n$, решение с наибольшей степенью

представляет собой строку $W = x^{\gcd(m,n)}$, так что этот алгоритм включает в себя как частный случай алгоритм поиска gcd целых чисел.)

▸ **18.** [*M24*] (*Алгоритм Евклида для строковых полиномов.*) Пусть V_1 и V_2 — строковые полиномы, не равные одновременно нулю и имеющие *общее левое кратное* (это означает, что существуют не равные одновременно нулю строковые полиномы U_1 и U_2, такие, что $U_1V_1 = U_2V_2$). Назначение данного упражнения — найти алгоритм для вычисления их *наибольшего общего правого делителя* НОПД(V_1, V_2) и *наименьшего общего левого кратного* НОЛК(V_1, V_2). Эти величины определяются следующим образом: НОПД(V_1, V_2) является общим правым делителем V_1 и V_2 (т. е. $V_1 = W_1$НОПД(V_1, V_2) и $V_2 = W_2$НОПД(V_1, V_2) для некоторых W_1 и W_2), и любой общий правый делитель V_1 и V_2 является правым делителем НОПД(V_1, V_2); НОЛК$(V_1, V_2) = Z_1V_1 = Z_2V_2$ для некоторых Z_1 и Z_2, и любое общее левое кратное V_1 и V_2 является левым кратным для НОЛК(V_1, V_2).

Например, пусть $U_1 = abbbab + abbab - bbab + ab - 1$, $V_1 = babab + abab + ab - b$; $U_2 = abb + ab - b$, $V_2 = babbabab + bababab + babab + abab - babb - 1$. Тогда имеем $U_1V_1 = U_2V_2 = abbbabbabab + abbabbabab + abbbababab + abbababab - bbabbabab + abbbabab - bbababab + 2abbabab - abbbabb + ababab - abbabb - bbabab - babab + bbabb - abb - ab + b$. Для этих строковых полиномов можно показать, что НОПД$(V_1, V_2) = ab + 1$ и НОЛК$(V_1, V_2) = U_1V_1$.

Алгоритм деления из упр. 17 можно сформулировать так: если V_1 и V_2 являются строковыми полиномами, причем $V_2 \neq 0$, и если $U_1 \neq 0$ и U_2 удовлетворяет соотношению $U_1V_1 = U_2V_2$, то существуют строковые полиномы Q и R, такие, что

$$V_1 = QV_2 + R, \qquad \text{где } \deg(R) < \deg(V_2).$$

Отсюда легко вывести, что Q и R определяются однозначно; они не зависят от данных U_1 и U_2. Кроме того, результат обладает право-левой симметрией в том смысле, что

$$U_2 = U_1Q + R', \qquad \text{где } \deg(R') = \deg(U_1) - \deg(V_2) + \deg(R) < \deg(U_1).$$

Покажите, что этот алгоритм для деления может быть расширен до алгоритма, вычисляющего НОЛК(V_1, V_2) и НОПД(V_1, V_2); фактически расширенный алгоритм находит строковые полиномы Z_1 и Z_2, такие, что $Z_1V_1 + Z_2V_2 =$ НОПД(V_1, V_2). [*Указание.* Используйте вспомогательные переменные u_1, u_2, v_1, v_2, w_1, w_2, w_1', w_2', z_1, z_2, z_1' и z_2', значения которых представляют собой строковые переменные; начните с установки $u_1 \leftarrow U_1$, $u_2 \leftarrow U_2$, $v_1 \leftarrow V_1$, $v_2 \leftarrow V_2$ и на протяжении алгоритма поддерживайте выполнение условий

$$\begin{aligned} U_1w_1 + U_2w_2 &= u_1, & z_1V_1 + z_2V_2 &= v_1,\\ U_1w_1' + U_2w_2' &= u_2, & z_1'V_1 + z_2'V_2 &= v_2,\\ u_1z_1 - u_2z_1' &= (-1)^nU_1, & w_1v_1 - w_1'v_2 &= (-1)^nV_1,\\ -u_1z_2 + u_2z_2' &= (-1)^nU_2, & -w_2v_1 + w_2'v_2 &= (-1)^nV_2 \end{aligned}$$

на n-й итерации. Это можно рассматривать, как "окончательное" расширение алгоритма Евклида.]

19. [*M39*] (*Общие делители квадратных матриц.*) В упр. 18 показано, что концепция наибольших общих правых делителей может иметь смысл при некоммутативном умножении. Докажите, что любые две целочисленные матрицы A и B размера $n \times n$ имеют наибольший общий правый матричный делитель D. [*Предложение.* Разработайте алгоритм, на вход которого поступают матрицы A и B, а на выходе получаются целочисленные матрицы D, P, Q, X, Y, где $A = PD$, $B = QD$ и $D = XA + YB$.] Найдите наибольший общий правый делитель матриц $\left(\begin{smallmatrix}1 & 2\\ 3 & 4\end{smallmatrix}\right)$ и $\left(\begin{smallmatrix}4 & 3\\ 2 & 1\end{smallmatrix}\right)$.

20. [*M40*] Исследуйте точность алгоритма Евклида: что можно сказать о вычислении наибольшего общего делителя полиномов с коэффициентами, представляющими числа с плавающей точкой?

21. [*M25*] Докажите, что время вычисления, требуемое алгоритмом C для поиска gcd двух полиномов n-й степени над кольцом целых чисел, составляет $O(n^4(\log Nn)^2)$, если коэффициенты полинома ограничены по абсолютному значению величиной N.

22. [*M23*] Докажите теорему Штурма. [*Указание.* Некоторые последовательности знаков невозможны.]

23. [*M22*] Докажите, что если $u(x)$ в (29) имеет $\deg(u)$ действительных корней, то

$$\deg(u_{j+1}) = \deg(u_j) - 1 \text{ для } 0 \le j \le k.$$

24. [*M21*] Покажите, что (19) упрощается до (20) и (23) упрощается до (24).

25. [*M24*] (В. С. Браун (W. S. Brown).) Докажите, что все полиномы $u_j(x)$ в (16) для $j \ge 3$ кратны $\gcd(\ell(u), \ell(v))$, и поясните, как в соответствии с этим можно улучшить алгоритм C.

▸ **26.** [*M26*] Назначение этого упражнения — найти аналог для полиномов того факта, что цепная дробь с целыми положительными элементами дает наилучшее приближение действительных чисел (упр. 4.5.3–42).

Пусть $u(x)$ и $v(x)$ — полиномы над полем с $\deg(u) > \deg(v)$ и пусть $a_1(x)$, $a_2(x)$, ... — частные полиномы, образующиеся в результате применения алгоритма Евклида к $u(x)$ и $v(x)$. Например, последовательность частных в (5) и (6) представляет собой $9x^2+7$, $5x^2+5$, $6x^3+5x^2+6x+5$, $9x+12$. Необходимо показать, что представления $p_n(x)/q_n(x)$ цепной дроби $//a_1(x), a_2(x), \ldots //$ являются "наилучшими приближениями" малых степеней рациональной функции $v(x)/u(x)$, где $p_n(x) = K_{n-1}(a_2(x), \ldots, a_n(x))$ и $q_n(x) = K_n(a_1(x), \ldots, a_n(x))$ — континуанты из 4.5.3–(4). По определению $p_0(x) = q_{-1}(x) = 0$, $p_{-1}(x) = q_0(x) = 1$.

Докажите, что если $p(x)$ и $q(x)$ являются полиномами, такими, что $\deg(q) < \deg(q_n)$ и $\deg(pu - qv) \le \deg(p_{n-1}u - q_{n-1}v)$ для некоторого $n \ge 1$, то $p(x) = cp_{n-1}(x)$ и $q(x) = cq_{n-1}(x)$ для некоторой константы c. В частности, каждый $q_n(x)$ является полиномом, "разбивающим запись" (record-breaking) в том смысле, что не существует ненулевого полинома $q(x)$ меньшей степени, такого, что для любого полинома $p(x)$ полином $p(x)u(x) - q(x)v(x)$ имел столь же малую степень, что и $p_n(x)u(x) - q_n(x)v(x)$.

27. [*M23*] Предложите путь ускорения деления $u(x)$ на $v(x)$, если заранее известно, что остаток будет нулевым.

*4.6.2. Разложение полиномов на множители

Рассмотрим теперь задачу не только поиска наибольшего общего делителя двух или более полиномов, но и *разложения* полиномов.

Разложение по модулю p. Как и в случае целых чисел (разделы 4.5.2 и 4.5.4), задача разложения представляется существенно сложнее поиска наибольшего общего делителя. Однако разложение полиномов по модулю некоторого простого целого числа p не настолько сложно, как кажется. Гораздо проще найти множители произвольного полинома степени n по модулю 2, чем использовать любой известный метод для поиска множителей произвольного n-битового двоичного числа. Эта неожиданная ситуация является следствием поучительного алгоритма разложения, открытого в 1967 году Элвином Р. Берлекампом (Elwyn R. Berlekamp) [*Bell System Technical J.* **46** (1967), 1853–1859].

Пусть p — простое число; все арифметические операции над полиномами в приведенном далее материале выполняются по модулю p. Предположим, задан полином $u(x)$, коэффициенты которого выбраны из множества $\{0, 1, \ldots, p-1\}$. Мы

можем считать, что $u(x)$ нормирован. Наша цель — выразить $u(x)$ в виде

$$u(x) = p_1(x)^{e_1} \dots p_r(x)^{e_r}, \tag{1}$$

где $p_1(x), \dots, p_r(x)$ — различные нормированные неприводимые полиномы.

В качестве первого шага можно использовать стандартную технологию определения, не является ли какой-либо показатель степени $e_1, \dots, e_r$ больше единицы. Если

$$u(x) = u_n x^n + \cdots + u_0 = v(x)^2 w(x), \tag{2}$$

то производная (найденная так, как обычно, но по модулю p) будет равна

$$u'(x) = n u_n x^{n-1} + \cdots + u_1 = 2v(x)v'(x)w(x) + v(x)^2 w'(x), \tag{3}$$

а это выражение кратно $v(x)$. Значит, наш первый шаг в разложении $u(x)$ должен состоять в поиске

$$\gcd\big(u(x), u'(x)\big) = d(x). \tag{4}$$

Если $d(x)$ равно 1, то, как мы только что убедились, $u(x)$ *свободен от квадратов*, т. е. представляет собой произведение различных простых $p_1(x) \dots p_r(x)$. Если $d(x)$ не равно 1 и $d(x) \neq u(x)$, то $d(x)$ является собственным множителем $u(x)$; соотношение между множителями $d(x)$ и множителями $u(x)/d(x)$ ускоряет процесс разложения для этого случая (см. упр. 34 и 36). И наконец, если $d(x) = u(x)$, то необходимо, чтобы $u'(x) = 0$; следовательно, коэффициент u_k при x^k ненулевой только тогда, когда k кратно p. Это означает, что $u(x)$ может быть записан как полином вида $v(x^p)$. В таком случае имеем

$$u(x) = v(x^p) = \big(v(x)\big)^p. \tag{5}$$

Процесс разложения может быть завершен нахождением неприводимых множителей $v(x)$ и возведением их в степень p.

Тождество (5) может показаться читателю несколько странным, однако это важный факт, лежащий в основе алгоритма Берлекампа и других методов, которые мы обсудим чуть позже. Его можно доказать следующим образом. Если $v_1(x)$ и $v_2(x)$ — некоторые полиномы по модулю p, то

$$\begin{aligned}\big(v_1(x)+v_2(x)\big)^p &= v_1(x)^p + \tbinom{p}{1} v_1(x)^{p-1} v_2(x) + \cdots + \tbinom{p}{p-1} v_1(x) v_2(x)^{p-1} + v_2(x)^p \\ &= v_1(x)^p + v_2(x)^p,\end{aligned}$$

поскольку биномиальные коэффициенты $\binom{p}{1}, \dots, \binom{p}{p-1}$ кратны p. Далее, если a — произвольное целое число, имеем $a^p \equiv a$ (по модулю p) согласно теореме Ферма. Таким образом, когда $v(x) = v_m x^m + v_{m-1} x^{m-1} + \cdots + v_0$, находим, что

$$\begin{aligned}v(x)^p &= (v_m x^m)^p + (v_{m-1} x^{m-1})^p + \cdots + (v_0)^p \\ &= v_m x^{mp} + v_{m-1} x^{(m-1)p} + \cdots + v_0 \;=\; v(x^p).\end{aligned}$$

Эти замечания показывают, что задача разложения сводится к задаче разложения свободных от квадратов полиномов. Таким образом, положим, что

$$u(x) = p_1(x) p_2(x) \dots p_r(x) \tag{6}$$

является произведением различных простых сомножителей. К какой хитрости следует прибегнуть, чтобы суметь найти $p_j(x)$, если дан только $u(x)$? Идея Берлекампа

состоит в применении китайской теоремы об остатках, которая справедлива для полиномов так же, как и для целых чисел (см. упр. 3). Если $(s_1, s_2, \ldots, s_r)$ является произвольным набором из r целых чисел по модулю p, из китайской теоремы об остатках вытекает, что *существует единственный полином* $v(x)$, *такой, что*

$$\begin{gathered} v(x) \equiv s_1 \ \text{(по модулю } p_1(x)\text{)}, \quad \ldots, \quad v(x) \equiv s_r \ \text{(по модулю } p_r(x)\text{)}, \\ \deg(v) < \deg(p_1) + \deg(p_2) + \cdots + \deg(p_r) = \deg(u). \end{gathered} \tag{7}$$

Запись "$g(x) \equiv h(x)$ (по модулю $f(x)$)", появляющаяся здесь, имеет тот же смысл, что и "$g(x) \equiv h(x)$ (по модулю $f(x)$ и p)" в упр. 3.2.2–11, поскольку мы рассматриваем полиномиальную арифметику по модулю p. Полином $v(x)$ в (7) предоставляет способ получения множителей $u(x)$, так как при $r \geq 2$ и $s_1 \neq s_2$ мы получим $\gcd\big(u(x), v(x) - s_1\big)$, делящийся на $p_1(x)$, но не на $p_2(x)$.

Поскольку эти наблюдения показывают, что информацию о множителях $u(x)$ можно получить из соответствующих решений $v(x)$ (7), проанализируем (7) более тщательно. Во-первых, можно заметить, что полином $v(x)$ удовлетворяет условию $v(x)^p \equiv s_j^p = s_j \equiv v(x)$ (по модулю $p_j(x)$) для $1 \leq j \leq r$. Таким образом,

$$v(x)^p \equiv v(x) \ \text{(по модулю } u(x)\text{)}, \qquad \deg(v) < \deg(u). \tag{8}$$

Во-вторых, имеется основное полиномиальное тождество

$$x^p - x \equiv (x-0)(x-1)\ldots(x-(p-1)) \ \text{(по модулю } p\text{)} \tag{9}$$

(см. упр. 6). Следовательно,

$$v(x)^p - v(x) = \big(v(x)-0\big)\big(v(x)-1\big)\ldots\big(v(x)-(p-1)\big) \tag{10}$$

является тождеством для любого полинома $v(x)$ при работе по модулю p. Если $v(x)$ удовлетворяет (8), то $u(x)$ делит левую часть (10), так что каждый неприводимый множитель $u(x)$ должен делить один из p взаимно простых множителей правой части (10). Другими словами, *все* решения (8) должны иметь вид (7) для некоторых $s_1, s_2, \ldots, s_r$; *имеется в точности* p^r *решений* (8).

Решения $v(x)$ уравнения (8), таким образом, дают ключ к разложению $u(x)$. Сначала поиск всех решений (8) может показаться более сложным, чем разложение $u(x)$, но в действительности это не так, поскольку множество решений (8) замкнуто по отношению к сложению. Пусть $\deg(u) = n$. Можно построить матрицу $n \times n$

$$Q = \begin{pmatrix} q_{0,0} & q_{0,1} & \cdots & q_{0,n-1} \\ \vdots & \vdots & & \vdots \\ q_{n-1,0} & q_{n-1,1} & \cdots & q_{n-1,n-1} \end{pmatrix}, \tag{11}$$

где

$$x^{pk} \equiv q_{k,n-1}x^{n-1} + \cdots + q_{k,1}x + q_{k,0} \ \text{(по модулю } u(x)\text{)}. \tag{12}$$

В таком случае $v(x) = v_{n-1}x^{n-1} + \cdots + v_1x + v_0$ является решением (8) тогда и только тогда, когда

$$(v_0, v_1, \ldots, v_{n-1})Q = (v_0, v_1, \ldots, v_{n-1}). \tag{13}$$

Последнее соотношение, в свою очередь, справедливо тогда и только тогда, когда

$$v(x) = \sum_j v_j x^j = \sum_j \sum_k v_k q_{k,j} x^j \equiv \sum_k v_k x^{pk} = v(x^p) \equiv v(x)^p \ \text{(по модулю } u(x)\text{)}.$$

Значит, алгоритм разложения Берлекампа выглядит следующим образом.

B1. Добиться, чтобы $u(x)$ был свободен от квадратов; другими словами, если $\gcd\big(u(x), u'(x)\big) \neq 1$, свести задачу к разложению $u(x)$, как указывалось ранее в этом разделе.

B2. Построить матрицу Q, определенную (11) и (12). Это можно сделать одним из двух способов в зависимости от того, очень ли велико p, как поясняется ниже.

B3. "Триангуляризовать" матрицу $Q-I$, где $I = (\delta_{ij})$ — единичная матрица размера $n \times n$, найти ее ранг $n-r$ и найти линейно независимые векторы $v^{[1]}, \ldots, v^{[r]}$, такие, что $v^{[j]}(Q-I) = (0, 0, \ldots, 0)$ для $1 \le j \le r$. (В качестве первого вектора $v^{[1]}$ всегда можно взять вектор $(1, 0, \ldots, 0)$, представляющий тривиальное решение уравнения (8) $v^{[1]}(x) = 1$. Вычисления могут быть проведены с использованием соответствующих операций со столбцами, как описывается в приведенном ниже алгоритме N.) *В этот момент r равно количеству неприводимых множителей* $u(x)$, потому что решениями (8) являются p^r полиномов, соответствующих векторам $t_1 v^{[1]} + \cdots + t_r v^{[r]}$ для всех выборов целых чисел $0 \le t_1, \ldots, t_r < p$. Таким образом, если $r = 1$, то известно, что $u(x)$ неприводим, и на этом работа алгоритма завершается.

B4. Вычислить $\gcd\big(u(x), v^{[2]}(x) - s\big)$ для $0 \le s < p$, где $v^{[2]}(x)$ — полином, представленный вектором $v^{[2]}$. Результатом будет нетривиальное разложение $u(x)$, потому что полином $v^{[2]}(x) - s$ ненулевой и имеет степень, меньшую, чем $\deg(u)$. Согласно упр. 7 имеем

$$u(x) = \prod_{0 \le s < p} \gcd\big(v(x) - s,\, u(x)\big) \tag{14}$$

в случае, если $v(x)$ удовлетворяет (8).

Если использование $v^{[2]}(x)$ не приводит к разложению $u(x)$ на r множителей, то дальнейшие множители могут быть получены посредством вычисления $\gcd\big(v^{[k]}(x) - s,\, w(x)\big)$ для $0 \le s < p$ и всех найденных множителей $w(x)$ при $k = 3, 4, \ldots$, пока не будут получены все r множителей. (Если в (7) выбрано $s_i \neq s_j$, то получим решение $v(x)$ уравнения (8), которое "отличает" $p_i(x)$ от $p_j(x)$; некоторый $v^{[k]}(x) - s$ будет делиться на $p_i(x)$, но не на $p_j(x)$, так что в результате этой процедуры в конечном счете будут найдены все множители.)

Если p равно 2 или 3, вычисления на этом шаге весьма эффективны; но если p больше, чем, скажем, 25, то можно использовать гораздо лучший способ, который рассматривается ниже. ▮

Историческая справка. М. Ч. Р. Батлер (M. C. R. Butler) [*Quart. J. Math.* **5** (1954), 102–107] обнаружил, что матрица $Q-I$, соответствующая свободному от квадратов полиному с r неприводимыми множителями, будет иметь ранг $n-r$ по модулю p. На самом деле этот факт неявно содержится в более общем результате К. Петра (K. Petr) [*Časopis pro Pěstování Matematiky a Fysiky* **66** (1937), 85–94], который определил характеристический полином для Q. См. также Š. Schwarz, *Quart. J. Math.* **7** (1956), 110–124.

В качестве примера работы алгоритма B найдем разложение полинома

$$u(x) = x^8 + x^6 + 10x^4 + 10x^3 + 8x^2 + 2x + 8 \tag{15}$$

по модулю 13 (этот полином появляется в ряде примеров в разделе 4.6.1). Быстрое вычисление с использованием алгоритма 4.6.1E показывает, что $\gcd\big(u(x), u'(x)\big) = 1$. Таким образом, $u(x)$ свободен от квадратов, и мы возвращаемся к шагу B2. На шаге B2 вычисляется матрица Q, которая в нашем случае представляет собой массив размера 8×8. Первая строка Q всегда равна $(1, 0, 0, \ldots, 0)$ и представляет полином $x^0 \bmod u(x) = 1$. Вторая строка представляет $x^{13} \bmod u(x)$, и, в общем, $x^k \bmod u(x)$ легко может быть определено следующим образом (для относительно небольших значений k). Если

$$u(x) = x^n + u_{n-1}x^{n-1} + \cdots + u_1 x + u_0$$

и если

$$x^k \equiv a_{k,n-1}x^{n-1} + \cdots + a_{k,1}x + a_{k,0} \quad \big(\text{по модулю } u(x)\big),$$

то

$$\begin{aligned} x^{k+1} &\equiv a_{k,n-1}x^n + \cdots + a_{k,1}x^2 + a_{k,0}x \\ &\equiv a_{k,n-1}(-u_{n-1}x^{n-1} - \cdots - u_1 x - u_0) + a_{k,n-2}x^{n-1} + \cdots + a_{k,0}x \\ &= a_{k+1,n-1}x^{n-1} + \cdots + a_{k+1,1}x + a_{k+1,0}, \end{aligned}$$

где

$$a_{k+1,j} = a_{k,j-1} - a_{k,n-1}u_j. \tag{16}$$

В этой формуле $a_{k,-1}$ трактуется как нуль, так что $a_{k+1,0} = -a_{k,n-1}u_0$. Простая рекуррентность наподобие регистра сдвига (16) упрощает вычисление $x^k \bmod u(x)$ для $k = 1, 2, 3, \ldots, (n-1)p$. В компьютере такое вычисление в общем случае производится с помощью одномерного массива $(a_{n-1}, \ldots, a_1, a_0)$, а также установок

$$t \leftarrow a_{n-1},\ a_{n-1} \leftarrow (a_{n-2} - tu_{n-1}) \bmod p,\ \ldots,\ a_1 \leftarrow (a_0 - tu_1) \bmod p$$

и $a_0 \leftarrow (-tu_0) \bmod p$. (Мы сталкивались с подобными процедурами в связи с генерированием случайных чисел; см. 3.2.2–(10).) Для используемого в качестве примера полинома $u(x)$ (15), получим такую последовательность коэффициентов $x^k \bmod u(x)$ с использованием арифметики по модулю 13.

k	$a_{k,7}$	$a_{k,6}$	$a_{k,5}$	$a_{k,4}$	$a_{k,3}$	$a_{k,2}$	$a_{k,1}$	$a_{k,0}$
0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1	0
2	0	0	0	0	0	1	0	0
3	0	0	0	0	1	0	0	0
4	0	0	0	1	0	0	0	0
5	0	0	1	0	0	0	0	0
6	0	1	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0
8	0	12	0	3	3	5	11	5
9	12	0	3	3	5	11	5	0
10	0	4	3	2	8	0	2	8
11	4	3	2	8	0	2	8	0
12	3	11	8	12	1	2	5	7
13	11	5	12	10	11	7	1	2

Таким образом, второй строкой матрицы Q является $(2,1,7,11,10,12,5,11)$. Аналогично можно определить $x^{26} \bmod u(x), \ldots, x^{91} \bmod u(x)$ и найти, что

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 7 & 11 & 10 & 12 & 5 & 11 \\ 3 & 6 & 4 & 3 & 0 & 4 & 7 & 2 \\ 4 & 3 & 6 & 5 & 1 & 6 & 2 & 3 \\ 2 & 11 & 8 & 8 & 3 & 1 & 3 & 11 \\ 6 & 11 & 8 & 6 & 2 & 7 & 10 & 9 \\ 5 & 11 & 7 & 10 & 0 & 11 & 7 & 12 \\ 3 & 3 & 12 & 5 & 0 & 11 & 9 & 12 \end{pmatrix},$$

$$Q - I = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 7 & 11 & 10 & 12 & 5 & 11 \\ 3 & 6 & 3 & 3 & 0 & 4 & 7 & 2 \\ 4 & 3 & 6 & 4 & 1 & 6 & 2 & 3 \\ 2 & 11 & 8 & 8 & 2 & 1 & 3 & 11 \\ 6 & 11 & 8 & 6 & 2 & 6 & 10 & 9 \\ 5 & 11 & 7 & 10 & 0 & 11 & 6 & 12 \\ 3 & 3 & 12 & 5 & 0 & 11 & 9 & 11 \end{pmatrix}. \tag{17}$$

Этим завершается шаг B2. На следующем шаге процедуры Берлекампа требуется найти ядро преобразования, осуществляемого матрицей $Q - I$. В общем, предположим, что A — матрица размера $n \times n$ над полем, ранг которой $n - r$ необходимо определить. Предположим также, что нужно определить линейно независимые векторы $v^{[1]}$, $v^{[2]}$, $\ldots$, $v^{[r]}$, такие, что $v^{[1]}A = v^{[2]}A = \cdots = v^{[r]}A = (0, \ldots, 0)$. Алгоритм для этого вычисления может быть основан на том наблюдении, что любой столбец матрицы A можно умножить на ненулевую величину и что это произведение можно добавить к любому другому столбцу матрицы без изменения ранга матрицы или векторов $v^{[1]}, \ldots, v^{[r]}$ (подобные преобразования равносильны замене матрицы A матрицей AB, где B представляет собой несингулярную матрицу). Таким образом, может быть использована следующая хорошо известная процедура "триангуляризации".

Алгоритм N (*Алгоритм ядра*). Пусть A — матрица размера $n \times n$, элементы которой a_{ij} принадлежат полю и имеют индексы в диапазоне $0 \le i, j < n$. Этот алгоритм дает r векторов $v^{[1]} \ldots v^{[r]}$, линейно независимых над полем и удовлетворяющих условию $v^{[j]}A = (0, \ldots, 0)$, где $n - r$ — ранг матрицы A.

N1. [Инициализация.] Установить $c_0 \leftarrow c_1 \leftarrow \cdots \leftarrow c_{n-1} \leftarrow -1$, $r \leftarrow 0$. (Во время вычислений $c_j \ge 0$ будет выполняться только тогда, когда $a_{c_j j} = -1$, а все другие элементы строки c_j будут нулевыми.)

N2. [Цикл по k.] Выполнить шаг N3 для $k = 0, 1, \ldots, n-1$, затем завершить работу алгоритма.

N3. [Проверка зависимости строк.] Если существует некоторое j из интервала $0 \le j < n$, такое, что $a_{kj} \ne 0$ и $c_j < 0$, то выполнить следующее. Умножить столбец j матрицы A на $-1/a_{kj}$ (так, чтобы a_{kj} стало равным -1), добавить умноженный на a_{ki} j-й столбец к i-му столбцу для всех $i \ne j$ и наконец установить $c_j \leftarrow k$ (поскольку, как нетрудно показать, что $a_{sj} = 0$ для всех $s < k$, эти операции не влияют на строки $0, 1, \ldots, k-1$ матрицы A).

С другой стороны, если не существует такого j из диапазона $0 \le j < n$, что $a_{kj} \neq 0$ и $c_j < 0$, следует установить $r \leftarrow r+1$ и вывести вектор

$$v^{[r]} = (v_0, v_1, \ldots, v_{n-1}),$$

определяемый правилом

$$v_j = \begin{cases} a_{ks}, & \text{если } c_s = j \ge 0; \\ 1, & \text{если } j = k; \\ 0 & \text{в противном случае.} \end{cases} \quad (18)$$

Лучше всего механизм работы этого алгоритма проиллюстрировать на конкретном примере. Пусть A — матрица $Q - I$ из (17) над полем целых чисел по модулю 13. При $k = 0$ получим вектор $v^{[1]} = (1,0,0,0,0,0,0,0)$. При $k = 1$ на шаге N3 можно принять j равным 0, 2, 3, 4, 5, 6 либо 7; выбор здесь абсолютно произволен, хотя он и повлияет на вид векторов, выдаваемых алгоритмом. При вычислениях вручную удобнее всего взять $j = 5$, поскольку $a_{15} = 12 = -1$. Операции над столбцами на шаге N3 преобразуют матрицу A в матрицу

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 \\ 11 & 6 & 5 & 8 & 1 & 4 & 1 & 7 \\ 3 & 3 & 9 & 5 & 9 & 6 & 6 & 4 \\ 4 & 11 & 2 & 6 & 12 & 1 & 8 & 9 \\ 5 & 11 & 11 & 7 & 10 & 6 & 1 & 10 \\ 1 & 11 & 6 & 1 & 6 & 11 & 9 & 3 \\ 12 & 3 & 11 & 9 & 6 & 11 & 12 & 2 \end{pmatrix}.$$

(Элемент в кружке на пересечении столбца 5 и строки 1 используется здесь для того, чтобы указать, что $c_5 = 1$. Помните, что алгоритм N нумерует строки и столбцы матрицы, начиная с 0, а не с 1.) Когда $k = 2$, можно выбрать $j = 4$ и аналогично получить следующие матрицы с тем же ядром, что и у $Q - I$.

$k = 2$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 & 0 \\ 8 & 1 & 3 & 11 & 4 & 9 & 10 & 6 \\ 2 & 4 & 7 & 1 & 1 & 5 & 9 & 3 \\ 12 & 3 & 0 & 5 & 3 & 5 & 4 & 5 \\ 0 & 1 & 2 & 5 & 7 & 0 & 3 & 0 \\ 11 & 6 & 7 & 0 & 7 & 0 & 6 & 12 \end{pmatrix}$$

$k = 3$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 & 0 \\ 0 & \textcircled{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ 9 & 9 & 8 & 9 & 11 & 8 & 8 & 5 \\ 1 & 10 & 4 & 11 & 4 & 4 & 0 & 0 \\ 5 & 12 & 12 & 7 & 3 & 4 & 6 & 7 \\ 2 & 7 & 2 & 12 & 9 & 11 & 11 & 2 \end{pmatrix}$$

$k = 4$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 & 0 \\ 0 & \textcircled{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textcircled{12} \\ 1 & 10 & 4 & 11 & 4 & 4 & 0 & 0 \\ 8 & 2 & 6 & 10 & 11 & 11 & 0 & 9 \\ 1 & 6 & 4 & 11 & 2 & 0 & 0 & 10 \end{pmatrix}$$

$k = 5$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & \textcircled{12} & 0 & 0 & 0 \\ 0 & \textcircled{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \textcircled{12} \\ \textcircled{12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 5 & 5 & 0 & 9 \\ 12 & 9 & 0 & 0 & 11 & 9 & 0 & 10 \end{pmatrix}$$

Теперь каждый столбец, в котором нет элемента в кружкé, полностью нулевой; так что при $k = 6$ и $k = 7$ алгоритм дает еще два вектора, а именно:

$$v^{[2]} = (0,5,5,0,9,5,1,0), \qquad v^{[3]} = (0,9,11,9,10,12,0,1).$$

Из вида матрицы A после пятого преобразования ясно, что эти векторы удовлетворяют уравнению $vA = (0,\ldots,0)$. Поскольку вычисление дает три линейно независимых вектора, $u(x)$ должен иметь ровно три неприводимых множителя.

В заключение можно перейти к шагу B4 процедуры разложения. Вычисление $\gcd\big(u(x), v^{[2]}(x) - s\big)$ для $0 \le s < 13$, где $v^{[2]}(x) = x^6 + 5x^5 + 9x^4 + 5x^2 + 5x$, дает $x^5 + 5x^4 + 9x^3 + 5x + 5$ в качестве ответа при $s = 0$ и $x^3 + 8x^2 + 4x + 12$ при $s = 2$; для остальных значений s gcd равен единице. Таким образом, $v^{[2]}(x)$ дает только два из трех множителей. Обратившись к $\gcd\big(v^{[3]}(x) - s,\ x^5 + 5x^4 + 9x^3 + 5x + 5\big)$, где $v^{[3]}(x) = x^7 + 12x^5 + 10x^4 + 9x^3 + 11x^2 + 9x$, получим множитель $x^4 + 2x^3 + 3x^2 + 4x + 6$ для $s = 6$, $x + 3$ для $s = 8$ и единицу в противном случае. Поэтому полное разложение имеет вид

$$u(x) = (x^4 + 2x^3 + 3x^2 + 4x + 6)(x^3 + 8x^2 + 4x + 12)(x + 3). \tag{19}$$

Оценим теперь время работы алгоритма Берлекампа при разложении полинома n-й степени по модулю p. Прежде всего, будем считать, что p относительно мало́, так что четыре арифметические операции по модулю p, по существу, выполняются за некоторый фиксированный промежуток времени (деление по модулю p может быть преобразовано в умножение путем хранения таблицы обратных величин, как предложено в упр. 9; например, при работе по модулю 13 имеем $\frac{1}{2} = 7$, $\frac{1}{3} = 9$ и т. д.). Для вычислений на шаге B1 необходимо $O(n^2)$ единиц времени; шаг B2 требует $O(pn^2)$ единиц. Для шага B3 мы воспользовались алгоритмом N, которому нужно не более $O(n^3)$ единиц времени. И наконец, на шаге B4 можно увидеть, что вычисление $\gcd\big(f(x), g(x)\big)$ по алгоритму Евклида требует $O\big(\deg(f)\,\deg(g)\big)$ единиц времени. Следовательно, вычисление $\gcd\big(v^{[j]}(x) - s,\ w(x)\big)$ при фиксированных j и s для всех найденных множителей $w(x)$ полинома $u(x)$ займет $O(n^2)$ единиц. Шаг B4, таким образом, требует не более $O(prn^2)$ единиц времени. *Процедура Берлекампа разлагает на множители по модулю p произвольный полином степени n за $O(n^3 + prn^2)$ шагов* при условии, что p — небольшое простое число; в упр. 5 показано, что среднее количество множителей r примерно равно $\ln n$. Таким образом, алгоритм Берлекампа гораздо быстрее любого известного метода разложения n-значного числа в системе счисления с основанием p.

Конечно, при малых n и p разложение методом проб и ошибок, аналогичное алгоритму 4.5.4A, будет еще более быстрым, чем метод Берлекампа. Из упр. 1 следует, что при малом p стоит отбросить множители малой степени, прежде чем переходить к более сложной процедуре, даже если n велико.

При больших p следовало бы использовать другую реализацию алгоритма Берлекампа. Деление по модулю p не нужно выполнять при помощи вспомогательной таблицы обратных чисел. Вместо этого, вероятно, следует применять метод из упр. 4.5.2–16, который требует $O\big((\log p)^2\big)$ шагов. Тогда для шага B1 понадобится $O\big(n^2(\log p)^2\big)$ единиц времени, а для шага B3 — $O\big(n^3(\log p)^2\big)$ единиц. На шаге B2 при больши́х p можно вычислить $x^p \bmod u(x)$ более эффективным способом, чем (16). В разделе 4.6.3 показано, что эту величину можно получить, по существу,

с помощью $O(\log p)$ операций возведения в квадрат по модулю $u(x)$, т. е. переходов от $x^k \bmod u(x)$ к $x^{2k} \bmod u(x)$, совместно с операциями умножения на x. Операция возведения в квадрат выполняется относительно просто, если сначала создать вспомогательную таблицу значений $x^m \bmod u(x)$ для $m = n,\ n+1,\ \ldots,\ 2n-2$. Если $x^k \bmod u(x) = c_{n-1}x^{n-1} + \cdots + c_1x + c_0$, то

$$x^{2k} \bmod u(x) = \left(c_{n-1}^2 x^{2n-2} + \cdots + (c_1c_0 + c_1c_0)x + c_0^2\right) \bmod u(x),$$

где $x^{2n-2},\ \ldots,\ x^n$ могут быть заменены полиномами из вспомогательной таблицы. Общее время вычисления $x^p \bmod u(x)$ сводится к $O\big(n^2(\log p)^3\big)$ единицам, и мы получаем вторую строку матрицы Q. Для получения следующих строк матрицы Q можно вычислить $x^{2p} \bmod u(x)$, $x^{3p} \bmod u(x)$, $\ldots$, выполнив многократное умножение на $x^p \bmod u(x)$ аналогично возведению в квадрат по модулю $u(x)$. Шаг B2 завершается за $O\big(n^3(\log p)^2\big)$ дополнительных единиц времени. Таким образом, шаги B1–B3 требуют в сумме $O\big(n^2(\log p)^3 + n^3(\log p)^2\big)$ единиц времени; эти три шага позволяют получить количество делителей $u(x)$.

Но на шаге B4 необходимо вычислить наибольший общий делитель для p различных значений s, что очень сложно даже при умеренно больших значениях p. Впервые это препятствие было преодолено Гансом Зассенхаузом (Hans Zassenhaus) [*J. Number Theory* **1** (1969), 291–311], который показал, как определить все "полезные" значения s (см. упр. 14). Еще лучший путь был найден Зассенхаузом и Кантором (Cantor) в 1980 году. Если $v(x)$ представляет собой некоторое решение (8), то $u(x)$ делит $v(x)^p - v(x) = v(x)\cdot\big(v(x)^{(p-1)/2}+1\big)\cdot\big(v(x)^{(p-1)/2}-1\big)$. Предполагается, что мы вычисляем

$$\gcd\big(u(x),\ v(x)^{(p-1)/2} - 1\big); \tag{20}$$

при небольшом везении (20) будет нетривиальным множителем $u(x)$. В действительности можно точно определить нужную степень везения, рассмотрев (7). Пусть $v(x) \equiv s_j$ по модулю $p_j(x)$ для $1 \le j \le r$. В таком случае $p_j(x)$ делит $v(x)^{(p-1)/2} - 1$ тогда и только тогда, когда $s_j^{(p-1)/2} \equiv 1$ по модулю p. Известно, что в точности $(p-1)/2$ целых s в диапазоне $0 \le s < p$ удовлетворяют условию $s^{(p-1)/2} \equiv 1$ (по модулю p); следовательно, около половины $p_j(x)$ появятся в gcd (20). Точнее, если $v(x)$ — случайное решение (8), где все p^r решений равновероятны, вероятность того, что gcd (20) равен $u(x)$, в точности равна

$$\big((p-1)/2p\big)^r,$$

а вероятность, что это равно 1, составляет $\big((p+1)/2p\big)^r$. Вероятность получения нетривиального множителя будет, таким образом, равна

$$1 - \left(\frac{p-1}{2p}\right)^r - \left(\frac{p+1}{2p}\right)^r = 1 - \frac{1}{2^{r-1}}\left(1 + \binom{r}{2}p^{-2} + \binom{r}{4}p^{-4} + \cdots\right) \ge \frac{4}{9}$$

для всех $r \ge 2$ и $p \ge 3$.

Таким образом, неплохой идеей является замена шага B4 следующей процедурой (кроме случаев, когда p мало́): установить $v(x) \leftarrow a_1v^{[1]}(x) + a_2v^{[2]}(x) + \cdots + a_rv^{[r]}(x)$, где коэффициенты a_j выбраны случайно в диапазоне $0 \le a_j < p$. Пусть текущее частичное разложение $u(x)$ представляет собой $u_1(x)\ldots u_t(x)$, где t изначально равно 1. Вычислим

$$g_i(x) = \gcd\big(u_i(x),\ v(x)^{(p-1)/2} - 1\big)$$

для всех i, таких, что $\deg(u_i) > 1$; заменим $u_i(x)$ на $g_i(x)\cdot\big(u_i(x)/g_i(x)\big)$ и будем увеличивать значение t после каждого найденного нетривиального gcd. Будем повторять процесс для различных вариантов $v(x)$ до тех пор, пока не получим $t = r$.

Если предположить, что потребуется всего $O(\log r)$ случайных решений $v(x)$ уравнения (8) (что вполне допустимо), то можно указать верхнюю границу времени, необходимого для выполнения этой альтернативы шагу B4. Она требует $O\big(rn(\log p)^2\big)$ шагов для вычисления $v(x)$, $O\big(d^2(\log p)^3\big)$ шагов для вычисления $v(x)^{(p-1)/2} \bmod u_i(x)$ (если $\deg(u_i) = d$) и $O\big(d^2(\log p)^2\big)$ шагов для поиска $\gcd\big(u_i(x), v(x)^{(p-1)/2} - 1\big)$. Таким образом, общее время составляет $O\big(n^2(\log p)^3 \log r\big)$.

Разложение с различными степенями. Теперь обратимся к несколько более простому пути поиска разложения по модулю p. Изложенные в этом разделе идеи включают множество поучительных обращений к вычислительной алгебре, и автор не извиняется перед читателем за их количество. Однако проблема разложения по модулю p фактически может быть решена без обращения к множеству концепций.

Во-первых, можно использовать тот факт, что неприводимый полином $q(x)$ степени d является делителем $x^{p^d} - x$ и не является делителем $x^{p^c} - x$ для $1 \le c < d$ (см. упр. 16). Таким образом, можно исключить неприводимые множители каждой степени раздельно, выбрав следующую стратегию.

D1. Исключить квадратные множители, как в алгоритме Берлекампа. Установить также $v(x) \leftarrow u(x)$, $w(x) \leftarrow$ "x" и $d \leftarrow 0$ (здесь $v(x)$ и $w(x)$ — переменные, имеющие в качестве значений полиномы).

D2. (Сейчас $w(x) = x^{p^d} \bmod v(x)$; все неприводимые множители $v(x)$ различны и имеют степень $> d$.) Если $d + 1 > \frac{1}{2}\deg(v)$, выполнение процедуры прекращается, поскольку либо $v(x) = 1$, либо $v(x)$ — неприводимый полином. В противном случае увеличить d на 1 и заменить $w(x)$ на $w(x)^p \bmod v(x)$.

D3. Найти $g_d(x) = \gcd\big(w(x) - x, v(x)\big)$. (Это произведение всех неприводимых множителей $u(x)$, степени которых равны d.) Если $g_d(x) \neq 1$, заменить $v(x)$ на $v(x)/g_d(x)$ и $w(x)$ на $w(x) \bmod v(x)$. Если степень $g_d(x)$ больше, чем d, использовать приведенный ниже алгоритм для поиска его множителей. Вернуться к шагу D2. ▮

Эта процедура позволяет определить произведение всех неприводимых множителей со степенью d и таким образом выяснить, сколько существует множителей конкретной степени. Поскольку три множителя в нашем примере полинома (19) имеют различные степени, все они могут быть найдены без разложения полиномов $g_d(x)$.

Для завершения метода необходим путь, предоставляющий возможность разделить полином $g_d(x)$ на неприводимые множители, когда $\deg(g_d) > d$. Майкл Рабин (Michael Rabin) в 1976 году доказал, что это можно сделать при помощи арифметических операций в поле из p^d элементов. Дэвид Д. Кантор (David G. Cantor) и Ганс Зассенхауз (Hans Zassenhaus) открыли в 1979 году, что существует еще более простой путь, основанный на следующем тождестве: если p — некоторое нечетное простое число, то имеем

$$g_d(x) = \gcd\big(g_d(x), t(x)\big)\ \gcd\big(g_d(x),\ t(x)^{(p^d-1)/2} + 1\big)\ \gcd\big(g_d(x),\ t(x)^{(p^d-1)/2} - 1\big) \quad (21)$$

для всех полиномов $t(x)$, поскольку $t(x)^{p^d} - t(x)$ кратно всем неприводимым полиномам степени d ($t(x)$ можно рассматривать как элемент поля размером p^d, когда такое поле состоит из всех полиномов по модулю неприводимого $f(x)$, как в упр. 16). В упр. 29 показано, что $\gcd\big(g_d(x),\ t(x)^{(p^d-1)/2}-1\big)$ будет нетривиальным множителем $g_d(x)$ примерно в половине случаев, если $t(x)$ — случайный полином степени $\le 2d-1$; следовательно, не придется предпринимать множество случайных попыток, чтобы найти все делители. Без потери общности можно положить, что $t(x)$ нормирован, поскольку целые кратные $t(x)$ не приводят к отличиям, кроме возможного изменения знака $t(x)^{(p^d-1)/2}$. Таким образом, когда $d = 1$, можно получить $t(x) = x + s$, где s выбрано случайно.

Иногда эта процедура будет в действительности успешной для $d > 1$, когда используются только линейные полиномы $t(x)$. Например, имеется восемь неприводимых полиномов $f(x)$ степени 3 по модулю 3 и для всех из них по-разному вычисляются $\gcd\big(f(x),\ (x+s)^{13}-1\big)$ для $0 \le s < 3$.

$f(x)$	$s=0$	$s=1$	$s=2$
$x^3 + 2x + 1$	1	1	1
$x^3 + 2x + 2$	$f(x)$	$f(x)$	$f(x)$
$x^3 + x^2 + 2$	$f(x)$	$f(x)$	1
$x^3 + x^2 + x + 2$	$f(x)$	1	$f(x)$
$x^3 + x^2 + 2x + 1$	1	$f(x)$	$f(x)$
$x^3 + 2x^2 + 1$	1	$f(x)$	1
$x^3 + 2x^2 + x + 1$	1	1	$f(x)$
$x^3 + 2x^2 + 2x + 2$	$f(x)$	1	1

В упр. 31 частично объясняется, почему линейные полиномы могут оказаться эффективными; однако, когда количество неприводимых полиномов степени d превышает 2^p, ясно, что будут существовать неприводимые полиномы, неразличимые посредством линейного выбора $t(x)$.

Альтернатива для (21), которая работает при $p = 2$, обсуждается в упр. 30. Более быстрый алгоритм разложения с различными степенями при очень больших p был найден Э. Калтофеном (E. Kaltofen) и В. Шаупом (V. Shoup); время его работы составляет $O(n^{2.5} + n^{1+\epsilon}\log p)$ арифметических операций по модулю p для чисел практического размера и $O(n^{(5+\omega+\epsilon)/4}\log p)$ таких операций при $n \to \infty$, когда ω является степенью "быстрого" умножения матриц в упр. 4.6.4–66. [См. *J. Symbolic Comp.* **20** (1995), 363–397; *Math. Comp.* **67** (1998), 1179–1197.]

Исторические справки. Идея поиска всех линейных множителей свободного от квадратов полинома $f(x)$ по модулю p посредством вычисления сначала $g(x) = \gcd\big(x^{p-1}-1,\ f(x)\big)$, а затем — $\gcd\big(g(x),\ (x+s)^{(p-1)/2}\pm 1\big)$ для произвольного s была высказана А. М. Лежандром (A. M. Legendre), *Mémoires Acad. Sci. Paris* (1785), 484–490. Поводом к этому послужил поиск всех целых решений Диофантовых уравнений вида $f(x) = py$, т. е. $f(x) \equiv 0$ (по модулю p). Более общая технология разделения степеней, воплощенная в алгоритме D, была открыта сначала К. Ф. Гауссом (C. F. Gauss) до 1800 года, но не публиковалась [см. его *Werke* **2** (1876), 237], а затем — Эваристом Галуа (Évariste Galois) в классической ныне работе, ставшей основой теории конечных полей [*Bulletin des Sciences Mathématiques, Physiques et Chimiques* **13** (1830), 428–435; перепечатана в *J. de Math. Pures et Appliquées* **11**

(1846), 398–407]. Однако эти работы Гаусса и Галуа опередили свое время и не были поняты, пока Дж. А. Серре (J. A. Serret) не дал детальное толкование несколько позже [*Mémoires Acad. Sci. Paris*, series 2, **35** (1866), 617–688; алгоритм D находится в §7]. Специальные процедуры для разделения $g_d(x)$ на неприводимые множители были разработаны последовательно различными авторами, однако универсальный метод, который оставался бы эффективным при больших p, по-видимому, не был открыт до появления компьютеров, потребовавших его разработки. Первый такой рандомизированный алгоритм со строгим анализом времени работы был опубликован Берлекампом [E. Berlekamp, *Math. Comp.* **24** (1970), 713–735]. Он был усовершенствован и упрощен в работах Robert T. Moenck, *Math. Comp.* **31** (1977), 235–250, M. O. Rabin, *SICOMP* **9** (1980), 273–280, и D. G. Cantor and H. J. Zassenhaus, *Math. Comp.* **36** (1981), 587–592]. Поль Камён (Paul Camion) независимо открыл обобщение для специальных классов полиномов от многих переменных [*Comptes Rendus Acad. Sci. Paris* **A291** (1980), 479–482; *IEEE Trans.* **IT-29** (1983), 378–385].

Среднее количество операций, требующихся для разложения случайного полинома по модулю p, было проанализировано в работе P. Flajolet, X. Gourdon и D. Panario, *Lecture Notes in Comp. Sci.* **1099** (1996), 232–243.

Разложение над кольцом целых чисел. Задача поиска полного разложения полиномов с целыми коэффициентами, когда работа выполняется *не* по модулю p, несколько сложнее предыдущей, но и в этом случае имеется ряд обоснованно эффективных методов решения.

Исаак Ньютон привел метод поиска линейных и квадратичных множителей полиномов с целыми коэффициентами в своей работе *Arithmetica Universalis* (1707). Его метод был расширен в 1793 году астрономом Фридрихом фон Шубертом (Friedrich von Schubert), который показал, как найти все множители степени n за конечное число шагов [см. M. Cantor, *Geschichte der Mathematik* **4** (Leipzig: Teubner, 1908), 136–137]. Л. Кронекер (L. Kronecker) независимо открыл метод Шуберта примерно 90 годами позже, но, к сожалению, этот метод крайне неэффективен при n, равном пяти или превышающем пять. Гораздо лучшие результаты могут быть получены при помощи представленных выше методов разложения по модулю p.

Предположим, что нужно найти неприводимые множители полинома

$$u(x) = u_n x^n + u_{n-1} x^{n-1} + \cdots + u_0, \qquad u_n \neq 0,$$

над кольцом целых чисел. В качестве первого шага можно разделить его на наибольший общий делитель коэффициентов полинома; в результате работа будет продолжена с *примитивным* полиномом. Можно также считать, что $u(x)$ свободен от квадратов (разделив его на $\gcd\big(u(x), u'(x)\big)$, как в упр. 34).

Теперь, если $u(x) = v(x)w(x)$, где каждый из полиномов имеет целые коэффициенты, мы, очевидно, имеем $u(x) \equiv v(x)w(x)$ (по модулю p) для всех простых p, так что существует нетривиальное разложение по модулю p, кроме случая, когда p делит $\ell(u)$. Эффективный алгоритм разложения $u(x)$ по модулю p может, таким образом, использоваться для того, чтобы попытаться реконструировать возможное разложение $u(x)$ над кольцом целых чисел.

Например, пусть

$$u(x) = x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5. \tag{22}$$

Мы уже видели в (19), что

$$u(x) \equiv (x^4 + 2x^3 + 3x^2 + 4x + 6)(x^3 + 8x^2 + 4x + 12)(x + 3) \text{ (по модулю 13)}, \quad (23)$$

а полное разложение $u(x)$ по модулю 2 показывает наличие двух множителей: одного — степени 6 и другого — степени 2 (см. упр. 10). Из (23) можно увидеть, что $u(x)$ не имеет множителей степени 2, так что он должен быть неприводим над кольцом целых чисел.

Этот частный пример, вероятно, был слишком прост; опыт показывает, что большинство неприводимых полиномов могут быть признаны таковыми путем исследования их множителей по модулю нескольких простых чисел, но установить неприводимость просто удается далеко *не* всегда. Например, существуют полиномы, такие, что они могут быть корректно разложены по модулю p для всех простых p с согласующимися степенями множителей, но при этом являющиеся неприводимыми над кольцом целых чисел (см. упр. 12).

Большое семейство неприводимых полиномов рассмотрено в упр. 38, а в упр. 27 доказывается, что почти все полиномы являются неприводимыми над кольцом целых чисел. Однако обычно мы не пытаемся раскладывать на множители случайные полиномы; вероятно, есть некоторая причина ожидать наличия нетривиального множителя у полинома, так что нас интересует метод определения множителей тогда, когда они существуют.

В общем случае найти множители $u(x)$, рассматривая разложения $u(x)$ по различным простым модулям, непросто. Например, если $u(x)$ — это произведение четырех квадратичных полиномов, то возникают трудности при согласовании их образов по отношению к различным простым модулям. Поэтому желательно выбрать одно простое число и посмотреть, сколько информации можно получить, используя его, особенно если кажется, что множители по модулю этого простого числа имеют верные степени.

Одна из идей состоит в использовании в качестве модуля *очень большого* простого числа, достаточно большого для того, чтобы коэффициенты любого корректного разложения $u(x) = v(x)w(x)$ над кольцом целых чисел в действительности находились в диапазоне от $-p/2$ до $p/2$. Тогда все возможные целые множители могут быть получены из множителей, вычисленных по известному нам методу по модулю p.

В упр. 20 показано, как получить неплохую оценку границ коэффициентов множителей полинома. Например, если (22) приводимо, то его множитель $v(x)$ имеет степень ≤ 4, а коэффициенты v не превышают 34 согласно результатам этого упражнения. Таким образом, все потенциальные множители $u(x)$ окажутся совершенно очевидными при работе по модулю любого простого числа $p > 68$. На самом деле полное разложение по модулю 71 равно

$$(x + 12)(x + 25)(x^2 - 13x - 7)(x^4 - 24x^3 - 16x^2 + 31x - 12),$$

и мы тут же видим, что никакие из полученных полиномов не могут быть множителями (22) над кольцом целых чисел, поскольку постоянные члены не делят 5. Кроме того, никаким образом не удается найти делитель (22), группируя два из полученных множителей, поскольку никакие из найденных постоянных членов 12×25, $12 \times (-7)$, $12 \times (-12)$ не равны ± 1 или ± 5 (по модулю 71).

Определение хороших границ коэффициентов множителей полиномов — нетривиальная задача, поскольку в процессе умножения полиномов может встретиться большое количество сокращений. Например, выглядящий совершенно безобидно полином $x^n - 1$ имеет неприводимые множители, коэффициенты которых превышают $\exp(n^{1/\lg\lg n})$ для неограниченно большого количества n. [См. R. C. Vaughan, *Michigan Math. J.* **21** (1974), 289–295.] Разложению полинома $x^n - 1$ посвящено упр. 32.

Вместо большого простого числа p, которое может оказаться просто громадным, если $u(x)$ имеет высокую степень или большие коэффициенты, можно также использовать малые p при условии, что $u(x)$ свободен от квадратов по модулю p. В этом случае для расширения разложения по модулю p единственным образом в разложение по модулю p^e для произвольно большого показателя степени e можно воспользоваться важным построением, известным как лемма Хенселя (Hensel) (см. упр. 22). Если применить лемму Хенселя к (23) с $p = 13$ и $e = 2$, получится единственное разложение

$$u(x) \equiv (x - 36)(x^3 - 18x^2 + 82x - 66)(x^4 + 54x^3 - 10x^2 + 69x + 84)$$

(по модулю 169). Обозначив эти множители как $v_1(x)v_3(x)v_4(x)$, мы видим, что ни $v_1(x)$ и $v_3(x)$ не являются множителями $u(x)$ над кольцом целых чисел, ни их произведение $v_1(x)v_3(x)$ с приведенными по модулю 169 к диапазону $(-\frac{169}{2}, \frac{169}{2})$ коэффициентами. Итак, мы использовали все возможности доказательства того, что $u(x)$ неприводим над кольцом целых чисел — на этот раз используя только его разложение по модулю 13.

Рассмотренный выше пример нетипичен в одном важном отношении: выполнялось разложение *нормированного* полинома $u(x)$ из (22), поэтому можно считать, что все его множители нормированы. Что же делать в случае, когда $u_n > 1$? В такой ситуации старший коэффициент одного из множителей полинома может почти произвольно варьироваться по модулю p^e; мы, конечно, не хотим рассматривать все имеющиеся возможности. Вероятно, читатель уже заметил эту проблему. К счастью, существует простой выход: разложение $u(x) = v(x)w(x)$ влечет за собой разложение $u_n u(x) = v_1(x)w_1(x)$, где $\ell(v_1) = \ell(w_1) = u_n = \ell(u)$. ("Простите, вы не забыли, что я умножил ваш полином на старший коэффициент перед разложением?") Теперь можно действовать так же, как и выше, с использованием $p^e > 2B$, где B ограничивает максимальный коэффициент множителя $u_n u(x)$, а не $u(x)$. Другой путь решения проблемы старшего коэффициента обсуждается в упр. 40.

Объединим весь рассмотренный материал в следующей процедуре.

F1. Найти единственное свободное от квадратов разложение

$$u(x) \equiv \ell(u)v_1(x)\dots v_r(x) \ \text{(по модулю } p^e),$$

где p^e достаточно велико, как пояснялось выше, и где $v_j(x)$ — нормированный полином. (Это будет возможно для некоторых простых чисел p; см. упр. 23.) Установить также $d \leftarrow 1$.

F2. Для каждой комбинации множителей $v(x) = v_{i_1}(x)\dots v_{i_d}(x)$ с $i_1 = 1$, если $d = \frac{1}{2}r$, построить уникальный полином $\bar{v}(x) \equiv \ell(u)v(x)$ (по модулю p^e), коэффициенты которого находятся в интервале $[-\frac{1}{2}p^e \mathinner{..} \frac{1}{2}p^e)$. Если $\bar{v}(x)$ делит $\ell(u)u(x)$, вывести множитель $\operatorname{pp}(\bar{v}(x))$, разделить на него $u(x)$, удалить соот-

ветствующий $v_i(x)$ из списка множителей по модулю p^e, уменьшить r на число удаленных множителей и завершить работу алгоритма, если $d > \frac{1}{2}r$.

F3. Увеличить d на 1 и вернуться к шагу F2, если $d \le \frac{1}{2}r$. ▮

В заключение этого процесса текущее значение $u(x)$ будет последним неприводимым множителем изначально заданного полинома. Заметьте, что, если $|u_0| < |u_n|$, предпочтительно выполнять всю работу с "обращенным полиномом" $u_0x^n + \cdots + u_n$, множители которого представляют собой "обращенные" множители $u(x)$.

Описанная процедура требует выполнения условия $p^e > 2B$, где B — граница коэффициентов *любого* делителя $u_nu(x)$, но можно использовать и гораздо меньшее значение B, если гарантировать, что оно будет верно для делителей степени $\le \frac{1}{2}\deg(u)$. В этом случае тест на делимость на шаге F2 должен применяться к $w(x) = v_1(x)\ldots v_r(x)/v(x)$, а не к $v(x)$ всякий раз, когда $\deg(v) > \frac{1}{2}\deg(u)$.

Можно еще больше снизить B, если гарантировать, что B ограничивает коэффициенты *по меньшей мере одного* корректного делителя $u(x)$ (например, при разложении составного целого N вместо полинома некоторые делители могут быть очень большими, но хотя бы один из них будет $\le \sqrt{N}$). Эта идея, предложенная в работе B. Beauzamy, V. Trevisan, P. S. Wang, *J. Symbolic Comp.* **15** (1993), 393–413, обсуждается в упр. 21. Проверка делимости на шаге F2 должна в таком случае быть применена и к $v(x)$, и к $w(x)$, но вычисления при этом будут выполняться быстрее, так как p^e будет иметь гораздо меньшее значение.

Описанный выше алгоритм имеет очевидное слабое место: может возникнуть необходимость в проверке для $2^{r-1} - 1$ потенциальных множителей $v(x)$. Среднее значение 2^r в случайной ситуации составляет порядка n или, возможно, $n^{1.5}$ (см. упр. 5), но в противном случае понадобится ускорить данную часть программы настолько, насколько это окажется возможно. Один из способов быстрого исключения ложных множителей состоит в первоначальном вычислении младшего коэффициента $\bar{v}(0)$ с продолжением, только если он делит $\ell(u)u(0)$. Сложности, рассмотренные в предыдущих абзацах, не возникают, если это условие делимости не выполнено, поскольку такая проверка корректна даже при $\deg(v) > \frac{1}{2}\deg(u)$.

Другой важный способ ускорения процедуры состоит в таком уменьшении r, чтобы оно отражало истинное количество множителей. Алгоритм разложения на различные степени, приведенный выше, приложим к различным малым простым числам p_j. Таким образом, для каждого простого числа получается множество D_j возможных степеней множителей по модулю p_j (см. упр. 26). Можно представить D_j в виде строки из n битов. Теперь вычислим пересечение $\bigcap D_j$, т. е. побитовое AND этих строк, и выполним шаг F2 только для

$$\deg(i_1) + \cdots + \deg(i_d) \in \bigcap D_j.$$

Кроме того, p выбирается таким образом, чтобы p_j имело наименьшее значение r. Эта технология разработана Дэвидом Р. Мюссером (David R. Musser), который, исходя из своего опыта, предложил испытывать около пяти простых чисел p_j [см. *JACM* **25** (1978), 271–282]. Конечно, следует немедленно остановиться, если текущее пересечение $\bigcap D_j$ показывает, что полином $u(x)$ является неприводимым.

Мюссер привел полное обсуждение метода разложения, подобного описанному выше, в *JACM* **22** (1975), 291–308. Шаги F1–F3 объединяются в усовершенствованном варианте, предложенном в 1978 году Дж. Э. Коллинзом (G. E. Collins).

Он заключается в поиске пробных делителей путем одновременного получения комбинаций из d множителей вместо комбинаций с общей степенью d. Это усовершенствование важно в связи со статистическим поведением множителей полиномов по модулю p, которые неприводимы над полем рациональных чисел (см. упр. 37).

А. К. Ленстра (A. K. Lenstra), Х. В. Ленстра (мл.) (H. W. Lenstra, Jr.) и Л. Ловас (L. Lovász) предложили свой известный "LLL-алгоритм" разложения полинома над кольцом целых чисел с точными границами количества вычислений в худшем случае [*Math. Annalen* **261** (1982), 515–534]. Их метод не требует случайных чисел, а время его работы для полинома $u(x)$ степени n составляет $O\bigl(n^{12} + n^9(\log\|u\|)^3\bigr)$ битовых операций, где $\|u\|$ определяется в упр. 20. Эта оценка включает время поиска подходящего простого числа p и всех множителей по модулю p при помощи алгоритма B. Конечно, эвристические методы, использующие рандомизацию, на практике работают значительно быстрее.

Наибольшие общие делители. Подобные технологии могут применяться и для вычисления наибольших общих делителей полиномов: если $\gcd\bigl(u(x),v(x)\bigr) = d(x)$ над кольцом целых чисел и если $\gcd\bigl(u(x),v(x)\bigr) = q(x)$ (по модулю p), где $q(x)$ — нормированный полином, то $d(x)$ является общим делителем $u(x)$ и $v(x)$ по модулю p; следовательно,

$$d(x) \text{ делит } q(x) \quad (\text{по модулю } p). \tag{24}$$

Если p не делит старшие коэффициенты ни u, ни v, то p не делит и старший коэффициент d; в таком случае $\deg(d) \le \deg(q)$. Если для такого простого p $q(x) = 1$, то $\deg(d) = 0$ и $d(x) = \gcd\bigl(\operatorname{cont}(u),\operatorname{cont}(v)\bigr)$. Это подтверждает сделанное в разделе 4.6.1 примечание о том, что простого вычисления $\gcd\bigl(u(x),v(x)\bigr)$ по модулю 13 в 4.6.1–(6) достаточно для доказательства того, что $u(x)$ и $v(x)$ взаимно просты над кольцом целых чисел; тем самым сравнительно трудоемких вычислений согласно алгоритму 4.6.1E или 4.6.1C можно избежать. Поскольку два случайных примитивных полинома почти всегда взаимно просты над кольцом целых чисел и поскольку они взаимно просты по модулю простого числа p с вероятностью $1 - 1/p$ в соответствии с упр. 4.6.1–5, вычисления обычно стоит производить по модулю p.

Как отмечалось ранее, нужны хорошие методы и для неслучайных полиномов, встречающихся на практике. Таким образом, мы хотим улучшить свои методы и научиться находить $\gcd\bigl(u(x),v(x)\bigr)$ в общем случае над кольцом целых чисел, основываясь только на информации, которая была получена при работе по модулю простых чисел p. Можно считать, что $u(x)$ и $v(x)$ — примитивные полиномы.

Вместо непосредственного вычисления $\gcd\bigl(u(x),v(x)\bigr)$ удобнее искать полином

$$\bar{d}(x) = c \cdot \gcd\bigl(u(x),v(x)\bigr), \tag{25}$$

где константа c выбирается таким образом, что

$$\ell(\bar{d}) = \gcd\bigl(\ell(u),\ell(v)\bigr). \tag{26}$$

Выбрав подходящее c, можно всегда достичь выполнения данного условия, поскольку старший коэффициент любого общего делителя $u(x)$ и $v(x)$ должен быть делителем $\gcd\bigl(\ell(u),\ell(v)\bigr)$. Как только будет найден $\bar{d}(x)$, удовлетворяющий этим условиям, можно будет легко вычислить $\operatorname{pp}\bigl(\bar{d}(x)\bigr)$, который и является истинным наибольшим общим делителем $u(x)$ и $v(x)$. Условие (26) удобно тем, что позволяет

избежать неопределенности кратности наибольшего общего делителя обратимым элементам; использовалась, по существу, та же идея для контроля над старшими коэффициентами в программе разложения на множители.

Если p — достаточно большое простое число, которое основано на границах коэффициентов из упр. 20, приложенных к $\ell(\bar{d})u(x)$ либо к $\ell(\bar{d})v(x)$, вычислим единственный полином $\bar{q}(x) \equiv \ell(\bar{d})q(x)$ (по модулю p), все коэффициенты которого находятся в диапазоне $[-\frac{1}{2}p\,..\,\frac{1}{2}p)$. Если $\operatorname{pp}\bigl(\bar{q}(x)\bigr)$ делит как $u(x)$, так и $v(x)$, то он должен быть равен $\gcd\bigl(u(x),v(x)\bigr)$ в соответствии с (24). С другой стороны, если он не делит $u(x)$ и $v(x)$, то $\deg(q) > \deg(d)$. Из алгоритма 4.6.1E следует, что этот случай возможен, только если p делит старший коэффициент одного из ненулевых остатков, вычисленных по этому алгоритму с использованием точной целой арифметики; в противном случае алгоритм Евклида по модулю p работает с той же последовательностью полиномов, что и алгоритм 4.6.1E, за исключением кратности ненулевой константе (по модулю p). Только малое число "неудачных" целых чисел может привести к отсутствию наибольшего общего делителя, и если продолжить попытки, то вскоре можно будет найти "удачное" простое число.

Если граница коэффициентов настолько велика, что простых чисел p с однократной точностью недостаточно, можно вычислять $\bar{d}(x)$ по модулю нескольких простых чисел p, пока она не будет определена с помощью алгоритма на основе китайской теоремы об остатках из раздела 4.3.2. Такой подход, предложенный В. С. Брауном (W. S. Brown) и Дж. Э. Коллинзом, детально описан в *JACM* **18** (1971), 478–504. Кроме того, как рекомендуется в работе J. Moses and D. Y. Y. Yun, *Proc. ACM Conf.* **28** (1973), 159–166, можно использовать для определения $\bar{d}(x)$ по модулю p^e для достаточно больших e метод Хенселя. Построение Хенселя выглядит с точки зрения вычислений превосходящим подход с использованием китайской теоремы об остатках, но непосредственно это верно только при

$$d(x) \perp u(x)/d(x) \qquad \text{или} \qquad d(x) \perp v(x)/d(x), \tag{27}$$

поскольку идея заключается в применении методик из упр. 22 к одному из разложений $\ell(\bar{d})u(x) \equiv \bar{q}(x)u_1(x)$ или $\ell(\bar{d})v(x) \equiv \bar{q}(x)v_1(x)$ (по модулю p). В упр. 34 и 35 показано, что при необходимости с помощью перестановки можно выполнить (27). (Запись

$$u(x) \perp v(x), \tag{28}$$

использованная в (27), означает, что $u(x)$ и $v(x)$ взаимно просты, по аналогии с обозначением, применяемым для взаимно простых чисел.)

Алгоритм наибольшего общего делителя, наброски которого приведены в этом разделе, выполняется значительно быстрее алгоритма из раздела 4.6.1 за исключением случая, когда последовательность полиномиальных остатков очень коротка. Возможно, лучшая обобщенная процедура должна начинаться с вычисления $\gcd\bigl(u(x),v(x)\bigr)$ по модулю небольшого простого числа p, не являющегося одновременно делителем $\ell(u)$ и $\ell(v)$. Если получен результат $q(x) = 1$, мы завершаем работу; если же он имеет высокую степень, используем алгоритм 4.6.1C. В противном случае применяем один из описанных выше методов, сначала вычисляя границу коэффициентов $\bar{d}(x)$ на основе коэффициентов $u(x)$ и $v(x)$ и малой степени полинома $q(x)$. Как и в задаче разложения на множители, если младшие коэффициенты полиномов

проще старших, необходимо применить эту процедуру к "обращенным" полиномам $u(x), v(x)$ и обратить полученный результат.

Полиномы от многих переменных. Подобные методы приводят к алгоритмам, применимым для разложения на множители или поиска наибольших общих делителей полиномов от нескольких переменных с целыми коэффициентами. С полиномом $u(x_1, \ldots, x_t)$ удобно работать по модулю неприводимых полиномов $x_2 - a_2, \ldots, x_t - a_t$, играющих в данном случае роль p из рассматривавшегося ранее материала. Поскольку $v(x) \bmod (x - a) = v(a)$, значение $u(x_1, \ldots, x_t) \bmod \{x_2 - a_2, \ldots, x_t - a_t\}$ является полиномом от одной переменной $u(x_1, a_2, \ldots, a_t)$. Если целые числа $a_2, \ldots, a_t$ выбраны так, что $u(x_1, a_2, \ldots, a_t)$ имеет ту же степень x_1, что и исходный полином $u(x_1, x_2, \ldots, x_t)$, подходящее обобщение построения Хенселя "поднимет" свободные от квадратов разложения этого полинома от одной переменной к разложениям по модулю $\{(x_2 - a_2)^{n_2}, \ldots, (x_t - a_t)^{n_t}\}$, где n_j — степень x_j в u. В то же время можно работать и по модулю подходящего целого простого числа p. Чтобы сохранялась разреженность промежуточных результатов, как можно большее количество a_j должно быть нулевым. Дополнительная информация приводится в упоминавшейся в этом разделе статье, а также в P. S. Wang, *Math. Comp.* **32** (1978), 1215–1231.

Со времен первых пионерских статей, процитированных выше, накоплен значительный вычислительный опыт. Для ознакомления с ним рекомендуется обратиться к работе R. E. Zippel, *Effective Polynomial Computation* (Boston: Kluwer, 1993), в которой приведен обзор последних важных публикаций. Кроме того, в настоящее время возможно разложение полиномов, даваемых неявно вычислительной процедурой "черного ящика", даже если они, будучи записанными явным образом, заполнят всю Вселенную [см. E. Kaltofen and B. M. Trager, *J. Symbolic Comp.* **9** (1990), 301–320; Y. N. Lakshman and B. David Saunders, *SICOMP* **24** (1995), 387–397].

Асимптотически лучшие алгоритмы зачастую оказываются наихудшим решением для всех задач, к которым они применимы.

— Д. Д. КАНТОР (D. G. CANTOR) и Г. ЗАССЕНХАУЗ (H. ZASSENHAUS) (1981)

УПРАЖНЕНИЯ

▸ **1.** [*M24*] Пусть p — простое число и пусть $u(x)$ — случайный полином степени n. Считаем, что все p^n нормированных полиномов равновероятны. Покажите, что, если $n \geq 2$, вероятность того, что $u(x)$ имеет линейный множитель по модулю p, находится между $(1+p^{-1})/2$ и $(2+p^{-2})/3$ включительно. Приведите точный вид этой вероятности при $n \geq p$. Чему равно среднее количество линейных множителей?

▸ **2.** [*M25*] (a) Покажите, что любой нормированный полином $u(x)$ над областью единственного разложения может быть единственным образом представлен в виде

$$u(x) = v(x)^2 w(x),$$

где $w(x)$ свободен от квадратов (не имеет множителей положительной степени вида $d(x)^2$) и оба полинома — $v(x)$ и $w(x)$ — нормированы. (b) (Э. Р. Берлекамп (E. R. Berlekamp).) Сколько нормированных полиномов степени n свободны от квадратов по модулю p, где p — простое число?

3. [*M25*] (*Китайская теорема об остатках для полиномов.*) Пусть $u_1(x), \ldots, u_r(x)$ — полиномы над полем S, где $u_j(x) \perp u_k(x)$ для всех $j \neq k$. Докажите, что для любых данных полиномов $w_1(x), \ldots, w_r(x)$ над S имеется единственный полином $v(x)$ над S, такой, что $\deg(v) < \deg(u_1) + \cdots + \deg(u_r)$ и $v(x) \equiv w_j(x) \bmod u_j(x)$ для $1 \leq j \leq r$. Справедливо ли это утверждение и в случае, когда S представляет собой множество всех целых чисел?

4. [*HM28*] Пусть a_{np} — количество нормированных неприводимых полиномов степени n по модулю простого числа p. Найдите формулу для производящей функции $G_p(z) = \sum_n a_{np} z^n$. [*Указание.* Докажите следующее утверждение, касающееся степенных рядов: $f(z) = \sum_{j \geq 1} g(z^j)/j^t$ тогда и только тогда, когда $g(z) = \sum_{n \geq 1} \mu(n) f(z^n)/n^t$.] Чему равен $\lim_{p \to \infty} a_{np}/p^n$?

5. [*HM30*] Пусть A_{np} — среднее количество неприводимых множителей выбранного случайным образом полинома степени n по модулю простого числа p. Покажите, что $\lim_{p \to \infty} A_{np} = H_n$. Чему равно предельное среднее значение 2^r, где r — число неприводимых множителей?

6. [*M21*] (Ж. Л. Лагранж (J. L. Lagrange), 1771.) Докажите тождество (9). [*Указание.* Разложите $x^p - x$ в поле из p элементов.]

7. [*M22*] Докажите (14).

8. [*HM20*] Как убедиться в том, что векторы, получаемые на выходе алгоритма N, линейно независимы?

9. [*20*] Объясните, каким наипростейшим образом можно построить таблицу обратных величин по модулю 101, если дано, что 2 является первообразным корнем числа 101.

▶ **10.** [*21*] Найдите полное разложение по модулю 2 полинома $u(x)$ из (22) с использованием процедуры Берлекампа.

11. [*22*] Найдите полное разложение полинома $u(x)$ из (22) по модулю 5.

▶ **12.** [*M22*] Используйте алгоритм Берлекампа для определения количества множителей $u(x) = x^4 + 1$ по модулю p для всех простых p. [*Указание.* Рассмотрите случаи, когда $p = 2$, $p = 8k + 1$, $p = 8k + 3$, $p = 8k + 5$, $p = 8k + 7$, отдельно. Чему при этом равна матрица Q? Вам не нужно находить множители; требуется найти только их количество.]

13. [*M25*] Продолжая предыдущее упражнение, найдите точные формулы для множителей полинома $x^4 + 1$ по модулю p для всех нечетных простых чисел p с использованием величин $\sqrt{-1}$, $\sqrt{2}$, $\sqrt{-2}$, если такие квадратные корни существуют по модулю p.

14. [*M25*] (Г. Зассенхауз (H. Zassenhaus).) Пусть $v(x)$ — решение (8) и пусть $w(x) = \prod(x - s)$, где произведение берется по всем $0 \leq s < p$, таким, что $\gcd(u(x), v(x) - s) \neq 1$. Объясните, как вычислить $w(x)$ по данным $u(x)$ и $v(x)$. [*Указание.* Из формулы (14) вытекает, что $w(x)$ является полиномом минимальной степени, таким, что $u(x)$ делит $w(v(x))$.]

▶ **15.** [*M27*] (*Квадратные корни по модулю простого числа.*) Разработайте алгоритм для вычисления квадратного корня целого числа u по модулю простого числа p, т. е. найдите число v, такое, что $v^2 \equiv u \bmod p$, если оно существует. Ваш алгоритм должен быть эффективен даже при очень больших целых числах p. (Для $p \neq 2$ решение этой задачи сводится к решению квадратного уравнения по модулю p с использованием обычной квадратичной формулы.) *Указание.* Рассмотрите, что произойдет после применения методов разложения из этого раздела к полиному $x^2 - u$.

16. [*M30*] (*Конечные поля.*) Назначение данного упражнения — доказать основные свойства полей, введенных Э. Галуа (É. Galois) в 1830 году.

а) Дано, что $f(x)$ — неприводимый по модулю простого числа p полином степени n. Докажите, что p^n полиномов степени, меньшей n, образуют поле с арифметикой по

модулю $f(x)$ и p. [*Указание.* Существование неприводимых полиномов любой степени доказано в упр. 4, поэтому поля с p^n элементами существуют для всех простых чисел p и всех $n \geq 1$.]

b) Покажите, что любое поле с p^n элементами имеет элемент "примитивный корень" ξ, такой, что элементами поля являются $\{0, 1, \xi, \xi^2, \ldots, \xi^{p^n-2}\}$. [*Указание.* В упр. 3.2.1.2–16 содержится доказательство для частного случая $n = 1$.]

c) Если $f(x)$ — неприводимый полином по модулю p степени n, докажите, что $x^{p^m} - x$ делится на $f(x)$ тогда и только тогда, когда m кратно n. (Отсюда следует, что можно быстро проверить неприводимость. Данный полином n-й степени $f(x)$ неприводим по модулю p тогда и только тогда, когда $x^{p^n} - x$ делится на $f(x)$ и $x^{p^{n/q}} - x \perp f(x)$ для всех простых q, которые делят n.)

17. [*M23*] Пусть F — поле с 13^2 элементами. Сколько элементов F имеют порядок f для каждого целого $1 \leq f < 13^2$? (*Порядком* элемента a является наименьшее положительное целое число m, такое, что $a^m = 1$.)

▸ **18.** [*M25*] Пусть $u(x) = u_n x^n + \cdots + u_0, u_n \neq 0$ является примитивным полиномом с целыми коэффициентами и пусть $v(x)$ — нормированный полином, определяемый как

$$v(x) = u_n^{n-1} \cdot u(x/u_n) = x^n + u_{n-1}x^{n-1} + u_{n-2}u_n x^{n-2} + \cdots + u_0 u_n^{n-1}.$$

(a) Дано, что $v(x)$ имеет полное разложение $p_1(x) \ldots p_r(x)$ над кольцом целых чисел, где каждый $p_j(x)$ нормирован. Каково полное разложение полинома $u(x)$ над кольцом целых чисел? (b) Если $w(x) = x^m + w_{m-1}x^{m-1} + \cdots + w_0$ является множителем $v(x)$, докажите, что w_k является множителем u_n^{m-1-k} при $0 \leq k < m$.

19. [*M20*] (*Критерий Эйзенштейна.*) Возможно, самый известный класс неприводимых полиномов над кольцом целых чисел был введен Т. Шёнеманном (T. Schönemann) в *Crelle* **32** (1846), 100, а популяризован Г. Эйзенштейном (G. Eisenstein) в *Crelle* **39** (1850), 166–169. Пусть p является простым числом и пусть полином $u(x) = u_n x^n + \cdots + u_0$ имеет следующие свойства: (i) u_n не делится на p; (ii) u_{n-1}, ..., u_0 делятся на p; (iii) u_0 не делится на p^2. Покажите, что $u(x)$ неприводим над кольцом целых чисел.

20. [*HM33*] Если $u(x) = u_n x^n + \cdots + u_0$ является некоторым полиномом над полем комплексных чисел, обозначим $\|u\| = (|u_n|^2 + \cdots + |u_0|^2)^{1/2}$.

a) Пусть $u(x) = (x - \alpha)w(x)$ и $v(x) = (\bar{\alpha}x - 1)w(x)$, где α — произвольное комплексное число, а $\bar{\alpha}$ — сопряженное ему. Докажите, что $\|u\| = \|v\|$.

b) Пусть $u_n(x - \alpha_1) \ldots (x - \alpha_n)$ представляет собой полное разложение $u(x)$ над полем комплексных чисел. Введем обозначение $M(u) = |u_n| \prod_{j=1}^n \max(1, |\alpha_j|)$. Докажите, что $M(u) \leq \|u\|$.

c) Покажите, что $|u_j| \leq \binom{n-1}{j} M(u) + \binom{n-1}{j-1}|u_n|$ для $0 \leq j \leq n$.

d) Объедините эти результаты для доказательства того, что если $u(x) = v(x)w(x)$ и $v(x) = v_m x^m + \cdots + v_0$, где u, v, w имеют целые коэффициенты, то коэффициенты v ограничены следующим образом:

$$|v_j| \leq \binom{m-1}{j}\|u\| + \binom{m-1}{j-1}|u_n|.$$

21. [*HM32*] Продолжая упр. 20, выведем полезные границы коэффициентов полиномов *от многих переменных* над кольцом целых чисел. Для удобства будем использовать полужирные символы, чтобы обозначить последовательности из t целых чисел. Таким образом, вместо записи

$$u(x_1, \ldots, x_t) = \sum_{j_1, \ldots, j_t} u_{j_1 \ldots j_t} x_1^{j_1} \ldots x_t^{j_t}$$

можно записать просто $u(\mathbf{x}) = \sum_{\mathbf{j}} u_{\mathbf{j}} \mathbf{x}^{\mathbf{j}}$. Обратите внимание на обозначение $\mathbf{x}^{\mathbf{j}}$; также обозначаем $\mathbf{j}! = j_1! \ldots j_t!$ и $\Sigma\mathbf{j} = j_1 + \cdots + j_t$.

a) Докажите тождество

$$\sum_{\mathbf{j},\mathbf{k}} \frac{1}{\mathbf{j}!\,\mathbf{k}!} \sum_{\mathbf{p},\mathbf{q}\geq 0} [\mathbf{p}-\mathbf{j}=\mathbf{q}-\mathbf{k}]\, a_{\mathbf{p}} b_{\mathbf{q}} \frac{\mathbf{p}!\,\mathbf{q}!}{(\mathbf{p}-\mathbf{j})!} \sum_{\mathbf{r},\mathbf{s}\geq 0} [\mathbf{r}-\mathbf{j}=\mathbf{s}-\mathbf{k}]\, c_{\mathbf{r}} d_{\mathbf{s}} \frac{\mathbf{r}!\,\mathbf{s}!}{(\mathbf{r}-\mathbf{j})!}$$
$$= \sum_{\mathbf{i}\geq 0} \mathbf{i}! \sum_{\mathbf{p},\mathbf{s}\geq 0} [\mathbf{p}+\mathbf{s}=\mathbf{i}]\, a_{\mathbf{p}} d_{\mathbf{s}} \sum_{\mathbf{q},\mathbf{r}\geq 0} [\mathbf{q}+\mathbf{r}=\mathbf{i}]\, b_{\mathbf{q}} c_{\mathbf{r}}\,.$$

b) Полином $u(\mathbf{x}) = \sum_{\mathbf{j}} u_{\mathbf{j}} \mathbf{x}^{\mathbf{j}}$ называется *однородным* полиномом степени n, если каждый член имеет общую степень n. Таким образом, $\Sigma\mathbf{j} = n$ при $u_{\mathbf{j}} \neq 0$. Рассмотрим взвешенную сумму коэффициентов $B(u) = \sum_{\mathbf{j}} \mathbf{j}!\,|u_{\mathbf{j}}|^2$. Используя п. (a), покажите, что $B(u) \geq B(v)B(w)$, если $u(\mathbf{x}) = v(\mathbf{x})w(\mathbf{x})$ однороден.

c) *Норма Бомбьери* $[u]$ полинома $u(\mathbf{x})$ определяется как $\sqrt{B(u)/n!}$, если u — однородный полином степени n. Она определена также для неоднородных полиномов посредством добавления новой переменной x_{t+1} и умножения каждого члена на степень x_{t+1}, такую, что u становится однородным без увеличения его максимальной степени. Например, пусть $u(x) = 4x^3 + x - 2$; соответствующий однородный полином — $4x^3 + xy^2 - 2y^3$ и $[u]^2 = (3!\,0!\,4^2 + 1!\,2!\,1^2 + 0!\,3!\,2^2)/3! = 16 + \frac{1}{3} + 4$. Если $u(x,y,z) = 3xy^3 - z^2$, аналогично получаем $[u]^2 = (1!\,3!\,0!\,0!\,3^2 + 0!\,0!\,2!\,2!\,1^2)/4! = \frac{9}{4} + \frac{1}{6}$. Что говорит п. (b) о связи между $[u]$, $[v]$ и $[w]$ при $u(\mathbf{x}) = v(\mathbf{x})w(\mathbf{x})$?

d) Докажите, что если $u(x)$ — приводимый полином степени n от одной переменной, то он имеет множитель, коэффициенты которого не превышают $n!^{1/4}[u]^{1/2}/(n/4)!$ по абсолютному значению. Чему равно соответствующее значение для однородных полиномов от t переменных?

e) Вычислите $[u]$ явно и асимптотически при $u(x) = (x^2 - 1)^n$.

f) Докажите, что $[u][v] \geq [uv]$.

g) Покажите, что $2^{-n/2}M(u) \leq [u] \leq 2^{n/2}M(u)$, если $u(x)$ — полином степени n и $M(u)$ — величина, определенная в упр. 20. (Вследствие этого грань в п. (d) примерно равна квадратному корню грани, полученной в упр. 20.)

▸ **22.** [*M24*] (*Лемма Хенселя.*) Пусть $u(x)$, $v_e(x)$, $w_e(x)$, $a(x)$, $b(x)$ представляют собой полиномы с целыми коэффициентами, удовлетворяющими соотношениям

$$u(x) \equiv v_e(x)w_e(x) \mod p^e, \qquad a(x)v_e(x) + b(x)w_e(x) \equiv 1 \mod p,$$

где p — простое число, $e \geq 1$, $v_e(x)$ — нормированный полином, $\deg(a) < \deg(w_e)$, $\deg(b) < \deg(v_e)$ и $\deg(u) = \deg(v_e) + \deg(w_e)$. Покажите, как вычислить полиномы $v_{e+1}(x) \equiv v_e(x)$ и $w_{e+1}(x) \equiv w_e(x)$ (по модулю p^e), удовлетворяющие тем же условиям с e, увеличенным на 1. Кроме того, докажите, что $v_{e+1}(x)$ и $w_{e+1}(x)$ единственны по модулю p^{e+1}.

Используйте свой метод для $p = 2$ для доказательства того, что (22) неприводим над кольцом целых чисел, начиная с его разложения по модулю 2, найденного в упр. 10. (Заметьте, что расширенный алгоритм Евклида из упр. 4.6.1–3 даст процесс, начинающийся с $e = 1$.)

23. [*HM23*] Пусть $u(x)$ является полиномом с целыми коэффициентами, свободным от квадратов. Докажите, что имеется только конечное число простых чисел p, таких, что этот полином $u(x)$ не является свободным от квадратов по модулю p.

24. [*M20*] В тексте раздела говорится о разложении над кольцом целых чисел, а не над полем рациональных чисел. Объясните, как найти полное разложение полинома с рациональными коэффициентами над полем рациональных чисел.

25. [*M25*] Каково полное разложение полинома $x^5+x^4+x^2+x+2$ над полем рациональных чисел?

26. [*20*] Пусть $d_1, \ldots, d_r$ — степени неприводимых множителей полинома $u(x)$ по модулю p с правильной кратностью, так что $d_1 + \cdots + d_r = n = \deg(u)$. Объясните, как найти множество $\{\deg(v) \mid u(x) \equiv v(x)w(x) \bmod p$ для некоторых $v(x), w(x)\}$, выполнив $O(r)$ операций над битовой строкой длины n.

27. [*HM30*] Докажите, что случайный примитивный полином над кольцом целых чисел в некотором определенном смысле "почти всегда" неприводим.

28. [*M25*] Процедура разложения с различными степенями "удачна", если существует не более одного неприводимого полинома каждой степени d. Тогда $g_d(x)$ никогда не должно разбиваться на множители. Чему равна вероятность такой удачной ситуации при разложении случайного полинома степени n по модулю p для фиксированного n при $p \to \infty$?

29. [*M22*] Пусть $g(x)$ — произведение двух или более различных неприводимых полиномов степени d по модулю нечетного простого числа p. Докажите, что

$$\gcd(g(x),\, t(x)^{(p^d-1)/2} - 1)$$

будет собственным множителем $g(x)$ с вероятностью $\geq 1/2 - 1/(2p^d)$ для любого фиксированного $g(x)$, когда $t(x)$ выбрано случайным образом из p^{2d} полиномов степени $< 2d$ по модулю p.

30. [*M25*] Докажите, что если $q(x)$ является неприводимым полиномом степени d по модулю p и если $t(x)$ является произвольным полиномом, то значение

$$(t(x) + t(x)^p + t(x)^{p^2} + \cdots + t(x)^{p^{d-1}}) \bmod q(x)$$

представляет собой целое число (т. е. полином степени ≤ 0). Используйте этот факт, чтобы создать рандомизированный алгоритм для разложения $g_d(x)$ в виде произведения неприводимых полиномов степени $-d$ аналогично (21) для случая, когда $p = 2$.

31. [*HM30*] Пусть p — нечетное простое число и пусть $d \geq 1$. Покажите, что существует число $n(p,d)$, обладающее следующими двумя свойствами. (i) Для всех целых t в точности $n(p,d)$ неприводимых полиномов $q(x)$ степени d по модулю p удовлетворяют соотношению $(x+t)^{(p^d-1)/2} \bmod q(x) = 1$. (ii) Для всех целых чисел $0 \leq t_1 < t_2 < p$ в точности $n(p,d)$ неприводимых полиномов $q(x)$ степени d по модулю p удовлетворяют соотношению $(x+t_1)^{(p^d-1)/2} \bmod q(x) = (x+t_2)^{(p^d-1)/2} \bmod q(x)$.

▸ **32.** [*M30*] (*Циклотомические полиномы.*) Пусть $\Psi_n(x) = \prod_{1\leq k\leq n,\, k\perp n}(x - \omega^k)$, где $\omega = e^{2\pi i/n}$. Тогда корни $\Psi_n(x)$ — это n-е комплексные корни единицы, не являющиеся m-ми корнями при $m < n$.

a) Докажите, что $\Psi_n(x)$ — полином с целыми коэффициентами и что

$$x^n - 1 = \prod_{d\backslash n} \Psi_d(x); \qquad \Psi_n(x) = \prod_{d\backslash n} (x^d - 1)^{\mu(n/d)}.$$

(См. упр. 4.5.2–10(b) и 4.5.3–28(c).)

b) Докажите, что полином $\Psi_n(x)$ неприводим над кольцом целых чисел. Следовательно, предыдущая формула является полным разложением $x^n - 1$ над кольцом целых чисел. [*Указание.* Если $f(x)$ является неприводимым множителем $\Psi_n(x)$ над кольцом целых чисел и если ζ — комплексное число, для которого $f(\zeta) = 0$, докажите, что $f(\zeta^p) = 0$ для всех простых чисел p, не делящих n. Вам может помочь тот факт, что $x^n - 1$ свободен от квадратов по модулю p для всех таких простых чисел.]

c) Обсудите вычисление $\Psi_n(x)$ и протабулируйте значения этой функции для $n \leq 15$.

33. [*M18*] Истинно ли следующее утверждение: если $u(x) \neq 0$ и полное разложение $u(x)$ по модулю p представляет собой $p_1(x)^{e_1} \dots p_r(x)^{e_r}$, то $u(x)/\gcd(u(x), u'(x)) = p_1(x) \dots p_r(x)$?

▸ **34.** [*M25*] (*Разложение, свободное от квадратов.*) Ясно, что любой примитивный полином из области единственного разложения может быть выражен в виде

$$u(x) = u_1(x)u_2(x)^2u_3(x)^3\dots,$$

где полиномы $u_i(x)$ свободны от квадратов и взаимно просты. Такое представление, в котором $u_j(x)$ является произведением всех неприводимых полиномов, делящих $u(x)$ ровно j раз, единственно с точностью до обратимого множителя; это полезный способ представления полиномов, участвующих в умножении, делении и поиске наибольшего общего делителя.

Пусть $\mathrm{GCD}(u(x), v(x))$ представляет собой процедуру, которая вычисляет три значения:

$$\mathrm{GCD}(u(x), v(x)) = (d(x), u(x)/d(x), v(x)/d(x)), \quad \text{где } d(x) = \gcd(u(x), v(x)).$$

Метод, описанный в тексте раздела, следующем за формулой (25), всегда заканчивается проверочным делением $u(x)/d(x)$ и $v(x)/d(x)$, которое позволяет убедиться, что не было использовано "неудачное" простое число, так что значения $u(x)/d(x)$ и $v(x)/d(x)$ представляют собой побочные продукты вычисления наибольшего общего делителя, т. е. можно вычислить $\mathrm{GCD}(u(x), v(x))$ с помощью указанного метода так же быстро, как и $\gcd(u(x), v(x))$.

Придумайте процедуру для получения свободного от квадратов представления $(u_1(x), u_2(x), \dots)$ данного примитивного полинома $u(x)$ над кольцом целых чисел. Ваш алгоритм должен выполнять в точности e вычислений GCD, где e — наибольший индекс, для которого $u_e(x) \neq 1$. К тому же каждое вычисление GCD должно удовлетворять условиям (27), чтобы можно было использовать построение Хенселя.

35. [*M22*] (Д. Ю. Е. Юнь (D. Y. Y. Yun).) Разработайте алгоритм, вычисляющий свободное от квадратов представление $(w_1(x), w_2(x), \dots)$ полинома $w(x) = \gcd(u(x), v(x))$ над кольцом целых чисел по свободным от квадратов представлениям $(u_1(x), u_2(x), \dots)$ и $(v_1(x), v_2(x), \dots)$ полиномов $u(x)$ и $v(x)$.

36. [*M27*] Расширьте процедуру упр. 34 так, чтобы получалось свободное от квадратов представление $(u_1(x), u_2(x), \dots)$ полинома $u(x)$ с арифметикой коэффициентов, выполняемой по модулю p.

37. [*HM24*] (Джордж Э. Коллинз (George E. Collins).) Пусть $d_1, \dots, d_r$ — положительные целые числа, сумма которых равна n, и пусть p — простое число. Чему равна вероятность того, что неприводимые множители случайного полинома степени n с целыми коэффициентами имеют при полном разложении по модулю p степени $d_1, \dots, d_r$? Покажите, что асимптотически данная вероятность аналогична вероятности того, что случайная перестановка n элементов имеет циклы длины $d_1, \dots, d_r$.

38. [*HM27*] (*Критерий Перрона.*) Пусть $u(x) = x^n + u_{n-1}x^{n-1} + \cdots + u_0$ — полином с целыми коэффициентами, такой, что $u_0 \neq 0$, и либо $|u_{n-1}| > 1 + |u_{n-2}| + \cdots + |u_0|$, либо $(u_{n-1} = 0$ и $u_{n-2} > 1 + |u_{n-3}| + \cdots + |u_0|)$. Покажите, что полином $u(x)$ неприводим над кольцом целых чисел. [*Указание.* Докажите, что почти все корни u меньше 1 по абсолютному значению.]

39. [*HM42*] (Дэвид Д. Кантор (David G. Cantor).) Покажите, что если полином $u(x)$ неприводим над кольцом целых чисел, то он имеет "укороченное" доказательство неприводимости в том смысле, что количество битов в доказательстве — это по крайней мере полином от $\deg(u)$ и длин коэффициентов. (Здесь требуется граница *длины* доказательства, как в упр. 4.5.4–17, а не граница времени, необходимого для поиска такого доказательства.) *Указание.* Если $v(x)$ неприводим и t является любым полиномом над кольцом целых чисел,

то все множители $v(t(x))$ имеют степень $\geq \deg(v)$. Критерий Перрона дает большой запас неприводимых полиномов $v(x)$.

▸ **40.** [*M20*] (П. Ш. Ванг (P. S. Wang).) Если u_n — старший коэффициент полинома $u(x)$ и B — граница коэффициентов некоторого множителя u, то для алгоритма разложения, приведенного в тексте раздела, требуется найти разложение по модулю p^e, где $p^e > 2|u_n|B$. Но $|u_n|$ может быть больше, чем B, когда B выбирается по методу из упр. 21. Покажите, что если полином $u(x)$ приводим, то существует способ восстановления одного из его истинных множителей по разложению по модулю p^e, когда $p^e \geq 2B^2$, с помощью алгоритма из упр. 4.5.3–51.

41. [*M47*] (Бизами (Beauzamy), Тревисан (Trevisan) и Ванг (Wang).) Докажите или опровергните следующее: существует константа c, такая, что если $f(x)$ — некоторый целый полином с коэффициентами, по абсолютному значению не превосходящими B, то один из его неприводимых множителей имеет коэффициенты, ограниченные величиной cB.

4.6.3. Вычисление степеней

В этом разделе рассматривается интересная задача — эффективное вычисление x^n по данным x и n, где n — положительное целое число. Предположим, например, что необходимо вычислить x^{16}. Можно просто начать с x и 15 раз умножить его на x. Но тот же ответ можно получить всего за четыре умножения, если несколько раз возвести в квадрат получающийся результат, последовательно вычисляя x^2, x^4, x^8, x^{16}.

Эта же идея, в целом, применима к любому значению n следующим образом. Запишем n в виде числа в двоичной системе счисления (убирая нули слева). Затем заменим каждую "1" парой символов SX, каждый "0" — символом S и вычеркнем крайнюю слева пару символов "SX". Результат представляет собой правило вычисления x^n, в котором "S" трактуется как операция *возведения в квадрат*, а "X" — как операция *умножения на* x. Например, $n = 23$ имеет двоичное представление 10111. Таким образом, мы формируем последовательность SX S SX SX SX, из которой удаляем начальную пару SX для получения окончательного правила SSXSXSX. Это правило гласит, что необходимо "возвести в квадрат, возвести в квадрат, умножить на x, возвести в квадрат, умножить на x, возвести в квадрат и умножить на x", т. е. последовательно вычислить значения x^2, x^4, x^5, x^{10}, x^{11}, x^{22}, x^{23}.

Этот бинарный метод можно легко обосновать, проанализировав последовательность степеней при вычислении: рассматривая каждое "S" как операцию умножения на 2, а "X" — как операцию прибавления 1 и начав с 1, а не с x, мы придем к вычислению n в соответствии со свойствами двоичной системы счисления. Этот метод очень древний; он появился до 200 г. до н. э. в классической индусской *Чанда-сутре* (*Chandaḥ-sûtra*) Пингалы (Piṅgala) [см. B. Datta and A. N. Singh, *History of Hindu Mathematics* **2** (Lahore: Motilal Banarsi Das, 1935), 76]. Похоже, что в течение следующего тысячелетия этот метод не упоминался нигде за пределами Индии. В 952 г. н. э. аль-Уклидиси (al-Uqlīdisī) из Дамаска четко пояснил, как эффективно вычислить 2^n для произвольного n. См. *The Arithmetic of al-Uqlīdisī* by A. S. Saidan (Dordrecht: D. Reidel, 1975), 341–342, где общие идеи проиллюстрированы на примере $n = 51$. См. также работу аль-Бируни (al-Bīrūnī) *Chronology of Ancient Nations*, переведенную и отредактированную Э. Сашо (E. Sachau) (London, 1879), 132–136. Эта арабская работа 11 века оказала сильное влияние на развитие математики.

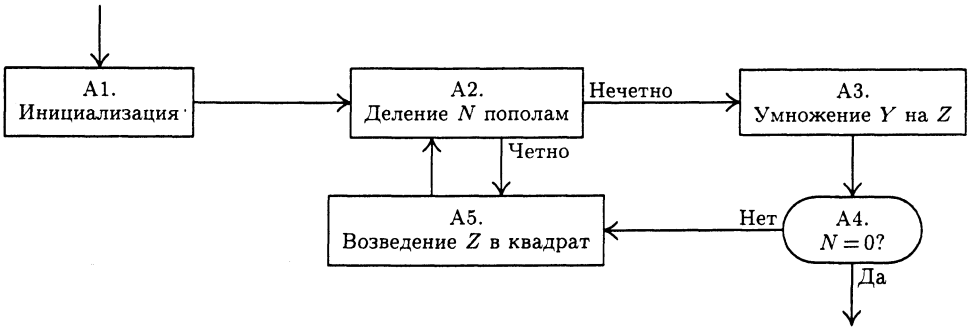


Рис. 13. Вычисление x^n, основанное на сканировании двоичной записи n справа налево.

Бинарный метод "S и X" для получения x^n не требует дополнительной памяти, за исключением памяти для хранения x и текущего промежуточного результата, а потому он удобен для аппаратной реализации на бинарном компьютере. Метод легко программируем, однако требует сканирования двоичного представления числа n слева направо, в то время как компьютерные программы обычно делают это в обратном направлении, в связи с тем, что доступные операции деления на 2 и взятия остатка по модулю 2 выводят двоичное представление числа справа налево. Поэтому зачастую более удобным оказывается следующий алгоритм, основанный на сканировании числа справа налево.

Алгоритм А (*Бинарный метод возведения в степень справа налево*). Этот алгоритм (рис. 13) вычисляет значение x^n, где n — положительное целое число (здесь x принадлежит любой алгебраической системе, в которой определено ассоциативное умножение с единичным элементом 1).

А1. [Инициализация.] Установить $N \leftarrow n$, $Y \leftarrow 1$, $Z \leftarrow x$.

А2. [Деление N пополам.] (В этот момент $x^n = Y Z^N$.) Установить $N \leftarrow \lfloor N/2 \rfloor$ и одновременно определить, было ли N четно. Если N было четно, перейти к шагу А5.

А3. [Умножение Y на Z.] Установить $Y \leftarrow Z$, умноженное на Y.

А4. [$N = 0$?] Если $N = 0$, то выполнение алгоритма прекращается с Y в качестве результата.

А5. [Возведение Z в квадрат.] Установить $Z \leftarrow Z$, умноженное на Z, и вернуться к шагу А2. ▮

В качестве примера применения алгоритма А рассмотрим пошаговое вычисление x^{23}.

	N	Y	Z
После шага А1	23	1	x
После шага А5	11	x	x^2
После шага А5	5	x^3	x^4
После шага А5	2	x^7	x^8
После шага А5	1	x^7	x^{16}
После шага А4	0	x^{23}	x^{16}

Соответствующая алгоритму А MIX-программа приведена в упр. 2.

Великий вычислитель аль-Каши (al-Kāshī) сформулировал алгоритм А в 1427 году [*Историко-математические исследования* **7** (1954), 256–257]. Этот алгоритм тесно связан с процедурой умножения, в действительности использовавшейся египетскими математиками за 2000 лет до н. э. Если заменить шаг А3 на "$Y \leftarrow Y + Z$", а шаг А5 на "$Z \leftarrow Z + Z$" и если установить Y равным нулю, а не единице на шаге А1, то в результате работы алгоритма получится $Y = nx$. [См. A. B. Chace, *The Rhind Mathematical Papyrus* (1927); W. W. Struve, *Quellen und Studien zur Geschichte der Mathematik* **A1** (1930).] Это практичный метод умножения чисел вручную, поскольку используются только простые операции удвоения, деления пополам и сложения. Часто его называют русским крестьянским методом умножения, так как Запад стал свидетелем его популярности в России в 19 веке.

Количество умножений, требуемых алгоритмом А, составляет

$$\lfloor \lg n \rfloor + \nu(n),$$

где $\nu(n)$ равно количеству единиц в двоичном представлении числа n. Это на одно умножение больше, чем при бинарном методе "слева направо", о котором упоминалось в самом начале данного раздела, так как при первом выполнении шага А3 просто производится умножение на единицу.

Из-за дополнительного времени на накладные расходы этого алгоритма метод не так хорош для малых значений n, скажем, для $n \leq 10$, если, конечно, время, необходимое для умножения, сравнительно невелико. Если значение n известно заранее, то предпочтителен двоичный метод "слева направо". Иногда, например при вычислении $x^n \bmod u(x)$, которое обсуждалось в разделе 4.6.2, гораздо проще умножать на x, чем выполнять обобщенное умножение или возведение в квадрат, так что двоичные методы для возведения в степень в таких случаях, в первую очередь, подходят для очень больших значений n. Для вычисления точного значения x^n с многократной точностью при целом x, большем, чем размер компьютерного слова, бинарные методы не слишком хороши до тех пор, пока n не станет столь огромным, что высокоскоростное умножение (см. раздел 4.3.3) будет неприменимо. Однако такие приложения очень редки. Точно так же двоичный метод мало пригоден для возведения в степень полиномов [см. в R. J. Fateman, *SICOMP* **3** (1974), 196–213, обзор публикаций по проблеме возведения полиномов в степень].

Главная идея этого примечания состоит в том, что двоичные методы хороши, но не являются панацеей. Они применимы в основном тогда, когда время умножения $x^j \cdot x^k$, по сути, не зависит от j и k (например, когда выполняется умножение с плавающей точкой или умножение по модулю m); в таких случаях порядок времени работы уменьшается с n до $\log n$.

Уменьшение количества операций умножения. Несколько авторов без доказательства опубликовали утверждение о том, что двоичный метод дает *минимально* возможное число умножений. Однако это утверждение неверно, и простейший контрпример — $n = 15$, при котором двоичный метод требует шести умножений, в то время как $y = x^3$ можно вычислить с помощью двух умножений и $x^{15} = y^5$ — с помощью еще трех, т. е. получить требуемый результат, выполнив всего пять умножений. Обсудим теперь несколько других процедур для вычисления x^n, в предположении, что n известно заранее. Такие процедуры особенно интересны, например, при генерировании машинного кода оптимизирующим компилятором.

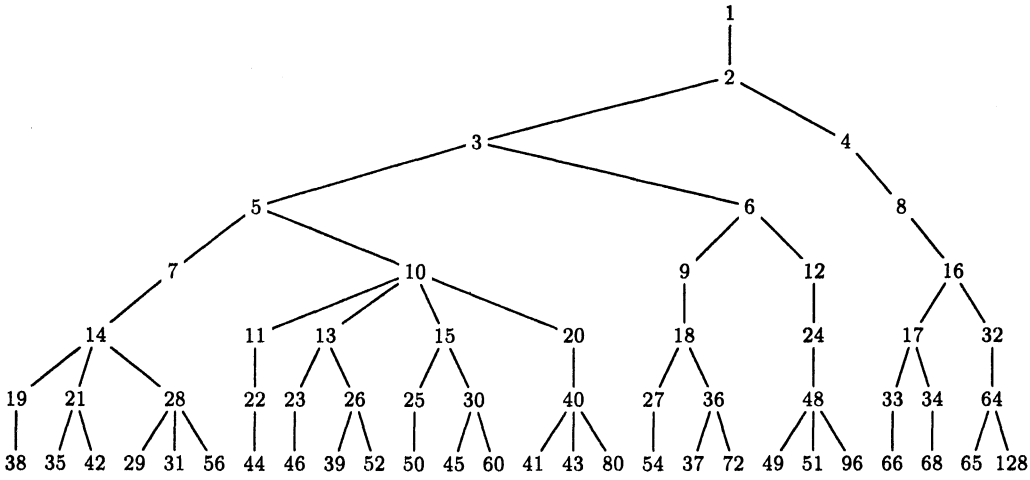


Рис. 14. "Дерево степеней."

Метод множителя основан на разложении n. Если $n = pq$, где p представляет собой наименьший простой множитель n и $q > 1$, x^n можно найти, вычислив x^p и возведя эту величину в степень q. Если n — простое число, можно вычислить x^{n-1} и умножить его на x. И конечно, при $n = 1$ получаем x^n безо всяких вычислений. Неоднократно применяя эти правила, можно получить процедуру вычисления x^n при любом данном значении n. Например, чтобы найти x^{55}, сначала нужно вычислить $y = x^5 = x^4x = (x^2)^2x$, а затем построить $y^{11} = y^{10}y = (y^2)^5y$. Весь процесс предусматривает выполнение восьми умножений, в то время как двоичному методу потребовалось бы девять умножений. Метод множителя в среднем превосходит двоичный метод, но в некоторых случаях (наименьший из них — $n = 33$) двоичный метод лучше.

Двоичный метод может быть обобщен в *m-арный метод* следующим образом. Пусть $n = d_0m^t + d_1m^{t-1} + \cdots + d_t$, где $0 \le d_j < m$ для $0 \le j \le t$. Вычисление начинается с построения x, x^2, x^3, ..., x^{m-1}. (На самом деле нужны только те степени x^{d_j}, у которых d_j появляются в представлении числа n, и это часто экономит наши усилия.) Затем возведем x^{d_0} в степень m и умножим на x^{d_1}. Таким образом мы вычислили $y_1 = x^{d_0m+d_1}$. Затем возведем y_1 в степень m, умножим на x^{d_2} и получим $y_2 = x^{d_0m^2+d_1m+d_2}$. Этот процесс продолжается до тех пор, пока не будет вычислено значение $y_t = x^n$. Когда $d_j = 0$, естественно, умножение на x^{d_j} не является необходимым. Заметим, что данный метод при $m = 2$ сводится к обсуждавшемуся ранее двоичному методу "слева направо", однако менее ясный m-арный метод "справа налево" требует большего объема памяти и несколько большего количества шагов (см. упр. 9). Если m представляет собой малое простое число, m-арный метод будет особенно эффективен для вычисления степеней одного полинома по модулю другого, когда коэффициенты рассматриваются по модулю m в соответствии с 4.6.2–(5).

Систематический метод, дающий минимальное количество умножений для всех относительно малых значений n (в частности, для большинства встречающихся в реальных приложениях значений n), проиллюстрирован на рис. 14. Для вычисле-

ния x^n найдите n в представленном на рисунке дереве; путь от корня дерева к n дает последовательность показателей степеней, встречающихся при эффективном вычислении x^n. Правило, по которому строится это "дерево степеней", приводится в упр. 5. Вычислительные тесты показали, что дерево степеней дает оптимальные результаты для всех указанных на рисунке n, однако для достаточно больших значений n метод дерева степеней не всегда оптимален (наименьшие примеры неоптимальности — $n = 77$, 154, 233). Первое значение, для которого дерево степеней превосходит и двоичный метод, и метод множителя, — $n = 23$. Первое значение, для которого метод множителя превосходит метод дерева степеней, — $n = 19879 = 103 \cdot 193$. Такие случаи весьма редки. (Для $n \leq 100\,000$ метод дерева степеней лучше метода множителя 88 803 раз, столь же хорош 11 191 раз и проигрывает ему только 6 раз.)

Аддитивные цепочки. Наиболее экономичный путь вычисления x^n с помощью операций умножения представляет собой математическую задачу с интересной историей. Сейчас мы приступим к ее изучению не только потому, что она классическая и интересна сама по себе, но и потому, что это превосходный пример теоретических вопросов, возникающих при изучении оптимальных методов вычислений.

Хотя мы имеем дело с умножением степеней x, проблему можно легко свести к сложению, поскольку при умножении показатели степеней складываются. Это приводит к следующей абстрактной формулировке: *аддитивной цепочкой для n* является последовательность целых чисел

$$1 = a_0, \quad a_1, \quad a_2, \quad \ldots, \quad a_r = n, \tag{1}$$

обладающих тем свойством, что

$$a_i = a_j + a_k \qquad \text{для некоторых } k \leq j < i, \tag{2}$$

для всех $i = 1, 2, \ldots, r$. Это определение можно трактовать следующим образом. Рассмотрим простой компьютер с аккумулятором, который способен выполнять три операции: `LDA`, `STA` и `ADD`. Машина начинает работу с 1 в аккумуляторе и затем вычисляет число n, складывая полученные ранее результаты. Заметьте, что a_1 должно быть равно 2, а a_2 равно 2, 3 либо 4.

Наименьшая длина r, для которой существует аддитивная цепочка для n, обозначается как $l(n)$. Наша цель в оставшейся части этого раздела состоит в изучении функции $l(n)$. Значения $l(n)$ для малых n приведены в дереве на рис. 15, которое показывает, как вычислить x^n с наименьшим возможным количеством умножений для всех $n \leq 100$.

Проблема определения $l(n)$ была, по-видимому, впервые поставлена в 1894 году Х. Деллаком (H. Dellac) и частично решена Э. де Жонкиэрес (E. de Jonquières) упомянутым методом множителя [см. *L'Intermédiaire des Mathématiciens* **1** (1894), 20, 162–164]. В своем решении де Жонкиэрес перечислил значения $l(p)$ для всех простых чисел $p < 200$, но в его таблице значения для $p = 107$, 149, 163, 179 были слишком велики.

Методом множителя можно получить неравенство

$$l(mn) \leq l(m) + l(n), \tag{3}$$

поскольку можно взять цепочки $1, a_1, \ldots, a_r = m$ и $1, b_1, \ldots, b_s = n$ и построить цепочку $1, a_1, \ldots, a_r, a_r b_1, \ldots, a_r b_s = mn$.

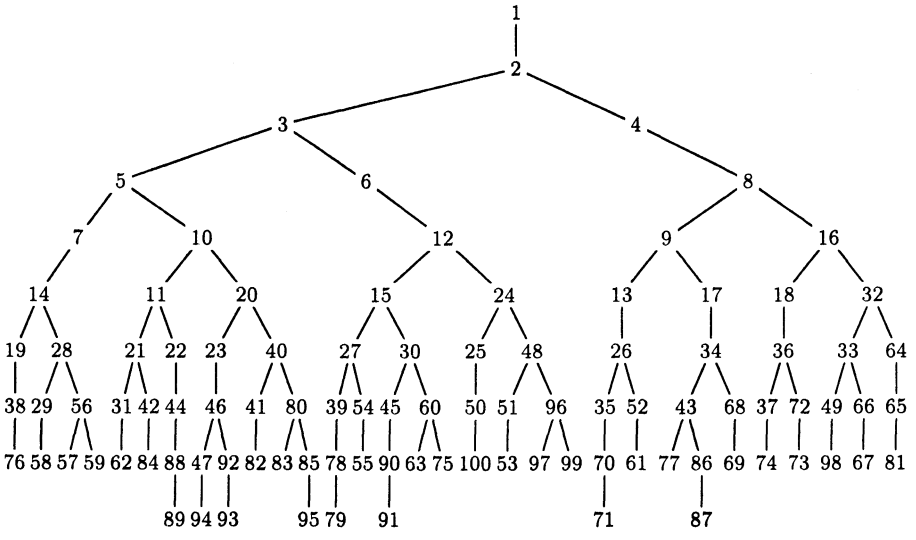


Рис. 15. Дерево, минимизирующее число умножений для $n \le 100$.

Можно также сформулировать m-арный метод в терминах аддитивных цепочек. Рассмотрим случай, когда $m = 2^k$, и запишем $n = d_0 m^t + d_1 m^{t-1} + \cdots + d_t$ в m-ичной системе счисления. Соответствующая аддитивная цепочка принимает вид

$$\begin{aligned} &1, 2, 3, \ldots, m-2, m-1, \\ &\quad 2d_0, 4d_0, \ldots, md_0, md_0 + d_1, \\ &\qquad 2(md_0+d_1), 4(md_0+d_1), \ldots, m(md_0+d_1), m^2 d_0 + md_1 + d_2, \\ &\qquad\quad \ldots, \qquad m^t d_0 + m^{t-1} d_1 + \cdots + d_t. \end{aligned} \tag{4}$$

Ее длина равна $m - 2 + (k+1)t$, и зачастую ее можно уменьшить, удалив некоторые элементы из первой строки (те, которые не встречаются среди коэффициентов d_j, а также элементов из $2d_0$, $4d_0$, ..., которые уже имеются в первой строке). Если какое-либо d_j равно нулю, последний член в правой части соответствующей строки, конечно, может быть опущен. Кроме того, как обнаружил Э. Г. Тюрбер (E. G. Thurber) [*Duke Math. J.* **40** (1973), 907–913], можно опустить все четные числа (за исключением 2) в первой строке, если привести значения вида $d_j/2^e$ в вычислении e шагами ранее.

Простейший случай m-арного метода — двоичный (бинарный) метод ($m = 2$), когда общая схема (4) упрощается до правила "S и X", упоминавшегося в начале этого раздела: бинарная аддитивная цепочка для $2n$ представляет собой бинарную аддитивную цепочку для n с добавлением числа $2n$; для $2n+1$ она является бинарной аддитивной цепочкой для $2n$ с последующим числом $2n + 1$. Из бинарного метода можно заключить, что

$$l(2^{e_0} + 2^{e_1} + \cdots + 2^{e_t}) \le e_0 + t, \qquad \text{если } e_0 > e_1 > \cdots > e_t \ge 0. \tag{5}$$

Определим теперь две вспомогательные функции для удобства последующего обсуждения:

$$\lambda(n) = \lfloor \lg n \rfloor; \tag{6}$$

$$\nu(n) = \text{количество единиц в двоичном представлении } n. \tag{7}$$

Таким образом, $\lambda(17) = 4$, $\nu(17) = 2$. Эти функции могут быть определены следующими рекуррентными соотношениями:

$$\lambda(1) = 0, \qquad \lambda(2n) = \lambda(2n+1) = \lambda(n) + 1; \tag{8}$$

$$\nu(1) = 1, \qquad \nu(2n) = \nu(n), \qquad \nu(2n+1) = \nu(n) + 1. \tag{9}$$

С их помощью можно записать, что бинарные аддитивные цепочки для n требуют ровно $\lambda(n) + \nu(n) - 1$ шагов, а (5) принимает вид

$$l(n) \le \lambda(n) + \nu(n) - 1. \tag{10}$$

Специальные классы цепочек. Не теряя общности, можно положить, что аддитивные цепочки *возрастающие*, т. е.

$$1 = a_0 < a_1 < a_2 < \cdots < a_r = n. \tag{11}$$

Если два числа из a одинаковы, то одно из них может быть опущено. Можно также переупорядочить последовательность (1) в порядке возрастания и удалить члены, бóльшие, чем n, не нарушая свойство аддитивной цепочки (2). *В дальнейшем будем рассматривать только возрастающие цепочки*, не оговаривая это соглашение явно.

Теперь удобно определить несколько специальных терминов, относящихся к аддитивным цепочкам. По определению для $1 \le i \le r$

$$a_i = a_j + a_k \tag{12}$$

для некоторых j и k, $0 \le k \le j < i$. Если это соотношение выполняется для более чем одной пары (j, k), полагаем, что j — наибольшее из возможных. Будем говорить, что шаг i цепочки (11) — *удвоение*, если $j = k = i - 1$. Тогда a_i имеет максимально возможное значение $2a_{i-1}$, которое может следовать за возрастающей цепочкой $1, a_1, \ldots, a_{i-1}$. Если j (но не обязательно k) равно $i - 1$, будем говорить, что шаг i — *звездный шаг* (*star step*). Важность звездных шагов поясняется ниже. И наконец, будем говорить, что шаг i представляет собой *малый шаг*, если $\lambda(a_i) = \lambda(a_{i-1})$. Поскольку $a_{i-1} < a_i \le 2a_{i-1}$, величина $\lambda(a_i)$ всегда равна либо $\lambda(a_{i-1})$, либо $\lambda(a_{i-1}) + 1$; отсюда следует, что *длина r любой цепочки* (11) *равна $\lambda(n)$ плюс количество малых шагов.*

Эти типы шагов удовлетворяют ряду элементарных соотношений. Шаг 1 всегда представляет собой удвоение. Ясно, что удвоение — звездный шаг, но никогда не малый шаг. За удвоением всегда следует звездный шаг. Кроме того, если шаг i *не* малый, то шаг $i + 1$ является либо малым, либо звездным, либо и тем, и другим одновременно. Рассматривая данное утверждение с другой стороны, получим, что если шаг $i + 1$ не является ни малым, ни звездным, то шаг i должен быть малым.

Звездной цепочкой (*star chain*) является аддитивная цепочка, включающая только звездные шаги. Это означает, что каждый член a_i представляет собой сумму a_{i-1} и предшествующего ему a_k; простой "компьютер", упоминавшийся ранее, после (2), в звездной цепочке использует только две операции, `STA` и `ADD` (без `LDA`),

поскольку каждый новый член последовательности использует предыдущий результат, находящийся в аккумуляторе. Большинство рассмотренных здесь аддитивных цепочек являются звездными. Минимальная длина звездной цепочки для n записывается как $l^*(n)$; понятно, что

$$l(n) \le l^*(n). \tag{13}$$

Теперь можно вывести несколько нетривиальных фактов об аддитивных цепочках. Сначала покажем, что в цепочке должно быть весьма много удвоений, если r не слишком сильно отличается от $\lambda(n)$.

Теорема А. *Если аддитивная цепочка* (11) *включает* d *удвоений и* $f = r - d$ *неудвоений, то*

$$n \le 2^{d-1} F_{f+3}. \tag{14}$$

Доказательство. Используя индукцию по $r = d + f$, находим, что (14) истинно при $r = 1$. При $r > 1$ имеется три случая. Если шаг r — удвоение, то $\frac{1}{2}n = a_{r-1} \le 2^{d-2} F_{f+3}$; следовательно, (14) выполняется. Если шаги r и $r-1$ не являются удвоениями, то $a_{r-1} \le 2^{d-1} F_{f+2}$ и $a_{r-2} \le 2^{d-1} F_{f+1}$; следовательно, $n = a_r \le a_{r-1} + a_{r-2} \le 2^{d-1}(F_{f+2} + F_{f+1}) = 2^{d-1} F_{f+3}$ по определению последовательности Фибоначчи. И наконец, если шаг r — неудвоение, а шаг $r-1$ — удвоение, то $a_{r-2} \le 2^{d-2} F_{f+2}$ и $n = a_r \le a_{r-1} + a_{r-2} = 3a_{r-2}$. Теперь $2F_{f+3} - 3F_{f+2} = F_{f+1} - F_f \ge 0$; следовательно, $n \le 2^{d-1} F_{f+3}$ для всех случаев. ▮

Использованный нами метод доказательства показывает, что неравенство (14) является "наилучшим возможным" при указанных предположениях; аддитивная цепочка

$$1, 2, \ldots, 2^{d-1}, 2^{d-1}F_3, 2^{d-1}F_4, \ldots, 2^{d-1}F_{f+3} \tag{15}$$

имеет d удвоений и f неудвоений.

Следствие. *Если аддитивная цепочка* (11) *включает* f *неудвоений и* s *малых шагов, то*

$$s \le f \le 3.271s. \tag{16}$$

Доказательство. Очевидно, что $s \le f$. Имеем

$$2^{\lambda(n)} \le n \le 2^{d-1} F_{f+3} \le 2^d \phi^f = 2^{\lambda(n)+s} (\phi/2)^f,$$

поскольку $d + f = \lambda(n) + s$ и $F_{f+3} \le 2\phi^f$ при $f \ge 0$. Значит, $0 \le s \ln 2 + f \ln(\phi/2)$ и (16) следует из того факта, что $\ln 2 / \ln(2/\phi) \approx 3.2706$. ▮

Значения $l(n)$ для специальных n. Легко показать при помощи индукции, что $a_i \le 2^i$, и, таким образом, $\lg n \le r$ в любой аддитивной цепочке (11). Следовательно,

$$l(n) \ge \lceil \lg n \rceil. \tag{17}$$

Эта нижняя граница вместе с верхней границей (10), полученной при помощи бинарного метода, дают значения

$$l(2^A) = A; \tag{18}$$

$$l(2^A + 2^B) = A + 1, \qquad \text{если } A > B. \tag{19}$$

Другими словами, бинарный метод является оптимальным при $\nu(n) \le 2$. При помощи некоторых вычислений можно расширить эти формулы для случая, когда $\nu(n) = 3$.

Теорема B. $l(2^A + 2^B + 2^C) = A + 2$, *если* $A > B > C$. (20)

Доказательство. В действительности можно доказать более строгий результат, который будет использован позже в этом разделе. *Все аддитивные цепочки с ровно одним малым шагом принадлежат одному из следующих шести типов* (здесь все шаги, указанные как "...", являются удвоениями).

Тип 1. $1, \ldots, 2^A, 2^A + 2^B, \ldots, 2^{A+C} + 2^{B+C}$; $A > B \ge 0$, $C \ge 0$.

Тип 2. $1, \ldots, 2^A, 2^A + 2^B, 2^{A+1} + 2^B, \ldots, 2^{A+C+1} + 2^{B+C}$; $A > B \ge 0$, $C \ge 0$.

Тип 3. $1, \ldots, 2^A, 2^A + 2^{A-1}, 2^{A+1} + 2^{A-1}, 2^{A+2}, \ldots, 2^{A+C}$; $A > 0$, $C \ge 2$.

Тип 4. $1, \ldots, 2^A, 2^A + 2^{A-1}, 2^{A+1} + 2^A, 2^{A+2}, \ldots, 2^{A+C}$; $A > 0$, $C \ge 2$.

Тип 5. $1, \ldots, 2^A, 2^A + 2^{A-1}, \ldots, 2^{A+C} + 2^{A+C-1}, 2^{A+C+1} + 2^{A+C-2}, \ldots$
$\ldots, 2^{A+C+D+1} + 2^{A+C+D-2}$; $A > 0$, $C > 0$, $D \ge 0$.

Тип 6. $1, \ldots, 2^A, 2^A + 2^B, 2^{A+1}, \ldots, 2^{A+C}$; $A > B \ge 0$, $C \ge 1$.

Непосредственные вычисления показывают, что эти шесть типов исчерпывают все возможности. Согласно следствию из теоремы A имеется не более трех неудвоений при наличии одного малого шага; этот максимум встречается только в последовательности третьего типа. Все приведенные выше цепочки — звездные, за исключением типа 6, когда $B < A - 1$.

Теперь теорема следует из того факта, что

$$l(2^A + 2^B + 2^C) \le A + 2,$$

и $l(2^A + 2^B + 2^C)$ должно быть больше, чем $A + 1$, поскольку ни один из шести возможных типов не имеет $\nu(n) > 2$. ∎

(Э. де Жонкиэрес (E. de Jonquières) в 1894 году без доказательства указал, что $l(n) \ge \lambda(n) + 2$, когда $\nu(n) > 2$. Впервые теорема B появилась в работе A. A. Gioia, M. V. Subbarao and M. Sugunamma, *Duke Math. J.* **29** (1962), 481–487.)

Вычислить $l(2^A + 2^B + 2^C + 2^D)$ при $A > B > C > D$ более сложно. По бинарному методу эта величина не превышает $A + 3$, а в соответствии с доказательством теоремы B она не меньше, чем $A + 2$. Величина $A + 2$ является допустимой, так как известно, что бинарный метод не оптимален при $n = 15$ или $n = 23$. Как мы сейчас увидим, при $\nu(n) = 4$ можно дать полное определение поведения этой величины.

Теорема C. *Если* $\nu(n) \ge 4$, *то* $l(n) \ge \lambda(n) + 3$ *за исключением следующих случаев, когда* $A > B > C > D$ *и* $l(2^A + 2^B + 2^C + 2^D)$ *равно* $A + 2$.

Случай 1. $A - B = C - D$. (Пример: $n = 15$.)

Случай 2. $A - B = C - D + 1$. (Пример: $n = 23$.)

Случай 3. $A - B = 3$, $C - D = 1$. (Пример: $n = 39$.)

Случай 4. $A - B = 5$, $B - C = C - D = 1$. (Пример: $n = 135$.)

Доказательство. Когда $l(n) = \lambda(n) + 2$, существует аддитивная цепочка для n, имеющая только два малых шага. Она состоит из одного из шести типов, перечисленных в доказательстве теоремы B, малого шага и последовательности немалых шагов. Будем говорить, что n "специально", если $n = 2^A + 2^B + 2^C + 2^D$ для одного из четырех случаев, перечисленных в теореме. Можно получить аддитивные цепочки требуемого вида для каждого специального n, как показано в упр. 13, поэтому остается доказать, что не существует цепочек с точно двумя малыми шагами, содержащих любые элементы с $\nu(a_i) \geq 4$, за исключением специального a_i.

Назовем цепочкой-контрпримером аддитивную цепочку с двумя малыми шагами, такую, что $\nu(a_r) \geq 4$, но a_r не является специальным. Если цепочка-контрпример существует, то рассмотрим цепочку $1 = a_0 < a_1 < \cdots < a_r = n$ наименьшей возможной длины. Тогда шаг r не является малым шагом, поскольку ни один из типов цепочек из доказательства теоремы B не может предшествовать малому шагу с $\nu(n) \geq 4$, за исключением специального n. Кроме того, шаг r не является удвоением, так как более коротким контрпримером была бы цепочка $a_0, \ldots, a_{r-1}$. Наконец, шаг r является звездным, так как иначе цепочка $a_0, \ldots, a_{r-2}, a_r$ представляла бы собой более короткую цепочку-контрпример. Таким образом,

$$a_r = a_{r-1} + a_{r-k}, \qquad k \geq 2, \qquad \text{и } \lambda(a_r) = \lambda(a_{r-1}) + 1. \tag{21}$$

Обозначим число переносов, встречающихся при сложении a_{r-1} и a_{r-k} в двоичной системе счисления по алгоритму 4.3.1A, как c. Используя фундаментальное соотношение

$$\nu(a_r) = \nu(a_{r-1}) + \nu(a_{r-k}) - c, \tag{22}$$

можно доказать, что *шаг $r-1$ не является малым* (см. упр. 14).

Пусть $m = \lambda(a_{r-1})$. Поскольку ни r, ни $r-1$ не является малым шагом, $c \geq 2$; равенство $c = 2$ может быть справедливо только тогда, когда $a_{r-1} \geq 2^m + 2^{m-1}$.

Теперь предположим, что $r-1$ — не звездный шаг. Тогда $r-2$ — малый шаг и $a_0, \ldots, a_{r-3}, a_{r-1}$ представляет собой цепочку с только одним малым шагом. Следовательно, $\nu(a_{r-1}) \leq 2$ и $\nu(a_{r-2}) \leq 4$. Соотношение (22) может выполняться, только если $\nu(a_r) = 4$, $\nu(a_{r-1}) = 2$, $k = 2$, $c = 2$, $\nu(a_{r-2}) = 4$. Из $c = 2$ можно заключить, что $a_{r-1} = 2^m + 2^{m-1}$; следовательно, $a_0, a_1, \ldots, a_{r-3} = 2^{m-1} + 2^{m-2}$ является аддитивной цепочкой с только одним малым шагом и эта цепочка должна быть первого типа, так что a_r относится к случаю 3. Таким образом, *$r-1$ является звездным шагом.*

Теперь предположим, что $a_{r-1} = 2^t a_{r-k}$ для некоторого t. Если $\nu(a_{r-1}) \leq 3$, то в соответствии с (22) $c = 2$, $k = 2$ и a_r должно относиться к случаю 3. С другой стороны, если $\nu(a_{r-1}) = 4$, то a_{r-1} будет специальным, и при рассмотрении каждого случая легко видеть, что a_r также относится к одному из четырех случаев. (Случай 4 возникает, например, когда $a_{r-1} = 90$, $a_{r-k} = 45$ или когда $a_{r-1} = 120$, $a_{r-k} = 15$.) Поэтому можно заключить, что $a_{r-1} \neq 2^t a_{r-k}$ для любого t.

Мы доказали, что $a_{r-1} = a_{r-2} + a_{r-q}$ для некоторого $q \geq 2$. Если $k = 2$, то $q > 2$ и $a_0, a_1, \ldots, a_{r-2}, 2a_{r-2}, 2a_{r-2} + a_{r-q} = a_r$ представляет контрпример последовательности, у которой $k > 2$. Поэтому можно считать, что $k > 2$.

Теперь предположим, что $\lambda(a_{r-k}) = m - 1$. Случай, когда $\lambda(a_{r-k}) < m - 1$, может быть исключен в результате подобного рассмотрения, как показано в упр. 14. Если $k = 4$, то и $r-2$, и $r-3$ являются малыми шагами; следовательно, $a_{r-4} = 2^{m-1}$

и (22) невозможно. Поэтому $k = 3$; шаг $r - 2$ является малым, $\nu(a_{r-3}) = 2$, $c = 2$, $a_{r-1} \geq 2^m + 2^{m-1}$ и $\nu(a_{r-1}) = 4$. Должно существовать как минимум два переноса при сложении чисел a_{r-2} и $a_{r-1} - a_{r-2}$. Значит, $\nu(a_{r-2}) = 4$ и a_{r-2} (будучи специальным и $\geq \frac{1}{2}a_{r-1}$) имеет вид $2^{m-1}+2^{m-2}+2^{d+1}+2^d$ для некоторого d. Теперь a_{r-1} равно либо $2^m+2^{m-1}+2^{d+1}+2^d$, либо $2^m+2^{m-1}+2^{d+2}+2^{d+1}$ и в обоих случаях a_{r-3} должно быть равно $2^{m-1}+2^{m-2}$, так что a_r относится к случаю 3. ▮

Э. Г. Тюрбер [*Pacific J. Math.* **49** (1973), 229–242] расширил теорему C для показа того, что $l(n) \geq \lambda(n) + 4$ при $\nu(n) > 8$. Представляется обоснованным предположение, что $l(n) \geq \lambda(n) + \lg\nu(n)$ в общем случае, поскольку А. Шёнхаге (A. Schönhage) очень близко подошел к доказательству этого факта (см. упр. 28).

***Асимптотические значения.** Из теоремы C следует, что получение точных значений $l(n)$ при больших n является, по всей видимости, очень трудной задачей. Однако можно определить приближенное поведение функции в предельном случае, когда $n \to \infty$.

Теорема D (A. Brauer, *Bull. Amer. Math. Soc.* **45** (1939), 736–739).

$$\lim_{n\to\infty} l^*(n)/\lambda(n) = \lim_{n\to\infty} l(n)/\lambda(n) = 1. \tag{23}$$

Доказательство. Аддитивная цепочка (4) для 2^k-арного метода является звездной, если удалить из нее все вторые вхождения каждого элемента, встречающегося в цепочке дважды. Если a_i — первый элемент среди $2d_0$, $4d_0$, ... второй строки, который отсутствует в первой строке, имеем $a_i \leq 2(m-1)$; следовательно, $a_i = (m-1) + a_j$ для некоторого a_j в первой строке. Суммируя общую длину цепочки, получаем

$$\lambda(n) \leq l(n) \leq l^*(n) < (1 + 1/k)\lg n + 2^k \tag{24}$$

для всех $k \geq 1$. Утверждение теоремы можно получить, если, например, выбрать $k = \lfloor \frac{1}{2}\lg\lambda(n) \rfloor$. ▮

Если положить $k = \lambda\lambda(n) - 2\lambda\lambda\lambda(n)$ в (24) для больших n, где $\lambda\lambda(n)$ означает $\lambda\big(\lambda(n)\big)$, получим более строгую асимптотическую границу

$$l(n) \leq l^*(n) \leq \lambda(n) + \lambda(n)/\lambda\lambda(n) + O\big(\lambda(n)\lambda\lambda\lambda(n)/\lambda\lambda(n)^2\big). \tag{25}$$

Второй член $\lambda(n)/\lambda\lambda(n)$ является, по существу, лучшим из того, что можно получить из (24). Можно провести более глубокий анализ нижних границ, чтобы показать, что этот член $\lambda(n)/\lambda\lambda(n)$ является неотъемлемой частью (25). Чтобы понять, почему это так, рассмотрим следующую теорему.

Теорема E (Paul Erdős, *Acta Arithmetica* **6** (1960), 77–81). *Пусть ϵ — положительное действительное число. Количество аддитивных цепочек* (11), *таких, что*

$$\lambda(n) = m, \qquad r \leq m + (1 - \epsilon)m/\lambda(m), \tag{26}$$

меньше, чем α^m, для некоторого $\alpha < 2$ для всех достаточно больших m. (Другими словами, число аддитивных цепочек, столь коротких, что удовлетворяется соотношение (26), существенно меньше количества значений n, таких, что $\lambda(n) = m$, при больших m.)

Доказательство. Чтобы оценить количество возможных аддитивных цепочек, сначала необходимо улучшить теорему A, и это позволит нам более успешно работать с неудвоениями.

Лемма P. *Пусть* $\delta < \sqrt{2} - 1$ *— фиксированное положительное действительное число. Назовем шаг* i *аддитивной цепочки мини-шагом, если это не удвоение и если* $a_i < a_j(1+\delta)^{i-j}$ *для некоторого* j, *где* $0 \le j < i$. *Если аддитивная цепочка содержит* s *малых шагов и* t *мини-шагов, то*

$$t \le s/(1-\theta), \qquad \text{где } (1+\delta)^2 = 2^\theta. \tag{27}$$

Доказательство. Для каждого мини-шага i_k, $1 \le k \le t$, имеем $a_{i_k} < a_{j_k}(1+\delta)^{i_k - j_k}$ для некоторых $j_k < i_k$. Пусть $I_1, \ldots, I_t$ — интервалы $(j_1 \,..\, i_1], \ldots, (j_t \,..\, i_t]$, где запись $(j \,..\, i\,]$ означает множество всех целых чисел k, таких, что $j < k \le i$. Можно найти (см. упр. 17) такие неперекрывающиеся интервалы $J_1, \ldots, J_h = (j'_1 \,..\, i'_1], \ldots, (j'_h \,..\, i'_h\,]$, что

$$\begin{aligned} I_1 \cup \cdots \cup I_t &= J_1 \cup \cdots \cup J_h, \\ a_{i'_k} < a_{j'_k}(1+\delta)^{2(i'_k - j'_k)} &\qquad \text{для } 1 \le k \le h. \end{aligned} \tag{28}$$

Теперь для всех шагов i, находящихся вне интервалов $J_1, \ldots, J_h$, имеем $a_i \le 2a_{i-1}$; следовательно, если положить

$$q = (i'_1 - j'_1) + \cdots + (i'_h - j'_h),$$

можно получить $2^{\lambda(n)} \le n \le 2^{r-q}(1+\delta)^{2q} = 2^{\lambda(n)+s-(1-\theta)q} \le 2^{\lambda(n)+s-(1-\theta)t}$. ▮

Возвращаясь к доказательству теоремы E, выберем $\delta = 2^{\epsilon/4} - 1$ и разделим r шагов каждой аддитивной цепочки на три класса:

$$t \text{ мини-шагов}, \qquad u \text{ удвоений}, \qquad v \text{ прочих шагов}, \qquad t + u + v = r. \tag{29}$$

При другом способе подсчета шагов получим s малых шагов, где $s+m = r$. Из наших предположений, теоремы A и леммы P получим соотношения

$$s \le (1-\epsilon)m/\lambda(m), \qquad t + v \le 3.271s, \qquad t \le s/(1-\epsilon/2). \tag{30}$$

Для данных s, t, u и v, удовлетворяющих этим условиям, существует

$$\binom{r}{t,u,v} = \binom{r}{t+v}\binom{t+v}{v} \tag{31}$$

способов назначения шагов определенным классам. При заданном распределении шагов рассмотрим, каким образом могут быть выбраны не мини-шаги. Если шаг i является одним из "других" шагов в (29), $a_i \ge (1+\delta)a_{i-1}$, так что $a_i = a_j + a_k$, где $\delta a_{i-1} \le a_k \le a_j \le a_{i-1}$. Кроме того, $a_j \le a_i/(1+\delta)^{i-j} \le 2a_{i-1}/(1+\delta)^{i-j}$, поэтому $\delta \le 2/(1+\delta)^{i-j}$. Это дает не более β выборов j, где β — константа, зависящая только от δ. Имеется также не более β выборов k, так что количество способов назначения j и k для каждого не мини-шага не превышает

$$\beta^{2v}. \tag{32}$$

И наконец, как только j и k выбраны для каждого из не мини-шагов, имеется менее

$$\binom{r^2}{t} \tag{33}$$

способов выбора j и k для мини-шагов: выбираем t различных пар $(j_1, k_1), \ldots, (j_t, k_t)$ индексов в диапазоне $0 \le k_h \le j_h < r$ меньшим числом способов, которые заданы в (33). Затем для каждого мини-шага i используем пару индексов (j_h, k_h), такую, что

a) $j_h < i$;

b) $a_{j_h} + a_{k_h}$ принимает наименьшее возможное значение среди еще неиспользованных для меньших мини-шагов i пар;

c) $a_i = a_{j_h} + a_{k_h}$ удовлетворяет определению мини-шага.

Если таких пар (j_h, k_h) не существует, аддитивная цепочка не будет получена; с другой стороны, любая аддитивная цепочка с мини-шагами на назначенных местах должна быть выбрана одним из этих способов, так что (33) представляет собой верхнюю границу имеющихся возможностей.

Таким образом, общее количество возможных аддитивных цепочек, удовлетворяющих (26), ограничено сверху величиной (31), умноженной на (32) и на (33), которая просуммирована по всем подходящим s, t, u и v. Доказательство теоремы E теперь может быть завершено стандартными оценками этих функций (см. упр. 18). ▮

Следствие. *Величина $l(n)$ асимптотически равна $\lambda(n) + \lambda(n)/\lambda\lambda(n)$ для почти всех n. Более строго говоря, существует функция $f(n)$, такая, что $f(n) \to 0$ при $n \to \infty$ и*

$$\Pr\Big(\big| l(n) - \lambda(n) - \lambda(n)/\lambda\lambda(n) \big| \ge f(n)\lambda(n)/\lambda\lambda(n) \Big) = 0. \tag{34}$$

(См. определение вероятности "Pr" в разделе 3.5.)

Доказательство. Верхняя граница (25) показывает, что (34) выполняется без знаков абсолютного значения. Нижняя граница вытекает из теоремы E, если положить следующее: $f(n)$ уменьшается до нуля достаточно медленно, чтобы при $f(n) \le \epsilon$ значение N было настолько велико, чтобы не более ϵN значений $n \le N$ имели $l(n) \le \lambda(n) + (1 - \epsilon)\lambda(n)/\lambda\lambda(n)$. ▮

***Звездные цепочки.** Оптимисты сочтут оправданным предположение о том, что $l(n) = l^*(n)$; трудно поверить, что по данной аддитивной цепочке минимальной длины $l(n)$ нельзя найти цепочку той же длины, удовлетворяющую условию звездности (по-видимому, слабому). Однако в 1958 году Вальтер Хансен (Walter Hansen) доказал замечательную теорему о том, что для определенных больших значений n значение $l(n)$ строго меньше, чем $l^*(n)$, а также ряд связанных с ней теорем, которые мы сейчас и рассмотрим.

Теоремы Хансена начинаются с исследования детальной структуры звездных цепочек. Пусть $n = 2^{e_0} + 2^{e_1} + \cdots + 2^{e_t}$, где $e_0 > e_1 > \cdots > e_t \ge 0$, и пусть $1 = a_0 < a_1 < \cdots < a_r = n$ — звездная цепочка для n. Если в этой цепочке имеется d удвоений, определим вспомогательную последовательность

$$0 = d_0 \le d_1 \le d_2 \le \cdots \le d_r = d, \tag{35}$$

где d_i — количество удвоений среди шагов 1, 2, ..., i. Также определим последовательность "мультимножеств" S_0, S_1, ..., S_r, которая будет отслеживать наличие степеней 2 в цепочке. (*Мультимножество* (*multiset*) представляет собой математический объект, подобный множеству, но он может содержать одинаковые элементы. Объект может быть элементом мультимножества несколько раз, причем кратность его появления в множестве имеет важное значение. Более подробно с

мультимножествами вы ознакомитесь в упр. 19.) Мультимножества S_i определены следующими правилами:

a) $S_0 = \{0\}$;

b) если $a_{i+1} = 2a_i$, то $S_{i+1} = S_i + 1 = \{x + 1 \mid x \in S_i\}$;

c) если $a_{i+1} = a_i + a_k$, $k < i$, то $S_{i+1} = S_i \uplus S_k$.

(Символ $\uplus$ означает, что мультимножества объединяются со сложением кратностей.) Из этого определения следует, что

$$a_i = \sum_{x \in S_i} 2^x, \tag{36}$$

где члены суммы необязательно различны, в частности

$$n = 2^{e_0} + 2^{e_1} + \cdots + 2^{e_t} = \sum_{x \in S_r} 2^x. \tag{37}$$

Число элементов в последней сумме не превышает 2^f, где $f = r - d$ — количество неудвоений.

Поскольку n имеет два различных бинарных представления в (37), мультимножество S_r можно разбить на мультимножества M_0, M_1, ..., M_t, такие, что

$$2^{e_j} = \sum_{x \in M_j} 2^x, \qquad 0 \le j \le t. \tag{38}$$

Данную операцию можно выполнить, разместив элементы S_r в порядке неубывания $x_1 \le x_2 \le \cdots$ и взяв $M_t = \{x_1, x_2, \ldots, x_k\}$, где $2^{x_1} + \cdots + 2^{x_k} = 2^{e_t}$. Это должно быть возможно, поскольку e_t — наименьшее из всех e. Аналогично $M_{t-1} = \{x_{k+1}, x_{k+2}, \ldots, x_{k'}\}$ и т. д. Процесс легко визуализируется в двоичной системе счисления, как показано в следующем примере.

Пусть M_j содержит m_j элементов (с учетом кратности); тогда $m_j \le 2^f - t$, поскольку S_r имеет не более 2^f элементов и разбито на $t + 1$ непустых мультимножеств. Из (38) видно, что

$$e_j \ge x > e_j - m_j \qquad \text{для всех } x \in M_j. \tag{39}$$

Наше исследование структуры звездных цепочек завершается построением мультимножеств M_{ij}, в которых записана история M_j. Мультимножество S_i разбито на $t + 1$ мультимножеств следующим образом:

a) $M_{rj} = M_j$;

b) если $a_{i+1} = 2a_i$, то $M_{ij} = M_{(i+1)j} - 1 = \{x - 1 \mid x \in M_{(i+1)j}\}$;

c) если $a_{i+1} = a_i + a_k$, $k < i$, то, поскольку $S_{i+1} = S_i \uplus S_k$, полагаем, что $M_{ij} = M_{(i+1)j}$ минус S_k, т. е. мы удаляем элементы S_k из $M_{(i+1)j}$. Если некоторый элемент S_k появляется в двух или более различных мультимножествах $M_{(i+1)j}$, удаляем его из множества с наибольшим возможным значением j. Это правило однозначно определяет M_{ij} для каждого j, когда i фиксировано.

Из данного определения следует, что

$$e_j + d_i - d \ge x > e_j + d_i - d - m_j \qquad \text{для всех } x \in M_{ij}. \tag{40}$$

В качестве примера такой детализированной конструкции рассмотрим звездную цепочку 1, 2, 3, 5, 10, 20, 23, для которой $t = 3$, $r = 6$, $d = 3$, $f = 3$. Получим следующий набор мультимножеств.

$(d_0, d_1, \ldots, d_6)$:	0	1	1	1	2	3	3		
$(a_0, a_1, \ldots, a_6)$:	1	2	3	5	10	20	23		
$(M_{03}, M_{13}, \ldots, M_{63})$:							0	M_3	$e_3 = 0$, $m_3 = 1$
$(M_{02}, M_{12}, \ldots, M_{62})$:							1	M_2	$e_2 = 1$, $m_2 = 1$
$(M_{01}, M_{11}, \ldots, M_{61})$:			0	0	1	2	2	M_1	$e_1 = 2$, $m_1 = 1$
$(M_{00}, M_{10}, \ldots, M_{60})$:	0	1	1	1	2	3	3	M_0	$e_0 = 4$, $m_0 = 2$
				1	2	3	3		
	S_0	S_1	S_2	S_3	S_4	S_5	S_6		

Таким образом, $M_{40} = \{2, 2\}$ и т. д. Из построения можно увидеть, что d_i является наибольшим элементом S_i; следовательно,

$$d_i \in M_{i0}. \tag{41}$$

Наиболее важная часть этой структуры следует из (40) и одним из непосредственных ее следствий является лемма K.

Лемма K. *Если M_{ij} и M_{uv} оба содержат общее число x, то*

$$-m_v < (e_j - e_v) - (d_u - d_i) < m_j. \quad \blacksquare \tag{42}$$

Хотя лемма K и не выглядит особо "сильной", она утверждает (когда m_j и m_v обоснованно малы и когда M_{ij} содержит общий с M_{uv} элемент), что количество удвоений между шагами u и i примерно равно разности между показателями e_v и e_j. Это производит впечатление некоторой регулярности в аддитивной цепочке и наводит на мысль о том, что можно доказать результат, аналогичный приведенной выше теореме B, а именно — что $l^*(n) = e_0 + t$, если показатели степеней e_j достаточно различны. Следующая теорема показывает, как это можно сделать.

Теорема H (W. Hansen, *Crelle* **202** (1959), 129–136). *Пусть $n = 2^{e_0} + 2^{e_1} + \cdots + 2^{e_t}$, где $e_0 > e_1 > \cdots > e_t \geq 0$. Если*

$$e_0 > 2e_1 + 2.271(t-1) \qquad \text{и} \qquad e_{i-1} \geq e_i + 2m \quad \text{для } 1 \leq i \leq t, \tag{43}$$

где $m = 2^{\lfloor 3.271(t-1) \rfloor} - t$, то $l^*(n) = e_0 + t$.

Доказательство. Положим, что $t > 2$, поскольку результат теоремы при $t \leq 2$ истинен без наложения ограничений на e. Допустим, что имеется звездная цепочка $1 = a_0 < a_1 < \cdots < a_r = n$ для n с $r \leq e_0 + t - 1$. Пусть ее структуру отражают целые числа d, f, d_0, $\ldots$, d_r и мультимножества M_{ij} и S_i, как было определено выше. Согласно следствию из теоремы A известно, что $f \leq \lfloor 3.271(t-1) \rfloor$. Поэтому значение m является настоящей верхней гранью числа элементов m_j каждого мультимножества M_j.

В сумме

$$a_i = \Big(\sum_{x \in M_{i0}} 2^x\Big) + \Big(\sum_{x \in M_{i1}} 2^x\Big) + \cdots + \Big(\sum_{x \in M_{it}} 2^x\Big)$$

не будет переносов из члена, соответствующего M_{ij}, к члену, соответствующему $M_{i(j-1)}$, если рассматривать ее как выполняющуюся в двоичной системе счисления, поскольку e достаточно далеко разнесены (см. (40)). В частности, сумма всех членов для $j \neq 0$ не даст переносов к членам для $j = 0$, так что мы должны получить

$$a_i \geq \sum_{x \in M_{i0}} 2^x \geq 2^{\lambda(a_i)}, \qquad 0 \leq i \leq r. \tag{44}$$

Для доказательства теоремы H необходимо показать, что в некотором смысле t дополнительных степеней n должны размещаться "по одной", чтобы можно было определить, на каком шаге каждый из этих членов, по существу, включается в аддитивную цепочку.

Пусть j представляет собой число между 1 и t. Поскольку M_{0j} пусто, а $M_{rj} = M_j$ не пусто, можно найти *первый* шаг i, для которого M_{ij} не пусто.

Из способа, которым определяется M_{ij}, известно, что шаг i не является удвоением: $a_i = a_{i-1} + a_u$ для некоторого $u < i - 1$. Также известно, что все элементы M_{ij} являются элементами S_u. Докажем, что a_u должно быть относительно мало́ по сравнению с a_i.

Пусть x_j является элементом M_{ij}. Тогда, поскольку $x_j \in S_u$, существует v, для которого $x_j \in M_{uv}$. Отсюда следует, что

$$d_i - d_u > m, \tag{45}$$

т. е. между шагами u и i встречается как минимум $m+1$ удвоений: если $d_i - d_u \leq m$, лемма K гласит, что $|e_j - e_v| < 2m$; следовательно, $v = j$. Однако это невозможно, потому что M_{uj} пусто в соответствии с нашим выбором шага i.

Все элементы S_u не превышают $e_1 + d_i - d$. Действительно, если $x \in S_u \subseteq S_i$ и $x > e_1 + d_i - d$, то $x \in M_{u0}$ и $x \in M_{i0}$ согласно (40), так что из леммы K следует, что $|d_i - d_u| < m$, а это противоречит (45). Значит, это рассуждение доказывает, что M_{i0} не имеет общих с S_u элементов и $M_{(i-1)0} = M_{i0}$. Из (44) имеем $a_{i-1} \geq 2^{\lambda(a_i)}$, и поэтому *шаг i является малым шагом.*

Теперь выведем ключевой факт во всем этом доказательстве: *все элементы S_u являются элементами M_{u0}.* Действительно, если это не так, пусть x будет элементом S_u, таким, что $x \notin M_{u0}$. Поскольку $x \geq 0$, из (40) следует, что $e_1 \geq d - d_u$ и

$$e_0 = f + d - s \leq 2.271s + d \leq 2.271(t-1) + e_1 + d_u.$$

Из гипотезы (43) получаем, что $d_u > e_1$. Однако $d_u \in S_u$ в соответствии с (41) и не может находиться в M_{i0}. Поэтому $d_u \leq e_1 + d_i - d \leq e_1$, что приводит к противоречию.

Возвращаясь к элементу x_j из M_{ij}, получаем, что $x_j \in M_{uv}$. К тому же было доказано, что $v = 0$. Поэтому, используя (40) еще раз, получим

$$e_0 + d_u - d \geq x_j > e_0 + d_u - d - m_0. \tag{46}$$

Теперь для всех $j = 1, 2, \ldots, t$ определено число x_j, удовлетворяющее (46), и малый шаг i, на котором в аддитивную цепочку вводится член 2^{e_j}. Если $j \neq j'$, шаг i, на котором это происходит, не может быть одним и тем же и для j, и для j'. Из (46) следует, что $|x_j - x_{j'}| < m$, в то время как элементы M_{ij} и $M_{ij'}$ должны отличаться более чем на m, поскольку e_j и $e_{j'}$ существенно различны. Мы вынуждены

заключить, что цепочка содержит как минимум t малых шагов, но это приводит к противоречию. ▮

Теорема F (В. Хансен (W. Hansen)).

$$l(2^A + xy) \le A + \nu(x) + \nu(y) - 1, \qquad \text{если } \lambda(x) + \lambda(y) \le A. \tag{47}$$

Доказательство. Аддитивная цепочка (в общем случае *не* звездная) может быть построена путем комбинирования бинарного метода и метода множителя. Пусть $x = 2^{x_1} + \cdots + 2^{x_u}$ и $y = 2^{y_1} + \cdots + 2^{y_v}$, где $x_1 > \cdots > x_u \ge 0$ и $y_1 > \cdots > y_v \ge 0$.

Первые шаги цепочки представляют собой последовательные степени 2, пока не достигнуто значение 2^{A-y_1}; между этими шагами в соответствующих местах вставляются дополнительные значения $2^{x_u-1} + 2^{x_u}$, $2^{x_u-2} + 2^{x_u-1} + 2^{x_u}$, ... и x. По достижении цепочкой $2^{A-y_i} + x(2^{y_1-y_i} + \cdots + 2^{y_{i-1}-y_i})$ продолжаем построение, добавив x и удвоив результирующую сумму $y_i - y_{i+1}$ раз, и получаем

$$2^{A-y_{i+1}} + x(2^{y_1-y_{i+1}} + \cdots + 2^{y_i-y_{i+1}}).$$

Если это построение выполнено для $i = 1, 2, \ldots, v$, приняв для удобства, что $y_{v+1} = 0$, получаем требовавшуюся аддитивную цепочку для $2^A + xy$. ▮

Теорема F позволяет найти значения n, для которых $l(n) < l^*(n)$, поскольку теорема H дает точное значение $l^*(n)$ в некоторых случаях. Например, пусть $x = 2^{1016} + 1$, $y = 2^{2032} + 1$ и пусть

$$n = 2^{6103} + xy = 2^{6103} + 2^{3048} + 2^{2032} + 2^{1016} + 1.$$

Согласно теореме F имеем $l(n) \le 6106$. Однако применима и теорема H с $m = 508$, а это доказывает, что $l^*(n) = 6107$.

Обширные компьютерные вычисления показали, что $n = 12509$ является наименьшим значением с $l(n) < l^*(n)$. Для него нет более короткой звездной цепочки, чем последовательность 1, 2, 4, 8, 16, 17, 32, 64, 128, 256, 512, 1024, 1041, 2082, 4164, 8328, 8345, 12509. Наименьшее n с $\nu(n) = 5$ и $l(n) \ne l^*(n)$ — 16537 $= 2^{14} + 9 \cdot 17$ (см. упр..15).

Ян Ван Лиивен (Jan van Leeuwen) обобщил теорему H, показав, что

$$l^*(k2^{e_0}) + t \le l^*(kn) \le l^*(k2^{e_t}) + e_0 - e_t + t$$

для всех фиксированных $k \ge 1$, если показатели степени $e_0 > \cdots > e_t$ достаточно различны [*Crelle* **295** (1977), 202–207].

Некоторые предположения. Хотя, на первый взгляд, разумно предположить, что $l(n) = l^*(n)$, мы убедились, что оно неверно. Другое правдоподобное предположение, впервые сделанное А. Голардом (A. Goulard) и "доказанное" Э. де Жонкиэресом (E. de Jonquières) в L'*Interméd. des math.* **2** (1895), 125–126, состоит в том, что $l(2n) = l(n)+1$. Удвоение настолько эффективно, что не представляется возможным, чтобы могла существовать некоторая более короткая цепочка для $2n$, чем цепочка, получаемая в результате добавления удвоения к кратчайшей цепочке для n. Однако компьютерные вычисления показывают, что это предположение также неверно, поскольку $l(191) = l(382) = 11$. (Не так трудно найти звездную цепочку длиной 11 для 382, например 1, 2, 4, 5, 9, 14, 23, 46, 92, 184, 198, 382. Число 191 — минимальное из

чисел, таких, что $l(n) = 11$, и доказательство того, что $l(191) > 10$, вручную представляется весьма нетривиальным. Так, компьютерное доказательство автором этого факта с использованием метода отката, который будет рассмотрен в разделе 7.2.2, требует детального изучения 948 случаев.) Наименьшими четырьмя значениями n, такими, что $l(2n) = l(n)$, являются $n = 191, 701, 743, 1111$. Э. Г. Тюрбер доказал в *Pacific J. Math.* **49** (1973), 229–242, что третье из этих чисел является членом бесконечного семейства таких n, а именно — $23{\cdot}2^k{+}7$ для всех $k \geq 5$. Представляется разумным предположение о том, что $l(2n) \geq l(n)$, но даже оно может оказаться ложным. Кевин Р. Хебб (Kevin R. Hebb) показал, что $l(n) - l(mn)$ может быть сколь угодно большим при всех фиксированных целых числах m, не являющихся степенями 2 [*Notices Amer. Math. Soc.* **21** (1974), A–294]. Наименьшим значением, для которого $l(mn) < l(n)$, является $l\big((2^{13}+1)/3\big) = 15$.

Обозначим через $c(r)$ наименьшее значение n, такое, что $l(n) = r$. Вычисление $l(n)$ представляется более трудным для этой последовательности n, начинающейся следующим образом.

r	$c(r)$	r	$c(r)$	r	$c(r)$
1	2	10	127	19	18287
2	3	11	191	20	34303
3	5	12	379	21	65131
4	7	13	607	22	110591
5	11	14	1087	23	196591
6	19	15	1903	24	357887
7	29	16	3583	25	685951
8	47	17	6271	26	1176431
9	71	18	11231	27	2211837

Для $r \leq 11$ значение $c(r)$ приблизительно равно $c(r-1) + c(r-2)$, и этот факт привел ряд исследователей к мысли о том, что $c(r)$ растет, как функция ϕ^r. Однако из результата теоремы D $\big($с $n = c(r)\big)$ вытекает, что $r/\lg c(r) \to 1$ при $r \to \infty$. Значения, перечисленные здесь для $r > 18$, были вычислены Ахимом Фламменкампом (Achim Flammenkamp), кроме вычисленного Даниэлем Блейхенбахером (Daniel Bleichenbacher) значения $c(24)$. Фламменкамп заметил, что $c(r)$ хорошо аппроксимируется формулой $2^r \exp(-\theta r/\lg r)$ для $10 \leq r \leq 27$, где θ близко к $\ln 2$. Это вполне согласуется с верхней гранью (25). Некоторые исследователи одно время полагали, что $c(r)$ всегда представляет собой простое число (исходя из метода множителя); однако $c(15)$, $c(18)$ и $c(21)$ делятся на 11. Возможно, любое предположение об аддитивных цепочках неверно!

Табуляция значений $l(n)$ показывает, что эта функция на удивление гладкая; например, для всех n в диапазоне $1125 \leq n \leq 1148$ значение функции $l(n) = 13$. Компьютерные вычисления показали, что таблица $l(n)$ может быть построена для значений $2 \leq n \leq 1000$ с использованием формулы

$$l(n) = \min(l(n-1)+1,\, l_n) - \delta_n, \tag{48}$$

где $l_n = \infty$, если n — целое число, в противном случае $l_n = l(p) + l(n/p)$, если p — наименьшее простое число, делящее n; и наконец, $\delta_n = 1$ для n из табл. 1, в противном случае — $\delta_n = 0$.

Таблица 1

ЗНАЧЕНИЯ n ДЛЯ СПЕЦИАЛЬНЫХ АДДИТИВНЫХ ЦЕПОЧЕК

23	163	229	319	371	413	453	553	599	645	707	741	813	849	903
43	165	233	323	373	419	455	557	611	659	709	749	825	863	905
59	179	281	347	377	421	457	561	619	667	711	759	835	869	923
77	203	283	349	381	423	479	569	623	669	713	779	837	887	941
83	211	293	355	382	429	503	571	631	677	715	787	839	893	947
107	213	311	359	395	437	509	573	637	683	717	803	841	899	955
149	227	317	367	403	451	551	581	643	691	739	809	845	901	983

Пусть $d(r)$ обозначает количество решений n уравнения $l(n) = r$. В следующей таблице перечислено несколько первых значений этой функции согласно Фламменкампу.

r	$d(r)$	r	$d(r)$	r	$d(r)$	r	$d(r)$	r	$d(r)$
1	1	6	15	11	246	16	4490	21	90371
2	2	7	26	12	432	17	8170	22	165432
3	3	8	44	13	772	18	14866	23	303475
4	5	9	78	14	1382	19	27128	24	558275
5	9	10	136	15	2481	20	49544	25	1028508

Конечно, $d(r)$ должно быть возрастающей функцией от r, но не существует ясного способа доказательства этого кажущегося таким простым утверждения и тем более — определения асимптотического роста $d(r)$ для больших r.

Наиболее знаменитая проблема, связанная с аддитивными цепочками и все еще нерешенная, — *гипотеза Шольца-Брауэра*, гласящая, что

$$l(2^n - 1) \leq n - 1 + l(n). \tag{49}$$

Компьютерные вычисления показывают, что в действительности в (49) выполняется равенство для $1 \leq n \leq 24$. Вычисления, проведенные Э. Г. Тюрбером [*Discrete Math.* **16** (1976), 279–289], показали, что равенство справедливо также для $n = 32$. Множество исследований аддитивных цепочек посвящено попыткам доказать (49). Аддитивные цепочки для чисел $2^n - 1$, которые имеют так много единиц в их двоичном представлении, вызывают особый интерес, поскольку они являются наихудшим случаем для бинарного метода. Арнольд Шольц (Arnold Scholz), придумавший название "аддитивные цепочки", поставил в 1937 году задачу (49) [*Jahresbericht der deutschen Mathematiker-Vereinigung*, Abteilung II, **47** (1937), 41–42]; Альфред Брауэр (Alfred Brauer) в 1939 году доказал, что

$$l^*(2^n - 1) \leq n - 1 + l^*(n). \tag{50}$$

Теоремы Хансена показывают, что $l(n)$ может быть меньше, чем $l^*(n)$, так что, определенно, потребуется много работы, чтобы доказать или опровергнуть (49). В качестве шага в этом направлении Хансен определил концепцию l^0-*цепочки*, которая лежит "между" l- и l^*-цепочками. В l^0-цепочке некоторые элементы подчеркнуты; условие заключается в том, что $a_i = a_j + a_k$, где a_j — наибольший подчеркнутый элемент, меньший, чем a_i.

В качестве примера l^0-цепочки (конечно, не минимальной) рассмотрим

$$\underline{1}, \underline{2}, \underline{4}, 5, \underline{8}, 10, 12, \underline{18}. \tag{51}$$

Легко убедиться, что разность между каждым элементом и предшествующим ему подчеркнутым элементом принадлежит цепочке. Обозначим через $l^0(n)$ минимальную длину l^0-цепочки для n. Ясно, что $l(n) \le l^0(n) \le l^*(n)$.

Цепочка, построенная в теореме F, является l^0-цепочкой (см. упр. 22); следовательно, имеем $l^0(n) < l^*(n)$ для некоторого n. Неизвестно, выполняется ли равенство $l(n) = l^0(n)$ во всех случаях. Если оно истинно, то предположение Шольца-Брауэра должно быть справедливо вследствие еще одной теоремы Хансена.

Теорема G. $l^0(2^n - 1) \le n - 1 + l^0(n)$.

Доказательство. Пусть $1 = a_0, a_1, \ldots, a_r = n$ — l^0-цепочка минимальной длины для n и пусть $1 = b_0, b_1, \ldots, b_t = n$ — последовательность подчеркнутых элементов. (Можно считать, что n подчеркнуто.) Тогда l^0-цепочку для $2^n - 1$ можно получить следующим образом.

a) Включить $l^0(n) + 1$ чисел $2^{a_i} - 1$ для $0 \le i \le r$, подчеркнуть их тогда и только тогда, когда a_i будет подчеркнуто.

b) Включить числа $2^i(2^{b_j} - 1)$ для $0 \le j < t$ и $0 < i \le b_{j+1} - b_j$ и подчеркнуть их все. (Теперь всего имеется $b_1 - b_0 + \cdots + b_t - b_{t-1} = n - 1$ чисел.)

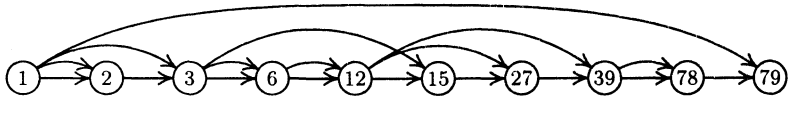
c) Рассортировать числа из (a) и (b) в порядке возрастания.

Можно легко проверить, что это дает l^0-цепочку: числа из (b) равны некоторым удвоенным элементам из (a) или (b); кроме того, данный элемент представляет собой предшествующий подчеркнутый элемент. Если $a_i = b_j + a_k$, где b_j — наибольший подчеркнутый элемент, меньший, чем a_i, то $a_k = a_i - b_j \le b_{j+1} - b_j$, поэтому $2^{a_k}(2^{b_j} - 1) = 2^{a_i} - 2^{a_k}$ в цепочке подчеркнут и предшествует $2^{a_i} - 1$. Поскольку $2^{a_i} - 1$ равно $(2^{a_i} - 2^{a_k}) + (2^{a_k} - 1)$, где оба эти значения содержатся в цепочке, имеем аддитивную цепочку со свойством l^0. ∎

Цепочка, соответствующая (51) и построенная в доказательстве теоремы G, такова:

$$\underline{1}, \underline{2}, \underline{3}, \underline{6}, \underline{12}, \underline{15}, \underline{30}, 31, \underline{60}, \underline{120}, \underline{240}, \underline{255}, \underline{510}, \underline{1020}, 1023, \underline{2040},$$
$$\underline{4080}, 4095, \underline{8160}, \underline{16320}, \underline{32640}, \underline{65280}, \underline{130560}, \underline{261120}, \underline{262143}.$$

Графическое представление. Аддитивная цепочка (1) естественным образом соответствует ориентированному графу, вершины которого помечены как a_i для $0 \le i \le r$ и в котором мы рисуем дуги от a_j к a_i и от a_k к a_i в качестве представления каждого шага $a_i = a_j + a_k$ из (2). Например, аддитивная цепочка 1, 2, 3, 6, 12, 15, 27, 39, 78, 79, которая может быть получена из рис. 15, соответствует ориентированному графу

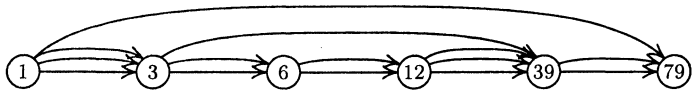


.

Если $a_i = a_j + a_k$ для более чем одной пары индексов (j, k), выбираем определенные j и k для нашего построения.

В целом, все вершины такого ориентированного графа, кроме первой, будут началом ровно двух дуг. Однако в действительности это не важное свойство графа, потому что оно маскирует тот факт, что различные аддитивные цепочки, по сути,

являются эквивалентными. Если вершина имеет выходную степень 1, то она используется только в одном последующем шаге, и поэтому подобный шаг представляет собой сумму $a_j+a_k+a_m$, которая может быть вычислена либо как $(a_j+a_k)+a_m$, либо как $a_j+(a_k+a_m)$, либо как $a_k+(a_j+a_m)$. Какой именно вариант выбран, не важно, но соглашения об аддитивных цепочках заставляют нас их различать. Можно избежать такой избыточности, удалив все вершины, выходная степень которых равна 1 и которые соединены дугами со своими предшественницей и преемницей. Например, приведенный выше граф должен принять вид



. (52)

Также можно удалить любую вершину, выходная степень которой равна 0, за исключением, естественно, конечной вершины a_r, поскольку она соответствует неиспользуемому шагу в аддитивной цепочке.

Таким образом, каждая аддитивная цепочка дает приводимый ориентированный граф, который содержит одну вершину-источник, помеченную как 1, и одну вершину-сток, помеченную как n. Каждая вершина, кроме источника, имеет входную степень ≥ 2, и каждая вершина, кроме стока, имеет выходную степень ≥ 2. И наоборот, любой такой ориентированный граф без ориентированных циклов соответствует минимум одной аддитивной цепочке, поскольку можно топологически рассортировать вершины и записать $d-1$ дополнительных шагов для каждой вершины входной степени $d>0$. Длина аддитивной цепочки без учета неиспользуемых шагов может быть определена в процессе просмотра приведенного графа. Она равна

$$(\text{число дуг})-(\text{число вершин})+1, \tag{53}$$

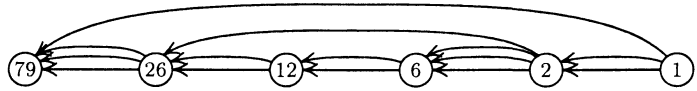
поскольку удаление вершины выходной степени 1 удаляет также одну дугу.

Две аддитивные цепочки *эквивалентны*, если они имеют один и тот же приведенный граф. Например, аддитивная цепочка 1, 2, 3, 6, 12, 15, 24, 39, 40, 79 эквивалентна цепочке, о которой шла речь в начале этого раздела, поскольку она также сводится к графу (52). Этот пример показывает, что незвездная цепочка может быть эквивалентна звездной цепочке. Аддитивная цепочка эквивалентна звездной цепочке тогда и только тогда, когда ее приведенный ориентированный граф может быть топологически рассортирован лишь одним-единственным образом.

Важное свойство такого графического представления аддитивных цепочек указано Н. Пиппенгером (N. Pippenger): метка каждой вершины в точности равна количеству ориентированных путей от источника к данной вершине. Таким образом, задача поиска оптимальной аддитивной цепочки для n эквивалентна задаче минимизации величины (53) по всем ориентированным графам, имеющим одну вершину-источник и одну вершину-сток и в точности n ориентированных путей от источника к стоку.

Такая характеристика имеет удивительное следствие в связи с симметричностью ориентированного графа. Если обратить направления всех дуг, источник и сток поменяются местами и получится другой ориентированный граф, соответствующий множеству аддитивных цепочек для того же значения n. Эти аддитивные цепочки имеют такую же длину (53), как и первоначальная цепочка. Например, если развернуть все стрелки в (52) справа налево и пометить вершины в соответствии с числом

путей от правой соседней вершины, получится



(54)

Одной из звездных цепочек, соответствующих этому приведенному ориентированному графу, является

$$1, 2, 4, 6, 12, 24, 26, 52, 78, 79;$$

ее можно назвать *дуальной* по отношению к исходной аддитивной цепочке.

В упр. 39 и 40 обсуждаются важные последствия такого графического представления и принципа дуальности.

УПРАЖНЕНИЯ

1. [*15*] Чему равно значение Z при завершении работы алгоритма A?

2. [*24*] Напишите MIX-программу для алгоритма A, вычисляющую $x^n \bmod w$ для данных целых n и x, где w — размер машинного слова. Считается, что MIX имеет бинарные операции SRB, JAE и др., описанные в разделе 4.5.2. Напишите еще одну программу, вычисляющую $x^n \bmod w$ последовательно (посредством многократного умножения на x), и сравните время работы этих программ.

▸ **3.** [*22*] Как вычисляется x^{975} при помощи (a) бинарного метода, (b) тернарного метода, (c) кватернарного (четверичного) метода и (d) метода множителя?

4. [*M20*] Найдите число n, для которого восьмеричный (2^3-арный) метод дает на десять умножений меньше, чем бинарный метод.

▸ **5.** [*24*] На рис. 14 показаны первые восемь уровней дерева степеней. В предположении, что k-й уровень дерева построен, его $(k+1)$-й уровень определяется следующим образом: берем каждый узел n на k-м уровне слева направо и присоединяем к нему снизу узлы

$$n+1,\ n+a_1,\ n+a_2,\ \ldots,\ n+a_{k-1}=2n$$

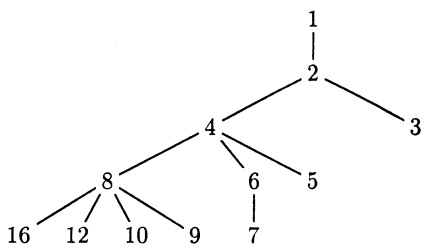
(именно в таком порядке), где $1, a_1, a_2, \ldots, a_{k-1}$ представляет собой путь от корня дерева до узла n. При этом, однако, устраняются все узлы, которые уже появлялись в дереве.

Разработайте эффективный алгоритм, который строит первые $r+1$ уровней дерева степеней. [*Указание.* Используйте два массива переменных, LINKU[j] и LINKR[j], для $0 \le j \le 2^r$; эти переменные указывают соответственно вверх и вправо для числа j в дереве.]

6. [*M26*] Если внести незначительные изменения в определение дерева степеней, данное в упр. 5, чтобы узлы, расположенные ниже n, присоединялись в порядке *убывания*

$$n+a_{k-1},\ \ldots,\ n+a_2,\ n+a_1,\ n+1,$$

а не возрастания, получится дерево, первые пять уровней которого выглядят как



Покажите, что это дерево дает метод вычисления x^n, который требует столько же умножений, сколько при двоичном методе. Поэтому модифицированное дерево, несмотря на то что оно строится почти так же, как и дерево степеней, дает более плохой метод вычисления степеней.

7. [*M21*] Докажите, что имеется бесконечно много значений n, для которых

a) метод множителя лучше бинарного метода;
b) бинарный метод лучше метода множителя;
c) метод дерева степеней лучше как бинарного метода, так и метода множителей.

(В данном случае понятие "лучше" означает вычисление x^n с использованием меньшего количества умножений.)

8. [*M21*] Докажите, что дерево степеней (упр. 5) никогда не дает для вычисления x^n большего количества умножений, чем бинарный метод.

▶ **9.** [*25*] Разработайте процедуру возведения в степень, аналогичную алгоритму A, но базирующуюся на $m = 2^e$. Ваш метод должен выполнять примерно $\lg n + \nu + m$ умножений, где ν — количество ненулевых цифр в m-арном представлении числа n.

10. [*10*] На рис. 15 показано дерево, которое указывает для каждого $n \le 100$ единственный путь вычисления x^n с минимально возможным количеством умножений. Каким образом это дерево можно расположить в памяти компьютера, используя только 100 ячеек памяти?

▶ **11.** [*M26*] Дерево на рис. 15 изображает аддитивные цепочки $a_0, a_1, \ldots, a_r$, у которых $l(a_i) = i$ для всех i в цепочке. Найдите все аддитивные цепочки для $n = 43$ и $n = 77$, имеющих это свойство. Покажите, что любое дерево, подобное изображенному на рис. 15, должно включать в себя либо путь 1, 2, 4, 8, 9, 17, 34, 43, 77, либо путь 1, 2, 4, 8, 9, 17, 34, 68, 77.

12. [*M10*] Нельзя ли расширить показанное на рис. 15 дерево до бесконечного дерева, которое давало бы правило вычисления x^n с наименьшим количеством умножений для всех положительных целых чисел n?

13. [*M21*] Найдите звездную цепочку длиной $A + 2$ для каждого из четырех случаев, перечисленных в теореме C (следовательно, теорема C остается справедливой и при замене l на l^*.)

14. [*M29*] Завершите доказательство теоремы C, показав, что (a) шаг $r - 1$ не является малым шагом; (b) $\lambda(a_{r-k})$ не может быть меньше, чем $m - 1$.

15. [*M43*] Напишите компьютерную программу для расширения теоремы C, определяющую все n, такие, что $l(n) = \lambda(n) + 3$, и все n, для которых $l^*(n) = \lambda(n) + 3$.

16. [*HM15*] Покажите, что теорему D трудно получить, используя бинарный метод; если $l^B(n)$ обозначает длину аддитивной цепочки для n, построенной бинарным SX-методом, отношение $l^B(n)/\lambda(n)$ не стремится к определенному пределу при $n \to \infty$.

17. [*M25*] Объясните, как найти интервалы $J_1, \ldots, J_h$, требующиеся в доказательстве леммы P.

18. [*HM24*] Пусть β — положительная константа. Покажите, что существует константа $\alpha < 2$, такая, что

$$\sum \binom{m+s}{t+v}\binom{t+v}{v}\beta^{2v}\binom{(m+s)^2}{t} < \alpha^m$$

для всех больших m, где сумма берется по всем s, t, v, удовлетворяющим (30).

19. [*M23*] "Мультимножество" подобно множеству, но оно может содержать один и тот же элемент конечное число раз. Если A и B — мультимножества, то мультимножества $A \uplus B$, $A \cup B$ и $A \cap B$ определяются следующим образом: элемент, встречающийся a раз в A и b раз в B, встречается в $A \uplus B$ $a+b$ раз, в $A \cup B$ — $\max(a,b)$ и в $A \cap B$ — $\min(a,b)$ раз. ("Множество" является мультимножеством, в котором нет элементов, встречающихся более одного раза; если A и B — множества, то множествами являются и $A \cup B$, и $A \cap B$ и данные здесь определения в этом случае согласуются с определениями объединения и пересечения множеств.)

a) Разложение натурального числа n на простые множители представляет собой мультимножество N, элементами которого являются простые числа, причем $\prod_{p \in N} = n$. Тот факт, что каждое натуральное число может быть единственным образом разложено на простые множители, дает взаимно однозначное соответствие между множеством натуральных чисел и конечными мультимножествами простых чисел, например если $n = 2^2 \cdot 3^3 \cdot 17$, соответствующим мультимножеством является $N = \{2, 2, 3, 3, 3, 17\}$. Если M и N — мультимножества, соответствующие m и n, то какие мультимножества соответствуют $\gcd(m,n)$, $\operatorname{lcm}(m,n)$ и mn?

b) Каждый нормированный полином $f(z)$ над полем комплексных чисел естественным образом соответствует мультимножеству его "корней"; имеем $f(z) = \prod_{\zeta \in F}(z-\zeta)$. Если $f(z)$ и $g(z)$ — полиномы, соответствующие конечным мультимножествам F и G комплексных чисел, то какие полиномы соответствуют $F \uplus G$, $F \cup G$ и $F \cap G$?

c) Найдите как можно больше интересных соотношений, выполняющихся при приложении к мультимножествам операций $\uplus$, $\cup$, $\cap$.

20. [*M20*] Какие последовательности S_i и M_{ij} $(0 \le i \le r,\ 0 \le j \le t)$ появляются при структурной декомпозиции Хансена звездных цепочек (a) типа 3 и (b) типа 5? (Шесть "типов" цепочек определены в доказательстве теоремы B.)

▸ **21.** [*M26*] (B. Хансен (W. Hansen).) Пусть q — некоторое натуральное число. Найдите значение n, такое, что $l(n) \le l^*(n) - q$.

22. [*M20*] Докажите, что аддитивная цепочка, построенная в доказательстве теоремы F, является l^0-цепочкой.

23. [*M20*] Докажите неравенство Брауэра (50).

▸ **24.** [*M22*] Обобщите доказательство теоремы G, чтобы показать, что

$$l^0((B^n-1)/(B-1)) \le (n-1)\,l^0(B) + l^0(n)$$

для любого целого $B > 1$; докажите также, что $l(2^{mn}-1) \le l(2^m-1) + mn - m + l^0(n)$.

25. [*20*] Пусть y является дробью $0 < y < 1$, выраженной в двоичной системе счисления как $y = (.d_1 \ldots d_k)_2$. Разработайте алгоритм для вычисления x^y с использованием операций умножения и извлечения квадратного корня.

▸ **26.** [*M25*] Разработайте эффективный алгоритм, вычисляющий n-е число Фибоначчи F_n по модулю m по данным большим целым числам n и m.

27. [*M21*] (A. Фламменкамп (A. Flammenkamp).) Чему равно наименьшее n, для которого каждая аддитивная цепочка содержит как минимум шесть малых шагов?

28. [*HM33*] (A. Шёнхаге (A. Schönhage).) Назначение данного упражнения — дать ясное и короткое доказательство того, что $l(n) \ge \lambda(n) + \lg \nu(n) - O(\log\log(\nu(n)+1))$.

a) Если $x = (x_k \ldots x_0.x_{-1} \ldots)_2$ и $y = (y_k \ldots y_0.y_{-1} \ldots)_2$ представляют собой действительные числа в двоичной записи, то введем обозначение $x \subseteq y$, если $x_j \le y_j$ для всех j. Приведите простое правило для построения наименьшего числа z, обладающего тем свойством, что из $x' \subseteq x$ и $y' \subseteq y$ вытекает, что $x' + y' \subseteq z$. Обозначив такое число как $x \nabla y$, докажите, что $\nu(x \nabla y) \le \nu(x) + \nu(y)$.

b) Дана произвольная аддитивная цепочка (11) с $r = l(n)$ и последовательность $d_0, d_1, \ldots, d_r$, определенная, как в (35). Определим последовательность $A_0, A_1, \ldots, A_r$ таким образом: $A_0 = 1$; если $a_i = 2a_{i-1}$, то $A_i = 2A_{i-1}$; в противном случае, если $a_i = a_j + a_k$ для некоторых $0 \le k \le j < i$, $A_i = A_{i-1} \nabla (A_{i-1}/2^{d_j - d_k})$. Докажите, что эта последовательность "покрывает" данную цепочку в том смысле, что $a_i \subseteq A_i$ для $0 \le i \le r$.

c) Пусть δ — натуральное число (которое будет выбрано позже). Назовем неудваивающий шаг $a_i = a_j + a_k$ небольшим шагом, если $d_j - d_k \ge \delta$, в противном случае назовем его близким шагом*. Пусть $B_0 = 1$; $B_i = 2B_{i-1}$, если $a_i = 2a_{i-1}$; $B_i = B_{i-1} \nabla (B_{i-1}/2^{d_j - d_k})$, если $a_i = a_j + a_k$ — небольшой шаг; $B_i = \rho(2B_{i-1})$ — в противном случае, где $\rho(x)$ — наименьшее число y, такое, что $x/2^e \subseteq y$ для $0 \le e \le \delta$. Покажите, что $A_i \subseteq B_i$ и $\nu(B_i) \le (1 + \delta c_i) 2^{b_i}$ для $0 \le i \le r$, где b_i и c_i означают соответственно число небольших шагов и число близких шагов $\le i$. [*Указание.* Покажите, что единицы в B_i появляются в последовательных блоках размером $\ge 1 + \delta c_i$.]

d) Теперь $l(n) = r = b_r + c_r + d_r$ и $\nu(n) \le \nu(B_r) \le (1 + \delta c_r) 2^{b_r}$. Объясните, как выбрать δ, чтобы получить неравенство, упомянутое в начале этого упражнения. [*Указание.* См. (16); обратите внимание на то, что $n \le 2^r \alpha^{b_r}$ для некоторого $\alpha < 1$, зависящего от δ.]

29. [*M49*] Справедливо ли $\nu(n) \le 2^{l(n) - \lambda(n)}$ для всех натуральных чисел n? (Если справедливо, то получим нижнюю границу $l(2^n - 1) \ge n - 1 + \lceil \lg n \rceil$; см. (17) и (49).)

30. [*20*] В *цепочке со сложением и вычитанием* вместо правила (2) имеется правило $a_i = a_j \pm a_k$. Воображаемый компьютер, описанный в тексте раздела, оснащается новой операцией — SUB. (На практике это соответствует тому, что при вычислении x^n используются и умножение, и деление.) Найдите такую цепочку для некоторого n, имеющую менее $l(n)$ шагов.

31. [*M46*] (Д. Г. Лехмер (D. H. Lehmer).) Исследуйте задачу минимизации $\epsilon q + (r - q)$ в аддитивной цепочке (1), где q — число простых шагов, в которых $a_i = a_{i-1} + 1$, при данном малом положительном весе ϵ. (Эта задача ближе к реальности для многих вычислений x^n, если умножение на x проще, чем общее умножение; см. приложения в разделе 4.6.2.)

32. [*M30*] (Э. Ч. Яо (A. C. Yao), Ф. Ф. Яо (F. F. Yao) и Р. Л. Грэхем (R. L. Graham).) Назначим "цену" $a_j a_k$ для каждого шага $a_i = a_j + a_k$ аддитивной цепочки (1). Покажите, что бинарный метод "слева направо" приводит к цепочке минимальной общей стоимости для всех натуральных n.

33. [*15*] Сколько аддитивных цепочек длиной 9 имеет (52) в качестве приведенного ориентированного графа?

34. [*M23*] Бинарная аддитивная цепочка для $n = 2^{e_0} + \cdots + 2^{e_t}$, когда $e_0 > \cdots > e_t \ge 0$, представляет собой $1, 2, \ldots, 2^{e_0 - e_1}, 2^{e_0 - e_1} + 1, \ldots, 2^{e_0 - e_2} + 2^{e_1 - e_2}, 2^{e_0 - e_2} + 2^{e_1 - e_2} + 1, \ldots, n$. Это соответствует методу "S и X", описанному в начале этого раздела, в то время как алгоритм A соответствует аддитивной цепочке, полученной посредством сортировки двух последовательностей — $(1, 2, 4, \ldots, 2^{e_0})$ и $(2^{e_t - 1} + 2^{e_t}, 2^{e_t - 2} + 2^{e_t - 1} + 2^{e_t}, \ldots, n)$ — в порядке возрастания. Докажите или опровергните следующее: каждая из этих аддитивных цепочек дуальна по отношению к другой.

35. [*M27*] Как много аддитивных цепочек без шагов, которые не используются, эквивалентны каждой из аддитивных цепочек, обсуждавшихся в упр. 34, если $e_0 > e_1 + 1$?

▸ **36.** [*25*] (Э. Г. Штраус (E. G. Straus).) Найдите способ вычисления обобщенного *мононома* $x_1^{n_1} x_2^{n_2} \ldots x_m^{n_m}$ за не более чем $2\lambda(\max(n_1, n_2, \ldots, n_m)) + 2^m - m - 1$ операций умножения.

---

* В оригинале — "baby step" и "close step" соответственно. — *Прим. перев.*

37. [*HM30*] (Э. Ч. Яо.) Пусть $l(n_1, \ldots, n_m)$ — длина наикратчайшей аддитивной цепочки, которая содержит m чисел $n_1 < \cdots < n_m$. Докажите, что $l(n_1, \ldots, n_m) \le \lambda(n_m) + m\lambda(n_m)/\lambda\lambda(n_m) + O(\lambda(n_m)\lambda\lambda\lambda(n_m)/\lambda\lambda(n_m)^2)$, тем самым обобщая (25).

38. [*M47*] Чему равно асимптотическое значение $l(1, 4, 9, \ldots, m^2) - m$ при $m \to \infty$ в обозначениях из упр. 37?

▸ **39.** [*M25*] (Х. Оливос (J. Olivos), 1979.) Пусть $l([n_1, n_2, \ldots, n_m])$ — минимальное количество умножений, требующихся для вычисления мононома $x_1^{n_1} x_2^{n_2} \ldots x_m^{n_m}$ в смысле упр. 36, где каждое n_i — натуральное. Докажите, что эта задача эквивалентна задаче из упр. 37, показав, что $l([n_1, n_2, \ldots, n_m]) = l(n_1, n_2, \ldots, n_m) + m - 1$. [*Указание.* Обобщите ориентированный граф, построив рассмотренный граф с более чем одной исходной вершиной.]

▸ **40.** [*M21*] (Х. Оливос.) Обобщая метод множителя и теорему F, докажите, что

$$l(m_1 n_1 + \cdots + m_t n_t) \le l(m_1, \ldots, m_t) + l(n_1, \ldots, n_t) + t - 1,$$

где $l(n_1, \ldots, n_t)$ определено в упр. 37.

41. [*M40*] (П. Давни (P. Downey), Б. Леон (B. Leong) и Р. Сети (R. Sethi).) Пусть G — связный граф с n вершинами $\{1, \ldots, n\}$ и m ребрами, где ребра объединяют u_j с v_j для $1 \le j \le m$. Докажите, что $l(1, 2, \ldots, 2^{An}, 2^{Au_1} + 2^{Av_1} + 1, \ldots, 2^{Au_m} + 2^{Av_m} + 1) = An + m + k$ для всех достаточно больших A, где k — минимальное число вершин в покрытии вершин для G (т. е. множество, содержащее либо u_j, либо v_j для $1 \le j \le m$).

42. [*M50*] Справедливо ли $l(2^n - 1) \le n - 1 + l(n)$ для всех натуральных n? Всегда ли выполняется равенство? Выполняется ли $l(n) = l^0(n)$?

4.6.4. Вычисление полиномов

Так как нам известен эффективный метод вычисления полинома специального вида x^n, рассмотрим общую задачу вычисления полинома n-й степени

$$u(x) = u_n x^n + u_{n-1} x^{n-1} + \cdots + u_1 x + u_0, \qquad u_n \ne 0, \tag{1}$$

для определенных значений x. Такая задача часто возникает на практике.

Затем сконцентрируем внимание на минимизации числа операций, требуемых для вычисления полиномов на компьютере, в идеальном случае предполагая, что все арифметические операции точны. Чаще всего полиномы вычисляются с использованием арифметики с плавающей точкой, которая не является точной, и различные схемы для оценки обычно дают различные ответы. Численный анализ достигаемой точности зависит от коэффициентов рассматриваемого полинома, но в данной книге он не проводится; читателю следует внимательно исследовать точность любых вычислений, полученных с использованием арифметики с плавающей точкой. В большинстве случаев будут описаны методы, вполне удовлетворительные с точки зрения численного анализа, но будет также рассмотрено достаточно много плохих примеров. [См. работу Webb Miller, *SICOMP* **4** (1975), 105–107, в которой приведены обзор литературы по устойчивости быстрых вычислений полинома и доказательство, что определенные виды численной устойчивости не могут быть гарантированными для некоторых семейств быстродействующих алгоритмов.]

Во всем этом разделе будем считать переменную x просто числом. Но важно помнить, что большинство рассматриваемых методов действительны также, когда переменные являются более сложными объектами, такими как числа с многократной точностью, полиномы и матрицы. В подобных случаях эффективная формула

приводит даже к большему выигрышу, в особенности, когда можно уменьшать количество операций умножения.

Начинающий программист, скорее всего, вычислит полином (1) способом, который прямо соответствует его традиционному виду: сначала вычислит $u_n x^n$, затем — $u_{n-1}x^{n-1}$, ..., $u_1 x$ и наконец суммирует все члены (1). Но даже если использовать эффективные методы из раздела 4.6.3 для вычисления степеней x этими методами, результирующее вычисление излишне медленно, кроме случая, когда почти все коэффициенты u_k равны нулю. Если не все коэффициенты равны нулю, то очевидной альтернативой должно быть вычисление (1) справа налево посредством вычисления значений x^k и $u_k x^k + \cdots + u_0$ для $k = 1, \ldots, n$. Такой процесс включает $2n - 1$ операций умножения и n операций сложения и, возможно, потребует дополнительных команд для сохранения промежуточных результатов в памяти и их извлечения из памяти.

Правило Горнера. Одной из первых забот начинающего программиста обычно является изучение элегантного способа перестройки этих вычислений следующим образом:

$$u(x) = \big(\ldots(u_n x + u_{n-1})x + \cdots\big)x + u_0 \tag{2}$$

(начать с u_n, умножить на x, добавить u_{n-1}, умножить на x, ..., умножить на x, добавить u_0). Эту модель вычисления обычно называют "правило Горнера" (либо "схема Горнера". — Прим. ред.). Мы уже знаем, как его использовать в связи с заменой основания системы счисления в разделе 4.4. Полный процесс требует n умножений и n сложений минус одно сложение для каждого нулевого коэффициента. Кроме того, нет необходимости запоминать промежуточные результаты, так как каждая величина, появляющаяся в процессе вычислений, немедленно используется после ее получения.

В. Дж. Горнер (W. G. Horner) предложил это правило в начале 19 века [*Philosophical Transactions*, Royal Society of London **109** (1819), 308–335] в связи с процедурой вычисления корней полинома. Известность последнего метода [см. J. L. Coolidge, *Mathematics of Great Amateurs* (Oxford, 1949), глава 15] объясняет то обстоятельство, что имя Горнера связывается с формулой (2), но в действительности Исаак Ньютон использовал такую же идею более чем на 150 лет раньше. Например, в общеизвестной работе *De Analysi per Æquationes Infinitas*, написанной в 1669 году, Ньютон записал

$$\overline{\overline{y - .4 \times y : + 5 \times y : - 12 \times y : + 17}}$$

для полинома $y^4 - 4y^3 + 5y^2 - 12y + 17$, что поясняет, почему позже метод приобрел известность как метод Ньютона нахождения корня. Тут ясно видно идею формулы (2), так как он часто обозначал группирование горизонтальными линиями и двоеточием, а не круглыми скобками. Ньютон использовал эту идею на протяжении нескольких лет в неопубликованных заметках. [См. *The Mathematical Papers of Isaac Newton*, edited by D. T. Whiteside, **1** (1967), 490, 531; **2** (1968), 222.] Независимо метод, эквивалентный правилу Горнера, фактически использовался в 13 веке китайцем Чин Чи Шао (Ch'in Chiu Shao) [см. Y. Mikami, *The Development of Mathematics in China and Japan* (1913), 73–77].

Предложим несколько обобщений правила Горнера. Сначала рассмотрим вычисление $u(z)$, когда z — комплексное число, в то время как коэффициенты u_k —

действительные числа. В частности, когда $z = e^{i\theta} = \cos\theta + i\sin\theta$, полином $u(z)$ является, по существу, двумя рядами Фурье (на самом деле — суммами. — *Прим. ред.*)

$$(u_0 + u_1\cos\theta + \cdots + u_n\cos n\theta) \;+\; i(u_1\sin\theta + \cdots + u_n\sin n\theta).$$

Комплексное сложение и умножение, очевидно, может быть сведено к последовательности обычных операций над вещественными числами.

вещественное + комплексное	требует	1 сложения
комплексное + комплексное	требует	2 сложений
вещественное × комплексное	требует	2 умножений
комплексное × комплексное	требует	4 умножений, 2 сложений
	или	3 умножений, 5 сложений

(См. упр. 41. Вычитание рассматривается здесь как эквивалент сложения.) Следовательно, правило Горнера (2) использует каждые $4n-2$ умножений и $3n-2$ сложений или $3n-1$ умножений и $6n-5$ сложений для вычисления $u(z)$, когда $z = x + iy$ — комплексное число. В действительности без $2n-4$ сложений можно обойтись, так как каждый раз выполняется умножение на то же число z. Предлагаем альтернативную процедуру вычисления $u(x+iy)$:

$$\begin{aligned} &a_1 = u_n, && b_1 = u_{n-1}, && r = x + x, \quad s = x^2 + y^2; \\ &a_j = b_{j-1} + ra_{j-1}, && b_j = u_{n-j} - sa_{j-1}, && 1 < j \le n. \end{aligned} \tag{3}$$

Тогда легко доказать по индукции, что $u(z) = za_n + b_n$. Эта схема [*BIT* **5** (1965), 142; также см. G. Goertzel, *AMM* **65** (1958), 34–35] требует только $2n+2$ умножений и $2n+1$ сложений, так как она является усовершенствованным правилом Горнера, когда $n \ge 3$. Для рядов Фурье, когда $z = e^{i\theta}$, имеем $s = 1$, так что число умножений уменьшается до $n+1$. Отсюда мораль: хороший программист не станет неразборчиво использовать встроенные характеристики комплексной арифметики языков программирования высокого уровня.

Рассмотрим процесс деления полинома $u(x)$ на $x - x_0$, используя алгоритм 4.6.1D, чтобы получить $u(x) = (x - x_0)q(x) + r(x)$, здесь $\deg(r) < 1$. В этом случае $r(x)$ — константа, не зависящая от x и $u(x_0) = 0 \cdot q(x_0) + r = r$. Проверка этого процесса деления позволяет обнаружить, что вычисления, по существу, такие же, как и правило Горнера для вычисления $u(x_0)$. Подобным образом, если мы делим $u(z)$ на полином $(z - z_0)(z - \bar{z}_0) = z^2 - 2x_0z + x_0^2 + y_0^2$, результат вычисления оказывается эквивалентным (3). Получаем $u(z) = (z - z_0)(z - \bar{z}_0)q(z) + a_nz + b_n$; отсюда $u(z_0) = a_nz_0 + b_n$.

Вообще, если разделить $u(x)$ на $f(x)$, получится $u(x) = f(x)q(x) + r(x)$, а если $f(x_0) = 0$, то $u(x_0) = r(x_0)$; это наблюдение приводит к дальнейшим обобщениям правила Горнера. Например, можно положить $f(x) = x^2 - x_0^2$, что дает правило Горнера "второго порядка":

$$\begin{aligned} u(x) = {} & \big(\ldots(u_{2\lfloor n/2\rfloor}x^2 + u_{2\lfloor n/2\rfloor-2})x^2 + \cdots\big)x^2 + u_0 \\ & + \big(\ldots(u_{2\lceil n/2\rceil-1}x^2 + u_{2\lceil n/2\rceil-3})x^2 + \cdots\big)x^2 + u_1\big)x. \end{aligned} \tag{4}$$

В правиле второго порядка используются $n+1$ умножение и n сложений (см. упр. 5), что с этой точки зрения оно не улучшает правило Горнера. Но существуют по

крайней мере два обстоятельства, в которых (4) полезно: если необходимо вычислить $u(x)$ и $u(-x)$ и этот подход дает $u(-x)$ только с помощью одной дополнительной операции сложения, два значения можно получить почти так же легко, как и одно. Кроме того, если компьютер позволяет выполнять параллельные вычисления, то имеется возможность обе строки (4) вычислять независимо, так что экономится около половины времени счета.

Если компьютер допускает параллельное вычисление k арифметических операций одновременно, то можно применять правило Горнера "k-го порядка" (положив $f(x) = x^k - x_0^k$). Другой привлекательный метод для параллельного вычисления предложил Дж. Эстрин (G. Estrin) [*Proc. Western Joint Computing Conf.* **17** (1960), 33–40]; для $n = 7$ метод Эстрина имеет следующий вид.

Процессор 1	Процессор 2	Процессор 3	Процессор 4	Процессор 5
$a_1 = u_7x + u_6$	$b_1 = u_5x + u_4$	$c_1 = u_3x + u_2$	$d_1 = u_1x + u_0$	x^2
$a_2 = a_1x^2 + b_1$		$c_2 = c_1x^2 + d_1$		x^4
$a_3 = a_2x^4 + c_2$				

Здесь $a_3 = u(x)$. Тем не менее интересный анализ В. Ш. Дорна (W. S. Dorn) [*IBM J. Res. and Devel.* **6** (1962), 239–245] показал, что эти методы в действительности не могут быть улучшением правила второго порядка, если каждая арифметическая операция должна обращаться к памяти, которая сообщается только с одним процессором.

Табулирование значений полинома. Чтобы вычислить полином n-й степени на множестве значений арифметической прогрессии (т. е. $u(x_0), u(x_0+h), u(x_0+2h), \ldots$), процесс можно свести только к сложению после первых нескольких шагов. Так, если начать с любой последовательности чисел $(\alpha_0, \alpha_1, \ldots, \alpha_n)$ и применить преобразование

$$\alpha_0 \leftarrow \alpha_0 + \alpha_1, \quad \alpha_1 \leftarrow \alpha_1 + \alpha_2, \quad \ldots, \quad \alpha_{n-1} \leftarrow \alpha_{n-1} + \alpha_n, \tag{5}$$

получим, что k применений (5) дают

$$\alpha_j^{(k)} = \binom{k}{0}\beta_j + \binom{k}{1}\beta_{j+1} + \binom{k}{2}\beta_{j+2} + \cdots, \qquad 0 \le j \le n,$$

где β_j — начальное значение α_j и $\beta_j = 0$ для $j > n$. В частности,

$$\alpha_0^{(k)} = \binom{k}{0}\beta_0 + \binom{k}{1}\beta_1 + \cdots + \binom{k}{n}\beta_n \tag{6}$$

представляет собой полином степени n от k. Правильно выбирая β_j, как показано в упр. 7, можно устроить так, что величина $\alpha_0^{(k)}$ будет требуемым значением $u(x_0+kh)$ для всех k. Другими словами, каждое выполнение n сложений в (5) будет давать следующее значение данного полинома.

Предостережение. Ошибки округления могут накапливаться после многочисленных повторений (5), и ошибка в α_j приводит к ошибке в коэффициентах $x^0, \ldots, x^j$ вычисляемого полинома. Поэтому значения α_j следовало бы "освежить" после большого числа итераций.

Производные и замена переменной. Иногда необходимо найти коэффициенты $u(x+x_0)$ с заданными постоянной x_0 и коэффициентами $u(x)$. Например, если $u(x)=3x^2+2x-1$, то $u(x-2)=3x^2-10x+7$. Это аналог проблемы преобразования системы счисления, преобразующей основание x в основание $x+2$. По теореме Тейлора новые коэффициенты заданы производными $u(x)$ в точке $x=x_0$, т. е.

$$u(x+x_0)=u(x_0)+u'(x_0)x+\big(u''(x_0)/2!\big)x^2+\cdots+\big(u^{(n)}(x_0)/n!\big)x^n, \qquad (7)$$

так что задача эквивалентна вычислению $u(x)$ и всех ее производных.

Если записать $u(x)=q(x)(x-x_0)+r$, то $u(x+x_0)=q(x+x_0)x+r$; в таком случае r есть постоянный коэффициент $u(x+x_0)$ и проблема сводится к нахождению коэффициентов $q(x+x_0)$, где $q(x)$ — известный полином степени $n-1$. Таким образом, предлагаем следующий алгоритм.

H1. Присвоить $v_j \leftarrow u_j$ для $0\le j\le n$.

H2. Для $k=0,\ 1,\ \ldots,\ n-1$ (в таком порядке) присвоить $v_j \leftarrow v_j+x_0v_{j+1}$ для $j=n-1,\ \ldots,\ k+1,\ k$ (в таком порядке). ▮

По окончании шага H2 получим $u(x+x_0)=v_nx^n+\cdots+v_1x+v_0$. Эта процедура была главной частью метода Горнера нахождения корней, и когда $k=0$, она является точным правилом (2) вычисления $u(x_0)$.

Метод Горнера требует $(n^2+n)/2$ умножений и $(n^2+n)/2$ сложений, но заметим, что, если $x_0=1$, все умножения опускаются. К счастью, можно свести общую задачу к случаю, когда $x_0=1$, введя сравнительно небольшое число умножений и делений.

S1. Вычислить и запомнить значения $x_0^2,\ \ldots,\ x_0^n$.

S2. Присвоить $v_j \leftarrow u_jx_0^j$ для $0\le j\le n$. (Сейчас $v(x)=u(x_0x)$.)

S3. Выполнить шаг H2, только с $x_0=1$. (Сейчас $v(x)=u\big(x_0(x+1)\big)=u(x_0x+x_0)$.)

S4. Присвоить $v_j \leftarrow v_j/x_0^j$ для $0<j\le n$. (Сейчас $v(x)=u(x+x_0)$ как и требовалось.) ▮

Эта идея, предложенная М. Шо (M. Shaw) и Ж. Ф. Траубом (J. F. Traub) [*JACM* **21** (1974), 161–167], обеспечивает такое же количество сложений и такую же численую устойчивость, как и метод Горнера, но при этом необходимы только $2n-1$ умножений и $n-1$ делений, так как $v_n=u_n$. Приблизительно $\frac{1}{2}n$ этих умножений можно по очереди избежать (см. упр. 6).

Шо и Трауб отметили, что существует дополнительный способ экономии времени счета, если нужны только несколько первых или последних производных. Например, если нужно вычислить только $u(x)$ и $u'(x)$, то для выполнения следующего алгоритма потребуется $2n-1$ сложений и около $n+\sqrt{2n}$ умножений и/или делений.

D1. Вычислить и запомнить значения $x^2,\ x^3,\ \ldots,\ x^t,\ x^{2t}$, где $t=\lceil\sqrt{n/2}\rceil$.

D2. Присвоить $v_j \leftarrow u_jx^{f(j)}$ для $0\le j\le n$, где $f(j)=t-1-\big((n-1-j)\bmod 2t\big)$ для $0\le j<n$, а $f(n)=t$.

D3. Присвоить $v_j \leftarrow v_j+v_{j+1}x^{g(j)}$ для $j=n-1,\ \ldots,\ 1,\ 0$; здесь $g(j)=2t$, когда $n-1-j$ — положительный множитель $2t$, иначе $g(j)=0$ и нет необходимости умножать на $x^{g(j)}$.

D4. Присвоить $v_j \leftarrow v_j + v_{j+1}x^{g(j)}$ для $j = n-1, \ldots, 2, 1$. Теперь $v_0/x^{f(0)} = u(x)$ и $v_1/x^{f(1)} = u'(x)$. ▮

Адаптация коэффициентов. Возвратимся к нашей первоначальной проблеме вычисления заданного полинома $u(x)$ для "случайных" значений x настолько быстро, насколько это возможно. Важность данной проблемы отчасти является следствием того факта, что стандартные функции, такие как $\sin x$, $\cos x$, e^x и т. д., обычно вычисляются подпрограммами, которые опираются на вычисление определенных полиномов. Поскольку такие полиномы часто вычисляются, желательно найти возможно наиболее быстрый метод вычислений.

Произвольные полиномы степени пять и выше можно вычислить, выполнив меньше операций, чем требуется по правилу Горнера, если сначала "адаптировать" коэффициенты $u_0, u_1, \ldots, u_n$. Как будет пояснено ниже, процесс адаптации может включать в себя уйму работы, но предварительное вычисление не является ненужной потерей времени, так как оно должно производиться только один раз, а полином вычисляется многократно. Примеры "адаптирования" полиномов для выполнения стандартных функций приводятся в работе В. Я. Пана, *СССР Вычисл. матем. и матем. физика* **2** (1963), 137–146.

Простейший случай, для которого адаптация коэффициентов полезна, встречается при использовании полиномов четвертой степени:

$$u(x) = u_4x^4 + u_3x^3 + u_2x^2 + u_1x + u_0, \qquad u_4 \neq 0. \tag{8}$$

Эту формулу можно переписать в виде, впервые предложенном Т. С. Моцкиным (T. S. Motzkin),

$$y = (x + \alpha_0)x + \alpha_1, \qquad u(x) = \big((y + x + \alpha_2)y + \alpha_3\big)\alpha_4, \tag{9}$$

для соответственно "адаптированных" коэффициентов α_0, α_1, α_2, α_3, α_4. Вычисления по этой схеме включают три операции умножения, пять операций сложения и (на машине с одним сумматором, подобной машине MIX) одно указание запомнить промежуточный результат y во временном запоминающем устройстве. По сравнению с правилом Горнера в этой схеме мы заменили умножение сложением и, возможно, командой запоминания. Даже эта сравнительно малая экономия имеет смысл, если полином вычисляется часто. (Конечно, если время умножения сравнимо со временем сложения, (9) не дает улучшения: для вычисления обычных полиномов четвертой степени всегда требуется по крайней мере восемь арифметических операций.)

Сравнив коэффициенты в (8) и (9), получим формулы для вычисления α_j, выраженных через u_k:

$$\begin{aligned} &\alpha_0 = \tfrac{1}{2}(u_3/u_4 - 1), \qquad \beta = u_2/u_4 - \alpha_0(\alpha_0 + 1), \qquad \alpha_1 = u_1/u_4 - \alpha_0\beta, \\ &\alpha_2 = \beta - 2\alpha_1, \qquad \alpha_3 = u_0/u_4 - \alpha_1(\alpha_1 + \alpha_2), \qquad \alpha_4 = u_4. \end{aligned} \tag{10}$$

Подобная схема вычисления полинома четвертой степени за такое же число шагов, как (9), предлагается в упр. 18; этот альтернативный метод дает в некоторых случаях большую точность, чем (9), хотя он уступает в точности другим.

Полиномы, встречающиеся на практике, часто имеют достаточно малый старший коэффициент, поэтому деление на u_4 в (10) ведет к неустойчивости. В таком

случае предпочтительней на первом шаге заменить x на $|u_4|^{1/4}x$, сводя (8) к полиному со старшим коэффициентом, равным ± 1. Подобное преобразование применимо к полиномам более высоких степеней. Эта идея предложена Ч. Т. Файком (C. T. Fike) [*CACM* **10** (1967), 175–178]; он рассмотрел несколько интересных примеров.

Любой полином пятой степени можно вычислить, используя четыре умножения, шесть сложений и одно запоминание согласно правилу $u(x) = U(x)x + u_0$, где $U(x) = u_5x^4 + u_4x^3 + u_3x^2 + u_2x + u_1$ вычисляется, как и в (9). Кроме того, можно произвести вычисление, выполнив четыре умножения, пять сложений и три запоминания, если вычисления осуществлялись по формуле

$$y = (x + \alpha_0)^2, \qquad u(x) = \big(((y + \alpha_1)y + \alpha_2)(x + \alpha_3) + \alpha_4\big)\alpha_5. \tag{11}$$

Для определения α_j теперь требуется решить кубическое уравнение (см. упр. 19).

На многих компьютерах количество требуемых формулами (11) операций "запомнить" меньше трех; например, мы, возможно, сумеем вычислить $(x + \alpha_0)^2$, не запоминая $x + \alpha_0$. Действительно, большинство современных компьютеров имеет более одного арифметического регистра для вычислений с плавающей точкой, так что вполне можно обойтись без запоминания. Поскольку различные компьютеры для выполнения арифметических действий предоставляют большие возможности, в этом разделе следует учитывать только арифметические операции, а не операции запоминания и загрузки сумматора. Вычислительные схемы обычно простым способом можно адаптировать к любому конкретному компьютеру так, что понадобится несколько дополнительных операций; с другой стороны, следует помнить, что эти операции могут свести на нет экономию одного или двух умножений, в особенности если программа компилируется машиной не оптимально.

Полином шестой степени $u(x) = u_6x^6 + \cdots + u_1x + u_0$ всегда можно вычислить, используя четыре операции умножения и семь операций сложений, по схеме

$$\begin{gathered} z = (x + \alpha_0)x + \alpha_1, \qquad w = (x + \alpha_2)z + \alpha_3, \\ u(x) = \big((w + z + \alpha_4)w + \alpha_5\big)\alpha_6. \end{gathered} \tag{12}$$

[D. E. Knuth, *CACM* **5** (1962), 595–599.] Такая схема позволяет избежать двух из шести умножений, требуемых по правилу Горнера. Здесь снова необходимо решить кубическое уравнение: так как $\alpha_6 = u_6$, можно предположить, что $u_6 = 1$. При этом предположении пусть

$$\begin{gathered} \beta_1 = (u_5 - 1)/2, \qquad \beta_2 = u_4 - \beta_1(\beta_1 + 1), \\ \beta_3 = u_3 - \beta_1\beta_2, \qquad \beta_4 = \beta_1 - \beta_2, \qquad \beta_5 = u_2 - \beta_1\beta_3. \end{gathered}$$

Допустим, что β_6 — вещественный корень кубического уравнения

$$2y^3 + (2\beta_4 - \beta_2 + 1)y^2 + (2\beta_5 - \beta_2\beta_4 - \beta_3)y + (u_1 - \beta_2\beta_5) = 0. \tag{13}$$

(Это уравнение всегда имеет вещественный корень, так как левая часть полинома стремится к $+\infty$ для больших положительных значений y и к $-\infty$ — для больших отрицательных значений y; оно должно принимать значение "нуль" где-то посередине.) Если сейчас определить

$$\beta_7 = \beta_6^2 + \beta_4\beta_6 + \beta_5, \qquad \beta_8 = \beta_3 - \beta_6 - \beta_7,$$

то окончательно получим

$$\alpha_0 = \beta_2 - 2\beta_6, \qquad \alpha_2 = \beta_1 - \alpha_0, \qquad \alpha_1 = \beta_6 - \alpha_0\alpha_2,$$
$$\alpha_3 = \beta_7 - \alpha_1\alpha_2, \qquad \alpha_4 = \beta_8 - \beta_7 - \alpha_1, \qquad \alpha_5 = u_0 - \beta_7\beta_8. \tag{14}$$

Эту процедуру можно пояснить на следующем примере: предположим, нужно вычислить $x^6 + 13x^5 + 49x^4 + 33x^3 - 61x^2 - 37x + 3$. Получим $\alpha_6 = 1$, $\beta_1 = 6$, $\beta_2 = 7$, $\beta_3 = -9$, $\beta_4 = -1$, $\beta_5 = -7$ и таким образом придем к кубическому уравнению

$$2y^3 - 8y^2 + 2y + 12 = 0. \tag{15}$$

Это уравнение имеет корень $\beta_6 = 2$. Находим

$$\beta_7 = -5, \qquad \beta_8 = -6,$$
$$\alpha_0 = 3, \quad \alpha_2 = 3, \quad \alpha_1 = -7, \quad \alpha_3 = 16, \quad \alpha_4 = 6, \quad \alpha_5 = -27.$$

Следовательно, окончательная схема такова:

$$z = (x + 3)x - 7, \qquad w = (x + 3)z + 16, \qquad u(x) = (w + z + 6)w - 27.$$

Благодаря явному совпадению дважды появляющихся величин $x + 3$ можно найти метод, использующий три умножения и шесть сложений.

Другие методы подхода к решению уравнения шестой степени предложены В. Я. Паном [*Проблемы кибернетики* **5** (1961), 17–29]. Один из них требует на одну операцию сложения больше, но включает только рациональные операции на первых шагах; однако кубическое уравнение необходимо решать. Можно записать процесс решения в следующем виде:

$$z = (x + \alpha_0)x + \alpha_1, \qquad w = z + x + \alpha_2,$$
$$u(x) = \big(((z - x + \alpha_3)w + \alpha_4)z + \alpha_5\big)\alpha_6. \tag{16}$$

Для определения α_j мы снова один раз делим полином на $u_6 = \alpha_6$ и таким образом приходим к нормированному многочлену $u(x)$. Затем можно проверить, что $\alpha_0 = u_5/3$ и

$$\alpha_1 = (u_1 - \alpha_0 u_2 + \alpha_0^2 u_3 - \alpha_0^3 u_4 + 2\alpha_0^5)/(u_3 - 2\alpha_0 u_4 + 5\alpha_0^3). \tag{17}$$

Заметим, что для метода Пана требуется, чтобы делитель в (17) не обращался в нуль. Другими словами, (16) можно использовать только тогда, когда

$$27u_3u_6^2 - 18u_6u_5u_4 + 5u_5^3 \neq 0; \tag{18}$$

в действительности это значение не может быть таким малым, поскольку α_1 станет слишком большим. После того как α_1 будет определено, остальные α_j можно определить из уравнений

$$\beta_1 = 2\alpha_0, \qquad \beta_2 = u_4 - \alpha_0\beta_1 - \alpha_1,$$
$$\beta_3 = u_3 - \alpha_0\beta_2 - \alpha_1\beta_1, \qquad \beta_4 = u_2 - \alpha_0\beta_3 - \alpha_1\beta_2,$$
$$\alpha_3 = \tfrac{1}{2}\big(\beta_3 - (\alpha_0 - 1)\beta_2 + (\alpha_0 - 1)(\alpha_0^2 - 1)\big) - \alpha_1,$$
$$\alpha_2 = \beta_2 - (\alpha_0^2 - 1) - \alpha_3 - 2\alpha_1, \qquad \alpha_4 = \beta_4 - (\alpha_2 + \alpha_1)(\alpha_3 + \alpha_1),$$
$$\alpha_5 = u_0 - \alpha_1\beta_4. \tag{19}$$

Мы детально обсуждаем случаи степеней $n = 4, 5, 6$, поскольку такие значения n чаще встречаются в приложениях. Сейчас рассмотрим общую схему вычисления полиномов n-й степени, метод, включающий максимум $\lfloor n/2 \rfloor + 2$ умножений и n сложений.

Теорема Е. *Каждый полином n-й степени* (1) *с действительными коэффициентами, $n \geq 3$, можно вычислить по схеме*

$$y = x + c, \qquad w = y^2; \qquad z = \begin{cases} (u_n y + \alpha_0)y + \beta_0, & n \text{ четное}, \\ u_n y + \beta_0, & n \text{ нечетное}, \end{cases}$$

$$u(x) = \big(\ldots((z(w-\alpha_1)+\beta_1)(w-\alpha_2)+\beta_2)\ldots\big)(w-\alpha_m)+\beta_m \tag{20}$$

при подходящих вещественных параметрах c, α_k и β_k, где $m = \lceil n/2 \rceil - 1$. На самом деле можно выбрать эти параметры таким образом, что $\beta_m = 0$.

Доказательство. Сначала рассмотрим условия, при которых α_j и β_j могут быть выбраны в (20) при фиксированном c. Пусть

$$p(x) = u(x-c) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0. \tag{21}$$

Покажем, что $p(x)$ имеет вид $p_1(x)(x^2 - \alpha_m) + \beta_m$ для некоторого полинома $p_1(x)$ и некоторых констант α_m, β_m. Если разделить $p(x)$ на $x^2 - \alpha_m$, то остаток β_m будет константой только в том случае, если вспомогательный полином

$$q(x) = a_{2m+1}x^m + a_{2m-1}x^{m-1} + \cdots + a_1, \tag{22}$$

сформированный из нечетных коэффициентов $p(x)$, кратен $x - \alpha_m$. Наоборот, если $q(x)$ имеет множитель $x - \alpha_m$, то $p(x) = p_1(x)(x^2 - \alpha_m) + \beta_m$ при определенных константах β_m, которые можно определить посредством деления.

Также необходимо, чтобы $p_1(x)$ имел вид $p_2(x)(x^2 - \alpha_{m-1}) + \beta_{m-1}$, и это эквивалентно тому, что $q(x)/(x - \alpha_m)$ является кратным $x - \alpha_{m-1}$; если $q_1(x)$ — полином, соответствующий $p_1(x)$, как $q(x)$ соответствует $p(x)$, то $q_1(x) = q(x)/(x - \alpha_m)$. Продолжая в том же духе, найдем, что параметры $\alpha_1, \beta_1, \ldots, \alpha_m, \beta_m$ существуют тогда и только тогда, когда

$$q(x) = a_{2m+1}(x - \alpha_1)\ldots(x - \alpha_m). \tag{23}$$

Другими словами, каждый полином $q(x)$ тождественно равен нулю (и это возможно, только когда n четное) или же $q(x)$ — полином степени m, имеющий все вещественные корни.

Поразительный факт был обнаружен Дж. Ивом (J. Eve) [*Numer. Math.* **6** (1964), 17–21]: *если $p(x)$ имеет по крайней мере $n-1$ комплексный корень, все вещественные части которых не отрицательны или не положительны, то соответствующий полином $q(x)$ тождественно равен нулю или имеет все вещественные корни* (см. упр. 23). Поскольку $u(x) = 0$ тогда и только тогда, когда $p(x + c) = 0$, необходимо просто выбрать параметр c достаточно большим, чтобы по крайней мере $n - 1$ корень $u(x) = 0$ имел вещественные части $\geq -c$, и (20) будет применяться всякий раз, как только $a_{n-1} = u_{n-1} - nc u_n \neq 0$.

Можно определить c таким образом, чтобы эти условия выполнялись, а также чтобы $\beta_m = 0$. Первые n корней уравнения $u(x) = 0$ определены. Если $a + bi$ — корень, имеющий наибольшую или наименьшую вещественную часть, и если $b \neq 0$,

то положим $c = -a$ и $\alpha_m = -b^2$; тогда $x^2 - \alpha_m$ является множителем $u(x - c)$. Если корень с наименьшей или наибольшей вещественной частью вещественный, но корень со *второй* наименьшей (или второй наибольшей) вещественной частью не вещественный, то применяется такое же преобразование. Если два корня с наименьшими (или наибольшими) вещественными частями вещественны, то их можно выразить в виде $a-b$ и $a+b$ соответственно. Пусть $c = -a$ и $\alpha_m = b^2$; снова $x^2-\alpha_m$ — множитель $u(x - c)$. (Однако часто возможны другие значения c; см. упр. 24.) Коэффициент a_{n-1} будет ненулевым для хотя бы одного из этих вариантов, если только $q(x)$ не будет тождественно равным нулю. ▮

Заметим, что этот метод доказательства обычно дает по крайней мере два значения c, переставлять $\alpha_1, \ldots, \alpha_{m-1}$ можно $(m - 1)!$ способом. Некоторые из этих вариантов, вероятно, дают большую точность, чем остальные.

Вопросы, связанные с точностью, конечно, не возникают при работе с целыми числами по модулю m, а не с действительными числами. Схема (9) работает при $n = 4$, когда m и $2u_4$ — взаимно простые числа, а (16) работает при $n = 6$, когда m — взаимно простое число с $6u_6$ и знаменателем (17). В упр. 44 показано, что $n/2 + O(\log n)$ умножений и $O(n)$ сложений достаточно для любого нормированного полинома n-й степени по любому модулю m.

***Цепочки полиномов (полиномиальные цепочки).** Рассмотрим вопросы оптимальности. Каковы *наилучшие схемы* вычисления полиномов различных степеней, выраженные в терминах минимального возможного числа арифметических операций? Этот вопрос впервые проанализировали А. М. Островский для случая, когда коэффициенты предварительно не адаптируются (опубликовано в *Studies in Mathematics and Mechanics Presented to R. von Mises* (New York: Academic Press, 1954), 40–48), и Т. С. Моцкин (T. S. Motzkin) — для адаптированных коэффициентов [см. *Bull. Amer. Math. Soc.* **61** (1955), 163].

Для исследования этого вопроса можно распространить понятие аддитивной цепочки из раздела 4.6.3 на понятие *цепочки полиномов*. Цепочка полиномов — это последовательность вида

$$x = \lambda_0, \quad \lambda_1, \quad \ldots, \quad \lambda_r = u(x), \tag{24}$$

где $u(x)$ — некоторый полином от x и для $1 \le i \le r$

$$\begin{aligned} &\text{либо } \lambda_i = (\pm\lambda_j) \circ \lambda_k, && 0 \le j, k < i, \\ &\text{либо } \lambda_i = \alpha_j \circ \lambda_k, && 0 \le k < i. \end{aligned} \tag{25}$$

Здесь символ "$\circ$" означает какую-либо из трех операций ("+", "−" или "×"), а α_j — так называемый параметр. Шаги первого вида называются *шагами цепочки*, а шаги второго вида — *шагами параметра*. Будем предполагать, что на каждом шаге параметра α_j используются разные параметры; если существует s шагов параметра, то они должны включать $\alpha_1, \alpha_2, \ldots, \alpha_s$ в таком порядке.

Следовательно, полином $u(x)$ в конце цепочки имеет вид

$$u(x) = q_n x^n + \cdots + q_1 x + q_0, \tag{26}$$

где $q_n, \ldots, q_1, q_0$ — полиномы от $\alpha_1, \alpha_2, \ldots, \alpha_s$ с целыми коэффициентами. Будем интерпретировать параметры $\alpha_1, \alpha_2, \ldots, \alpha_s$, как действительные числа, и, следо-

вательно, будем ограничиваться вычислением полиномов с действительными коэффициентами. *Множество результатов* R полиномиальной цепочки определяется, как множество всех возможных векторов $(q_n, \ldots, q_1, q_0)$ действительных чисел, когда $\alpha_1, \alpha_2, \ldots, \alpha_s$ независимо принимают все возможные действительные значения.

Если для каждого выбора $t+1$ различного целого числа $j_0, \ldots, j_t \in \{0, 1, \ldots, n\}$ существует ненулевой полином от многих переменных $f_{j_0 \ldots j_t}$ с целыми коэффициентами, такой, что $f_{j_0 \ldots j_t}(q_{j_0}, \ldots, q_{j_t}) = 0$ для всех $(q_n, \ldots, q_1, q_0)$, принадлежащих R, то мы говорим, что множество результатов R имеет максимум t *степеней свободы* и что цепочка (24) имеет максимум t степеней свободы. Мы также говорим, что цепочка (24) *вычисляет* данный полином $u(x) = u_n x^n + \cdots + u_1 x + u_0$, если $(u_n, \ldots, u_1, u_0)$ принадлежит R. Значит, цепочка полиномов, число степеней свободы которой не больше n, не может вычислять все полиномы n-й степени (см. упр. 27).

Как пример цепочки полиномов рассмотрим следующую цепочку, соответствующую теореме E, где n нечетное:

$$\begin{aligned}
\lambda_0 &= x \\
\lambda_1 &= \alpha_1 + \lambda_0 \\
\lambda_2 &= \lambda_1 \times \lambda_1 \\
\lambda_3 &= \alpha_2 \times \lambda_1 \\
\left.\begin{aligned}
\lambda_{1+3i} &= \alpha_{1+2i} + \lambda_{3i} \\
\lambda_{2+3i} &= \alpha_{2+2i} + \lambda_2 \\
\lambda_{3+3i} &= \lambda_{1+3i} \times \lambda_{2+3i}
\end{aligned}\right\} & \quad 1 \le i < n/2.
\end{aligned} \tag{27}$$

Здесь $\lfloor n/2 \rfloor + 2$ умножений и n сложений, $\lfloor n/2 \rfloor + 1$ шагов цепочки и $n+1$ шагов параметра. По теореме E множество результатов R включает множество всех $(u_n, \ldots, u_1, u_0)$ при $u_n \neq 0$, так что (27) вычисляет все полиномы степени n. Доказать, что множество R имеет максимум n степеней свободы, невозможно, поскольку множество результатов имеет $n+1$ независимую компоненту.

Полиномиальная цепочка с s шагами параметра имеет максимум s степеней свободы. В известной мере это очевидно: нельзя вычислить функцию с t степенями свободы, используя меньше чем t произвольных параметров. Однако этот интуитивно понятный факт нелегко доказать формально; например, существуют непрерывные функции ("заполняющие пространство кривые"), отображающие действительные прямые на плоскость, которые отображают один параметр на два независимых параметра. Для наших целей необходимо проверить, что нет полиномиальных функций с целыми коэффициентами, которые обладают таким свойством; доказательство можно найти в упр. 28.

Если этот факт имеет место, можно продолжить доказательство требуемых утверждений.

Теорема M (Т. С. Моцкин, 1954). *Полиномиальная цепочка с числом умножений $m > 0$ имеет максимум $2m$ степеней свободы.*

Доказательство. Пусть $\mu_1, \mu_2, \ldots, \mu_m$ — это λ_i-цепочки, которые являются операцией умножения. Тогда

$$\mu_i = S_{2i-1} \times S_{2i} \quad \text{для} \quad 1 \le i \le m \qquad \text{и} \qquad u(x) = S_{2m+1}, \tag{28}$$

где каждое S_j равно некоторой сумме μ_i, x_i и α_i. Запишем $S_j = T_j + \beta_j$, где T_j — сумма μ_i и x_i, тогда как β_j равно сумме α_i.

Сейчас $u(x)$ выражен в виде полинома от x, $\beta_1, \ldots, \beta_{2m+1}$ с целыми коэффициентами. Поскольку β_i можно выразить как линейные функции от $\alpha_1, \ldots, \alpha_s$, множество значений, представленных всеми действительными значениями $\beta_1, \ldots, \beta_{2m+1}$, содержит множество результатов цепочки. Следовательно, существует максимум $2m + 1$ степеней свободы; как показано в упр. 30, этот результат можно улучшить, получив $2m$, когда $m > 0$. ▮

В упр. 25 приведен пример построения согласно теореме М. Подобный результат можно доказать для сложения.

Теорема А (Э. Г. Белага, 1958). *Цепочка полинома, содержащая q операций сложения и вычитания, имеет максимум $q + 1$ степеней свободы.*

Доказательство. [*Проблемы кибернетики* **5** (1961), 7–15.] Пусть $\kappa_1, \ldots, \kappa_q$ — это λ_i-цепочки, которые соответствуют операциям сложения или вычитания. Тогда

$$\kappa_i = \pm T_{2i-1} \pm T_{2i} \quad \text{для} \quad 1 \le i \le q \qquad \text{и} \qquad u(x) = T_{2q+1}, \tag{29}$$

где каждое T_j — произведение κ_i, x_i и α_i. Можно записать $T_j = A_j B_j$, где A_j — произведение α_i и B_j — произведение κ_i и x_i. Следующее преобразование можно последовательно произвести по отношению к цепочке для $i = 1, 2, \ldots, q$: пусть $\beta_i = A_{2i}/A_{2i-1}$, тогда $\kappa_i = A_{2i-1}(\pm B_{2i-1} \pm \beta_i B_{2i})$. Затем заменим κ_i на $\pm B_{2i-1} \pm \beta_i B_{2i}$ и каждое появившееся κ_i в формулах $T_{2i+1}, T_{2i+2}, \ldots, T_{2q+1}$ на $A_{2i-1}\kappa_i$. (Эта замена может изменить значения $A_{2i+1}, A_{2i+2}, \ldots, A_{2q+1}$.)

После того как преобразование проделано для всех i, положим $\beta_{q+1} = A_{2q+1}$; тогда $u(x)$ можно выразить в виде полинома от $\beta_1, \ldots, \beta_{q+1}$ и x с целыми коэффициентами. Доказательство почти завершено, однако следует быть осторожными, потому что полиномы, полученные, как $\beta_1, \ldots, \beta_{q+1}$, и определенные для всех действительных значений, могут не включать все полиномы, представленные первоначальной цепочкой (см. упр. 26); возможно получение $A_{2i-1} = 0$ для некоторых значений α_j, но это делает неопределенным β_i.

Чтобы закончить доказательство, заметим, что множество результатов R первоначальной цепочки можно записать в виде $R = R_1 \cup R_2 \cup \cdots \cup R_q \cup R'$, где R_i — множество результатов возможных векторов, когда $A_{2i-1} = 0$, и R' — множество результатов возможных векторов, когда все α_i не равны нулю. Выше было доказано, что R' имеет максимум $q + 1$ степень свободы. Если $A_{2i-1} = 0$, то $T_{2i-1} = 0$. Таким образом, число шагов сложения κ_i может быть уменьшено, чтобы получить другие цепочки вычисляемого множества результатов R_i. По индукции можно доказать, что каждое множество R_i имеет максимум q степеней свободы. Следовательно, согласно упр. 29 R имеет максимум $q + 1$ степень свободы. ▮

Теорема С. *Если цепочка полинома* (24) *вычисляет все полиномы n-й степени $u(x) = u_n x^n + \cdots + u_0$ для некоторого $n \ge 2$, то она включает по крайней мере $\lfloor n/2 \rfloor + 1$ операций умножения и по крайней мере n операций сложения-вычитания.*

Доказательство. Пусть существует m шагов умножения. По теореме М цепочка имеет максимум $2m$ степеней свободы; таким образом, $2m \ge n + 1$. Аналогично по теореме А существует $\ge n$ сложений-вычитаний. ▮

Теорема утверждает, что не существует *ни одного* метода, имеющего меньше $\lfloor n/2 \rfloor + 1$ умножений или меньше n сложений, с помощью которого можно вычислить все возможные полиномы степени n. Результат упр. 29 позволяет нам усилить это утверждение и сказать, что не существует ограниченной совокупности таких цепочек полиномов, которые достаточны для всех полиномов заданной степени. Конечно, некоторые специальные полиномы можно вычислить более эффективно; мы действительно полностью доказали, что полиномы с *алгебраически независимыми* коэффициентами в том смысле, что они не удовлетворяют нетривиальному полиномиальному уравнению, требуют $\lfloor n/2 \rfloor + 1$ умножений и n сложений. К несчастью, коэффициенты, с которыми мы имеем дело на компьютере, — всегда рациональные числа, так что приведенные выше теоремы не имеют реального применения. На самом деле, в упр. 42 показано, что всегда можно достичь $O(\sqrt{n}\,)$ умножений (и, скорее всего, огромного числа сложений). С практической точки зрения ограничения теоремы С относятся к "почти всем" коэффициентам, и они, оказывается, применимы ко всем разумным схемам вычисления. Более того, можно получить нижние грани, соответствующие теореме С, даже в рациональном случае. Согласно приведенному выше усиленному доказательству У. Штрассен (V. Strassen) показал, например, что полином

$$u(x) = \sum_{k=0}^{n} 2^{2^{kn^3}} x^k \tag{30}$$

нельзя вычислить любой полиномиальной цепочкой длины $< n^2/\lg n$, если цепочка не имеет хотя бы $\frac{1}{2}n - 2$ умножений и $n - 4$ сложений [*SICOMP* **3** (1974), 128–149]. Коэффициенты (30) очень велики; однако можно найти такие полиномы, коэффициенты которых равны только нулям и единицам и каждая вычисляющая их цепочка полиномов включает по крайней мере $\sqrt{n}/(4 \lg n)$ умножений в цепочке для всех достаточно больших n, даже когда параметры α_j могут быть произвольными комплексными числами. [См. R. J. Lipton, *SICOMP* **7** (1978), 61–69; C.-P. Schnorr, *Lecture Notes in Comp. Sci.* **53** (1977), 135–147.] Жан-Поль Ван де Виль (Jean-Paul van de Wiele) показал, что оценки определенных 0–1 полиномов требуют, в общем, $cn/\log n$ арифметических операций для некоторого $c > 0$ [*FOCS* **19** (1978), 159–165].

Все еще существующее расхождение между нижней гранью теоремы С и действительным числом операций, как известно, достигается во всех ситуациях, кроме тривиального случая, когда $n = 2$. Теорема Е дает $\lfloor n/2 \rfloor + 2$ умножений, а не $\lfloor n/2 \rfloor + 1$, хотя она доводит до минимума количество сложений. Наши специальные методы для $n = 4$ и $n = 6$ имеют минимальное число умножений, но одно дополнительное сложение. Когда n нечетное, то несложно доказать, что нижних граней теоремы С нельзя одновременно достичь для умножений и для сложений (см. упр. 33). Для $n = 3$, 5 и 7 можно показать, что необходимо по крайней мере $\lfloor n/2 \rfloor + 2$ умножений. В упр. 35 и 36 показано, что обе нижние грани теоремы С не могут быть достигнуты одновременно при $n = 4$ или $n = 6$; таким образом, обсуждаемые методы будут наилучшими для $n < 8$. Для четного n Моцкин (Motzkin) доказал, что достаточно $\lfloor n/2 \rfloor + 1$ умножений, но его конструкция включает неопределенное количество сложений (см. упр. 39). Оптимальную схему для $n = 8$ нашел В. Я. Пан, доказавший, что в этом случае необходимо и достаточно $n + 1$ сложений, когда существует $\lfloor n/2 \rfloor + 1$ умножений; он также показал, что $\lfloor n/2 \rfloor + 1$ умножений и

$n+2$ сложений достаточно для всех четных $n \geq 10$. В работе Пана [*STOC* **10** (1978), 162–172] также установлено необходимое точное минимальное количество умножений и сложений, когда вычисления производятся полностью с комплексными числами, а не с действительными для всех степеней n. В упр. 40 обсуждается интересная ситуация, возникающая при нечетных значениях $n \geq 9$.

Ясно, что результаты, полученные для цепочек полиномов с одной переменной, можно без труда применить к полиномам с многими переменными. Например, если нужно найти оптимальную схему вычисления полинома *без* адаптации коэффициентов, можно рассматривать полином $u(x)$ как полином от $n+2$ переменных $x, u_n, \ldots, u_1, u_0$; в упр. 38 показано, что в этом случае необходимо n умножений и n сложений. В самом деле, А. Бородин [*Theory of Machines and Computations*, edited by Z. Kohavi] и А. Паз (A. Paz) [New York: Academic Press, 1971, 45–58] доказали, что правило Горнера (2) является, по существу, *только* методом вычисления $u(x)$ за $2n$ операций без предварительных условий.

С незначительными изменениями приведенные выше методы могут быть так же хорошо обобщены для цепочек, включающих деление, т. е. для рациональных функций, как и для полиномов. Любопытно, что цепные дроби, аналогичные правилу Горнера, оказались оптимальными с точки зрения вычислительных операций, если скорости выполнения операций умножения и деления одинаковы, даже при дополнительных условиях (см. упр. 37).

Иногда деление полезно во время вычисления полиномов, даже если полиномы определены только в терминах умножения и сложения (соответствующие примеры приводятся в алгоритмах Шо-Трауба для производных полинома). Другим примером является

$$x^n + \cdots + x + 1.$$

Поскольку этот полином можно записать в виде $(x^{n+1} - 1)/(x - 1)$, его можно вычислить, выполнив $l(n+1)$ операций умножения (см. раздел 4.6.3), две операции вычитания и одно деление, в то время как технический прием во избежание деления, кажется, требует приблизительно втрое больше операций (см. упр. 43).

Специальные полиномы от нескольких переменных. *Определитель* матрицы размера $n \times n$ можно рассматривать как полином от n^2 переменных x_{ij}, $1 \leq i, j \leq n$. Если $x_{11} \neq 0$, то

$$\det \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ x_{31} & x_{32} & \ldots & x_{3n} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nn} \end{pmatrix} = x_{11} \det \begin{pmatrix} x_{22} - (x_{21}/x_{11})x_{12} & \ldots & x_{2n} - (x_{21}/x_{11})x_{1n} \\ x_{32} - (x_{31}/x_{11})x_{12} & \ldots & x_{3n} - (x_{31}/x_{11})x_{1n} \\ \vdots & & \vdots \\ x_{n2} - (x_{n1}/x_{11})x_{12} & \ldots & x_{nn} - (x_{n1}/x_{11})x_{1n} \end{pmatrix}. \tag{31}$$

Поэтому определитель матрицы размера $n \times n$ можно найти, вычислив определитель матрицы размера $(n-1) \times (n-1)$ и выполнив дополнительно $(n-1)^2 + 1$ умножений, $(n-1)^2$ сложений и $n-1$ делений. Поскольку определитель размера 2×2 можно вычислить с помощью двух операций умножения и одной операции сложения, определитель почти всех матриц (т. е. матриц, для которых нет необходимости в делении на нуль) можно вычислить, выполнив максимум $(2n^3 - 3n^2 + 7n - 6)/6$ операций умножения, $(2n^3 - 3n^2 + n)/6$ операций сложения и $(n^2 - n - 2)/2$ делений.

Определитель даже легче вычислить, когда появляется нуль. Например, если $x_{11} = 0$, но $x_{21} \neq 0$, получим

$$\det \begin{pmatrix} 0 & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ x_{31} & x_{32} & \dots & x_{3n} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{pmatrix} = x_{21} \det \begin{pmatrix} x_{12} & \dots & x_{1n} \\ x_{32} - (x_{31}/x_{21})x_{22} & \dots & x_{3n} - (x_{31}/x_{21})x_{2n} \\ \vdots & & \vdots \\ x_{n2} - (x_{n1}/x_{21})x_{22} & \dots & x_{nn} - (x_{n1}/x_{21})x_{2n} \end{pmatrix}. \tag{32}$$

Здесь сокращение размера определителя до $(n-1) \times (n-1)$ приводит к экономии $n-1$ умножений и $n-1$ сложений, используемых в (31); это компенсация за дополнительную информацию, необходимую для того, чтобы отличать данный случай. Таким образом, любой определитель можно вычислить, выполнив приблизительно $\frac{2}{3}n^3$ арифметических операций (включая деление). Это замечательно, поскольку это полином с $n!$ членами и n переменными в каждом члене.

Для вычисления определителя матрицы с *целыми* элементами процедуры (31) и (32) кажутся непривлекательными, так как они требуют рациональной арифметики. Однако можно воспользоваться методом вычисления определителя по mod p для любого простого числа p, поскольку возможно деление по mod p (упр. 4.5.2–16). Если это проделать для достаточного количества простых чисел, то точное значение определителя можно найти так, как объясняется в разделе 4.3.2, поскольку неравенство Адамара 4.6.1–(25) дает верхнюю грань.

Коэффициенты *характеристического полинома* $\det(xI - X)$ матрицы X размера $n \times n$ также можно вычислить за $O(n^3)$ шагов; см. J. H. Wilkinson, *The Algebraic Eigenvalue Problem* (Oxford: Clarendon Press, 1965), 353–355, 410–411. В упр. 70 обсуждается интересный свободный от деления метод, не использующий операции деления и включающий $O(n^4)$ шагов.

Перманентом матрицы называется полином, который очень похож на определитель, но все его коэффициенты при произведениях членов матрицы равны $+1$. Таким образом,

$$\operatorname{per} \begin{pmatrix} x_{11} & \dots & x_{1n} \\ \vdots & & \vdots \\ x_{n1} & \dots & x_{nn} \end{pmatrix} = \sum x_{1j_1} x_{2j_2} \dots x_{nj_n}, \tag{33}$$

где сумма берется по всем перестановкам $j_1\, j_2 \dots j_n$ от $\{1, 2, \dots, n\}$. Казалось бы, эту функцию даже легче вычислить, чем ее более сложную с виду сестру, но не существует метода вычисления перманента матрицы, такого же эффективного, как известные методы для определителя. В упр. 9 и 10 показано, что достаточно существенно меньшего количества операций $n!$ для больших n, но время счета всех известных методов все еще возрастает по экспоненте с ростом размера матрицы. На самом деле Лесли Г. Велиант (Leslie G. Valiant) показал, что вычислить заданную 0–1-матрицу так же трудно, как подсчитать число допустимых вычислений машины Тьюринга с недетерминированным полиномиальным временем, если игнорировать полиномиальный характер времени вычислений. Следовательно, проблемы, связанные с полиномиальным временем вычисления перманента, должны включать множество других хорошо известных проблем, сопротивляющихся эффективному решению за полиномиальное время. С другой стороны, Велиант доказал, что пер-

манент целочисленной матрицы размера $n \times n$ можно вычислить по модулю 2^k за $O(n^{4k-3})$ шагов для всех $k \geq 2$. [См. *Theoretical Comp. Sci.* **8** (1979), 189–201.]

Другой связанной с матрицами фундаментальной операцией, конечно, является *умножение матриц:* если $X = (x_{ij})$ — матрица размера $m \times n$, $Y = (y_{jk})$ — матрица размера $n \times s$ и $Z = (z_{ik})$ — матрица размера $m \times s$, то формула $Z = XY$ означает, что

$$z_{ik} = \sum_{j=1}^{n} x_{ij} y_{jk}, \qquad 1 \leq i \leq m, \qquad 1 \leq k \leq s. \tag{34}$$

Это равенство можно рассматривать как вычисление одновременно ms полиномов от $mn + ns$ переменных; каждый полином — "скалярное произведение" двух n-мерных векторов. Непосредственно вычисление включает mns умножений и $ms(n-1)$ сложений, но Ш. Виноград (S. Winograd) в 1967 году обнаружил, что можно заменить около половины умножений сложениями:

$$z_{ik} = \sum_{1 \leq j \leq n/2} (x_{i,2j} + y_{2j-1,k})(x_{i,2j-1} + y_{2j,k}) - a_i - b_k + x_{in} y_{nk} [n \text{ odd}];$$
$$a_i = \sum_{1 \leq j \leq n/2} x_{i,2j} x_{i,2j-1}; \qquad b_k = \sum_{1 \leq j \leq n/2} y_{2j-1,k} y_{2j,k}. \tag{35}$$

В этой схеме используется $\lceil n/2 \rceil ms + \lfloor n/2 \rfloor (m + s)$ умножений и $(n + 2)ms + (\lfloor n/2 \rfloor - 1)(ms + m + s)$ сложений или вычитаний; общее число операций возрастает незначительно, но число умножений сокращается приблизительно вдвое. [См. *IEEE Trans.* **C-17** (1968), 693–694.] Поразительная конструкция Винограда побудила многих ученых более внимательно взглянуть на проблему умножения матрицы, что привело к широко распространенному предположению о том, что $n^3/2$ умножений, возможно, необходимо выполнить для умножения матриц размера $n \times n$, потому что довольно похожая нижняя грань, как известно, имеет место для полиномов с одной переменной.

Еще лучшую схему для больших n открыл в 1968 году Уолкер Штрассен (Volker Strassen); он нашел способ вычисления произведения матриц размера 2×2 с помощью всего семи операций умножения, без коммутативности умножения, как в (35). Так как матрицы размера $2n \times 2n$ можно разбить на четыре матрицы размера $n \times n$, его идею можно использовать рекурсивно для получения произведения матриц размера $2^k \times 2^k$ только с 7^k умножениями вместо $(2^k)^3 = 8^k$. Порядок роста числа сложений равен 7^k. Первоначально в методе Штрассена умножения матриц размера 2×2 [*Numer. Math.* **13** (1969), 354–356] использовалось 7 умножений и 18 сложений; Ш. Виноград позже обнаружил следующую более экономную формулу:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} A & C \\ B & D \end{pmatrix} = \begin{pmatrix} aA+bB & w+v+(a+b-c-d)D \\ w+u+d(B+C-A-D) & w+u+v \end{pmatrix}, \tag{36}$$

где $u = (c-a)(C-D)$, $v = (c+d)(C-A)$, $w = aA+(c+d-a)(A+D-C)$. Если соответствующие промежуточные результаты сохранены, то используется 7 умножений и только 15 сложений; индукцией по k легко показать, что можно умножать матрицы размера $2^k \times 2^k$ с помощью 7^k операций умножения и $5(7^k - 4^k)$ операций сложения. Общее число операций, необходимых для умножения матриц размера $n \times n$, следовательно, можно сократить от порядка n^3 до $O(n^{\lg 7}) = O(n^{2.8074})$. Подобное

сокращение применяется также к вычислению определителей и обратной матрицы; см. J. R. Bunch and J. E. Hopcroft, *Math. Comp.* **28** (1974), 231–236.

До 1978 года несмотря на многочисленные попытки никому не удавалось уменьшить показатель степени Штрассена $\lg 7$, пока Виктор Пан не обнаружил, что он может быть уменьшен до $\log_{70} 143640 \approx 2.795$ (см. упр. 60). Этот новый прорыв привел к дальнейшему интенсивному исследованию проблемы, и совместные усилия Д. Бини (D. Bini), М. Каповани (M. Capovani), Д. Копперсмита (D. Coppersmith), Г. Лотти (G. Lotti), Ф. Романи (F. Romani), А. Шёнхаге (A. Schönhage), В. Пана и Ш. Винограда привели к драматическому уменьшению асимптотики времени счета. В упр. 60–67 обсуждаются интересные технические приемы, благодаря которым установлены такие верхние грани; в частности, в упр. 66 приведено достаточно простое доказательство того, что достаточно $O(n^{2.55})$ операций. Известная до 1997 года лучшая верхняя грань, равная $O(n^{2.376})$, предложена в работе Копперсмита и Винограда [*J. Symbolic Comp.* **9** (1990), 251–280]. По контрасту лучшая находящаяся в обращении нижняя грань равна $2n^2 - 1$ (см. упр. 12).

Эти теоретические результаты совершенно поразительны, но с практической точки зрения они редко используются, так как n должно быть очень большим для того, чтобы можно было преодолеть влияние добавочных факторов. Ричард Брент (Richard Brent) [Stanford Computer Science report CS157 (March, 1970), см. также *Numer. Math.* **16** (1970), 145–156] нашел, что аккуратное выполнение схемы Винограда (35) с соответствующим масштабированием численной устойчивости будет лучше традиционного метода только при $n \geq 40$. К тому же будет сэкономлено всего 7% времени счета, когда $n = 100$. Для комплексной арифметики ситуация отчасти иная; схема (35) становится благоприятной для $n > 20$, и экономится 18% времени при $n = 100$. Брент оценил, что схема Штрассена (36) не превосходит (35) до тех пор, пока $n \approx 250$. Такие огромные матрицы на практике встречаются нечасто, поэтому применяются другие технические приемы. Кроме того, известные методы порядка n^ω, где $\omega < 2.7$, имеют такую большую константу пропорциональности, что потребуется более 10^{23} умножений, прежде чем они превзойдут (36).

По контрасту следующие методы, которые мы обсудим, очень практичны и находят широкое применение. *Дискретное преобразование Фурье* f комплекснозначной функции F от n переменных по соответствующей области $m_1, \ldots, m_n$ элементов определяется равенством

$$f(s_1, \ldots, s_n) = \sum_{\substack{0 \le t_1 < m_1 \\ \cdots \\ 0 \le t_n < m_n}} \exp\left(2\pi i \left(\frac{s_1 t_1}{m_1} + \cdots + \frac{s_n t_n}{m_n}\right)\right) F(t_1, \ldots, t_n) \tag{37}$$

для $0 \le s_1 < m_1, \ldots, 0 \le s_n < m_n$. Слово "преобразование" оправдано, так как можно восстановить значения $F(t_1, \ldots, t_n)$ по значениям $f(s_1, \ldots, s_n)$, как показано в упр. 13. В особо важном случае, когда все $m_j = 2$, получаем

$$f(s_1, \ldots, s_n) = \sum_{0 \le t_1, \ldots, t_n \le 1} (-1)^{s_1 t_1 + \cdots + s_n t_n} F(t_1, \ldots, t_n) \tag{38}$$

для $0 \le s_1, \ldots, s_n \le 1$, и этот метод можно рассматривать, как одновременное вычисление 2^n линейных полиномов от 2^n переменных $F(t_1, \ldots, t_n)$. Общеизвестный технический прием, предложенный Ф. Ятсом (F. Yates) [*The Design and Analysis of Factorial Experiments* (Harpenden: Imperial Bureau of Soil Sciences, 1937)], можно использовать для уменьшения числа сложений в (38) от $2^n(2^n - 1)$ до $n2^n$. Метод Ятса можно понять, рассмотрев случай, когда $n = 3$: пусть $x_{t_1t_2t_3} = F(t_1, t_2, t_3)$.

Задано	Первый шаг	Второй шаг	Третий шаг
x_{000}	$x_{000}+x_{001}$	$x_{000}+x_{001}+x_{010}+x_{011}$	$x_{000}+x_{001}+x_{010}+x_{011}+x_{100}+x_{101}+x_{110}+x_{111}$
x_{001}	$x_{010}+x_{011}$	$x_{100}+x_{101}+x_{110}+x_{111}$	$x_{000}-x_{001}+x_{010}-x_{011}+x_{100}-x_{101}+x_{110}-x_{111}$
x_{010}	$x_{100}+x_{101}$	$x_{000}-x_{001}+x_{010}-x_{011}$	$x_{000}+x_{001}-x_{010}-x_{011}+x_{100}+x_{101}-x_{110}-x_{111}$
x_{011}	$x_{110}+x_{111}$	$x_{100}-x_{101}+x_{110}-x_{111}$	$x_{000}-x_{001}-x_{010}+x_{011}+x_{100}-x_{101}-x_{110}+x_{111}$
x_{100}	$x_{000}-x_{001}$	$x_{000}+x_{001}-x_{010}-x_{011}$	$x_{000}+x_{001}+x_{010}+x_{011}-x_{100}-x_{101}-x_{110}-x_{111}$
x_{101}	$x_{010}-x_{011}$	$x_{100}+x_{101}-x_{110}-x_{111}$	$x_{000}-x_{001}+x_{010}-x_{011}-x_{100}+x_{101}-x_{110}+x_{111}$
x_{110}	$x_{100}-x_{101}$	$x_{000}-x_{001}-x_{010}+x_{011}$	$x_{000}+x_{001}-x_{010}-x_{011}-x_{100}-x_{101}+x_{110}+x_{111}$
x_{111}	$x_{110}-x_{111}$	$x_{100}-x_{101}-x_{110}+x_{111}$	$x_{000}-x_{001}-x_{010}+x_{011}-x_{100}+x_{101}+x_{110}-x_{111}$

Для вычисления от колонки "Задано" к колонке "Первый шаг" требуется четыре сложения и четыре вычитания; интересной особенностью метода Ятса является то, что точно такое же преобразование, которое действует от колонки "Задано" к колонке "Первый шаг", будет действовать от колонки "Первый шаг" к колонке "Второй шаг" и от колонки "Второй шаг" к колонке "Третий шаг". В каждом случае выполняется четыре сложения и затем четыре вычитания, а после трех шагов магически получается требуемое преобразование Фурье $f(s_1, s_2, s_3)$ на месте, первоначально занимаемом $F(s_1, s_2, s_3)$.

Этот особый случай часто называют *преобразованием Уолша* 2^n данных элементов, поскольку соответствующая модель знаков изучена Дж. Л. Уолшем (J. L. Walsh) [*Amer. J. Math.* **45** (1923), 5–24]. Заметим, что число перемен знаков слева направо в колонке "Третий шаг" принимает соответствующие значения

$$0,\ 7,\ 3,\ 4,\ 1,\ 6,\ 2,\ 5.$$

Это перестановки чисел $\{0, 1, 2, 3, 4, 5, 6, 7\}$. Уолш заметил, что будет точно $0, 1, \ldots, 2^n - 1$ изменений знаков в общем случае, если соответствующим образом переставить преобразованные элементы. При этом коэффициенты обеспечивают дискретную аппроксимацию синусоидами с различными частотами. (См. работу H. F. Harmuth, *IEEE Spectrum* **6**, 11 (November, 1969), 82–91, в которой речь идет об использовании этого свойства; дополнительно коэффициенты Уолша обсуждаются в разделе 7.2.1.)

Метод Уолша можно обобщить для оценки любого дискретного преобразования Фурье и фактически для оценки любого числа сумм, которые можно записать в общем виде

$$f(s_1, s_2, \ldots, s_n) = \sum_{\substack{0 \le t_1 < m_1 \\ \ldots \\ 0 \le t_n < m_n}} g_1(s_1, s_2, \ldots, s_n, t_1) g_2(s_2, \ldots, s_n, t_2) \ldots g_n(s_n, t_n) F(t_1, t_2, \ldots, t_n) \quad (39)$$

для $0 \le s_j < m_j$ и заданных функций $g_j(s_j, \ldots, s_n, t_j)$. Мы поступим следующим образом:

$$\begin{aligned}
f_0(t_1, t_2, t_3, \ldots, t_n) &= F(t_1, t_2, t_3, \ldots, t_n);\\
f_1(s_n, t_1, t_2, \ldots, t_{n-1}) &= \sum_{0 \le t_n < m_n} g_n(s_n, t_n) f_0(t_1, t_2, \ldots, t_n);\\
f_2(s_{n-1}, s_n, t_1, \ldots, t_{n-2}) &= \sum_{0 \le t_{n-1} < m_{n-1}} g_{n-1}(s_{n-1}, s_n, t_{n-1}) f_1(s_n, t_1, \ldots, t_{n-1});\\
&\vdots\\
f_n(s_1, s_2, s_3, \ldots, s_n) &= \sum_{0 \le t_1 < m_1} g_1(s_1, \ldots, s_n, t_1) f_{n-1}(s_2, s_3, \ldots, s_n, t_1);\\
f(s_1, s_2, s_3, \ldots, s_n) &= f_n(s_1, s_2, s_3, \ldots, s_n). \qquad (40)
\end{aligned}$$

Для метода Ятса, как показано выше, $g_j(s_j, \ldots, s_n, t_j) = (-1)^{s_j t_j}$; $f_0(t_1, t_2, t_3)$ представляет колонку "Задано", $f_1(s_3, t_1, t_2)$ — колонку "Первый шаг" и т. д. Всякий раз, когда множество сумм можно представить в виде (39) для умеренно простых функций $g_j(s_j, \ldots, s_n, t_j)$, схема (40) будет уменьшать количество вычислений от порядка N^2 до порядка $N \log N$ или около того, где $N = m_1 \ldots m_n$ — количество данных точек. К тому же данная схема идеально подходит для параллельного вычисления. Важный особый случай одномерного преобразования Фурье обсуждается в упр. 14 и 53; одномерный случай также приводится в разделе 4.3.3C.

Рассмотрим более частный случай вычисления полинома. *Интерполяционный полином Лагранжа* порядка n, который мы запишем в виде

$$\begin{aligned}
u_{[n]}(x) = y_0 \frac{(x-x_1)(x-x_2)\ldots(x-x_n)}{(x_0-x_1)(x_0-x_2)\ldots(x_0-x_n)} + y_1 \frac{(x-x_0)(x-x_2)\ldots(x-x_n)}{(x_1-x_0)(x_1-x_2)\ldots(x_1-x_n)}\\
+ \cdots + y_n \frac{(x-x_0)(x-x_1)\ldots(x-x_{n-1})}{(x_n-x_0)(x_n-x_1)\ldots(x_n-x_{n-1})}, \qquad (41)
\end{aligned}$$

является только полиномом от x степени $\le n$, который принимает соответствующие значения $y_0, y_1, \ldots, y_n$ в $n+1$ различной точке $x = x_0, x_1, \ldots, x_n$. (Из (41) очевидно, что $u_{[n]}(x_k) = y_k$ для $0 \le k \le n$. Если $f(x)$ — любой такой полином степени $\le n$, то $g(x) = f(x) - u_{[n]}(x)$ имеет степень $\le n$ и $g(x)$ равно нулю для $x = x_0, x_1, \ldots, x_n$; следовательно, $g(x)$ должен быть кратен полиному $(x-x_0)(x-x_1)\ldots(x-x_n)$. Степень последнего полинома больше n, поэтому $g(x) = 0$.) Если предположить, что значения функции табулированы и хорошо аппроксимируются полиномом, то формулу (41) можно использовать для "интерполирования" значений функции в точках x, не занесенных в таблицу. Лагранж предложил (41) своим ученикам в Paris École Normale в 1795 году [см. *Œuvres* **7** (Paris, 1877), 286], однако Эдвард Уоринг (Edward Waring) из Кембриджского университета к тому времени уже представил такую же формулу, совершенно точно и ясно сформулированную в *Philosophical Transactions* **69** (1779), 59–67.

На первый взгляд кажется, что в формулах Уоринга и Лагранжа совсем немного сложений, вычитаний, умножений и делений; на самом деле — точно n сложений,

$2n^2+2n$ вычитаний, $2n^2+n-1$ умножений и $n+1$ делений. Но, к счастью (как мы подозревали), возможно улучшение.

Основная идея упрощения (41) учитывает тот факт, что $u_{[n]}(x)-u_{[n-1]}(x)=0$ для $x=x_0,\dots,x_{n-1}$; таким образом, $u_{[n]}(x)-u_{[n-1]}(x)$ — полином степени n или меньше, кратный $(x-x_0)\dots(x-x_{n-1})$. Можно сделать вывод, что

$$u_{[n]}(x)=\alpha_n(x-x_0)\dots(x-x_{n-1})+u_{[n-1]}(x),$$

где α_n — константа. Это приводит к *интерполяционной формуле Ньютона*

$$\begin{aligned}u_{[n]}(x)=\alpha_n(x-x_0)(x-x_1)\dots(x-x_{n-1})+\cdots\\ +\alpha_2(x-x_0)(x-x_1)+\alpha_1(x-x_0)+\alpha_0,\end{aligned}\tag{42}$$

где α_k — коэффициенты, которые необходимо определить по заданным числам x_0, x_1, …, x_n, y_0, y_1, …, y_n. Заметим, что эта формула имеет место для всех n; коэффициент α_k не зависит от $x_{k+1},\dots,x_n$ или от $y_{k+1},\dots,y_n$. Когда α_k известны, интерполяционная формула Ньютона удобна для вычисления, так как можно снова обобщить правило Горнера и записать

$$u_{[n]}(x)=\big(\dots(\alpha_n(x-x_{n-1})+\alpha_{n-1})(x-x_{n-2})+\cdots\big)(x-x_0)+\alpha_0\big).\tag{43}$$

Это потребует n умножений и $2n$ сложений. Иначе можно вычислить каждый отдельный член (42) справа налево; выполнив $2n-1$ умножений и $2n$ сложений, мы вычислим все значения $u_{[0]}(x)$, $u_{[1]}(x)$, …, $u_{[n]}(x)$, что во всяком случае покажет, будет ли сходиться интерполяционный процесс.

Коэффициенты α_k в формуле Ньютона можно найти, вычисляя *отношения разностей* по следующей таблице (показано для $n=3$).

$$\begin{array}{lllll}y_0\\ & (y_1-y_0)/(x_1-x_0)=y_1'\\ y_1 & & (y_2'-y_1')/(x_2-x_0)=y_2''\\ & (y_2-y_1)/(x_2-x_1)=y_2' & & (y_3''-y_2'')/(x_3-x_0)=y_3'''\\ y_2 & & (y_3'-y_2')/(x_3-x_1)=y_3''\\ & (y_3-y_2)/(x_3-x_2)=y_3'\\ y_3\end{array}\tag{44}$$

Можно доказать, что $\alpha_0=y_0$, $\alpha_1=y_1'$, $\alpha_2=y_2''$ и т. д., и показать, что отношения разностей имеют тесную связь с производными интерполируемой функции (см. упр. 15). Значит, следующие вычисления (соответствующие (44)) можно использовать, чтобы получить α_k.

Начать с $(\alpha_0,\alpha_1,\dots,\alpha_n)\leftarrow(y_0,y_1,\dots,y_n)$;
затем для $k=1,2,\dots,n$ (в таком порядке)
  присвоить $\alpha_j\leftarrow(\alpha_j-\alpha_{j-1})/(x_j-x_{j-k})$ для $j=n,n-1,\dots,k$ (в таком порядке).

Для этого процесса требуется $\frac{1}{2}(n^2+n)$ делений и n^2+n вычитаний, и экономится около трех четвертей работы по сравнению с (41).

Например, необходимо вычислить 1.5! по значениям 0!, 1!, 2! и 3!, используя кубический полином. Отношения разностей равны

$$\begin{array}{c|c|cccc} x & y & y' & y'' & y''' \\ 0 & 1 & & & \\ & & 0 & & \\ 1 & 1 & & \frac{1}{2} & \\ & & 1 & & \frac{1}{3} \\ 2 & 2 & & \frac{3}{2} & \\ & & 4 & & \\ 3 & 6 & & & \end{array} .$$

Таким образом, $u_{[0]}(x) = u_{[1]}(x) = 1$, $u_{[2]}(x) = \frac{1}{2}x(x-1)+1$, $u_{[3]}(x) = \frac{1}{3}x(x-1)(x-2)+\frac{1}{2}x(x-1)+1$. Подставляя $x = 1.5$ в $u_{[3]}(x)$, получаем $-.125+.375+1 = 1.25$; требуемое "правильное" значение равно $\Gamma(2.5) = \frac{3}{4}\sqrt{\pi} \approx 1.33$. (Но существует, конечно, много других последовательностей, которые начинаются с чисел 1, 1, 2 и 6.)

Чтобы интерполировать несколько полиномов, имеющих те же точки интерполирования x_0, x_1, ..., x_n, но разные значения y_0, y_1, ..., y_n, желательно переписать (41) в виде, предложенном В. Дж. Тейлором (W. J. Taylor) [*J. Research Nat. Bur. Standards* **35** (1945), 151–155]:

$$u_{[n]}(x) = \Big(\frac{y_0 w_0}{x-x_0} + \cdots + \frac{y_n w_n}{x-x_n}\Big) \Big/ \Big(\frac{w_0}{x-x_0} + \cdots + \frac{w_n}{x-x_n}\Big) \tag{45}$$

и $x \notin \{x_0, x_1, \ldots, x_n\}$, где

$$w_k = 1/(x_k - x_0)\ldots(x_k - x_{k-1})(x_k - x_{k+1})\ldots(x_k - x_n). \tag{46}$$

Такой вид (41) также рекомендуется в связи с численной устойчивостью [см. работу P. Henrici, *Essentials of Numerical Analysis* (New York: Wiley, 1982), 237–243]. Знаменатель (45) равен частичным отношениям дроби $1/(x-x_0)(x-x_1)\ldots(x-x_n)$.

Важное и в некоторой степени неожиданное применение полиномиальной интерполяции открыто Ади Шамиром (Adi Shamir) [*CACM* **22** (1979), 612–613], который заметил, что полиномы по модулю p можно использовать для "засекречивания". Иначе говоря, существует возможность разработки системы скрытых ключей или паролей, такой, что, зная любые $n+1$ ключей, можно эффективно вычислить магическое число N, допустим, открывающее дверь, но, зная любые n ключей, получить какую-либо информацию о N невозможно. Поразительно простое решение Шамира этой проблемы состоит в выборе случайного полинома $u(x) = u_n x^n + \cdots + u_1 x + u_0$, где $0 \le u_i < p$ и p—большие простые числа. Каждая часть секрета является целым числом x из интервала $0 < x < p$ вместе со значением $u(x) \bmod p$, а сверхсекретное число N равно постоянному члену u_0. Задав $n+1$ значение $u(x_i)$, можно получить N путем интерполирования. Однако, если только n значений $u(x_i)$ заданы, всегда существует единственный полином $u(x)$, имеющий заданный постоянный член, но такие же значения в точках x_1, ..., x_n, следовательно, n значений не делают одно определенное N более вероятным, чем любое другое.

Полезно заметить, что интерполирование полинома—только частный случай китайской теоремы об остатках из раздела 4.3.2 и упр. 4.6.2–3, так как нам известны значения $u_{[n]}(x)$ по модулю взаимно простых полиномов $x-x_0$, ..., $x-x_n$. (Как видно из раздела 4.6.2 и обсуждения, следующего за (3), $f(x)$ по модулю $(x-x_0) = f(x_0)$.) В такой интерпретации формула Ньютона (42) точно равна "представлению

со смешанным основанием" формулы 4.3.2–(25), а 4.3.2–(24) дает другой способ вычисления $\alpha_0, \ldots, \alpha_n$ с помощью такого же числа операций, как и (44).

Используя быстрое преобразование Фурье, можно уменьшить время счета при интерполировании до $O\bigl(n\,(\log n)^2\bigr)$. Подобное сокращение можно сделать и для таких родственных алгоритмов, как решение китайской теоремы об остатках и вычисление полиномов n-й степени в n различных точках. [См. E. Horowitz, *Inf. Proc. Letters* **1** (1972), 157–163; A. Borodin and R. Moenck, *J. Comp. Syst. Sci.* **8** (1974), 336–385; A. Borodin, *Complexity of Sequential and Parallel Numerical Algorithms*, edited by J. F. Traub (New York: Academic Press, 1973), 149–180; D. Bini and V. Pan, *Polynomial and Matrix Computations* **1** (Boston: Birkhäuser, 1994), гл. 1.] Однако эти исследования представляют, главным образом, теоретический интерес, поскольку известные алгоритмы требуют достаточно больших затрат времени на другие операции, что делает их непривлекательными, если n не слишком велико.

Замечательное расширение метода разностных отношений, который так же хорошо применим к отношению полиномов, как и к самим полиномам, введено в 1909 году Т. Н. Тьеле (T. N. Thiele). Метод Тьеле "обратных разностей" обсуждается в книге Л. М. Милна-Томпсона (L. M. Milne-Thompson, *Calculus of Finite Differences* (London: MacMillan, 1933), гл. 5; см. также работу R. W. Floyd, *CACM* **3** (1960), 508).

***Билинейные формы.** Некоторые из проблем, рассмотренных в данном разделе, — это частные случаи общей проблемы вычисления множества *билинейных форм*

$$z_k = \sum_{i=1}^{m}\sum_{j=1}^{n} t_{ijk}x_iy_j \qquad \text{для } 1 \le k \le s, \tag{47}$$

где t_{ijk} — определенные коэффициенты, принадлежащие некоторому заданному полю. Трехмерная матрица (t_{ijk}) называется *тензором* размера $m\times n\times s$. Его можно изобразить, записывая s матриц размера $m \times n$ по одной для каждого значения k. Например, проблема умножения комплексных чисел, т. е. вычисления

$$z_1 + iz_2 = (x_1 + ix_2)(y_1 + iy_2) = (x_1y_1 - x_2y_2) + i(x_1y_2 + x_2y_1), \tag{48}$$

является проблемой вычисления билинейной формы, точно определенной тензором размера $2 \times 2 \times 2$:

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Умножение матриц, как это определено в (34), — это проблема вычисления семейства билинейных форм, соответствующих особому тензору размера $mn \times ns \times ms$. Преобразования Фурье (37) также можно отнести к этой проблеме, несмотря на то что они не билинейны, а линейны, если допустить, что x_k — это постоянные, а не переменные.

Вычисление билинейных форм легче всего изучать, если ограничиться тем, что можно назвать *нормальными* вычислительными схемами, в которых все умножения в цепочке происходят между линейными комбинациями x-в и линейными комбинациями y-в. Таким образом, мы строим r произведений

$$w_l = (a_{1l}x_1 + \cdots + a_{ml}x_m)(b_{1l}y_1 + \cdots + b_{nl}y_n) \qquad \text{для } 1 \le l \le r \tag{49}$$

и получаем z_k в виде линейных комбинаций этих произведений:

$$z_k = c_{k1}w_1 + \cdots + c_{kr}w_r \qquad \text{для } 1 \le k \le s. \tag{50}$$

Здесь все a_k, b_k и c_{kr} принадлежат заданному полю коэффициентов. Сравнив (50) с (47), получаем, что нормальные вычислительные схемы корректны для тензора (t_{ijk}) тогда и только тогда, когда

$$t_{ijk} = a_{i1}b_{j1}c_{k1} + \cdots + a_{ir}b_{jr}c_{kr} \tag{51}$$

для $1 \le i \le m$, $1 \le j \le n$ и $1 \le k \le s$.

Ненулевой тензор (t_{ijk}) называют тензором ранга "единица", если существуют три таких вектора $(a_1, \ldots, a_m)$, $(b_1, \ldots, b_n)$, $(c_1, \ldots, c_s)$, что $t_{ijk} = a_i b_j c_k$ для всех i, j, k. Это определение можно распространить на все тензоры, утверждая, что *рангом тензора* (t_{ijk}) *является такое минимальное число* r, *что* (t_{ijk}) *выражается в виде суммы* r *тензоров единичного ранга* над заданным полем. Сравнив это определение с равенством (51), покажем, что ранг тензора есть минимальное число умножений в цепочке при нормальном вычислении соответствующих билинейных форм. Между прочим, когда $s = 1$, тензор (t_{ijk}) — всего лишь обычная матрица и ранг тензора (t_{ij1}) равен рангу матрицы (см. упр. 49). Понятие ранга тензора введено Ф. Л. Хичкоком (F. L. Hitchcock, *J. Math. and Physics* **6** (1927), 164–189); его применение к проблеме сложности вычисления полинома приведено в важной статье V. Strassen, *Crelle* **264** (1973), 184–202.

Схема Винограда (35) для умножения матриц является "анормальной", так как она смешивает значения x и y до их умножения. С другой стороны, схема Штрассена-Винограда (36) не опирается на коммутативность умножения, поэтому она нормальна. На самом деле (36) соответствует следующему способу представления тензора размера $4 \times 4 \times 4$ для умножения матриц размера 2×2 в виде суммы семи тензоров единичного ранга:

$$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\1&0&0&0\\0&1&0&0\end{pmatrix}\begin{pmatrix}0&0&1&0\\0&0&0&1\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&1&0&0\\0&0&0&1\end{pmatrix} = \begin{pmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}$$
$$+\begin{pmatrix}0&0&0&0\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&1&1\\0&0&0&0\\0&0&1&\bar{1}\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&1&\bar{1}\\0&0&0&0\\0&0&1&\bar{1}\\0&0&0&0\end{pmatrix}$$
$$+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\\bar{1}&1&1&\bar{1}\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\\bar{1}&0&1&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\\bar{1}&0&1&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\\bar{1}&0&1&0\end{pmatrix}$$
$$+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&1\\0&0&0&\bar{1}\end{pmatrix}\begin{pmatrix}0&0&0&1\\0&0&0&\bar{1}\\0&0&0&\bar{1}\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}+\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}\bar{1}&0&1&\bar{1}\\0&0&0&0\\1&0&\bar{1}&1\\1&0&\bar{1}&1\end{pmatrix}\begin{pmatrix}\bar{1}&0&1&\bar{1}\\0&0&0&0\\1&0&\bar{1}&1\\1&0&\bar{1}&1\end{pmatrix}\begin{pmatrix}\bar{1}&0&1&\bar{1}\\0&0&0&0\\1&0&\bar{1}&1\\1&0&\bar{1}&1\end{pmatrix}. \tag{52}$$

(Здесь $\bar{1}$ обозначает -1.)

Тот факт, что (51) симметрично по i, j, k и инвариантно относительно множества преобразований, делает изучение рангов тензоров простым с математической точки зрения и приводит к некоторым удивительным выводам о билинейных формах. Можно, изменяя порядок индексов i, j, k, получить "транспонированные"

билинейные формы, и транспонированный тензор, понятно, имеет такой же ранг, но соответствующие билинейные формы являются, в принципе, совершенно иными. Например, нормальная схема вычисления произведения матрицы размера $(m \times n)$ на матрицу размера $(n \times s)$ предусматривает существование нормальной схемы вычисления произведения матрицы размера $(n \times s)$ и матрицы размера $(s \times m)$, если используется такое же число умножений по цепочке. В терминах матриц эти две проблемы едва ли кажутся как-то связанными — они включают различное число умножений на векторы различной размерности, но в тензорной терминологии они эквивалентны. [См. В. Я. Пан, *Успехи мат. наук* **27**, 5 (сентябрь-октябрь 1972), 249–250; J. E. Hopcroft and J. Musinski, *SICOMP* **2** (1973), 159–173.]

Если тензор (t_{ijk}) можно представить в виде суммы (51) r одноранговых тензоров и A, B, C — матрицы (a_{il}), (b_{jl}), (c_{kl}) соответствующего размера $m \times r$, $n \times r$, $s \times r$, то мы говорим, что (A, B, C) — *реализация* тензора (t_{ijk}). Например, реализация умножения матриц размера 2×2 в (52) может быть точно определена матрицами

$$A = \begin{pmatrix} 1&0&\bar{1}&0&0&1&\bar{1} \\ 0&1&0&0&0&1&0 \\ 0&0&1&0&1&\bar{1}&1 \\ 0&0&0&1&1&\bar{1}&1 \end{pmatrix}, \quad B = \begin{pmatrix} 1&0&0&\bar{1}&\bar{1}&0&1 \\ 0&1&0&1&0&0&0 \\ 0&0&1&1&1&0&\bar{1} \\ 0&0&\bar{1}&\bar{1}&0&1&\bar{1} \end{pmatrix}, \quad C = \begin{pmatrix} 1&1&0&0&0&0&0 \\ 1&0&1&1&0&0&1 \\ 1&0&0&0&1&1&1 \\ 1&0&1&0&1&0&1 \end{pmatrix}. \tag{53}$$

Тензор (t_{ijk}) размера $m \times n \times s$ также можно представить в виде матрицы, объединив ее индексы. Обозначим $(t_{(ij)k})$ для матрицы размера $mn \times s$, строки которой указаны парами индексов $\langle i, j \rangle$, а столбцы имеют индекс k. Аналогично $(t_{k(ij)})$ станет матрицей размера $s \times mn$, содержащей t_{ijk} в строке k и в столбце $\langle i, j \rangle$; $(t_{(ik)j})$ является матрицей размера $ms \times n$ и т. д. Индексы матрицы — необязательно целые числа, и здесь упорядоченные пары используются как индексы. С помощью этих обозначений можно получить следующую простую, но полезную нижнюю грань ранга тензора.

Лемма Т. *Пусть* (A, B, C) *— реализация тензора* (t_{ijk}) *размера* $m \times n \times s$. *Тогда* $\operatorname{rank}(A) \geq \operatorname{rank}(t_{i(jk)})$, $\operatorname{rank}(B) \geq \operatorname{rank}(t_{j(ik)})$ *и* $\operatorname{rank}(C) \geq \operatorname{rank}(t_{k(ij)})$; *следовательно,*

$$\operatorname{rank}(t_{ijk}) \geq \max\bigl(\operatorname{rank}(t_{i(jk)}), \operatorname{rank}(t_{j(ik)}), \operatorname{rank}(t_{k(ij)})\bigr).$$

Доказательство. Из соображений симметрии достаточно показать, что $r \geq \operatorname{rank}(A) \geq \operatorname{rank}(t_{i(jk)})$. Так как матрица A имеет размер $m \times r$, то, очевидно, она не может иметь ранг, больший, чем r. К тому же согласно (51) матрица $(t_{i(jk)})$ равна AQ, где Q — матрица размера $r \times ns$ вида $Q_{l\langle j,k\rangle} = b_{jl}c_{kl}$. Если x — любая вектор-строка, такая, что $xA = 0$, то $xAQ = 0$. Следовательно, все линейные зависимости в A появляются и в AQ и $\operatorname{rank}(AQ) \leq \operatorname{rank}(A)$. ▮

В качестве примера использования леммы Т рассмотрим проблему умножения полиномов. Предположим, необходимо умножить обычный полином степени 2 на обычный полином степени 3, вычисляя коэффициенты произведения:

$$(x_0 + x_1u + x_2u^2)(y_0 + y_1u + y_2u^2 + y_3u^3) = z_0 + z_1u + z_2u^2 + z_3u^3 + z_4u^4 + z_5u^5. \tag{54}$$

Это сводится к проблеме вычисления шести билинейных форм, соответствующих тензору размера $3 \times 4 \times 6$:

$$\begin{pmatrix}1&0&0&0\\0&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&1&0&0\\1&0&0&0\\0&0&0&0\end{pmatrix}\begin{pmatrix}0&0&1&0\\0&1&0&0\\1&0&0&0\end{pmatrix}\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&1&0&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0&0&0&0\\0&0&0&0\\0&0&0&1\end{pmatrix}. \tag{55}$$

Для краткости можно записать (54) в виде $x(u)y(u) = z(u)$, где $x(u)$ — полином $x_0 + x_1u + x_2u^2$ и т. д. (Мы замкнули круг, вернувшись к началу данного раздела, но в отличие от (1), где полином обозначается как $u(x)$, мы обозначаем его как $x(u)$; мы изменили обозначения, потому что сейчас нас интересуют переменные *коэффициенты* полиномов.)

Если каждую из шести матриц в (55) рассматривать как вектор размерности 12 с индексами $\langle i, j\rangle$, то ясно, что векторы линейно независимы, так как их нули занимают разные позиции. Следовательно, по лемме T ранг (55) равен по крайней мере шести. Обратно, можно получить коэффициенты z_0, z_1, ..., z_5 только с помощью шести цепочек умножений. Например, вычисляя

$$x(0)y(0),\ x(1)y(1),\ \ldots,\ x(5)y(5) \tag{56}$$

при заданных значениях $z(0)$, $z(1)$, ..., $z(5)$ и по интерполяционной формуле, приведенной выше, получим коэффициенты $z(u)$. Вычисление $x(j)$ и $y(j)$ можно выполнить всецело в терминах сложений и/или умножений параметров, и интерполяционная формула просто использует линейные комбинации этих значений. Таким образом, все цепочки умножений показаны в (56), а ранг (55) равен шести. (По существу, здесь используется та же техника, что и при умножении чисел с большой точностью в алгоритме 4.3.3T.)

Оказывается, реализация (A, B, C) (55), набросок которой приведен в данном разделе выше, имеет вид

$$\begin{pmatrix}1&1&1&1&1&1\\0&1&2&3&4&5\\0&1&4&9&16&25\end{pmatrix},\ \begin{pmatrix}1&1&1&1&1&1\\0&1&2&3&4&5\\0&1&4&9&16&25\\0&1&8&27&64&125\end{pmatrix},\ \begin{pmatrix}120&0&0&0&0&0\\-274&600&-600&400&-150&24\\225&-770&1070&-780&305&-50\\-85&355&-590&490&-205&35\\15&-70&130&-120&55&-10\\-1&5&-10&10&-5&1\end{pmatrix}\times\tfrac{1}{120}. \tag{57}$$

Таким образом, схема действительно выполняет минимальное количество цепочек умножений, но она совершенно непрактична, потому что включает в себя очень много операций сложения и умножения коэффициентов. Рассмотрим практический подход к построению более эффективных схем, предложенных Ш. Виноградом.

Во-первых, чтобы вычислить коэффициенты $x(u)y(u)$, когда $\deg(x) = m$ и $\deg(y) = n$, можно воспользоваться тождеством

$$x(u)y(u) = \big(x(u)y(u) \bmod p(u)\big) + x_m y_n p(u), \tag{58}$$

где $p(u)$ — любой нормированный многочлен степени $m + n$. Полином $p(u)$ следует выбрать так, чтобы коэффициенты $x(u)y(u) \bmod p(u)$ легко вычислялись.

Во-вторых, чтобы вычислить коэффициенты $x(u)y(u) \bmod p(u)$, когда полином $p(u)$ может быть множителем $q(u)r(u)$, где $\gcd\big(q(u), r(u)\big) = 1$, следует воспользо-

ваться тождеством

$$x(u)y(u) \bmod q(u)r(u) = \big(a(u)r(u)(x(u)y(u) \bmod q(u)) + b(u)q(u)(x(u)y(u) \bmod r(u))\big) \bmod q(u)r(u), \quad (59)$$

где $a(u)r(u) + b(u)q(u) = 1$; это, по существу, применение к полиномам китайской теоремы об остатках.

В-третьих, всегда можно вычислять коэффициенты полинома $x(u)y(u) \bmod p(u)$, используя тривиальное тождество

$$x(u)y(u) \bmod p(u) = \big(x(u) \bmod p(u)\big)\big(y(u) \bmod p(u)\big) \bmod p(u). \quad (60)$$

Повторно применив (58)–(60), можно, как будет показано ниже, получить эффективную схему.

Например, для задачи (54) выберем $p(u) = u^5 - u$ и применим (58); основания для такого выбора $p(u)$ будут приведены в дальнейшем. Если записать $p(u) = u(u^4 - 1)$, правило (59) приведет к

$$x(u)y(u) \bmod u(u^4 - 1) = \big(-(u^4 - 1)x_0y_0 + u^4(x(u)y(u) \bmod (u^4 - 1))\big) \bmod (u^5 - u). \quad (61)$$

Здесь использован тот факт, что $x(u)y(u) \bmod u = x_0y_0$; вообще, это хорошая идея — выбрать $p(u)$ так, чтобы $p(0) = 0$, поэтому можно использовать данное упрощение. Если сейчас удастся определить коэффициенты w_0, w_1, w_2, w_3 полинома $x(u)y(u) \bmod (u^4 - 1) = w_0 + w_1u + w_2u^2 + w_3u^3$, то задача будет решена, поскольку

$$u^4\big(x(u)y(u) \bmod (u^4 - 1)\big) \bmod (u^5 - u) = w_0u^4 + w_1u + w_2u^2 + w_3u^3,$$

и комбинация (58) и (61) приведет к

$$x(u)y(u) = x_0y_0 + (w_1 - x_2y_3)u + w_2u^2 + w_3u^3 + (w_0 - x_0y_0)u^4 + x_2y_3u^5. \quad (62)$$

(Конечно, эту формулу можно проверить непосредственно.)

Остается решить проблему вычисления $x(u)y(u) \bmod (u^4 - 1)$ (эта небольшая задача интересна сама по себе). На минутку допустим, что полином $x(u)$ имеет степень 3, а не 2. Тогда коэффициенты $x(u)y(u) \bmod (u^4 - 1)$ равны

$$x_0y_0 + x_1y_3 + x_2y_2 + x_3y_1, \quad x_0y_1 + x_1y_0 + x_2y_3 + x_3y_2,$$
$$x_0y_2 + x_1y_1 + x_2y_0 + x_3y_3, \quad x_0y_3 + x_1y_2 + x_2y_1 + x_3y_0$$

и соответствующий тензор равен

$$\begin{pmatrix}1&0&0&0\\0&0&0&1\\0&0&1&0\\0&1&0&0\end{pmatrix}\begin{pmatrix}0&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0&0&1&0\\0&1&0&0\\1&0&0&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}0&0&0&1\\0&0&1&0\\0&1&0&0\\1&0&0&0\end{pmatrix}. \quad (63)$$

Вообще, когда $\deg(x) = \deg(y) = n - 1$, коэффициенты $x(u)y(u) \bmod (u^n - 1)$ называют *циклической сверткой* $(x_0, x_1, \ldots, x_{n-1})$ и $(y_0, y_1, \ldots, y_{n-1})$. k-й коэффициент w_k является билинейной формой $\sum x_iy_j$, где сумма берется по всем i и j с $i + j \equiv k$ по модулю n.

Циклическую свертку степени 4 можно получить, применив правило (59). Первым шагом будет нахождение множителей $u^4 - 1$, т. е. $(u-1)(u+1)(u^2+1)$. Это можно

записать в виде $(u^2-1)(u^2+1)$, затем применить правило (59) и снова использовать (59) по модулю $(u^2-1)=(u-1)(u+1)$. Однако легче обобщить китайскую теорему об остатках (59) непосредственно для нескольких взаимно простых множителей. Например,

$$\begin{aligned} x(u)y(u) \bmod q_1(u)q_2(u)q_3(u) \\ = \big(a_1(u)q_2(u)q_3(u)\big(x(u)y(u) \bmod q_1(u)\big) + a_2(u)q_1(u)q_3(u)\big(x(u)y(u) \bmod q_2(u)\big) \\ + a_3(u)q_1(u)q_2(u)\big(x(u)y(u) \bmod q_3(u)\big)\big) \bmod q_1(u)q_2(u)q_3(u), \end{aligned} \tag{64}$$

где $a_1(u)q_2(u)q_3(u)+a_2(u)q_1(u)q_3(u)+a_3(u)q_1(u)q_2(u)=1$. (Это соотношение можно также истолковать другим способом, заметив, что представление в виде частичных отношений $1/q_1(u)q_2(u)q_3(u)$ равно $a_1(u)/q_1(u)+a_2(u)/q_2(u)+a_3(u)/q_3(u)$.) Из (64) получим

$$\begin{aligned} x(u)y(u) \bmod (u^4-1) = \big(\tfrac14(u^3+u^2+u+1)x(1)y(1) - \tfrac14(u^3-u^2+u-1)x(-1)y(-1) \\ - \tfrac12(u^2-1)\big(x(u)y(u) \bmod (u^2+1)\big)\big) \bmod (u^4-1). \end{aligned} \tag{65}$$

Остается вычислить $x(u)y(u) \bmod (u^2+1)$ и обратиться к правилу (60). Сначала преобразуем $x(u)$ и $y(u) \bmod (u^2+1)$ и получим $X(u)=(x_0-x_2)+(x_1-x_3)u$, $Y(u)=(y_0-y_2)+(y_1-y_3)u$. Затем по правилу (60) вычислим $X(u)Y(u)=Z_0+Z_1u+Z_2u^2$ и, преобразовав его по модулю (u^2+1), получим $(Z_0-Z_2)+Z_1u$. Вычислить $X(u)Y(u)$ несложно: можно воспользоваться правилом (58) с $p(u)=u(u+1)$ и получить

$$Z_0 = X_0Y_0, \quad Z_1 = X_0Y_0 - (X_0-X_1)(Y_0-Y_1) + X_1Y_1, \quad Z_2 = X_1Y_1.$$

(Тем самым мы заново открыли трюк 4.3.3–(2) более систематическим способом.) Объединив все вместе, получим следующую реализацию (A,B,C) степени 4 циклической свертки:

$$\begin{pmatrix} 1&1&1&0&1\\ 1&\bar1&0&1&\bar1\\ 1&1&\bar1&0&\bar1\\ 1&\bar1&0&\bar1&1 \end{pmatrix}, \quad \begin{pmatrix} 1&1&1&0&1\\ 1&\bar1&0&1&\bar1\\ 1&1&\bar1&0&\bar1\\ 1&\bar1&0&\bar1&1 \end{pmatrix}, \quad \begin{pmatrix} 1&1&2&\bar2&0\\ 1&\bar1&2&2&\bar2\\ 1&1&\bar2&2&0\\ 1&\bar1&\bar2&\bar2&2 \end{pmatrix} \times \frac14. \tag{66}$$

Здесь $\bar1$ обозначает -1 и $\bar2$ обозначает -2.

Тензор для циклической свертки степени n удовлетворяет равенству

$$t_{i,j,k} = t_{k,-j,i}; \tag{67}$$

нижние индексы рассматриваются по модулю n, поскольку $t_{ijk}=1$ тогда и только тогда, когда $i+j\equiv k$ по модулю n, поэтому, если (a_{il}), (b_{jl}), (c_{kl}) — реализация циклической свертки, т. е. (c_{kl}), $(b_{-j,l})$, (a_{il}); в частности, можно представить (63), преобразовав (66) в

$$\begin{pmatrix} 1&1&2&\bar2&0\\ 1&\bar1&2&2&\bar2\\ 1&1&\bar2&2&0\\ 1&\bar1&\bar2&\bar2&2 \end{pmatrix} \times \frac14, \quad \begin{pmatrix} 1&1&1&0&1\\ 1&\bar1&0&\bar1&1\\ 1&1&\bar1&0&\bar1\\ 1&\bar1&0&1&\bar1 \end{pmatrix}, \quad \begin{pmatrix} 1&1&1&0&1\\ 1&\bar1&0&1&\bar1\\ 1&1&\bar1&0&\bar1\\ 1&\bar1&0&\bar1&1 \end{pmatrix}. \tag{68}$$

Сейчас все сложные скалярные величины оказались в матрице A. Это важно для практических целей, так как часто необходимо вычислить свертку для нескольких значений y_0, y_1, y_2, y_3, но с фиксированным набором x_0, x_1, x_2, x_3. В такой

ситуации арифметика для x_j может быть произведена для всех значений сразу и нет необходимости ее пересчитывать. Таким образом, (68) приводит к следующей схеме вычисления циклической свертки w_0, w_1, w_2, w_3, где x_0, x_1, x_2, x_3 заранее известны:

$$s_1 = y_0 + y_2, \quad s_2 = y_1 + y_3, \quad s_3 = s_1 + s_2, \quad s_4 = s_1 - s_2,$$
$$s_5 = y_0 - y_2, \quad s_6 = y_3 - y_1, \quad s_7 = s_5 - s_6;$$
$$m_1 = \tfrac{1}{4}(x_0 + x_1 + x_2 + x_3)\cdot s_3, \quad m_2 = \tfrac{1}{4}(x_0 - x_1 + x_2 - x_3)\cdot s_4,$$
$$m_3 = \tfrac{1}{2}(x_0 + x_1 - x_2 - x_3)\cdot s_5, \quad m_4 = \tfrac{1}{2}(-x_0 + x_1 + x_2 - x_3)\cdot s_6, \quad m_5 = \tfrac{1}{2}(x_3 - x_1)\cdot s_7;$$
$$t_1 = m_1 + m_2, \quad t_2 = m_3 + m_5, \quad t_3 = m_1 - m_2, \quad t_4 = m_4 - m_5;$$
$$w_0 = t_1 + t_2, \quad w_1 = t_3 + t_4, \quad w_2 = t_1 - t_2, \quad w_3 = t_3 - t_4. \tag{69}$$

Здесь 5 операций умножения и 15 — сложения, хотя определение циклической свертки включает в себя 16 умножений и 12 сложений. Позже будет доказано, что необходимо 5 операций умножения.

Возвратимся к нашей первоначальной проблеме умножения в (54). Воспользовавшись (62), получим реализацию

$$\begin{pmatrix} 4 & 0 & 1 & 1 & 2 & \bar{2} & 0 \\ 0 & 0 & 1 & \bar{1} & 2 & 2 & \bar{2} \\ 0 & 4 & 1 & 1 & \bar{2} & 2 & 0 \end{pmatrix} \times \frac{1}{4}, \qquad \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & \bar{1} & 0 & \bar{1} & 1 \\ 0 & 0 & 1 & 1 & \bar{1} & 0 & \bar{1} \\ 0 & 1 & 1 & \bar{1} & 0 & 1 & \bar{1} \end{pmatrix}, \qquad \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \bar{1} & 1 & \bar{1} & 0 & 1 & \bar{1} \\ 0 & 0 & 1 & 1 & \bar{1} & 0 & \bar{1} \\ 0 & 0 & 1 & \bar{1} & 0 & \bar{1} & 1 \\ \bar{1} & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{70}$$

В этой схеме используется на одну цепочку больше, чем минимальное число умножений в цепочке, но требуется намного меньше операций умножения параметров, чем (57). Конечно, это допустимо, так как схема все еще достаточно сложна: если нашей целью является простое вычисление коэффициентов $z_0, z_1, \ldots, z_5$ произведения двух заданных полиномов $(x_0 + x_1u + x_2u^2)(y_0 + y_1u + y_2u^2 + y_3u^3)$ как одноразовая задача, то лучше всего держать пари, что можно использовать очевидный метод с 12 умножениями и 6 сложениями, если, скажем, x_k и y_i — не матрицы. Другая умеренно привлекательная схема, требующая 8 умножений и 18 сложений, приведена в упр. 58, (b). Заметим, что если x_k фиксированы, когда y_i изменяются, то (70) производит вычисления с 7 операциями умножения и 17 операциями сложения. Хотя даже эта схема не особенно пригодна в том виде, в котором она построена, наш вывод иллюстрирует важную технику, полезную в различных ситуациях. Например, Виноград воспользовался этим подходом для вычисления преобразования Фурье с помощью значительно меньшего количества операций умножения, чем необходимо для алгоритма быстрого преобразования Фурье (см. упр. 53).

В завершение этого раздела найдем точный ранг тензора размера $n \times n \times n$, который соответствует умножению двух полиномов по модулю третьего:

$$z_0 + z_1u + \cdots + z_{n-1}u^{n-1} = (x_0 + x_1u + \cdots + x_{n-1}u^{n-1})(y_0 + y_1u + \cdots + y_{n-1}u^{n-1}) \bmod p(u). \tag{71}$$

Здесь $p(u)$ означает любой заданный нормированный полином степени n; в частности, $p(u)$ может быть $u^n - 1$, тогда одним результатом нашего исследования будет

нахождение ранга тензора, соответствующего циклической свертке степени n. Будет удобно записать $p(u)$ в виде

$$p(u) = u^n - p_{n-1}u^{n-1} - \cdots - p_1 u - p_0. \tag{72}$$

Таким образом, $u^n \equiv p_0 + p_1 u + \cdots + p_{n-1}u^{n-1}$ $\big($по модулю $p(u)\big)$.

Элемент тензора t_{ijk} равен коэффициенту u^k в $u^{i+j} \bmod p(u)$ и равен элементу в строке i и столбце k матрицы P^j, где

$$P = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ p_0 & p_1 & p_2 & \cdots & p_{n-1} \end{pmatrix} \tag{73}$$

называется *сопровождающей матрицей* $p(u)$. (Индексы i, j, k в нашей ситуации пробегают значения от 0 до $n-1$, а не от 1 до n.) Тензор удобен для транспонирования, если $T_{ijk} = t_{ikj}$ — частные слои (T_{ijk}) для $k = 0, 1, 2, \ldots, n-1$ просто заданы матрицами

$$I \qquad P \qquad P^2 \qquad \ldots \qquad P^{n-1}. \tag{74}$$

Первые строки матриц в (74) — это соответственно единичные векторы $(1,0,0,\ldots,0)$, $(0,1,0,\ldots,0)$, $(0,0,1,\ldots,0)$, $\ldots$, $(0,0,0,\ldots,1)$; следовательно, линейная комбинация $\sum_{k=0}^{n-1} v_k P^k$ будет нулевой матрицей тогда и только тогда, когда все коэффициенты v_k равны нулю. Кроме того, большинство этих линейных комбинаций в действительности — невырожденные матрицы, для которых

$$(w_0, w_1, \ldots, w_{n-1}) \sum_{k=0}^{n-1} v_k P^k = (0,0,\ldots,0)$$

тогда и только тогда, когда $\quad v(u)w(u) \equiv 0$ $\big($по модулю $p(u)\big)$,

где $v(u) = v_0 + v_1 u + \cdots + v_{n-1}u^{n-1}$ и $w(u) = w_0 + w_1 u + \cdots + w_{n-1}u^{n-1}$. Таким образом, $\sum_{k=0}^{n-1} v_k P^k$ является невырожденной матрицей тогда и только тогда, когда полином $v(u)$ кратен некоторому множителю $p(u)$. Сейчас мы готовы доказать требуемый результат.

Теорема W (Ш. Виноград, 1975). *Пусть $p(u)$ — нормированный полином степени n и его полное разложение на множители в данном бесконечном поле равно*

$$p(u) = p_1(u)^{e_1} \ldots p_q(u)^{e_q}. \tag{75}$$

Тогда ранг тензора (74), *соответствующего билинейным формам* (71), *равен $2n - q$ над этим полем.*

Доказательство. Билинейные формы можно вычислить только с $2n-q$ умножений в цепочке, используя правила (58)–(60) в соответствующем стиле. Таким образом, необходимо только доказать, что ранг $r \geq 2n - q$. Выше был установлен тот факт, что $\operatorname{rank}(T_{(ij)k}) = n$; следовательно, по лемме T любая $(n \times r)$-реализация (A, B, C) тензора (T_{ijk}) имеет $\operatorname{rank}(C) = n$ ("rank" в переводе означает "ранг". — *Прим. перев.*). Наша стратегия — снова использовать лемму T для нахождения вектора $(v_0, v_1, \ldots, v_{n-1})$, который обладает следующими двумя свойствами.

i) Вектор $(v_0, v_1, \ldots, v_{n-1})C$ имеет самое большее $q + r - n$ ненулевых коэффициентов.

ii) Матрица $v(P) = \sum_{k=0}^{n-1} v_k P^k$.

Эти свойства и лемма T доказывают, что $q + r - n \geq n$. Следовательно, тождество

$$\sum_{l=1}^{r} a_{il} b_{jl} \left(\sum_{k=0}^{n-1} v_k c_{kl} \right) = v(P)_{ij}$$

показывает, как реализовать тензор $v(P)$ размера $n \times n \times 1$ ранга n с помощью $q+r-n$ умножений в цепочке.

Для удобства можно предположить, что первые n столбцов матрицы C линейно независимы. Пусть D — такая матрица размера $n \times n$, что первые n столбцов матрицы DC равны тождественной матрице. Наша цель будет достигнута, если существует линейная комбинация $(v_0, v_1, \ldots, v_{n-1})$ максимум q строк из D, что $v(P)$ не вырожден; данный вектор удовлетворяет условиям (i) и (ii).

Поскольку строки D линейно независимы, неприводимый множитель $p_\lambda(u)$ не может делить полиномы, соответствующие каждой строке. Пусть задан вектор

$$w = (w_0, w_1, \ldots, w_{n-1}),$$

covered(w) означает множество всех λ, такое, что $w(u)$ не кратно $p_\lambda(u)$. Можно найти линейную комбинацию v и w $v + \alpha w$ двух векторов, такую, что

$$\text{covered}(v + \alpha w) = \text{covered}(v) \cup \text{covered}(w) \tag{76}$$

для некоторого α, принадлежащего полю. Смысл этого состоит в том, что если λ покрыты v или w, но не обоими, то λ покрыта $v + \alpha w$ для всех ненулевых α; если λ покрыты обоими векторами v и w, но λ не покрыта $v + \alpha w$, то λ покрыта $v + \beta w$ для всех $\beta \neq \alpha$. Если проверить $q + 1$ различное значение α, то по крайней мере одно будет удовлетворять (76). Таким способом можно систематически строить линейную комбинацию самое большее q строк матрицы D, покрывающих все λ для $1 \leq \lambda \leq q$. ▮

Одним из наиболее важных следствий теоремы W является то, что ранг тензора может зависеть от поля, из которого получаются элементы реализации (A, B, C). Например, рассмотрим тензор, соответствующий циклической свертке степени 5; это эквивалентно умножению полиномов mod $p(u) = u^5 - 1$. Над полем рациональных чисел полным разложением $p(u)$ согласно упр. 4.6.2–32 является $(u - 1) \times (u^4 + u^3 + u^2 + u + 1)$. Таким образом, ранг тензора равен $10 - 2 = 8$. С другой стороны, полное разложение над действительными числами в терминах числа $\phi = \frac{1}{2}(1 + \sqrt{5})$ имеет вид $(u - 1)(u^2 + \phi u + 1)(u^2 - \phi^{-1} u + 1)$, поэтому ранг равен только 7, если допустить появление в A, B, C произвольных действительных чисел. При разложении над комплексными числами ранг равен 5. Этот феномен не имеет места для двумерных тензоров (матриц), где ранг можно определить, вычислив определители подматриц и проверив, не равен ли он нулю. Ранг матрицы не изменяется, когда поле, содержащее ее элементы, является вложенным в большее поле, однако ранг тензора *может* уменьшиться, когда поле становится большим.

В статье, в которой впервые была приведена теорема W [*Math. Systems Theory* **10** (1977), 169–180], Виноград идет дальше, показывая, что *все* реализации (71)

в $2n - q$ умножений в цепочке соответствуют использованию (59), когда q больше 1. Кроме того, он показал, что единственный метод вычисления коэффициентов $x(u)y(u)$ с $\deg(x) + \deg(y) + 1$ умножений в цепочке — это использование интерполяции или (58) с полиномом, который расщепляется на различные линейные множители в поле. Наконец, Виноград доказал, что метод вычисления $x(u)y(u) \bmod p(u)$ с $2n - 1$ умножений в цепочке, где $q = 1$, по существу, является (60). Этот результат справедлив для *всех* цепочек полиномов, если только они "нормальные". Виноград расширил свои результаты на полиномы от нескольких переменных в работе *SICOMP* **9** (1980), 225–229.

Ранг произвольных тензоров размера $m \times n \times 2$ в соответствующем большом поле определен Джозефом Яя (Joseph Ja'Ja'), *SICOMP* **8** (1979), 443–462; *JACM* **27** (1980), 822–830; см. также его интересное обсуждение коммутативных билинейных форм в работе *SICOMP* **9** (1980), 713–728). Однако проблема вычисления ранга произвольного тензора размера $n \times n \times n$ над любым конечным полем является NP-полной [J. Håstad, *Journal of Algorithms* **11** (1990), 644–654].

Для дополнительного чтения. В этом разделе была кратко затронута очень большая область, в которой возникает множество прекрасных теорий. Значительно более исчерпывающий материал можно найти в книгах А. Бородина и Дж. Мунро [A. Borodin and I. Munro, *Computational Complexity of Algebraic and Numeric Problems* (New York: American Elsevier, 1975)]; Д. Бини и В. Пана [D. Bini and V. Pan, *Polynomial and Matrix Computations* **1** (Boston: Birkhäuser, 1994)]; П. Бергиссера, М. Клаусена и М. Амина Чокролахи [P. Bürgisser, M. Clausen, and M. Amin Shokrollahi, *Algebraic Complexity Theory* (Heidelberg: Springer, 1997)].

УПРАЖНЕНИЯ

1. [*15*] Каков "хороший" метод вычисления "нечетного" полинома

$$u(x) = u_{2n+1}x^{2n+1} + u_{2n-1}x^{2n-1} + \cdots + u_1x?$$

▶ **2.** [*M20*] Вместо того чтобы вычислять $u(x + x_0)$ с помощью шагов H1 и H2, приведенных в разделе, обсудите применение правила Горнера (2), когда умножение и сложение *полиномов* используются вместо арифметики в области коэффициентов.

3. [*20*] Предложите метод, аналогичный правилу Горнера, для вычисления полинома от двух переменных $\sum_{i+j\le n} u_{ij}x^iy^j$. (Этот полином имеет $(n + 1)(n + 2)/2$ коэффициентов, и его "общая степень" равна n.) Вычислите используемое вами количество операций сложения и умножения.

4. [*M20*] В разделе показано, что схема (3) превосходит правило Горнера, когда вычисляется полином с вещественными коэффициентами в комплексной точке z. Сравните (3) с правилом Горнера, когда и коэффициенты, и переменная z — комплексные числа. Как много (вещественных) умножений и сложений-вычитаний требуется для каждого метода?

5. [*M15*] Подсчитайте количество умножений и сложений, необходимых по правилу второго порядка (4).

6. [*22*] (Л. де Джонг (L. de Jong) и Я. Ван Лиивен (J. van Leeuwen).) Покажите, как можно улучшить шаги S1, ..., S4 алгоритма Шо-Трауба (Shaw-Traub), вычисляя Джонг Л. С. (Jong, Lieuwe Sytse de) Ван Лиивен Ж. (van Leeuwen, Jan) Шо М. М. (Shaw, Mary Margaret) Трауб Ж. Ф. (Traub, Joseph Frederick) только приблизительно $\frac{1}{2}n$ степеней x_0.

7. [*M25*] Как можно вычислить $\beta_0, \ldots, \beta_n$, когда (6) имеет значение $u(x_0 + kh)$ для всех целых k?

8. [*M20*] Факториальная степень $x^{\underline{k}}$ определяется таким образом:

$$k!\binom{x}{k} = x(x-1)\ldots(x-k+1).$$

Объясните, как вычислить $u_n x^{\underline{n}} + \cdots + u_1 x^{\underline{1}} + u_0$ с максимум n умножениями и $2n-1$ сложениями, начиная с x и $n+3$ постоянных $u_n, \ldots, u_0, 1, n-1$.

9. [*M25*] (Г. Дж. Райзер (H. J. Ryser).) Покажите, что если $X = (x_{ij})$ — матрица размера $n \times n$, то

$$\operatorname{per}(X) = \sum (-1)^{n-\epsilon_1-\cdots-\epsilon_n} \prod_{1\le i\le n} \sum_{1\le j\le n} \epsilon_j x_{ij}.$$

Суммирование осуществляется по всем 2^n наборам $\epsilon_1, \ldots, \epsilon_n$, независимо равным 0 или 1. Подсчитайте количество операций сложения и умножения, требуемых для вычисления $\operatorname{per}(X)$ по приведенной формуле.

10. [*M21*] Перманент матрицы $X = (x_{ij})$ размера $n \times n$ можно вычислить следующим образом: начать с n величин $x_{11}, x_{12}, \ldots, x_{1n}$. Для $1 \le k < n$ предположим, что $\binom{n}{k}$ величин A_{kS} вычислены для всех подмножеств множества S, $\{1, 2, \ldots, n\}$, содержащих k элементов, где $A_{kS} = \sum x_{1j_1} \ldots x_{kj_k}$ суммируется по всем $k!$ перестановкам $j_1 \ldots j_k$ элементов множества S; затем составляются все суммы вида

$$A_{(k+1)S} = \sum_{j\in S} A_{k(S\setminus\{j\})}\, x_{(k+1)j}.$$

Мы получим $\operatorname{per}(X) = A_{n\{1,\ldots,n\}}$. Сколько операций сложения и умножения потребуется для этого метода? Сколько ячеек понадобится для временного хранения?

11. [*M46*] Существует ли какой-нибудь метод вычисления перманента обычной матрицы размера $n \times n$, использующий менее 2^n арифметических операций?

12. [*M50*] Каково минимальное количество умножений, необходимых для получения произведения двух матриц размера $n \times n$? Чему равен наименьший показатель ω, такой, что $O(n^{\omega+\epsilon})$ умножений достаточно для всех $\epsilon > 0$? (Найдите хорошие верхнюю и нижнюю грани как для малых, так и для больших n.)

13. [*M23*] Найдите преобразование, обратное обычному дискретному преобразованию Фурье (37), выразив $F(t_1, \ldots, t_n)$ в терминах значений $f(s_1, \ldots, s_n)$. [*Указание.* См. уравнение 1.2.9–(13).]

▸ **14.** [*HM28*] (*Быстрое преобразование Фурье.*) Покажите, что с помощью схемы . (40) можно вычислить одномерное дискретное преобразование Фурье

$$f(s) = \sum_{0\le t<2^n} F(t)\,\omega^{st}, \qquad \omega = e^{2\pi i/2^n}, \qquad 0 \le s < 2^n,$$

используя арифметику комплексных чисел. Оцените число выполняемых арифметических операций.

▸ **15.** [*HM28*] *n-е разностное отношение* $f(x_0, x_1, \ldots, x_n)$ функции $f(x)$ в $n+1$ различных точках $x_0, x_1, \ldots, x_n$ определяется формулой

$$f(x_0, x_1, \ldots, x_n) = \big(f(x_0, x_1, \ldots, x_{n-1}) - f(x_1, \ldots, x_{n-1}, x_n)\big)/(x_0 - x_n)$$

для $n > 0$. Таким образом, $f(x_0, x_1, \ldots, x_n) = \sum_{k=0}^{n} f(x_k)/\prod_{0\le j\le n,\, j\ne k}(x_k - x_j)$ — симметричная функция $n+1$ аргументов. (а) Докажите, что $f(x_0, \ldots, x_n) = f^{(n)}(\theta)/n!$ для

некоторого θ, лежащего между $\min(x_0,\ldots,x_n)$ и $\max(x_0,\ldots,x_n)$, если n-я производная $f^{(n)}(x)$ существует и конечна. [*Указание.* Докажите тождество

$$f(x_0,x_1,\ldots,x_n)=\int_0^1 dt_1\int_0^{t_1}dt_2\ldots\int_0^{t_{n-1}}dt_n f^{(n)}(x_0(1-t_1)+x_1(t_1-t_2)+\cdots \\ +x_{n-1}(t_{n-1}-t_n)+x_n(t_n-0)).$$

Эта формула также определяет $f(x_0,x_1,\ldots,x_n)$, когда x_j не различны.] (b) Если $y_j=f(x_j)$, покажите, что $\alpha_j=f(x_0,\ldots,x_j)$ в интерполяционном полиноме Ньютона (42).

16. [*M22*] Как можно легко вычислить коэффициенты $u_{[n]}(x)=u_nx^n+\cdots+u_0$, если заданы значения x_0, x_1, $\ldots$, x_{n-1}, α_0, α_1, $\ldots$, α_n в интерполяционном полиноме Ньютона (42)?

17. [*M20*] Покажите, что интерполяционная формула (45) сводится к очень простому выражению, включающему биномиальные коэффициенты, когда $x_k=x_0+kh$ для $0\le k\le n$. [*Указание.* См. упр. 1.2.6–48.]

18. [*M20*] Если в схеме четвертой степени (9) сделать замену

$$y=(x+\alpha_0)x+\alpha_1,\qquad u(x)=((y-x+\alpha_2)y+\alpha_3)\alpha_4,$$

какие формулы для вычисления α_j в терминах u_k следует взять вместо (10)?

▸ **19.** [*M24*] Объясните, как определить адаптированные коэффициенты α_0, α_1, $\ldots$, α_5 в (11) из коэффициентов u_5, $\ldots$, u_1, u_0 $u(x)$, и найдите α_j для полинома $u(x)=x^5+5x^4-10x^3-50x^2+13x+60$.

▸ **20.** [*21*] Напишите программу для вычисления полинома пятой степени согласно схеме (11) для машины MIX. Попытайтесь сделать программу настолько эффективной, насколько это возможно, слегка модифицировав (11). Используйте в машине MIX арифметические операторы с плавающей точкой FADD и FMUL, описанные в разделе 4.2.1.

21. [*20*] Найдите два дополнительных способа вычисления полинома $x^6+13x^5+49x^4+33x^3-61x^2-37x+3$ по схеме (12), используя два корня (15), которые не рассмотрены в разделе.

22. [*18*] В какой схеме вычисления $x^6-3x^5+x^4-2x^3+x^2-3x-1$ используется метод Пана (16)?

23. [*HM30*] (Дж. Ив (J. Eve).) Пусть $f(z)=a_nz^n+a_{n-1}z^{n-1}+\cdots+a_0$ — полином степени n с действительными коэффициентами, имеющий по крайней мере $n-1$ корней с неотрицательной действительной частью. Пусть

$$g(z)=a_nz^n+a_{n-2}z^{n-2}+\cdots+a_{n\bmod 2}z^{n\bmod 2},$$
$$h(z)=a_{n-1}z^{n-1}+a_{n-3}z^{n-3}+\cdots+a_{(n-1)\bmod 2}z^{(n-1)\bmod 2}.$$

Предположим, что $h(z)$ не равно тождественно нулю.

a) Покажите, что $g(z)$ имеет по крайней мере $n-2$ мнимых корней (т. е. корней, действительная часть которых равна нулю) и $h(z)$ имеет по крайней мере $n-3$ мнимых корней. [*Указание.* Рассмотрите, сколько раз траектория $f(z)$ обходит начало координат, когда z перемещается по пути, показанному на рис. 16, для достаточно большого радиуса R.]

b) Докажите, что квадраты корней $g(z)=0$ и $h(z)=0$ — все действительные числа.

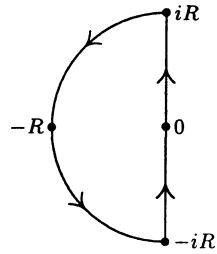


Рис. 16. Доказательство теоремы Ива.

▸ **24.** [*M24*] Найдите значения c и α_k, β_k, удовлетворяющие условиям теоремы E, для полинома $u(x) = (x+7)(x^2+6x+10)(x^2+4x+5)(x+1)$. Выберите эти значения так, чтобы $\beta_2 = 0$. Приведите два различных решения.

25. [*M20*] Когда конструкция в доказательстве теоремы M применяется к неэффективной цепочке полиномов

$$\lambda_1 = \alpha_1 + \lambda_0, \qquad \lambda_2 = -\lambda_0 - \lambda_0, \qquad \lambda_3 = \lambda_1 + \lambda_1, \qquad \lambda_4 = \alpha_2 \times \lambda_3,$$
$$\lambda_5 = \lambda_0 - \lambda_0, \qquad \lambda_6 = \alpha_6 - \lambda_5, \qquad \lambda_7 = \alpha_7 \times \lambda_6, \qquad \lambda_8 = \lambda_7 \times \lambda_7,$$
$$\lambda_9 = \lambda_1 \times \lambda_4, \qquad \lambda_{10} = \alpha_8 - \lambda_9, \qquad \lambda_{11} = \lambda_3 - \lambda_{10},$$

как можно $\beta_1, \beta_2, \ldots, \beta_9$ выразить в терминах $\alpha_1, \ldots, \alpha_8$?

▸ **26.** [*M21*] (a) Получите цепочку полиномов, соответствующих правилу Горнера вычисления полиномов степени $n = 3$. (b) Используя конструкцию из доказательства теоремы A, выразите κ_1, κ_2, κ_3 и полином в конце цепочки $u(x)$ в терминах β_1, β_2, β_3, β_4 и x. (c) Покажите, что множество полиномов в конце цепочки, полученное в (b), когда все β_1, β_2, β_3 и β_4 независимо принимают все действительные значения, не содержит некоторые элементы множества, полученного в конце цепочки (a).

27. [*M22*] Пусть R — множество, которое включает все строки размерности $(n+1)$ действительных чисел $(q_n, \ldots, q_1, q_0)$, таких, что $q_n \neq 0$. Докажите, что R имеет более n степеней свободы.

28. [*HM20*] Покажите, что если $f_0(\alpha_1, \ldots, \alpha_s)$, $\ldots$, $f_s(\alpha_1, \ldots, \alpha_s)$ — полиномы от многих переменных с целыми коэффициентами, то существует ненулевой полином $g(x_0, \ldots, x_s)$ с целыми коэффициентами, такой, что $g(f_0(\alpha_1, \ldots, \alpha_s), \ldots, f_s(\alpha_1, \ldots, \alpha_s)) = 0$ для всех действительных $\alpha_1, \ldots, \alpha_s$. (Следовательно, любая цепочка полиномов с s параметрами имеет максимум s степеней свободы.) [*Указание.* Воспользуйтесь теоремами об "алгебраической зависимости", которые можно найти, например, у Б. Л. Ван дер Вардена (B. L. van der Waerden, *Modern Algebra*, translated by Fred Blum (New York: Ungar, 1949, раздел 64)). Русский перевод: Ван дер Варден, *Современная алгебра*. — М.: ОГИЗ, 1947.]

▸ **29.** [*M20*] Пусть все $R_1, R_2, \ldots, R_m$ множества строк действительных чисел размерности $(n+1)$ имеют максимум t степеней свободы. Покажите, что объединение $R_1 \cup R_2 \cup \cdots \cup R_m$ также имеет максимальное число степеней свободы t.

▸ **30.** [*M28*] Докажите, что цепочка полиномов с m_c операциями умножения в цепочке и m_p операциями умножения параметров имеет максимальное число степеней свободы, равное $2m_c + m_p + \delta_{0m_c}$. [*Указание.* Обобщение теоремы M показывает, что первое умножение в цепочке и каждое умножение параметров может, по существу, вводить только один новый параметр в множество результатов.]

31. [*M23*] Докажите, что цепочка полиномов, допускающая вычисление всех *нормированных* полиномов степени n, имеет по крайней мере $\lfloor n/2 \rfloor$ умножений и n сложений-вычитаний.

32. [*M24*] Найдите цепочку полиномов минимальной возможной длины, которая может вычислить все полиномы вида $u_4x^4 + u_2x^2 + u_0$. Докажите, что эта цепочка минимальна.

▸ **33.** [*M25*] Пусть $n \geq 3$ нечетное. Докажите, что цепочка полиномов с $\lfloor n/2 \rfloor + 1$ шагами умножений не может вычислить все полиномы степени n, если она не имеет хотя бы $n+2$ шага сложения-умножения. [*Указание.* См. упр. 30.]

34. [*M26*] Пусть $\lambda_0, \lambda_1, \ldots, \lambda_r$ — цепочка полиномов, в которой все шаги сложений и вычитаний — шаги параметра и среди которых имеется по крайней мере один параметр умножения. Предположим, что эта схема имеет m умножений и $k = r - m$ сложений-вычитаний и что вычисление полинома по цепочке имеет максимальную степень n. Докажите, что все полиномы, вычисляемые по этой цепочке, коэффициенты которых при x^n не равны нулю, могут быть вычислены по другой цепочке, которая имеет максимум m умножений, максимум k сложений и не имеет вычитаний. Кроме того, последний шаг новой цепочки должен быть только умножением параметра.

▸ **35.** [*M25*] Покажите, что любая цепочка полиномов, которая вычисляет полином общего вида четвертой степени, используя три умножения, должна иметь по крайней мере пять сложений-вычитаний. [*Указание.* Предположите, что существует только четыре сложения-вычитания, и покажите, что применимо упр. 34, поэтому схема должна быть частного вида, который не представляет все полиномы четвертой степени.]

36. [*M27*] Продолжая предыдущее упражнение, покажите, что любая цепочка полиномов, которая вычисляет обычный полином седьмой степени и использует только четыре умножения, должна иметь по крайней мере семь сложений-вычитаний.

37. [*M21*] (T. C. Моцкин (T. S. Motzkin).) Покажите, что "почти все" рациональные функции вида

$$(u_nx^n + u_{n-1}x^{n-1} + \cdots + u_1x + u_0)/(x^n + v_{n-1}x^{n-1} + \cdots + v_1x + v_0)$$

с коэффициентами из поля S можно вычислить, воспользовавшись схемой

$$\alpha_1 + \beta_1/(x + \alpha_2 + \beta_2/(x + \cdots + \beta_n/(x + \alpha_{n+1})\ldots))$$

для соответствующих α_j, β_j из S. (Эта схема цепных дробей имеет n делений и $2n$ сложений; под "почти всеми" рациональными функциями понимаются все функции, за исключением тех, коэффициенты которых удовлетворяют некоторым нетривиальным полиномиальным уравнениям.) Определите α_k и β_j для рациональной функции $(x^2+10x+29)/(x^2+8x+19)$.

▸ **38.** [*HM32*] (В. Я. Пан, 1962.) Назначение данного упражнения — доказать, что правило Горнера действительно оптимально, если предварительно не адаптировать коэффициент. Необходимо произвести n умножений и n сложений для вычисления $u_nx^n + \cdots + u_1x + u_0$, если заданы переменные $u_n, \ldots, u_1, u_0, x$ и произвольные постоянные. Рассмотрим цепочки, такие, как раньше, но так, что $u_n, \ldots, u_1, u_0, x$ являются переменными. Например, можно сказать, что $\lambda_{-j-1} = u_j$, $\lambda_0 = x$. Чтобы показать, что правило Горнера наилучшее, удобно доказать более общую теорему. Пусть $A = (a_{ij})$, $0 \leq i \leq m$, $0 \leq j \leq n$, — действительная матрица размера $(m+1) \times (n+1)$ и ранга $n+1$ и пусть $B = (b_0, \ldots, b_m)$ — действительный вектор. Докажите, что *любая цепочка полиномов, которая вычисляет*

$$P(x; u_0, \ldots, u_n) = \sum_{i=0}^{m} (a_{i0}u_0 + \cdots + a_{in}u_n + b_i)x^i,$$

включает по крайней мере n умножений в цепочке. (Заметим, что это не только означает, что рассматривается несколько фиксированных цепочек, параметры α_j которых — определенные значения, зависящие от A и B. Это означает, что u цепочка, u значения

α_j могут зависеть от заданных матрицы A и вектора B. Не важно, как выбраны A, B и значения α_j; невозможно вычислить $P(x; u_0, \ldots, u_n)$, не выполнив n "умножений в цепочке".) Из предположения, что ранг матрицы A равен $n+1$, вытекает, что $m \geq n$. [*Указание.* Покажите, что из любой такой схемы можно вывести другую схему, которая имеет меньше умножений в цепочке и n уменьшено на единицу.]

39. [*M29*] (Т. С. Моцкин, 1954.) Покажите, что схему вида

$$w_1 = x(x+\alpha_1)+\beta_1, \qquad w_k = w_{k-1}(w_1+\gamma_k x+\alpha_k)+\delta_k x+\beta_k \quad \text{для } 1<k\leq m,$$

где α_k, β_k — действительные числа и γ_k, δ_k — целые числа, можно использовать для вычисления всех нормированных полиномов степени $2m$ над полем действительных чисел. (Для различных полиномов можно по-разному выбирать α_k, β_k, γ_k и δ_k.) Попытайтесь, когда возможно, предположить, что $\delta_k = 0$.

40. [*M41*] Можно ли нижнюю грань количества операций умножения в теореме С поднять от $\lfloor n/2\rfloor+1$ до $\lceil n/2\rceil+1$ (см. упр. 33)?

41. [*22*] Покажите, что действительную и мнимую части $(a+bi)(c+di)$ можно получить с помощью трех умножений и пяти сложений действительных чисел, где два сложения включают только a и b.

42. [*36*] (М. Патерсон (M. Paterson) и Л. Штокмейер (L. Stockmeyer).) (a) Докажите, что цепочка полиномов с $m \geq 2$ умножениями имеет максимум m^2+1 степеней свободы. (b) Покажите, что для всех $n \geq 2$ существуют такие полиномы степени n, все коэффициенты которых — нули или единицы, которые не могут быть вычислены любой цепочкой полиномов с менее чем $\lfloor\sqrt{n}\rfloor$ умножениями, если все параметры α_j должны быть целыми числами. (c) Покажите, что любой полином степени n с целыми коэффициентами можно вычислить при помощи полностью целочисленного алгоритма, который выполняет максимум $2\lfloor\sqrt{n}\rfloor$ умножений, если количество произведенных сложений не имеет значения.

43. [*22*] Объясните, как вычислить $x^n+\cdots+x+1$, выполнив $2l(n+1)-2$ операций умножения и $l(n+1)$ сложения (не операций деления или вычитания), где $l(n)$ — функция, рассматриваемая в разделе 4.6.3.

▸ **44.** [*M25*] Покажите, что любой нормированный полином $u(x) = x^n + u_{n-1}x^{n-1}+\cdots+u_0$ можно вычислить, выполнив $\frac{1}{2}n+O(\log n)$ умножений и $\leq \frac{5}{4}n$ сложений и использовав параметры $\alpha_1, \alpha_2, \ldots$ — полиномы от $u_{n-1}, u_{n-2}, \ldots$ с целыми коэффициентами. [*Указание.* Сначала рассмотрите случай, когда $n = 2^l$.]

▸ **45.** [*HM22*] Пусть (t_{ijk}) — тензор размера $m\times n\times s$ и пусть F, G, H — невырожденные матрицы соответственно размера $m\times m$, $n\times n$ и $s\times s$. Если

$$T_{ijk} = \textstyle\sum_{i'=1}^{m}\sum_{j'=1}^{n}\sum_{k'=1}^{s} F_{ii'}G_{jj'}H_{kk'}t_{i'j'k'}$$

для всех i, j, k, докажите, что тензор (T_{ijk}) имеет такой же ранг, как и (t_{ijk}). [*Указание.* Рассмотрите, что произойдет, когда F^{-1}, G^{-1}, H^{-1} применить таким же способом к (T_{ijk}).]

46. [*M28*] Докажите, что все пары (z_1, z_2) билинейных форм от (x_1, x_2) и (y_1, y_2) можно вычислить с максимум тремя умножениями в цепочке. Другими словами, покажите, что каждый тензор размера $2\times 2\times 2$ имеет ранг ≤ 3.

47. [*M25*] Докажите, что для всех m, n и s существует тензор размера $m\times n\times s$, ранг которого равен по крайней мере $\lceil mns/(m+n+s)\rceil$. Обратно, покажите, что каждый тензор размера $m\times n\times s$ имеет ранг, не превышающий $mns/\max(m,n,s)$.

48. [*M21*] Если (t_{ijk}) и (t'_{ijk}) — тензоры размера $m\times n\times s$ и $m'\times n'\times s'$ соответственно, то их *прямая сумма* $(t_{ijk})\oplus(t'_{ijk}) = (t''_{ijk})$ является тензором размера $(m+m')\times(n+n')\times(s+s')$, определенным как $t''_{ijk} = t_{ijk}$, если $i\leq m$, $j\leq n$, $k\leq s$, или как $t''_{ijk} = t'_{i-m,j-n,k-s}$,

если $i > m$, $j > n$, $k > s$; в остальных случаях $t''_{ijk} = 0$. Их прямое произведение $(t_{ijk}) \otimes (t'_{ijk}) = (t'''_{ijk})$ — это тензор размера $mm' \times nn' \times ss'$, определенный как $t_{\langle ii'\rangle\langle jj'\rangle\langle kk'\rangle} = t_{ijk}t'_{i'j'k'}$. Получите верхние грани $\operatorname{rank}(t'''_{ijk}) \le \operatorname{rank}(t_{ijk}) + \operatorname{rank}(t'_{ijk})$ и $\operatorname{rank}(t'''_{ijk}) \le \operatorname{rank}(t_{ijk}) \cdot \operatorname{rank}(t'_{ijk})$.

▸ **49.** [*HM25*] Покажите, что ранг тензора (t_{ijk}) размера $m \times n \times 1$ такой же, как его ранг, когда он записан в виде матрицы (t_{ij1}) размера $m \times n$, соответствующий традиционному определению ранга матрицы как максимального числа линейно независимых строк.

50. [*HM20*] (Ш. Виноград.) Пусть (t_{ijk}) — тензор размера $mn \times n \times m$, соответствующий умножению матрицы размера $m \times n$ на вектор-столбец размерности $n \times 1$. Докажите, что ранг тензора (t_{ijk}) равен mn.

51. [*M24*] (Ш. Виноград.) Придумайте алгоритм для циклической свертки степени 2, который использует 2 умножения и 4 сложения, не считая операций с x_i. Подобно этому придумайте алгоритм для степени 3, используя 4 умножения и 11 сложений. (См. соотношение (69), которое позволяет решить аналогичную задачу для степени 4.)

52. [*M25*] (Ш. Виноград.) Пусть $n = n'n''$, где $n' \perp n''$. Пусть заданы нормальные схемы для циклических сверток степени n' и n'', использующие соответственно (m', m'') умножений в цепочке, (p', p'') умножений параметров и (a', a'') сложений. Покажите, как построить нормальную схему для циклической свертки степени n, используя $m'm''$ умножений в цепочке, $p'n'' + m'p''$ умножений параметров и $a'n'' + m'a''$ сложений.

53. [*HM40*] (Ш. Виноград.) Пусть ω — m-й комплексный корень из единицы. Рассмотрим одномерное дискретное преобразование Фурье

$$f(s) = \sum_{t=1}^{m} F(t)\,\omega^{st}, \qquad \text{для } 1 \le s \le m.$$

a) Когда $m = p^e$ — степень нечетного простого числа, покажите, что эффективные нормальные схемы вычисления циклических сверток степеней $(p-1)p^k$ для $0 \le k < e$ приводят к эффективным алгоритмам вычисления преобразования Фурье m комплексных чисел. Предложите подобную конструкцию для случая, когда $p = 2$.

b) Когда $m = m'm''$ и $m' \perp m''$, покажите, что алгоритмы преобразования Фурье для m' и m'' можно скомпоновать так, чтобы получить алгоритм преобразования Фурье для m элементов.

54. [*M23*] Теорема W доказана для бесконечных полей. Сколько элементов должно содержаться в конечном поле, чтобы доказательство теоремы W оставалось справедливым?

55. [*HM22*] Определите ранг тензора (74), когда P — *произвольная* матрица размера $n \times n$.

56. [*M32*] (У. Штрассен (V. Strassen).) Покажите, что любая цепочка полиномов, которая вычисляет множество *квадратичных форм* $\sum_{i=1}^{n}\sum_{j=1}^{n}\tau_{ijk}x_ix_k$ для $1 \le k \le s$, должна использовать, в целом, по крайней мере $\frac{1}{2}\operatorname{rank}(\tau_{ijk}+\tau_{jik})$ умножений в цепочке. [*Указание.* Покажите, что минимальное число умножений в цепочке равно минимальному рангу (t_{ijk}), взятому по всем тензорам (t_{ijk}), таким, что $t_{ijk} + t_{jik} = \tau_{ijk} + \tau_{jik}$ для всех i, j, k.] Воспользуйтесь этим для доказательства того, что любая цепочка полиномов, вычисляющая множество билинейных форм вида (47) и соответствующая тензору (t_{ijk}), нормален он или анормален, должна использовать хотя бы $\frac{1}{2}\operatorname{rank}(t_{ijk})$ умножений в цепочке.

57. [*M20*] Покажите, что быстрое преобразование Фурье можно использовать для вычисления коэффициентов произведения двух заданных полиномов степени n $x(u)y(u)$, произведя $O(n \log n)$ операций (точно) сложений и умножений комплексных чисел. [*Указание.* Рассмотрите произведение коэффициентов преобразования Фурье.]

58. [*HM28*] (a) Покажите, что любая реализация (A, B, C) тензора умножения полиномов (55) должна иметь следующие свойства: любая ненулевая линейная комбинация трех строк матрицы A должна быть вектором по крайней мере с четырьмя ненулевыми элементами и любая ненулевая комбинация четырех строк матрицы B должна иметь по крайней мере три ненулевых элемента. (b) Найдите реализацию (A, B, C) (55), которая использует в качестве элементов только 0, +1 и −1, где $r = 8$. Попытайтесь использовать настолько много нулей, насколько это возможно.

▸ **59.** [*M40*] (Г. Ж. Нусбаумер (H. J. Nussbaumer), 1980.) В разделе определено, что циклическая свертка двух последовательностей $(x_0, x_1, \ldots, x_{n-1})$ и $(y_0, y_1, \ldots, y_{n-1})$ — это последовательность $(z_0, z_1, \ldots, z_{n-1})$, где $z_k = x_0y_k + \cdots + x_ky_0 + x_{k+1}y_{n-1} + \cdots + x_{n-1}y_{k+1}$. Аналогично определим *отрицательную циклическую свертку*, но с

$$z_k = x_0y_k + \cdots + x_ky_0 - (x_{k+1}y_{n-1} + \cdots + x_{n-1}y_{k+1}).$$

Постройте эффективные алгоритмы для циклической и отрицательной циклической сверток для целых чисел, когда n — степень 2. Ваш алгоритм полностью должен обходиться целыми числами, и должно выполняться максимум $O(n \log n)$ умножений и максимум $O(n \log n \log\log n)$ сложений, вычитаний или делений четных чисел на 2. [*Указание.* Циклическая свертка порядка $2n$ может быть сведена к циклической и отрицательной циклической свертке порядка n, если воспользоваться (59).]

60. [*M27*] (В. Я. Пан.) Задача умножения матрицы размера $(m \times n)$ на матрицу размера $(n \times s)$ соответствует тензору $(t_{\langle i,j'\rangle\langle j,k'\rangle\langle k,i'\rangle})$ размера $mn \times ns \times sm$, где $t_{\langle i,j'\rangle\langle j,k'\rangle\langle k,i'\rangle} = 1$ тогда и только тогда, когда $i' = i$, $j' = j$ и $k' = k$. Ранг этого тензора $T(m, n, s)$ равен наименьшему числу r, такому, что числа $a_{ij'l}$, $b_{jk'l}$, $c_{ki'l}$ существуют и удовлетворяют соотношению

$$\sum_{\substack{1\le i\le m\\ 1\le j\le n\\ 1\le k\le s}} x_{ij}y_{jk}z_{ki} = \sum_{1\le l\le r} \Big(\sum_{\substack{1\le i\le m\\ 1\le j'\le n}} a_{ij'l}x_{ij'}\Big)\Big(\sum_{\substack{1\le j\le n\\ 1\le k'\le s}} b_{jk'l}y_{jk'}\Big)\Big(\sum_{\substack{1\le k\le s\\ 1\le i'\le m}} c_{ki'l}z_{ki'}\Big).$$

Пусть $M(n)$ — ранг тензора $T(n, n, n)$. Назначение упражнения — использовать симметрию такого трилинейного представления для эффективного вычисления реализации умножения матриц, элементы которых — целые числа, когда $m = n = s = 2\nu$. Для удобства разделим индексы $\{1, \ldots, n\}$ на два подмножества $O = \{1, 3, \ldots, n-1\}$ и $E = \{2, 4, \ldots, n\}$ с ν элементами в каждом и установим взаимно однозначное соответствие между подмножествами O и E по правилу: $\tilde{\imath} = i + 1$, если $i \in O$, $\tilde{\imath} = i - 1$, если $i \in E$. Таким образом получаем $\tilde{\tilde{\imath}} = i$ для всех индексов i.

a) Из тождества

$$abc + ABC = (a + A)(b + B)(c + C) - (a + A)bC - A(b + B)c - aB(c + C)$$

следует, что

$$\sum_{1\le i,j,k\le n} x_{ij}y_{jk}z_{ki} = \sum_{(i,j,k)\in S} (x_{ij} + x_{\tilde{k}\tilde{\imath}})(y_{jk} + y_{\tilde{\imath}\tilde{\jmath}})(z_{ki} + z_{\tilde{\jmath}\tilde{k}}) - \Sigma_1 - \Sigma_2 - \Sigma_3,$$

где $S = E{\times}E{\times}E \cup E{\times}E{\times}O \cup E{\times}O{\times}E \cup O{\times}E{\times}E$ — множество всех тройных индексов, включающих максимум один нечетный индекс; Σ_1 — сумма всех членов вида $(x_{ij} + x_{\tilde{k}\tilde{\imath}})y_{jk}z_{\tilde{\jmath}\tilde{k}}$ для $(i, j, k) \in S$ и Σ_2, Σ_3 — подобные суммы членов $x_{\tilde{k}\tilde{\imath}}(y_{jk} + y_{\tilde{\imath}\tilde{\jmath}})z_{ki}$, $x_{ij}y_{\tilde{\imath}\tilde{\jmath}}(z_{ki} + z_{\tilde{\jmath}\tilde{k}})$. Ясно, что S имеет $4\nu^3 = \frac{1}{2}n^3$ членов. Покажите, что каждую из сумм Σ_1, Σ_2, Σ_3 можно реализовать как сумму $3\nu^2$ трилинейных членов; кроме того, если 3ν тройных индексов вида $(i, i, \tilde{\imath})$, $(i, \tilde{\imath}, i)$ и $(\tilde{\imath}, i, i)$ не содержатся в S, можно

модифицировать Σ_1, Σ_2 и Σ_3 таким способом, что тождество останется справедливым без добавления каких-нибудь новых трилинейных членов. Следовательно, $M(n) \le \frac{1}{2}n^3 + \frac{9}{4}n^2 - \frac{3}{2}n$, когда n четное.

b) Используйте метод (a), чтобы показать, что две *независимые* задачи умножения матриц размера $m \times n \times s$ можно выполнить с $mns + mn + ns + sm$ некоммутативными умножениями.

61. [*M26*] Пусть (t_{ijk}) — тензор над произвольным полем. Определим $\operatorname{rank}_d(t_{ijk})$ как минимальное значение r, такое, что существует реализация вида

$$\sum_{l=1}^{r} a_{il}(u)b_{jl}(u)c_{kl}(u) = t_{ijk}u^d + O(u^{d+1}),$$

где $a_{il}(u)$, $b_{jl}(u)$, $c_{kl}(u)$ — полиномы от u над полем. Таким образом, rank_0 — обычный ранг тензора. Докажите, что

a) $\operatorname{rank}_{d+1}(t_{ijk}) \le \operatorname{rank}_d(t_{ijk})$;

b) $\operatorname{rank}(t_{ijk}) \le \binom{d+2}{2} \operatorname{rank}_d(t_{ijk})$;

c) $\operatorname{rank}_d((t_{ijk}) \oplus (t'_{ijk})) \le \operatorname{rank}_d(t_{ijk}) + \operatorname{rank}_d(t'_{ijk})$ в смысле упр. 48;

d) $\operatorname{rank}_{d+d'}((t_{ijk}) \otimes (t'_{ijk})) \le \operatorname{rank}_d(t_{ijk}) \cdot \operatorname{rank}_{d'}(t'_{ijk})$;

e) $\operatorname{rank}_{d+d'}((t_{ijk}) \otimes (t'_{ijk})) \le \operatorname{rank}_{d'}(r(t'_{ijk}))$, где $r = \operatorname{rank}_d(t_{ijk})$ и rT означает прямую сумму $T \oplus \cdots \oplus T$ r копий T.

62. [*M24*] *Гранью ранга* тензора (t_{ijk}), обозначенной через $\underline{\operatorname{rank}}(t_{ijk})$, называется $\min_{d \ge 0} \operatorname{rank}_d(t_{ijk})$, где rank_d определен в упр. 61. Докажите, что тензор $\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)\left(\begin{smallmatrix}0&1\\0&0\end{smallmatrix}\right)$ имеет ранг 3, но грань ранга 2 над каждым полем.

63. [*HM30*] Пусть $T(m,n,s)$ — тензор умножения матрицы, как в упр. 60, и пусть $M(N)$ — ранг $T(N,N,N)$.

a) Покажите, что $T(m,n,s) \otimes T(M,N,S) = T(mM,nN,sS)$.

b) Покажите, что $\operatorname{rank}_d(T(mN,nN,sN)) \le \operatorname{rank}_d(M(N)T(m,n,s))$ (см. упр. 61, (e)).

c) Если $T(m,n,s)$ имеет ранг $\le r$, покажите, что $M(N) = O(N^{\omega(m,n,s,r)})$ при $N \to \infty$, где $\omega(m,n,s,r) = 3\log r/\log mns$.

d) Если грань ранга тензора $T(m,n,s) \le r$, покажите, что $M(N) = O(N^{\omega(m,n,s,r)}(\log N)^2)$.

64. [*M30*] (A. Шёнхаге (A. Schönhage).) Покажите, что $\operatorname{rank}_2(T(3,3,3)) \le 21$, так как $M(N) = O(N^{2.78})$.

▸ **65.** [*M27*] (A. Шёнхаге.) Покажите, что

$$\operatorname{rank}_2(T(m,1,n) \oplus T(1,(m-1)(n-1),1)) = mn+1.$$

Указание. Рассмотрите трилинейную форму

$$\sum_{i=1}^{m}\sum_{j=1}^{n}(x_i + uX_{ij})(y_j + uY_{ij})(Z + u^2 z_{ij}) - (x_1 + \cdots + x_m)(y_1 + \cdots + y_n)Z,$$

когда $\sum_{i=1}^m X_{ij} = \sum_{j=1}^n Y_{ij} = 0$.

66. [*HM33*] Воспользуйтесь результатом упр. 65 для уточнения асимптотических граней из упр. 63.

a) Докажите, что существует предел $\omega = \lim_{n\to\infty} \log M(n)/\log n$.

b) Докажите неравенство $(mns)^{\omega/3} \le \underline{\operatorname{rank}}(T(m,n,s))$.

c) Пусть t — тензор $T(m,n,s) \oplus T(M,N,S)$. Докажите, что $(mns)^{\omega/3} + (MNS)^{\omega/3} \le \underline{\operatorname{rank}}(t)$. *Указание.* Рассмотрите прямое произведение t с самим собой.

d) Докажите, следовательно, неравенство $16^{\omega/3} + 9^{\omega/3} \le 17$ и получите, что $\omega < 2.55$.

67. [*HM40*] (Д. Копперсмит (D. Coppersmith) и Ш. Виноград.) Обобщая упр. 65 и 66, можно получить даже лучшую верхнюю грань ω.

a) Скажем, тензор (t_{ijk}) является *невырожденным*, если $\operatorname{rank}(t_{i(jk)}) = m$, $\operatorname{rank}(t_{j(ki)}) = n$ и $\operatorname{rank}(t_{k(ij)}) = s$ в обозначениях леммы T. Докажите, что тензор $T(m,n,s)$ для умножения матриц размера $mn \times ns$ является невырожденным.

b) Покажите, что прямая сумма невырожденных тензоров является невырожденной.

c) Тензор t размера $m \times n \times s$ с реализацией (A,B,C) длиной r называется *допускающим улучшение*, если он невырожден и существуют ненулевые элементы $d_1, \ldots, d_r$, такие, что $\sum_{l=1}^{r} a_{il}b_{jl}d_l = 0$ для $1 \le i \le m$ и $1 \le j \le n$. Докажите, что в подобном случае грань ранга $t \oplus T(1,q,1) \le r$, где $q = r - m - n$. *Указание.* Существуют матрицы V и W размера $q \times r$, такие, что $\sum_{l=1}^{r} v_{il}b_{jl}d_l = \sum_{l=1}^{r} a_{il}w_{jl}d_l = 0$ и $\sum_{l=1}^{r} v_{il}w_{jl}d_l = \delta_{ij}$ для всех соответствующих i и j.

d) Объясните, почему результат упр. 65 является частным случаем (c).

e) Докажите, что неравенство $\operatorname{rank}(T(m,n,s)) \le r$ влечет

$$\operatorname{rank}_2(T(m,n,s) \oplus T(1, r - n(m+s-1), 1)) \le r + n.$$

f) Докажите, следовательно, что ω действительно меньше, чем $\log M(n)/\log n$ для всех $n > 1$.

g) Обобщите (c) на случай, когда (A,B,C) — реализация t, только в слабом смысле упр. 61.

h) Из (d) следует, что $\underline{\operatorname{rank}}(T(3,1,3) \oplus T(1,4,1)) \le 10$. Тогда из упр. 61, (d) также получим $\underline{\operatorname{rank}}(T(9,1,9) \oplus 2T(3,4,3) \oplus T(1,16,1)) \le 100$. Докажите, что, если просто удалить строки A и B, соответствующие $16 + 16$ переменным $T(1,16,1)$, можно получить реализацию $T(9,1,9) \oplus 2T(3,4,3)$, которая допускает улучшение. Значит, фактически получим $\underline{\operatorname{rank}}(T(9,1,9) \oplus 2T(3,4,3) \oplus T(1,34,1)) \le 100$.

i) Обобщив упр. 66, (c), покажите, что

$$\sum_{p=1}^{t} (m_p n_p s_p)^{\omega/3} \le \underline{\operatorname{rank}}\Big(\bigoplus_{p=1}^{t} T(m_p, n_p, s_p)\Big).$$

j) Докажите, следовательно, что $\omega < 2.5$.

68. [*M45*] Можно ли вычислить полином

$$\sum_{1 \le i < j \le n} x_i x_j = x_1x_2 + \cdots + x_{n-1}x_n$$

с меньшим, чем $n-1$, количеством умножений и $2n-4$ сложений? (Существует $\binom{n}{2}$ членов.)

▸ **69.** [*HM27*] (В. Штрассен (V. Strassen), 1973.) Покажите, что определитель (31) матрицы размера $n \times n$ можно вычислить, произведя $O(n^5)$ умножений и $O(n^5)$ сложений или вычитаний, но не делений. [*Указание.* Рассмотрите $\det(I + Y)$, где $Y = X - I$.]

▸ **70.** [*HM25*] *Характеристический полином* $f_X(\lambda)$ матрицы X определяется как $\det(\lambda I - X)$. Докажите, что если $X = \left(\begin{smallmatrix} x & u \\ v & Y \end{smallmatrix}\right)$, где X, u, v и Y соответственно имеют размеры $n \times n$, $1 \times (n-1)$, $(n-1) \times 1$ и $(n-1) \times (n-1)$, то

$$f_X(\lambda) = f_Y(\lambda)\left(\lambda - x - \frac{uv}{\lambda} - \frac{uYv}{\lambda^2} - \frac{uY^2v}{\lambda^3} - \cdots\right).$$

Покажите, что это отношение позволяет вычислить коэффициенты f_X с приблизительно $\frac{1}{4}n^4$ умножениями, $\frac{1}{4}n^4$ сложениями-вычитаниями и без делений. *Указание.* Используйте равенство

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & D \end{pmatrix}\begin{pmatrix} A - BD^{-1}C & B \\ 0 & I \end{pmatrix}\begin{pmatrix} I & 0 \\ D^{-1}C & I \end{pmatrix},$$

которое справедливо для любых матриц A, B, C и D размеров $l \times l$, $l \times m$, $m \times l$ и $m \times m$ соответственно, когда D невырождена.

▶ **71.** [*HM30*] *Цепочка отношений полиномов* подобна цепочке полиномов, но она допускает деление, как и сложение, вычитание и умножение. Докажите, что если $f(x_1, \ldots, x_n)$ можно вычислить цепочкой отношений полиномов, имеющей m умножений в цепочке и d делений, то $f(x_1, \ldots, x_n)$ и все n ее частные производные $\partial f(x_1, \ldots, x_n)/\partial x_k$ для $1 \le k \le n$ можно вычислить единственной цепочкой отношений полиномов, которая имеет максимум $3m + d$ умножений в цепочке и $2d$ делений. (Например, любой эффективный метод вычисления определителя матрицы приводит к эффективному методу вычисления всех ее алгебраических дополнений, а значит, и к эффективному методу вычисления обратной матрицы.)

72. [*M48*] Можно ли определить ранг любого данного тензора (t_{ijk}) над, скажем, полем рациональных чисел за конечное число шагов?

73. [*HM25*] (Я. Моргенстерн (J. Morgenstern), 1973.) Докажите, что любая цепочка полиномов для дискретного преобразования Фурье (37) имеет по крайней мере $\frac{1}{2} m_1 \ldots m_n \lg m_1 \ldots m_n$ сложений-вычитаний, если не существует умножений в цепочке и если каждый параметр умножения является комплексной постоянной с $|\alpha_j| \le 1$. *Указание.* Рассмотрите матрицы линейных преобразований, вычисленных за первые k шагов.

74. [*HM35*] (А. Нозаки (A. Nozaki), 1978.) Большая часть теории, посвященной вычислению полиномов, касается умножения в цепочке, однако умножение нецелочисленных постоянных также может быть весьма важным. Назначение этого упражнения — развить соответствующую теорию. Скажем, что векторы $v_1, \ldots, v_s$ действительных чисел *Z-зависимы*, если существуют целые числа $(k_1, \ldots, k_s)$, такие, что $\gcd(k_1, \ldots, k_s) = 1$ и $k_1 v_1 + \cdots + k_s v_s$ — полностью целочисленный вектор. Если не существует таких целых чисел $(k_1, \ldots, k_s)$, то векторы $v_1, \ldots, v_s$ называются *Z-независимыми*.

a) Докажите, что если столбцы матрицы V размера $r \times s$ Z-независимы, то существуют столбцы матрицы VU, где U — любая унимодулярная матрица размера $s \times s$ (матрица из целых чисел с определителем, равным ± 1).

b) Пусть V — матрица размера $r \times s$ с Z-независимыми столбцами. Докажите, что цепочка полиномов, которая вычисляет элементы Vx, зависящие от $x_1, \ldots, x_s$, где $x = (x_1, \ldots, x_s)^T$, производит по крайней мере s умножений.

c) Пусть V — матрица размера $r \times t$, имеющая s Z-независимых столбцов. Докажите, что цепочка полиномов, которая позволяет вычислить элементы Vx, зависящие от $x_1, \ldots, x_t$, где $x = (x_1, \ldots, x_t)^T$, обязательно производит по крайней мере s умножений.

d) Покажите, как вычислить пары значений $\{x/2 + y,\ x + y/3\}$, которые зависят от x и y, используя только одно умножение, несмотря на то, что необходимы два умножения для вычисления пары $\{x/2 + y,\ x + y/2\}$.

*4.7. ОПЕРАЦИИ СО СТЕПЕННЫМИ РЯДАМИ

Если заданы два степенных ряда

$$U(z) = U_0 + U_1 z + U_2 z^2 + \cdots, \qquad V(z) = V_0 + V_1 z + V_2 z^2 + \cdots \tag{1}$$

с коэффициентами, принадлежащими некоторому полю, то можно образовывать их сумму, произведение и иногда их отношения, а также получать новые степенные ряды. Полиномы, очевидно, являются частным случаем степенных рядов, число членов которых конечно.

Безусловно, только конечное число членов можно представлять и хранить в компьютере. В таком случае уместно спросить: "Какая арифметика степенных рядов возможна на компьютерах, и, если возможна, чем она отличается от арифметики полиномов?". Ответ таков: "Мы работаем только с первыми N коэффициентами степенных рядов, где N — параметр, который, в принципе, может быть произвольно большим. Вместо обычной арифметики полиномов мы, по существу, используем арифметику по модулю z^N, что часто приводит к различным точкам зрения. К тому же отдельные операции наподобие "обращения" можно выполнять со степенными рядами, но не с полиномами, так как полиномы не замкнуты относительно этих операций".

Операции со степенными рядами находят много применений в численном анализе, но, возможно, наиболее важным является нахождение асимптотического разложения (как видно из раздела 1.2.11.3) или вычисление некоторых величин, определяемых производящими функциями. Именно последние, а не арифметика с плавающей точкой, делают их удобными для точного вычисления коэффициентов. Все алгоритмы данного раздела, кроме очевидных исключений, можно выполнять только с использованием рациональных операций. Таким образом, методы из раздела 4.5.1 можно будет применять для получения точных результатов, когда потребуется.

Вычисление $W(z) = U(z) \pm V(z)$, конечно, тривиально, поскольку $W_n = [z^n]W(z) = U_n \pm V_n$ для $n = 0, 1, 2, \ldots$. Так же легко вычислить коэффициенты $W(z) = U(z)V(z)$, воспользовавшись хорошо известным правилом свертки

$$W_n = \sum_{k=0}^{n} U_k V_{n-k} = U_0 V_n + U_1 V_{n-1} + \cdots + U_n V_0. \tag{2}$$

Отношение $W(z) = U(z)/V(z)$, когда $V_0 \neq 0$, можно получить, если поменять местами U и W в (2). Таким образом, получим правило

$$\begin{aligned} W_n &= \left(U_n - \sum_{k=0}^{n-1} W_k V_{n-k}\right) \Big/ V_0 \\ &= (U_n - W_0 V_n - W_1 V_{n-1} - \cdots - W_{n-1} V_1)/V_0. \end{aligned} \tag{3}$$

Это рекуррентное соотношение для W_k позволяет легко последовательно вычислить W_0, W_1, W_2, ..., не вводя U_n и V_n до вычисления W_{n-1}. Алгоритм с такими свойствами, оперирующий степенными рядами, обычно называют *интерактивным* (*online*). С его помощью можно определить N коэффициентов $W_0, W_1, \ldots, W_{N-1}$, не зная заранее N. Значит, можно, в принципе, выполнять алгоритм, как угодно

долго, и полностью вычислять степенные ряды. Можно также выполнять интерактивный алгоритм до тех пор, пока не будет соблюдено любое требуемое условие. (Противоположность "online" — "offline" ("автономный").)

Если коэффициенты U_k и V_k — целые, а W_k — не целые, рекуррентное соотношение (3) включает вычисления с дробями. Этого можно избежать, если воспользоваться полностью целочисленным подходом, предложенным в упр. 2.

Рассмотрим операцию вычисления $W(z) = V(z)^\alpha$, где α — "произвольная" степень. Например, можно вычислить квадратный корень из $V(z)$ при $\alpha = \frac{1}{2}$ или найти $V(z)^{-10}$ либо даже $V(z)^\pi$. Если V_m — первый не равный нулю коэффициент $V(z)$, получаем

$$\begin{aligned} V(z) &= V_m z^m \big(1 + (V_{m+1}/V_m)z + (V_{m+2}/V_m)z^2 + \cdots\big), \\ V(z)^\alpha &= V_m^\alpha z^{\alpha m} \big(1 + (V_{m+1}/V_m)z + (V_{m+2}/V_m)z^2 + \cdots\big)^\alpha. \end{aligned} \tag{4}$$

Это выражение будет степенным рядом тогда и только тогда, когда αm — неотрицательное целое число. Из (4) следует, что задачу вычисления произвольных степеней можно свести к случаю, когда $V_0 = 1$, т. е. к вычислению коэффициентов:

$$W(z) = (1 + V_1 z + V_2 z^2 + V_3 z^3 + \cdots)^\alpha. \tag{5}$$

Очевидно, что $W_0 = 1^\alpha = 1$.

Естественным методом нахождения коэффициентов выражения (5) является использование биномиальной теоремы 1.2.9–(19) или (если α — положительное целое число) методов из раздела 4.6.3. Но Леонард Эйлер открыл более простой и более эффективный метод получения степеней степенных рядов [*Introductio in Analysin Infinitorum* **1** (1748), §76]: если $W(z) = V(z)^\alpha$, то, дифференцируя, получим

$$W_1 + 2W_2 z + 3W_3 z^2 + \cdots = W'(z) = \alpha V(z)^{\alpha-1} V'(z); \tag{6}$$

следовательно,

$$W'(z)V(z) = \alpha W(z)V'(z). \tag{7}$$

Составив уравнение для коэффициентов при z^{n-1} в (7), получим, что

$$\sum_{k=0}^{n} k W_k V_{n-k} = \alpha \sum_{k=0}^{n} (n-k) W_k V_{n-k}, \tag{8}$$

а это дает удобное правило вычислений, справедливое для всех $n \geq 1$:

$$\begin{aligned} W_n &= \sum_{k=1}^{n} \Big(\Big(\frac{\alpha+1}{n} \Big) k - 1 \Big) V_k W_{n-k} \\ &= \big((\alpha+1-n)V_1 W_{n-1} + (2\alpha+2-n)V_2 W_{n-2} + \cdots + n\alpha V_n W_0 \big)/n. \end{aligned} \tag{9}$$

Из соотношения (9) можно получить простой интерактивный алгоритм для последовательного определения W_1, W_2, ..., используя приблизительно $2n$ умножений для вычисления n-го коэффициента. Отметим особый случай, когда $\alpha = -1$; (9) становится частным случаем (3) при $U(z) = V_0 = 1$.

Аналогичные методы можно использовать для образования $f(V(z))$, когда f — любая функция, удовлетворяющая простому дифференциальному уравнению (см., например, упр. 4). Для решения дифференциальных уравнений часто используется сравнительно прямой "метод степенных рядов". Он приводится почти во всех учебниках по дифференциальным уравнениям.

Обращение рядов. Возможно, наиболее интересным преобразованием степенных рядов является обращение рядов. Это преобразование позволяет решить уравнение

$$z = t + V_2 t^2 + V_3 t^3 + V_4 t^4 + \cdots \tag{10}$$

относительно t, т. е. получить коэффициенты степенного ряда

$$t = z + W_2 z^2 + W_3 z^3 + W_4 z^4 + \cdots. \tag{11}$$

Известны особо интересные способы выполнения такого обращения. Можно сказать, что "классический" метод основан на замечательной формуле обращения Лагранжа [*Mémoires Acad. Royale des Sciences et Belles-Lettres de Berlin* **24** (1768), 251–326], устанавливающей, что

$$W_n = \frac{1}{n}[t^{n-1}](1 + V_2 t + V_3 t^2 + \cdots)^{-n}. \tag{12}$$

Например, имеем $(1-t)^{-5} = \binom{4}{4} + \binom{5}{4}t + \binom{6}{4}t^2 + \cdots$, тогда пятый коэффициент, W_5, обращения $z = t - t^2$ равен $\binom{8}{4}/5 = 14$. Это проверяется с помощью формулы перечисления бинарных деревьев, приведенной в разделе 2.3.4.4.

В соотношении (12), которое имеет простое алгоритмическое доказательство (см. упр. 16), показано, что можно обратить ряд (10), если последовательно вычислять отрицательные степени $(1 + V_2 t + V_3 t^2 + \cdots)^{-n}$ для $n = 1, 2, 3, \ldots$. Непосредственное применение этой идеи должно привести к интерактивному алгоритму обращения, который использует около $N^3/2$ умножений для вычисления N коэффициентов, но соотношение (9) предоставляет возможность работать только с первыми n коэффициентами $(1+V_2t+V_3t^2+\cdots)^{-n}$, получаемыми интерактивным алгоритмом, которому требуется лишь около $N^3/6$ умножений.

Алгоритм L (*Обращение степенного ряда методом Лагранжа*). В этом интерактивном алгоритме (рис. 17) вводится значение V_n из (10) и выводится значение W_n из (11) для $n = 2, 3, 4, \ldots, N$. (Нет необходимости заранее знать точное число N; алгоритм можно завершить в соответствии с любым критерием.)

L1. [Инициализация.] Присвоить $n \leftarrow 1$, $U_0 \leftarrow 1$. (Соотношение

$$(1 + V_2 t + V_3 t^2 + \cdots)^{-n} = U_0 + U_1 t + \cdots + U_{n-1} t^{n-1} + O(t^n) \tag{13}$$

сохраняется на все время работы алгоритма.)

L2. [Ввод V_n.] Увеличить n на 1. Если $n > N$, работа алгоритма завершается, иначе — ввести следующий коэффициент V_n.

L3. [Деление.] Присвоить $U_k \leftarrow U_k - U_{k-1}V_2 - \cdots - U_1 V_k - U_0 V_{k+1}$ для $k = 1, 2, \ldots, n-2$ (в таком порядке); затем присвоить

$$U_{n-1} \leftarrow -2U_{n-2}V_2 - 3U_{n-3}V_3 - \cdots - (n-1)U_1 V_{n-1} - nU_0 V_n.$$

(В результате получаем $U(z)$, деленное на $V(z)/z$, см. (3) и (9).)

L4. [Вывод W_n.] Вывести U_{n-1}/n, которое равно W_n, и возвратиться к шагу L2. ∎

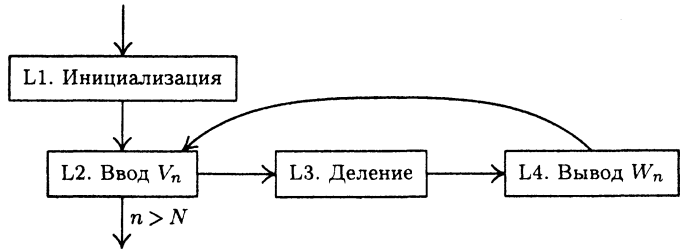


Рис. 17. Алгоритм L обращения степенного ряда.

Если алгоритм L применить к примеру $z = t - t^2$, можно вычислить следующее.

n	V_n	U_0	U_1	U_2	U_3	U_4	W_n
1	1	1					1
2	−1	1	2				1
3	0	1	3	6			2
4	0	1	4	10	20		5
5	0	1	5	15	35	70	14

В упр. 8 показано, что небольшая модификация алгоритма L позволит решать значительно более общие задачи, прилагая только чуть больше усилий.

Рассмотрим сейчас решение уравнения

$$U_1 z + U_2 z^2 + U_3 z^3 + \cdots = t + V_2 t^2 + V_3 t^3 + \cdots \tag{14}$$

относительно t и получим коэффициенты степенного ряда

$$t = W_1 z + W_2 z^2 + W_3 z^3 + W_4 z^4 + \cdots. \tag{15}$$

Уравнение (10) — это частный случай (14) при $U_1 = 1$, $U_2 = U_3 = \cdots = 0$. Если $U_1 \neq 0$, можно предположить, что $U_1 = 1$, заменив z на $(U_1 z)$, но мы рассмотрим общее уравнение (14), поскольку U_1 может равняться нулю.

Алгоритм T (*Обобщенное обращение степенных рядов*). В этом интерактивном алгоритме вводятся значения U_n и V_n из (14) и выводится значение W_n из (15) для $n = 1, 2, 3, \ldots, N$. При вычислениях используется вспомогательная матрица T_{mn}, $1 \le m \le n \le N$.

T1. [Инициализация.] Присвоить $n \leftarrow 1$. Занесем два первых введенных значения (а именно — U_1 и V_1) в T_{11} и V_1 соответственно. (Должно выполняться равенство $V_1 = 1$.)

T2. [Вывод W_n.] Вывести значение T_{1n}, которое равно W_n.

T3. [Ввод U_n, V_n.] Увеличить n на 1. Если $n > N$, алгоритм заканчивает работу, иначе — запоминает два следующих введенных значения (а именно — U_n и V_n) в T_{1n} и V_n.

T4. [Умножение.] Присвоить

$$T_{mn} \leftarrow T_{11} T_{m-1,n-1} + T_{12} T_{m-1,n-2} + \cdots + T_{1,n-m+1} T_{m-1,m-1}$$

и $T_{1n} \leftarrow T_{1n} - V_m T_{mn}$ для $2 \le m \le n$. (После этого шага для $1 \le m \le n$ получим

$$t^m = T_{mm} z^m + T_{m,m+1} z^{m+1} + \cdots + T_{mn} z^n + O(z^{n+1}). \tag{16}$$

Легко проверить (16) индукцией по $m \ge 2$ и, когда $m = 1$, получить $U_n = T_{1n} + V_2 T_{2n} + \cdots + V_n T_{nn}$ согласно (14) и (16).) Возвратиться к шагу T2. ▮

Соотношение (16) объясняет механизм этого алгоритма, предложенного Генри К. Тэчером (мл.) (Henry C. Thacher, Jr.) [*CACM* **9** (1966), 10–11]. Время счета алгоритма, по существу, такое же, как и у алгоритма L, но требуется значительно больший объем памяти для хранения данных. Пример работы алгоритма приведен в упр. 9.

Другой подход к обращению степенных рядов предложен в работе R. P. Brent and H. T. Kung, *JACM* **25** (1978), 581–595. Он основан на том факте, что стандартные итерационные процедуры, которые используются для нахождения корней уравнений с действительными числами, можно также применять к уравнениям для степенных рядов. В частности, можно рассмотреть метод Ньютона для приближенного вычисления действительного числа t, такого, что $f(t) = 0$, а заданная функция f хорошо ведет себя в окрестности t: если x является хорошим приближением к t, то $\phi(x) = x - f(x)/f'(x)$ будет даже лучшим приближением. Записав $x = t + \epsilon$, получим $f(x) = f(t) + \epsilon f'(t) + O(\epsilon^2)$, $f'(x) = f'(t) + O(\epsilon)$; следовательно, $\phi(x) = t + \epsilon - \big(0 + \epsilon f'(t) + O(\epsilon^2)\big)/\big(f'(t) + O(\epsilon)\big) = t + O(\epsilon^2)$. Применим эту идею к степенным рядам. Пусть $f(x) = V(x) - U(z)$, где U и V — степенные ряды из (14). Найдем степенной ряд t от z, такой, что $f(t) = 0$. Пусть $x = W_1 z + \cdots + W_{n-1} z^{n-1} = t + O(z^n)$ — "приближение" к t порядка n, тогда $\phi(x) = x - f(x)/f'(x)$ будет приближением порядка $2n$, поскольку для этих f и t выполняются предположения метода Ньютона.

Другими словами, можно воспользоваться следующей процедурой.

Алгоритм N (*Обобщенное обращение степенного ряда методом Ньютона*). Данный "полиинтерактивный" алгоритм вводит значения U_n и V_n согласно (14) для $2^k \le n < 2^{k+1}$ и выводит значения W_n согласно (15) для $2^k \le n < 2^{k+1}$, получая ответы группами по 2^k значений одновременно для $k = 0, 1, 2, \ldots, K$.

N1. [Инициализация.] Присвоить $N \leftarrow 1$. (Получим $N = 2^k$.) Ввести первые коэффициенты U_1 и V_1 ($V_1 = 1$) и присвоить $W_1 \leftarrow U_1$.

N2. [Вывод.] Вывести W_n для $N \le n < 2N$.

N3. [Ввод.] Присвоить $N \leftarrow 2N$. Если $N > 2^K$, алгоритм закончил работу, иначе — ввести значения U_n и V_n для $N \le n < 2N$.

N4. [Шаг Ньютона.] Воспользуемся алгоритмом для композиции степенных рядов (см. упр. 11), чтобы вычислить коэффициенты Q_j и R_j ($0 \le j < N$) степенного ряда

$$\begin{aligned} U_1 z + \cdots + U_{2N-1} z^{2N-1} - V(W_1 z + \cdots + W_{N-1} z^{N-1}) \\ = R_0 z^N + R_1 z^{N+1} + \cdots + R_{N-1} z^{2N-1} + O(z^{2N}), \\ V'(W_1 z + \cdots + W_{N-1} z^{N-1}) = Q_0 + Q_1 z + \cdots + Q_{N-1} z^{N-1} + O(z^N), \end{aligned}$$

где $V(x) = x + V_2x^2 + \cdots$ и $V'(x) = 1 + 2V_2x + \cdots$. Затем возьмем $W_N, \ldots, W_{2N-1}$ в качестве коэффициентов степенного ряда

$$\frac{R_0 + R_1z + \cdots + R_{N-1}z^{N-1}}{Q_0 + Q_1z + \cdots + Q_{N-1}z^{N-1}} = W_N + \cdots + W_{2N-1}z^{N-1} + O(z^N)$$

и возвратимся к шагу N2. ▮

Время работы данного алгоритма при получении $N = 2^K$ коэффициентов равно $T(N)$, где

$$T(2N) = T(N) + (\text{время на выполнение шага N4}) + O(N). \tag{17}$$

Прямые алгоритмы для композиции и деления на шаге N4 имеют порядка N^3 шагов, значит, алгоритм N работает медленнее алгоритма T. Тем не менее Брент (Brent) и Кунг (Kung) нашли метод, которым требуемая композиция степенных рядов выполняется с помощью $O(N \log N)^{3/2}$ арифметических операций (в упр. 6 приведен алгоритм для деления, работающий еще быстрее). Таким образом, в (17) показано, что обращение степенных рядов можно выполнить только с помощью $O(N \log N)^{3/2}$ операций, когда $N \to \infty$. (С другой стороны, константа пропорциональности такова, что N должно быть действительно большим, прежде чем алгоритмы L и T перестанут относиться к "быстродействующим" методам.)

Историческая справка. Ж. Н. Брамхел (J. N. Bramhall) и М. А. Чаппл (M. A. Chapple) впервые опубликовали метод обращения степенных рядов, требующий $O(N^3)$ операций, в *CACM* **4** (1961), 317–318, 503. Это, по существу, автономный алгоритм, эквивалентный методу, который приведен в упр. 16, с таким же временем счета, как у алгоритмов L и T.

Итерация рядов. Чтобы изучить поведение итеративного процесса $x_n \leftarrow f(x_{n-1})$, следует изучить n-кратную композицию данной функции f саму с собой, т. е. $x_n = f\big(f(\ldots f(x_0)\ldots)\big)$. Определим $f^{[0]}(x) = x$ и $f^{[n]}(x) = f\big(f^{[n-1]}(x)\big)$ так, что

$$f^{[m+n]}(x) = f^{[m]}\big(f^{[n]}(x)\big) \tag{18}$$

для всех целых m, $n \geq 0$. Во многих случаях обозначение $f^{[n]}(x)$ имеет смысл и для отрицательных целых n. Если $f^{[n]}$ и $f^{[-n]}$ — взаимно обратные функции, а именно — $x = f^{[n]}\big(f^{[-n]}(x)\big)$, и если обратные функции определены единственным образом, то (18) справедливо для *всех* целых m и n. Обращение рядов — это, по существу, операция нахождения обратного степенного ряда $f^{[-1]}(x)$. Например, соотношения (10) и (11) устанавливают, что $z = V\big(W(z)\big)$ и $t = W\big(V(t)\big)$; таким образом, $W = V^{[-1]}$.

Предположим, что заданы два степенных ряда, $V(z) = z + V_2z^2 + \cdots$ и $W(z) = z + W_2z^2 + \cdots$, таких, что $W = V^{[-1]}$. Пусть u — любая не равная нулю постоянная. Рассмотрим функцию

$$U(z) = W\big(uV(z)\big). \tag{19}$$

Легко видеть, что $U\big(U(z)\big) = W\big(u^2V(z)\big)$ и вообще

$$U^{[n]}(z) = W\big(u^nV(z)\big) \tag{20}$$

для всех целых n. Следовательно, имеем простое выражение для n-й итерации $U^{[n]}$, которую можно вычислить приблизительно с одинаковыми затратами труда

для всех n. Кроме того, можно даже воспользоваться (20), чтобы определить $U^{[n]}$ для нецелых значений n. Например, "полуитерация" $U^{[1/2]}$ — это такая функция, что $U^{[1/2]}\big(U^{[1/2]}(z)\big) = U(z)$. (Существуют две такие функции $U^{[1/2]}$, полученные в результате использования $\sqrt{u}$ и $-\sqrt{u}$ в качестве значения $u^{1/2}$ в (20).)

Мы получили простые соотношения в (20), которые, начиная с V и u, определяют U. Но на практике обычно применяется другой метод: начиная с некоторой заданной функции U, найти такие V и u, чтобы выполнялось (19), т. е. чтобы

$$V\big(U(z)\big) = uV(z). \tag{21}$$

Такая функция V называется *функцией Шрёдера* U, поскольку она была введена Эрнстом Шрёдером (Ernst Schröder, *Math. Annalen* **3** (1871), 296–322). Рассмотрим задачу нахождения функции Шрёдера $V(z) = z + V_2z^2 + \cdots$ заданного степенного ряда $U = U_1z + U_2z^2 + \cdots$. Очевидно, что $u = U_1$, если выполняется (21).

Подставив в (21) $u = U_1$ и собрав коэффициенты при z, придем к последовательности уравнений, начинающихся с

$$\begin{aligned} U_1^2V_2 + U_2 &= U_1V_2 \\ U_1^3V_3 + 2U_1U_2V_2 + U_3 &= U_1V_3 \\ U_1^4V_4 + 3U_1^2U_2V_3 + 2U_1U_3V_2 + U_2^2V_2 + U_4 &= U_1V_4 \end{aligned}$$

и т. д. Ясно, что не существует решения, когда $U_1 = 0$ (кроме тривиального случая, когда $U_2 = U_3 = \cdots = 0$), но существует единственное решение, если U_1 не является корнем из единицы. Можно предположить, что произойдет что-нибудь забавное, когда $U_1^n = 1$, так как из (20) видно, что $U^{[n]}(z) = z$, если функция Шрёдера в этом случае существует. Предположим на минуту, что U_1 не равно нулю и не равно корню из единицы. Тогда функция Шрёдера существует и возникает следующий вопрос: "Как ее вычислить, не прилагая слишком много усилий?".

Следующая процедура предложена Р. П. Брентом (R. P. Brent) и Ж. Ф. Траубом (J. F. Traub). Уравнение (21) приводит к подобной подзадаче, но более сложного вида. Таким образом, мы поставили более общую задачу, подзадача которой имеет такой же вид. Попытаемся найти $V(z) = V_0 + V_1z + \cdots + V_{n-1}z^{n-1}$, такое, что

$$V\big(U(z)\big) = W(z)V(z) + S(z) + O(z^n), \tag{22}$$

где $U(z)$, $W(z)$, $S(z)$ и n заданы, n — степень двойки и $U(0) = 0$. Для $n = 1$ просто положим $V_0 = S(0)/\big(1 - W(0)\big)$ с $V_0 = 1$, если $S(0) = 0$ и $W(0) = 1$. Кроме того, возможен переход от n к $2n$: сначала найдем $R(z)$, такое, что

$$V\big(U(z)\big) = W(z)V(z) + S(z) - z^nR(z) + O(z^{2n}), \tag{23}$$

затем вычислим

$$\hat{W}(z) = W(z)\big(z/U(z)\big)^n + O(z^n), \qquad \hat{S}(z) = R(z)\big(z/U(z)\big)^n + O(z^n) \tag{24}$$

и найдем $\hat{V}(z) = V_n + V_{n+1}z + \cdots + V_{2n-1}z^{n-1}$, такое, что

$$\hat{V}\big(U(z)\big) = \hat{W}(z)\hat{V}(z) + \hat{S}(z) + O(z^n). \tag{25}$$

Следовательно, функция $V^*(z) = V(z) + z^n\hat{V}(z)$ удовлетворяет

$$V^*\big(U(z)\big) = W(z)V^*(z) + S(z) + O(z^{2n}),$$

что и требовалось.

Время работы данной процедуры $T(n)$ удовлетворяет соотношению

$$T(2n) = 2T(n) + C(n), \tag{26}$$

где $C(n)$ — время вычисления $R(z)$, $\hat{W}(z)$ и $\hat{S}(z)$. Функция $C(n)$ отнимает основное время при вычислении $V\big(U(z)\big)$ по модулю z^{2n}, порядок роста $C(n)$ предположительно больше, чем $n^{1+\epsilon}$; следовательно, решение $T(n)$ рекуррентного соотношения (26) будет иметь порядок $C(n)$. Например, если $C(n) = cn^3$, получим $T(n) \approx \frac{4}{3}cn^3$ или, если $C(n)$ равно $O(n \log n)^{3/2}$, с помощью "быстрой" композиции получим $T(n) = O(n \log n)^{3/2}$.

Процедура не работает, когда $W(0) = 1$ и $S(0) \neq 0$, поэтому необходимо исследовать, когда это происходит. Легко доказать индукцией по n, что решение (22) согласно методу Брента-Трауба влечет рассмотрение точно n подзадач, в которых коэффициенты $V(z)$ правой части уравнения принимают соответствующие значения $W(z)\big(z/U(z)\big)^j + O(z^n)$ для $0 \leq j < n$ в некотором порядке. Если $W(0) = U_1$ и U_1 — не корень из единицы, получаем $W(0) = 1$ только тогда, когда $j = 1$; в этом случае процедура не работает, если (22) не имеет решения для $n = 2$.

Следовательно, функцию Шрёдера для U можно найти, решая уравнение (22) для $n = 2, 4, 8, 16, \ldots$ с $W(z) = U_1$ и $S(z) = 0$, всякий раз, когда U_1 не равно нулю и не является корнем из единицы.

Если $U_1 = 1$, функция Шрёдера не существует, кроме случая, когда $U(z) = z$. Однако Брент и Трауб сумели найти быстрый метод вычисления $U^{[n]}(z)$, даже когда $U_1 = 1$, благодаря использованию функции $V(z)$, такой, что

$$V\big(U(z)\big) = U'(z)V(z). \tag{27}$$

Если обе функции $U(z)$ и $\hat{U}(z)$ удовлетворяют (27) для тех же V, легко проверить, что их композиция $U\big(\hat{U}(z)\big)$ также удовлетворяет (27); следовательно, все итерации $U(z)$ являются решениями (27). Предположим, выполняется равенство $U(z) = z + U_k z^k + U_{k+1} z^{k+1} + \cdots$, где $k \geq 2$ и $U_k \neq 0$. Тогда можно показать, что существует единственный степенной ряд вида $V(z) = z^k + V_{k+1} z^{k+1} + V_{k+2} z^{k+2} + \cdots$, который удовлетворяет (27). Значит, если задана такая функция $V(z)$ и если заданы $k \geq 2$ и U_k, существует единственный ряд вида $U(z) = z + U_k z^k + U_{k+1} z^{k+1} + \cdots$, удовлетворяющий (27). Требуемая итерация $U^{[n]}(z)$ является единственным степенным рядом $P(z)$, удовлетворяющим

$$V\big(P(z)\big) = P'(z)V(z), \tag{28}$$

где $P(z) = z + nU_k z^k + \cdots$. Обе функции, $V(z)$ и $P(z)$, можно найти с помощью подходящих алгоритмов (см. упр. 14).

Если U_1 — k-й корень из единицы, не равный 1, то такой же метод можно применить к функции $U^{[k]}(z) = z + \cdots$ и $U^{[k]}(z)$ можно найти из $U(z)$, произведя $l(k)$ операций композиции (см. раздел 4.6.3). Можно также рассмотреть случай, когда $U_1 = 0$: если $U(z) = U_k z^k + U_{k+1} z^{k+1} + \cdots$, где $k \geq 2$ и $U_k \neq 0$, то идея состоит в том, чтобы найти решение уравнения $V\big(U(z)\big) = U_k V(z)^k$. Тогда

$$U^{[n]}(z) = V^{[-1]}\big(U_k^{[(k^n-1)/(k-1)]} V(z)^{k^n}\big). \tag{29}$$

И наконец, если $U(z) = U_0 + U_1 z + \cdots$, где $U_0 \neq 0$, пусть α — "неподвижная точка", такая, что $U(\alpha) = \alpha$, и пусть

$$\hat{U}(z) = U(\alpha + z) - \alpha = zU'(\alpha) + z^2 U''(\alpha)/2! + \cdots. \tag{30}$$

Тогда $U^{[n]}(z) = \hat{U}^{[n]}(z - \alpha) + \alpha$. Детали можно найти в работе Brent and Traub, *SICOMP* **9** (1980), 54–66. Функция V из (27), по существу, рассмотрена в книге M. Kuczma, *Functional Equations in a Single Variable* (Warsaw: PWN-Polish Scientific, 1968), лемма 9.4, и, безусловно, Э. Жаботинским (E. Jabotinsky) на несколько лет раньше (см. упр. 23).

Алгебраические функции. Коэффициенты каждого степенного ряда $W(z)$, удовлетворяющего общему уравнению вида

$$A_n(z)W(z)^n + \cdots + A_1(z)W(z) + A_0(z) = 0, \tag{31}$$

где каждое $A_i(z)$ — полином, можно эффективно вычислить методом, предложенным в работе H. T. Kung and J. F. Traub, *JACM* **25** (1978), 245–260. См. также работу Д. В. Чудновского и Г. В. Чудновского (D. V. Chudnovsky and G. V. Chudnovsky, *J. Complexity* **2** (1986), 271–294; **3** (1987), 1–25).

УПРАЖНЕНИЯ

1. [*M10*] В разделе объяснено, как разделить $U(z)$ на $V(z)$, когда $V_0 \neq 0$. Как произвести деление, когда $V_0 = 0$?

2. [*20*] Когда коэффициенты $U(z)$ и $V(z)$ — целые и $V_0 \neq 0$, найдите рекуррентное соотношение для целых $V_0^{n+1} W_n$, где W_n определено в (3). Как можно этим воспользоваться для деления степенных рядов?

3. [*M15*] Будет ли правильным результат, приведенный в формуле (9), когда $\alpha = 0$ и когда $\alpha = 1$?

▶ **4.** [*HM23*] Покажите, что простая модификация (9) может быть использована для вычисления $e^{V(z)}$, когда $V_0 = 0$, и $\ln V(z)$, когда $V_0 = 1$.

5. [*M00*] Что произойдет, если степенной ряд обратить дважды, т. е. если выходные значения алгоритма L или T обратить снова?

▶ **6.** [*M21*] (X. T. Кунг (H. T. Kung).) Примените метод Ньютона к вычислению $W(z) = 1/V(z)$, когда $V(0) \neq 0$, определив корень в виде степенного ряда уравнения $f(x) = 0$, где $f(x) = x^{-1} - V(z)$.

7. [*M23*] Воспользовавшись формулой обращения Лагранжа (12), найдите простое выражение для коэффициента W_n в обращении $z = t - t^m$.

▶ **8.** [*M25*] Для $W(z) = W_1 z + W_2 z^2 + W_3 z^3 + \cdots = G_1 t + G_2 t^2 + G_3 t^3 + \cdots = G(t)$, где $z = V_1 t + V_2 t^2 + V_3 t^3 + \cdots$ и $V_1 \neq 0$, Лагранж доказал, что

$$W_n = \frac{1}{n}[t^{n-1}]\, G'(t)/(V_1 + V_2 t + V_3 t^2 + \cdots)^n.$$

(Соотношение (12) является частным случаем предыдущего, когда $G_1 = V_1 = 1$, $G_2 = G_3 = \cdots = 0$.) Расширьте алгоритм L таким образом, чтобы для данной более общей ситуации он вычислял коэффициенты $W_1, W_2, \ldots$ без значительного увеличения времени работы алгоритма.

9. [*11*] Используя алгоритм T, найдите значения T_{mn} как первые пять коэффициентов обращения $z = t - t^2$.

10. [*M20*] Задано $y = x^\alpha + a_1 x^{\alpha+1} + a_2 x^{\alpha+2} + \cdots$, $\alpha \neq 0$. Покажите, как вычислить коэффициенты в разложении $x = y^{1/\alpha} + b_2 y^{2/\alpha} + b_3 y^{3/\alpha} + \cdots$.

▶ **11.** [*M25*] (*Композиция степенных рядов.*) Пусть

$$U(z) = U_0 + U_1 z + U_2 z^2 + \cdots \quad \text{и} \quad V(z) = V_1 z + V_2 z^2 + V_3 z^3 + \cdots.$$

Составьте план алгоритма, который вычисляет первые N коэффициентов $U(V(z))$.

12. [*M20*] Найдите связь между делением полиномов и делением степенных рядов. Заданы полиномы $u(x)$ и $v(x)$ степеней m и n соответственно над полем. Покажите, как найти полиномы $q(x)$ и $r(x)$, такие, что $u(x) = q(x)v(x) + r(x)$ и $\deg(r) < n$, используя только операции со степенными рядами.

13. [*M27*] (*Аппроксимация рациональных функций.*) Иногда необходимо найти полиномы, отношения которых имеют такие же начальные члены, как и данные степенные ряды. Например, если $W(z) = 1 + z + 3z^2 + 7z^3 + \cdots$, то существует четыре различных способа представления $W(z)$ в виде $w_1(z)/w_2(z) + O(z^4)$, где $w_1(z)$ и $w_2(z)$ — полиномы с $\deg(w_1) + \deg(w_2) < 4$:

$$(1 + z + 3z^2 + 7z^3) \,/\, 1 = 1 + z + 3z^2 + 7z^3 + 0z^4 + \cdots,$$
$$(3 - 4z + 2z^2) \,/\, (3 - 7z) = 1 + z + 3z^2 + 7z^3 + \tfrac{49}{3} z^4 + \cdots,$$
$$(1 - z) \,/\, (1 - 2z - z^2) = 1 + z + 3z^2 + 7z^3 + 17z^4 + \cdots,$$
$$1 \,/\, (1 - z - 2z^2 - 2z^3) = 1 + z + 3z^2 + 7z^3 + 15z^4 + \cdots.$$

Рациональные функции такого рода обычно называют *аппроксимацией Паде*, так как они широко изучены Г. Е. Паде (H. E. Padé) [*Annales Scient. de l'École Normale Supérieure* (3) **9** (1892), S1–S93; (3) **16** (1899), 395–426].

Покажите, что все аппроксимации Паде $W(z) = w_1(z)/w_2(z) + O(z^N)$ с $\deg(w_1) + \deg(w_2) < N$ можно получить, применяя обобщенный алгоритм Евклида к полиномам z^N и $W_0 + W_1 z + \cdots + W_{N-1} z^{N-1}$, и составьте целочисленный алгоритм для случая, когда каждое W_i — целое. [*Указание.* См. упр. 4.6.1–26.]

▶ **14.** [*HM30*] Используя (27) и (28), запишите подробно метод вычисления $U^{[n]}(z)$ Брента и Трауба, когда $U(z) = z + U_k z^k + \cdots$.

15. [*HM20*] Какой вид должна иметь функция $U(z)$, если в (27) $V(z)$ имеет простую форму z^k? Какой вывод можно сделать относительно итераций $U(z)$?

16. [*HM21*] Пусть, как в упр 8, $W(z) = G(t)$. "Очевидный" метод нахождения коэффициентов W_1, W_2, W_3, ... таков: присвоить $n \leftarrow 1$ и $R_1(t) \leftarrow G(t)$, а затем сохранить соотношение $W_n V(t) + W_{n+1} V(t)^2 + \cdots = R_n(t)$, неоднократно присваивая $W_n \leftarrow [t]\, R_n(t)/V_1$, $R_{n+1}(t) \leftarrow R_n(t)/V(t) - W_n$, $n \leftarrow n + 1$.

Докажите формулу Лагранжа из упр. 8, показав, что

$$\frac{1}{n}[t^{n-1}]\, R'_{k+1}(t)\, t^n / V(t)^n = \frac{1}{n+1}[t^n]\, R'_k(t)\, t^{n+1} / V(t)^{n+1} \quad \text{для всех } n \geq 1 \text{ и } k \geq 1.$$

▶ **17.** [*M20*] Задан степенной ряд $V(z) = V_1 z + V_2 z^2 + V_3 z^3 + \cdots$. Определим *степенную матрицу* V как бесконечную таблицу коэффициентов $v_{nk} = \frac{n!}{k!}[z^n] V(z)^k$; n-й *степеноид* (*poweroid*) V определяется как $V_n(x) = v_{n0} + v_{n1} x + \cdots + v_{nn} x^n$. Докажите, что степеноид удовлетворяет общему закону свертки

$$V_n(x + y) = \sum_k \binom{n}{k} V_k(x) V_{n-k}(y).$$

(Например, когда $V(z) = z$, получаем $V_n(x) = x^n$, и это биномиальная теорема. Когда $V(z) = \ln(1/(1-z))$, согласно 1.2.9–(26) получаем $v_{nk} = \left[{n \atop k}\right]$. Следовательно, степеноид

$V_n(x)$ равен $x^{\overline{n}}$, и тождество совпадает с результатом, доказанным в упр. 1.2.6–33. Когда $V(z) = e^z - 1$, получаем $V_n(x) = \sum_k \left\{ {n \atop k} \right\} x^k$ и данная формула эквивалентна равенству

$$\binom{l+m}{m} \left\{ {n \atop l+m} \right\} = \sum_k \binom{n}{k} \left\{ {k \atop l} \right\} \left\{ {n-k \atop m} \right\},$$

которого ранее у нас не было. Другие треугольные таблицы коэффициентов, которые возникают в комбинаторной математике и анализе алгоритмов, также оказываются степенными матрицами степенных рядов.)

18. [*HM22*] Продолжая упр. 17, докажите, что степеноид также удовлетворяет уравнению

$$xV_n(x+y) = (x+y) \sum_k \binom{n-1}{k-1} V_k(x) V_{n-k}(y).$$

[*Указание.* Рассмотрите производную $e^{xV(z)}$.]

19. [*M25*] Продолжая упр. 17, выразите все числа v_{nk} в терминах чисел $v_n = v_{n1} = n! V_n$ первого столбца и найдите простую рекуррентную формулу, которая позволит получить все столбцы из последовательности $v_1, v_2, \ldots$. В частности, покажите, что если все v_n — целые, то все v_{nk} также целые.

20. [*HM20*] Продолжая упр. 17, предположим, что $W(z) = U(V(z))$ и $U_0 = 0$. Докажите, что степенная матрица W равна произведению степенных матриц V и U: $w_{nk} = \sum_j v_{nj} u_{jk}$.

▸ **21.** [*HM27*] Продолжая предыдущее упражнение, предположим, что $V_1 \neq 0$, и пусть $W(z) = -V^{[-1]}(-z)$. Назначение данного упражнения — показать, что степенные матрицы V и W "двойственны" одна другой; например, когда $V(z) = \ln(1/(1-z))$, то $V^{[-1]}(z) = 1 - e^{-z}$, $W(z) = e^z - 1$ и соответствующие степенные матрицы — это хорошо известные треугольники Стирлинга $v_{nk} = \left[{n \atop k} \right]$ и $w_{nk} = \left\{ {n \atop k} \right\}$.

a) Докажите, что формула обращения 1.2.6–(47) для чисел Стирлинга обычно выполняется в более общем случае:

$$\sum_k v_{nk} w_{km} (-1)^{n-k} = \sum_k w_{nk} v_{km} (-1)^{n-k} = \delta_{mn}.$$

b) Из соотношения $v_{n(n-k)} = n^{\underline{k}} [z^k] \, (V(z)/z)^{n-k}$ для фиксированных k следует, что величина $v_{n(n-k)}/V_1^n$ является полиномом от n степени $\leq 2k$. Поэтому можно определить

$$v_{\alpha(\alpha-k)} = \alpha^{\underline{k}} [z^k] \, (V(z)/z)^{\alpha-k}$$

для произвольного α, когда k — неотрицательное целое число, как было определено для чисел Стирлинга в разделе 1.2.6. Докажите, что $v_{(-k)(-n)} = w_{nk}$. (Это обобщение формулы 1.2.6–(58).)

▸ **22.** [*HM27*] Задан ряд $U(z) = U_0 + U_1 z + U_2 z^2 + \cdots$ с $U_0 \neq 0$; *α-индуцированная функция* $U^{\{\alpha\}}(z)$ — это степенной ряд $V(z)$, полностью определенный уравнением

$$V(z) = U(zV(z)^\alpha).$$

a) Докажите, что $U^{\{0\}}(z) = U(z)$ и $U^{\{\alpha\}\{\beta\}}(z) = U^{\{\alpha+\beta\}}(z)$.

b) Пусть $B(z)$ — простой биномиальный ряд $1 + z$. Где раньше встречалось $B^{\{2\}}(z)$?

c) Докажите, что $[z^n] U^{\{\alpha\}}(z)^x = \frac{x}{x+n\alpha} [z^n] U(z)^{x+n\alpha}$. *Указание.* Если $W(z) = z/U(z)^\alpha$, то получим $U^{\{\alpha\}}(z) = (W^{[-1]}(z)/z)^{1/\alpha}$.

d) Докажите, следовательно, что любой степеноид $V_n(x)$ удовлетворяет не только равенствам из упр. 17 и 18, но и равенствам

$$\frac{(x+y)V_n(x+y+n\alpha)}{x+y+n\alpha} = \sum_k \binom{n}{k} \frac{xV_k(x+k\alpha)}{x+k\alpha} \frac{yV_{n-k}(y+(n-k)\alpha)}{y+(n-k)\alpha};$$

$$\frac{V_n(x+y)}{y-n\alpha} = (x+y)\sum_k \binom{n-1}{k-1} \frac{V_k(x+k\alpha)}{x+k\alpha} \frac{V_{n-k}(y-k\alpha)}{y-k\alpha}.$$

[Частные случаи включают биномиальную теорему Абеля, формулу 1.2.6–(16) и равенства Рота 1.2.6–(26) и 1.2.6–(30): сумму Торелли, упр. 1.2.6–34.]

23. [*HM35*] (Э. Жаботинский.) Продолжая в том же духе, предположим, что $U = (u_{nk})$ — степенная матрица $U(z) = z + U_2 z^2 + \cdots$. Пусть $u_n = u_{n1} = n!\,U_n$.

a) Объясните, как вычислить матрицу $\ln U$, чтобы степенная матрица $U^{[\alpha]}(z)$ равнялась $\exp(\alpha \ln U) = I + \alpha \ln U + (\alpha \ln U)^2/2! + \cdots$.

b) Пусть l_{nk} — элемент матрицы $\ln U$, находящийся на пересечении строки n и столбца k, и пусть

$$l_n = l_{n1}, \qquad L(z) = l_2 \frac{z^2}{2!} + l_3 \frac{z^3}{3!} + l_4 \frac{z^4}{4!} + \cdots.$$

Докажите, что $l_{nk} = \binom{n}{k-1} l_{n+1-k}$ для $1 \le k \le n$. [*Указание.* $U^{[\epsilon]}(z) = z + \epsilon L(z) + O(\epsilon^2)$.]

c) Рассмотрите $U^{[\alpha]}(z)$ как функцию от α и z и докажите, что

$$\frac{\partial}{\partial \alpha} U^{[\alpha]}(z) = L(z) \frac{\partial}{\partial z} U^{[\alpha]}(z) = L(U^{[\alpha]}(z)).$$

(Следовательно, $L(z) = (l_k/k!)V(z)$, где $V(z)$ — функция в (27) и (28).)

d) Покажите, что, если $u_2 \neq 0$, числа l_n можно вычислить по рекуррентной формуле

$$l_2 = u_2, \qquad \sum_{k=2}^{n} \binom{n}{k} l_k u_{n+1-k} = \sum_{k=2}^{n} l_k u_{nk}.$$

Как следует использовать данную рекуррентную формулу, когда $u_2 = 0$?

e) Докажите равенство

$$u_n = \sum_{m=0}^{n-1} \frac{n!}{m!} \sum_{\substack{k_1+\cdots+k_m=n+m-1 \\ k_1,\ldots,k_m \ge 2}} \frac{n_0}{k_1!} \frac{n_1}{k_2!} \cdots \frac{n_{m-1}}{k_m!} l_{k_1} l_{k_2} \ldots l_{k_m},$$

где $n_j = 1 + k_1 + \cdots + k_j - j$.

24. [*HM25*] Задан степенной ряд $U(z) = U_1 z + U_2 z^2 + \cdots$, где U_1 не равно корню из единицы. Пусть $U = (u_{nk})$ — степенная матрица $U(z)$.

a) Объясните, как вычислить матрицу $\ln U$ таким образом, чтобы степенная матрица $U^{[\alpha]}(z)$ равнялась $\exp(\alpha \ln U) = I + \alpha \ln U + (\alpha \ln U)^2/2! + \cdots$.

b) Покажите, что если $W(z)$ тождественно не равно нулю и если $U(W(z)) = W(U(z))$, то $W(z) = U^{[\alpha]}(z)$ для некоторого комплексного числа α.

25. [*M24*] При $U(z) = z + U_k z^k + U_{k+1} z^{k+1} + \cdots$ и $V(z) = z + V_l z^l + V_{l+1} z^{l+1} + \cdots$, где $k \ge 2$, $l \ge 2$, $U_k \neq 0$, $V_l \neq 0$ и $U(V(z)) = V(U(z))$, докажите, что $k = l$ и $V(z) = U^{[\alpha]}(z)$ для $\alpha = V_k/U_k$.

26. [*M22*] Покажите, что, если $U(z) = U_0 + U_1 z + U_2 z^2 \cdots$ и $V(z) = V_1 z + V_2 z^2 + \cdots$ — степенные ряды, все коэффициенты которых равны либо 0, либо 1, можно получить первые N коэффициентов $U(V(z))$ по модулю 2 за $O(N^{1+\epsilon})$ шагов для любого $\epsilon > 0$.

27. [*M22*] (Д. Зейлбергер (D. Zeilberger).) Найдите рекуррентную формулу, аналогичную (9), для вычисления коэффициентов $W(z) = V(z)V(qz)\dots V(q^{m-1}z)$ при заданных q и m и коэффициентов $V(z) = 1 + V_1 z + V_2 z^2 + \cdots$. Предполагается, что q не равно корню из единицы.

▸ **28.** [*HM26*] *Ряд Дирихле* — это сумма вида $V(z) = V_1/1^z + V_2/2^z + V_3/3^z + \cdots$; произведение $U(z)V(z)$ двух таких рядов — это ряд Дирихле $W(z)$, у которого

$$W_n = \sum_{d\backslash n} U_d V_{n/d}\,.$$

Обычные степенные ряды — частный случай рядов Дирихле, так как $V_0 + V_1 z + V_2 z^2 + V_3 z^3 + \cdots = V_0/1^s + V_1/2^s + V_2/4^s + V_3/8^s + \cdots$, когда $z = 2^{-s}$. На самом деле ряды Дирихле эквивалентны степенным рядам вида $V(z_1, z_2, \dots)$ с произвольным множеством переменных, где $z_k = p_k^{-s}$ и p_k — k-е простое число.

Найдите рекуррентное соотношение, с помощью которого можно обобщить (9) и формулу из упр. 4, если предположить, что задан ряд Дирихле $V(z)$ и что нужно вычислить (a) $W(z) = V(z)^\alpha$, когда $V_1 = 1$; (b) $W(z) = \exp V(z)$, когда $V_1 = 0$; (c) $W(z) = \ln V(z)$, когда $V_1 = 1$. [*Указание.* Пусть $t(n)$ — общее число простых множителей n, включая кратные, и пусть $\delta \sum_n V_n/n^z = \sum_n t(n)V_n/n^z$. Покажите, что операция δ — аналогична производной, например $\delta e^{V(z)} = e^{V(z)}\delta V(z)$.]

"Это, безусловно, мысль, — с некоторым интересом произнес Пуаро. — Да, да, я играю роль компьютера, который питается информацией."

"И, предположим, вы получаете все неправильные ответы", — сказала миссис Оливер.

"Это невозможно, — возразил Эркюль Пуаро. — Компьютеры всего лишь сортируют факты."

"Это не предполагается, — сказала миссис Оливер, — но следует удивляться вещам, которые иногда происходят."

— АГАТА КРИСТИ (AGATHA CHRISTIE),
Прием в честь дня всех святых (1969)

Кажется невозможным, *что некоторый факт становится реальным после* ряда *фактов без той власти, которая впервые их создала.*

— ЭДВАРД СТИЛЛИНГФЛИТ (EDWARD STILLINGFLEET),
Начала таинства, 2:3:2 (1662)

ОТВЕТЫ К УПРАЖНЕНИЯМ

Это единственная ветвь математики, я полагаю, в которой хорошие авторы неоднократно получали совершенно ошибочные результаты.
. . . Сомневаюсь, что есть хотя бы один пространный трактат о вероятностях, существующих в природе, который не содержал бы решений, абсолютно недоказуемых.

— Ч. С. ПИРС (C. S. PEIRCE), *Popular Science Monthly* (1878)

ПРИМЕЧАНИЯ К УПРАЖНЕНИЯМ

1. Задача средней сложности для читателей с математическим уклоном.

3. (Решение Роджера Фрея (Roger Frye), полученное после около 110 часов вычислений на Connection Machine в 1987 году): $95800^4 + 217519^4 + 414560^4 = 422481^4$.

4. Один из читателей рукописи этой книги сообщил, что нашел поистине замечательное доказательство. Но, к сожалению, размер его заметки был слишком мал, чтобы вместить его.

РАЗДЕЛ 3.1

1. (a) Вообще говоря, вы потерпите неудачу, так как владельцы телефонов по возможности выбирают "круглые" номера. Вероятно, в некоторых районах телефонные номера устанавливаются случайным образом. Но в любом случае было бы ошибкой пытаться получить несколько последовательных случайных чисел на одной и той же странице, так как один и тот же телефонный номер часто вносится в список несколько раз подряд.

(b) Вы воспользуетесь левой или правой страницей? Допустим, вы выберете номер на левой странице, разделите его на 2 и возьмете правую цифру. Общему числу страниц следовало бы быть кратным 20, но даже если это так, данный метод все равно обладает определенными недостатками.

(c) Маркировка на гранях кости несколько нарушает симметрию игральной кости, но для практических целей этот метод вполне удовлетворителен (и был использован автором при подготовке нескольких примеров для этой книги). См. *Math. Comp.* **15** (1961), 94–95, где обсуждаются игральные кости в виде икосаэдра.

(d) (Этот сложный вопрос включен в качестве сюрприза.) Число не вполне равномерно распределено. Если среднее число зарегистрированных частиц в минуту равно m, вероятность того, что счетчик зарегистрирует k частиц, равна $e^{-m}m^k/k!$ (пуассоновское распределение); так что цифра "0" выпадает с вероятностью $e^{-m}\sum_{k\ge 0} m^{10k}/(10k)!$ и т. д. В частности, младшая цифра будет четной с вероятностью $e^{-m}\cosh m = \frac{1}{2} + \frac{1}{2}e^{-2m}$ и она никогда не равна $\frac{1}{2}$ (хотя ошибка пренебрежительно мала, когда m велико).

Однако совершенно законно взять десять наблюдений $(m_0, \ldots, m_9)$ и затем выбрать то j, для которого m_j строго меньше, чем m_i для всех $i \neq j$; повторить операцию, если минимальное значение появляется более одного раза (см. (h)).

(e) О'кей; подходит, если время, прошедшее с момента выбора цифры, случайно. Тем не менее возможны искажения в граничных случаях.

(f, g) Нет. Люди обычно выбирают определенные цифры с большей вероятностью (см. 7).

(h) О'кей; при таком распределении номеров вероятность того, что заданная цифра присвоена выигравшей лошади, равна $\frac{1}{10}$.

2. Число таких последовательностей равно полиномиальному коэффициенту $1000000! \,/\, (100000!)^{10}$; таким образом, искомая вероятность равна этому числу, деленному на $10^{1000000}$, число всех последовательностей, содержащих миллион цифр. Используя формулу Стирлинга, найдем, что вероятность близка к $1/(16\pi^4 10^{22}\sqrt{2\pi}) \approx 2.56 \times 10^{-26}$. Грубо говоря, это происходит в одном случае из 4×10^{25}.

3. 3040504030.

4. (a) На шаг K11 мы попадаем только после шага K10 или K2, и в любом случае легко обнаружить, что X не может равняться нулю. Если бы X мог принимать значение "нуль" на этом шаге, то алгоритм никогда бы не закончился (программа зациклилась бы).

(b) Если X установлено равным 3830951656, то вычисления подобны таким же вычислениям шагов из табл. 1. Разница состоит в том, что мы попадаем в состояние K11 $Y + 1 = 7$ раз вместо $Y + 1 = 4$ раз; следовательно, $3830951656 \to 5870802097$. Аналогично $5870802097 \to 1226919902 \to 3172562687 \to 3319967479 \to 6065038420 \to 6065038420 \to \cdots$.

5. Существует только 10^{10} десятизначных чисел; некоторые значения X должны повториться на первых $10^{10} + 1$ шагах. И коль скоро какое-нибудь значение повторится, последовательность повторит свое прежнее поведение (появится периодичность).

6. (a) Используя те же аргументы, что и в предыдущем случае, легко показать, что в последовательности должно в конечном счете повториться одно из значений. Пусть в первый раз такое повторение произойдет на шаге $\mu + \lambda$, где $X_{\mu+\lambda} = X_\mu$. (Это условие определяет μ и λ.) Мы получим $0 \le \mu < m$, $0 < \lambda \le m$, $\mu + \lambda \le m$. Значения $\mu = 0$, $\lambda = m$ достигаются тогда и только тогда, когда f является циклической перестановкой; и $\mu = m - 1$, $\lambda = 1$ встретится, например, если $X_0 = 0$, $f(x) = x + 1$ для $x < m - 1$ и $f(m - 1) = m - 1$.

(b) Для $r > n$ имеем $X_r = X_n$ тогда и только тогда, когда $r - n$ кратно λ и $n \ge \mu$. Значит, $X_{2n} = X_n$ тогда и только тогда, когда n кратно λ и $n \ge \mu$. Ожидаемые результаты следуют теперь незамедлительно. [*Замечание.* Это, в сущности, доказательство известного математического утверждения: "Степени элемента конечной полугруппы включают единственный идемпотентный элемент: возьмите $X_1 = a$, $f(x) = ax$".]

(c) Как только n найдено, генерируем X_i и X_{n+i} для $i \ge 0$ до тех пор, пока найдется первое $X_i = X_{n+i}$; тогда $\mu = i$. Если ни одно из значений X_{n+i} при $0 < i < \mu$ не равно X_n, значит, $\lambda = n$, иначе λ — наименьшее из таких i.

7. (a) Наименьшее значение $n > 0$, такое, что $n - (\ell(n) - 1)$ кратно λ и $\ell(n) - 1 \ge \mu$, равно $n = 2^{\lceil \lg \max(\mu+1, \lambda) \rceil} - 1 + \lambda$. [Это можно сравнить с наименьшим $n > 0$, при котором $X_{2n} = X_n$, т. е. $\lambda(\lceil \mu/\lambda \rceil + \delta_{\mu 0})$.]

(b) Начните со значения $X = Y = X_0$, $k = m = 1$. (На ключевом месте в этом алгоритме получим $X = X_{2m-k-1}$, $Y = X_{m-1}$ и $m = \ell(2m - k)$.) Для генерирования следующего случайного числа выполним такие шаги. Установим $X \leftarrow f(X)$ и $k \leftarrow k - 1$. Если $X = Y$, остановка (длина периода λ равна $m - k$). Если $k = 0$, установим $Y \leftarrow X$, $m \leftarrow 2m$, $k \leftarrow m$. Получим X.

Замечания. Брент (Brent) также рассматривал более общий метод, в котором следующие одно за другим значения $Y = X_{n_i}$ удовлетворяют $n_1 = 0$, $n_{i+1} = 1 + \lfloor pn_i \rfloor$, где p — любое число, большее 1. Он показал, что наилучшее значение p, приближенно равное 2.4771, сокращает около 3% итераций по сравнению с ситуацией, когда $p = 2$. (См. упр. 4.5.4–4.)

Однако метод, описанный в (b), имеет серьезные недостатки, так как можно получить много неслучайных чисел, прежде чем прекратится генерирование, например в особенно неудачном случае, когда $\lambda = 1$, $\mu = 2^k$. Метод основан на идее Флойда (Floyd) из упр. 6, (b): присваивать значения $Y = X_{2n}$ и $X = X_n$ при $n = 0, 1, 2, \ldots$. Потребуется несколько больше функциональных оценок, чем для метода Брента, но при использовании метода Флойда работа остановится прежде, чем любое число появится дважды.

С другой стороны, если f неизвестно (например, если получены значения $X_0, X_1, \ldots$ из постороннего источника) или если сложно использовать f, то в этом случае предпочтительнее применять следующий обнаруживающий циклы алгоритм, который был предложен Р. В. Госпером (R. W. Gosper): для того чтобы получить X_n, используют вспомогательную таблицу значений T_0, T_1, $\ldots$, T_m, где $m = \lfloor \lg n \rfloor$. Сначала присвоим $T_0 \leftarrow X_0$. Для $n = 1, 2, \ldots$ сравниваем X_n с каждым из $T_0, \ldots, T_{\lfloor \lg n \rfloor}$; если в этой таблице не найдется числа, равного X_n, то присваиваем $T_{e(n)} \leftarrow X_n$, где $e(n) = \max\{e \mid 2^e \text{ делит на } n+1\}$. Но если найдется такое T_k, что $X_n = T_k$, то $\lambda = n - \max\{l \mid l < n \text{ и } e(l) = k\}$. После присвоения X_n значения $T_{e(n)}$ последовательно сравниваем это значение с X_{n+1}, $X_{n+2}, \ldots, X_{n+2^{e(n)}+1}$. Поэтому процедура остановится сразу же после генерирования $X_{\mu+\lambda+j}$, где $j \geq 0$ является минимальным среди тех j, для которых выполняется неравенство $e(\mu + j) \geq \lceil \lg \lambda \rceil - 1$. При использовании этого метода ни одно из значений X не появляется более двух раз и не более чем $\max(1, 2^{\lceil \lg \lambda \rceil - 1})$ значений случайных чисел появится более одного раза. [MIT AI Laboratory Memo 239 (February 29, 1972), Hack 132.]

Р. Седгевик (R. Sedgewick), Т. Г. Шиманский (T. G. Szymanski) и Э. Ч. Яо (A. C. Yao) проанализировали более сложный алгоритм, основанный на использовании параметров $m \geq 2$ и $g \geq 1$: вспомогательные таблицы размерности m включают X_0, X_b, $\ldots$, X_{qb} на момент, когда X_n вычислено, где $b = 2^{\lceil \lg n/m \rceil}$ и $q = \lceil n/b \rceil - 1$. Если $n \bmod gb < b$, X_n сравнивается с табличными данными; в конечном счете равенство выполняется и мы вычисляем μ и λ, произведя не более чем $(g+1)2^{\lceil \lg(\mu+\lambda) \rceil + 1}$ вычислений f. Если вычисление f отнимает время τ и если проверка равенства X_n с табличными данными отнимает время σ, то g может быть выбрано так, что общее время счета будет равно $(\mu + \lambda)(\tau + O(\frac{\sigma\tau}{m})^{1/2})$; если $\sigma/\tau = O(m)$, то время счета оптимально. Более того, X_n не вычисляется, пока не выполнится неравенство $\mu + \lambda > mn/(m + 4g + 2)$. Итак, этот "диалоговый" (online) метод можно использовать для гарантированного получения различных случайных чисел, так что для получения одного числа требуется только $2 + O(m^{-1/2})$ вычислений функции. [*SICOMP* **11** (1982), 376–390.]

8. (a,b) $00, 00, \ldots$ [62 начальных значения]; $10, 10, \ldots$ [19]; $60, 60, \ldots$ [15]; $50, 50, \ldots$ [1]; $24, 57, 24, 57, \ldots$ [3]. (c) 42 или 69; оба эти числа дают множества, содержащие 15 различных значений, а именно: 42 или 69, 76, 77, 92, 46, 11, 12, 14, 19, 36, 29, 84, 05, 02, 00.

9. С того момента, когда $X < b^n$, получаем $X^2 < b^{2n}$, и средины квадратов равны $\lfloor X^2/b^n \rfloor \leq X^2/b^n$. Если $X > 0$, то $X^2/b^n < Xb^n/b^n = X$.

10. Если $X = ab^n$, то следующее число последовательности имеет ту же форму; оно равно $(a^2 \bmod b^n)b^n$. Если a кратно всем простым множителям b, последовательность вскоре выродится в нуль; иначе она выродится в циклы чисел такой же формы, как число X.

Другие факты о методе средин квадратов найдены Б. Йенссеном (B. Jansson, *Random Number Generators* (Stockholm: Almqvist & Wiksell, 1966), Section 3A). Тем, кто интересуется магическими свойствами чисел, небезынтересно будет узнать, что число 3792

является самовоспроизводящим числом в четырехзначном методе средин квадратов, так как $3792^2 = 14379264$; более того, как заметил Йенссен, оно также "самовоспроизводимо" в другом смысле, поскольку его разложение на простые множители имеет вид $3 \cdot 79 \cdot 2^4$!

11. Вероятность того, что $\lambda = 1$ и $\mu = 0$, — это вероятность того, что $X_1 = X_0$, а именно — $1/m$. Вероятность того, что $\lambda = 1$, $\mu = 1$ или $\lambda = 2$, $\mu = 0$, — это вероятность того, что $X_1 \neq X_0$, и того, что X_2 принимает определенные значения, т. е. равна $(1 - 1/m)(1/m)$. Аналогично вероятность того, что последовательность имеет любые заданные μ и λ, является функцией только от $\mu + \lambda$, а именно:

$$P(\mu, \lambda) = \frac{1}{m} \prod_{1 \le k < \mu + \lambda} \Big(1 - \frac{k}{m}\Big).$$

Вероятность того, что $\lambda = 1$, задается в виде

$$\sum_{\mu \ge 0} \frac{1}{m} \prod_{k=1}^{\mu} \Big(1 - \frac{k}{m}\Big) = \frac{1}{m} Q(m),$$

где $Q(m)$ определено в разделе 1.2.11.3, равенство (2). По равенству (25) из того же раздела данная вероятность приближенно равна $\sqrt{\pi/2m} \approx 1.25/\sqrt{m}$. Шанс, что алгоритм K будет вести себя, как в рассмотренном примере, равен одному из 80000; автора постигла явная неудача. См. другие комментарии о "колоссальности" в упр. 15.

12. $$\sum_{\substack{1 \le \lambda \le m \\ 0 \le \mu < m}} \lambda P(\mu, \lambda) = \frac{1}{m} \left(1 + 3\Big(1 - \frac{1}{m}\Big) + 6\Big(1 - \frac{1}{m}\Big)\Big(1 - \frac{2}{m}\Big) + \cdots\right) = \frac{1 + Q(m)}{2}.$$

(См. предыдущий ответ. В общем случае, если $f(a_0, a_1, \dots) = \sum_{n \ge 0} a_n \prod_{k=1}^{n} (1 - k/m)$, то $f(a_0, a_1, \dots) = a_0 + f(a_1, a_2, \dots) - f(a_1, 2a_2, \dots)/m$; применим это тождество с $a_n = (n + 1)/2$.) Поэтому среднее значение λ (и в силу симметрии $P(\mu, \lambda)$, а также $\mu + 1$) приближенно равно $\sqrt{\pi m/8} + \frac{1}{3}$. Среднее значение $\mu + \lambda$ равняется точно $Q(m)$, что приближенно равно $\sqrt{\pi m/2} - \frac{1}{3}$. [Альтернативные выводы и другие результаты, включая асимптотические значения моментов, приводятся в A. Rapoport, *Bull. Math. Biophysics* **10** (1948), 145–157, и B. Harris, *Annals Math. Stat.* **31** (1960), 1045–1062; см. также Соболь И. М., *Теория вероятностей и ее применения* **9** (1964), 333–338. Соболь обсуждает асимптотику длины периода для более общей последовательности $X_{n+1} = f(X_n)$, если $n \not\equiv 0 \pmod m$, $X_{n+1} = g(X_n)$, если $n \equiv 0 \pmod m$, когда одновременно и f, и g случайны.]

13. [Пардом П. (Paul Purdom) и Вильямс Дж. (John Williams), *Trans. Amer. Math. Soc.* **133** (1968), 547–551.] Пусть T_{mn} — число функций, имеющих n циклов единичной длины и не имеющих более длинных циклов. Тогда

$$T_{mn} = \binom{m-1}{n-1} m^{m-n}.$$

(Это совпадает с $\binom{m}{n} r(m, m-n)$ в упр. 2.3.4.4-25.) *Любая* такая функция получается из такой же функции в результате перестановки n циклов единичной длины. Отсюда $\sum_{n \ge 1} T_{mn}\, n! = m^m$.

Пусть P_{nk} — число перестановок n элементов, самый длинный цикл которых имеет длину k. Тогда число функций с максимальным циклом длиной k равно $\sum_{n \ge 1} T_{mn} P_{nk}$. Чтобы получить среднее значение k, вычислим сумму $\sum_{k \ge 1} \sum_{n \ge 1} k T_{mn} P_{nk}$, которая, как следует из упр. 1.3.3–23, равна $\sum_{n \ge 1} T_{mn}\, n! (cn + \frac{1}{2}c + O(n^{-1}))$, где $c \approx .62433$. Суммируя, получим, что среднее значение равно $cQ(m) + \frac{1}{2}c + O(m^{1/2})$. (Это выражение, в сущности, не больше, чем среднее значение k, когда X_0 выбирается наудачу. Среднее значение $\max \mu$

асимптотически равно $Q(m)\ln 4$ и среднее значение $\max(\mu+\lambda)$ асимптотически равно $1.9268Q(m)$; см. Flajolet and Odlyzko, *Lecture Notes in Comp. Sci.* **434** (1990), 329–354.)

14. Пусть $c_r(m)$ — число функций, имеющих точно r различных финальных циклов. Из рекуррентного соотношения $c_1(m) = (m-1)! - \sum_{k>0}\binom{m}{k}(-1)^k(m-k)^k c_1(m-k)$, которое можно получить в результате подсчета числа функций, образ которых содержит не более чем $m-k$ элементов, следует, что $c_1(m) = m^{m-1}Q(m)$. (См. упр. 1.2.11.3–16.) Другой способ получения $c_1(m)$, который, возможно, более элегантен и показателен, приведен в упр. 2.3.4.4–17. Значение $c_r(m)$ может быть определено так же, как в упр. 13:

$$c_r(m) = \sum_{n\ge 1} T_{mn}\begin{bmatrix} n \\ r \end{bmatrix} = m^{m-1}\left(\frac{1}{0!}\begin{bmatrix} 1 \\ r \end{bmatrix} + \frac{1}{1!}\begin{bmatrix} 2 \\ r \end{bmatrix}\frac{m-1}{m} + \frac{1}{2!}\begin{bmatrix} 3 \\ r \end{bmatrix}\frac{m-1}{m}\frac{m-2}{m} + \cdots\right).$$

Теперь можно вычислить требуемое среднее значение (см. упр. 12):

$$\begin{aligned} E_m &= \frac{1}{m}\left(H_1 + 2H_2\frac{m-1}{m} + 3H_3\frac{m-1}{m}\frac{m-2}{m} + \cdots\right) \\ &= 1 + \frac{1}{2}\frac{m-1}{m} + \frac{1}{3}\frac{m-1}{m}\frac{m-2}{m} + \cdots. \end{aligned}$$

Последняя формула была получена несколько иным путем М. Д. Крускалом [см. Martin D. Kruskal, *AMM* **61** (1954), 392–397]. Используя интегральное представление

$$E_m = \int_0^\infty \left(\left(1+\frac{x}{m}\right)^m - 1\right)e^{-x}\,\frac{dx}{x},$$

он доказал асимптотическое соотношение $\lim_{m\to\infty}(E_m - \frac{1}{2}\ln m) = \frac{1}{2}(\gamma + \ln 2)$. Другие результаты, а также ссылки на литературу, можно найти в работе John Riordan, *Annals Math. Stat.* **33** (1962), 178–185.

15. Вероятность того, что $f(x) \ne x$ для всех x, равна $(m-1)^m/m^m$, а это приблизительно равно $1/e$. Существование самоповторяющегося значения в алгоритме, подобном алгоритму K, не является поэтому "колоссальным"; его вероятность равна $1 - 1/e \approx .63212$. "Колоссальным" было только то, что автору удалось получить такое значение X_0 при случайном выборе (см. упр. 11).

16. Последовательность повторится, когда пара следующих один за другим элементов встретится второй раз. Максимальный период равен m^2. (См. следующее упражнение.)

17. После произвольного выбора $X_0, \ldots, X_{k-1}$ пусть $X_{n+1} = f(X_n, \ldots, X_{n-k+1})$, где $0 \le x_1, \ldots, x_k < m$ влечет то, что $0 \le f(x_1, \ldots, x_k) < m$. Максимальный период равен m^k. Это очевидная верхняя граница, однако не очевидно, что она достижима, для конструктивного доказательства того, что она достижима для подходящей функции f (обратитесь к упр. 3.2.2–17 и 3.2.2–21; численный метод приводится в упр. 2.3.4.2–23).

18. Метод такой же, как в упр. 7, но используйте цепочку из k элементов $(X_n, \ldots, X_{n-k+1})$ вместо элемента X_n.

20. Достаточно рассмотреть простое отображение $g(X)$, определяемое на шагах K2–K13. Следуя в обратном порядке от числа 6065038420, получим в общей сложности 597 решений; наименьшим числом является 0009612809, а наибольшим — 9995371004.

21. Можно работать с $g(X)$, как и в предыдущем упражнении, но сейчас необходимо запустить функцию в прямом порядке. Существует интересная взаимосвязь между временем и пространством. Используя несколько мегабайтов памяти, автор в 1994 году проверил это предположение для всех начальных значений, меньших, чем 0000165181, но решил подождать несколько лет, прежде чем продолжить исследования, так как компьютеры становились все больше и мощнее. Заметим, что механизм шага K1 способствует уменьшению

периода. Существуют X с большим числом значений, переходящих в них, например 512 значений $X = *6********$ на шаге K2 приведут к шагу K10 со значением $X \leftarrow 0500000000$.

С. Флюхер (S. Fluhrer) обнаружил *другие* фиксированные точки алгоритма K, а именно — значение 5008502835(!). Он также нашел третий цикл $0225923640 \rightarrow 2811514413 \rightarrow 0590051662 \rightarrow 0225923640$, объединяющий семь циклов. Только 128 начальных чисел ведут к повторению значения 5008502835. Алгоритм K — это *ужасный* генератор случайных чисел!

22. Если f — действительно случайная функция, то это было бы идеально; но как построить такую f? Функция, определенная алгоритмом K, работала бы намного лучше по этой схеме, хотя она действительно имеет неслучайные свойства (см. предыдущий ответ).

23. Функция f переставляет свои циклические элементы; пусть $(x_0, \ldots, x_{k-1})$ является "необычным" представлением, обратным к этой перестановке. Затем продолжим определять $x_k, \ldots, x_{m-1}$, как в упр. 2.3.4.4–18. [См. *J. Combinatorial Theory* **8** (1970), 361–375.]

Так, если $m = 10$ и $(f(0), \ldots, f(9)) = (3, 1, 4, 1, 5, 9, 2, 6, 5, 4)$, получим $(x_0, \ldots, x_9) = (4, 9, 5, 1, 1, 3, 4, 2, 6, 5)$; если $(x_0, \ldots, x_9) = (3, 1, 4, 1, 5, 9, 2, 6, 5, 4)$, получим $(f(0), \ldots, f(9)) = (6, 4, 9, 3, 1, 1, 2, 5, 4, 5)$.

РАЗДЕЛ 3.2.1

1. Выберите числа X_0 и a четными, а c — нечетным. Тогда X_n будет нечетным при $n > 0$.

2. Пусть X_r является первым повторившимся значением последовательности. Если X_r равно X_k для некоторого k, где $0 < k < r$, можно доказать, что $X_{r-1} = X_{k-1}$, так как X_n единственным образом определяет X_{n-1}, где a взаимно просто с m. Отсюда $k = 0$.

3. Если d — наибольший общий делитель a и m, величина aX_n может принимать не более m/d значений. Возможна даже худшая ситуация; например, если $m = 2^e$ и a — четное число, формула (6) показывает, что последовательность в конечном счете является константой.

4. Индукция по k.

5. Если числа a и m взаимно простые, то существует число a', для которого $aa' \equiv 1$ (по модулю m). Тогда $X_{n-1} = (a'X_n - a'c) \bmod m$ и в общем случае

$$\begin{aligned} X_{n-k} &= ((a')^k X_n - c(a' + \cdots + (a')^k)) \bmod m \\ &= \Big((a')^k X_n + ((a')^k - 1)c/b\Big) \bmod m, \end{aligned}$$

где $k > 0$, $n - k \geq 0$. Если a и m не являются взаимно простыми числами, то определить X_{n-1}, когда задано X_n, невозможно; числа, кратные $m/\gcd(a, m)$, могут быть добавлены к X_{n-1} без изменения величины X_n. (См. также упр. 3.2.1.3–7.)

РАЗДЕЛ 3.2.1.1

1. Пусть c' — решение уравнения $ac' \equiv c$ по модулю m. (Так, $c' = a'c \bmod m$, если a' — число в ответе к упр. 3.2.1–5.) Тогда

```
LDA X;  ADD CPRIME;  MUL A.
```

Переполнение возможно на операции суммирования. (Из результатов, полученных ниже в этой главе, следует, что, возможно, лучше сберечь единицу времени, положив $c = a$ и заменив операцию `ADD` операцией "`INCA 1`". Затем, если $X_0 = 0$, переполнение не произойдет до конца периода, так что практически его не будет.)

2.
```
RANDM STJ  1F
      LDA  XRAND
      MUL  2F
      SLAX 5
      ADD  3F      (Или INCA c, если c мало́)
      STA  XRAND
1H    JNOV *
      JMP  *-1
XRAND CON  X0
2H    CON  a
3H    CON  c
```
∎

3. Пусть $a' = aw \bmod m$ и пусть m' таково, что $mm' \equiv 1$ (по модулю w). Положим $y \leftarrow \text{lomult}(a', x)$, $z \leftarrow \text{himult}(a', x)$, $t \leftarrow \text{lomult}(m', y)$, $u \leftarrow \text{himult}(m, t)$. Тогда получим $mt \equiv a'x$ (по модулю w). Значит, $a'x - mt = (z - u)w$ и $ax \equiv z - u$ (по модулю m); отсюда следует, что $ax \bmod m = z - u + [z < u]m$.

4. Определим операцию $x \bmod 2^e = y$ тогда и только тогда, когда $x \equiv y$ (по модулю 2^e) и $-2^{e-1} \le y < 2^{e-1}$. Конгруэнтная последовательность $\langle Y_n \rangle$, определенная следующим образом

$$Y_0 = X_0 \bmod 2^{32}, \qquad Y_{n+1} = (aY_n + c) \bmod 2^{32},$$

легко вычисляется на машинах серии System/370, так как младшие разряды произведения y и z равны $(yz) \bmod 2^{32}$ для всех двоичных дополнений чисел y и z; и поскольку при суммировании не принимается во внимание переполнение, она также представляет результат сравнимым по $\bmod 2^{32}$. Эта последовательность обладает всеми свойствами случайности стандартной линейной конгруэнтной последовательности $\langle X_n \rangle$, так как $Y_n \equiv X_n$ по модулю 2^{32}. В самом деле, представление в виде двоичного дополнения Y_n *идентично* двоичному представлению X_n для всех n. [Дж. Марсалья (G. Marsaglia) и Т. А. Брей (T. A. Bray) впервые подчеркнули это в *CACM* **11** (1968), 757–759.]

5. (a) Вычитание: `LDA X; SUB Y; JANN *+2; ADD M`. (b) Суммирование: `LDA X; SUB M; ADD Y; JANN *+2; ADD M`. (Заметим, что если m больше половины длины слова, то операция "`SUB M`" должна предшествовать операции "`ADD Y`".)

6. Эти последовательности мало различаются, так как добавление константы $(m - c)$ дает тот же эффект, что и вычитание константы c. Данная операция должна сочетаться с операцией умножения, так что процесс вычитания имеет небольшие преимущества перед процессом сложения (по крайней мере, для машины MIX). Исключение составляет случай, когда необходимо избежать переполнения.

7. Простые делители $z^k - 1$ появляются в факторизации $z^{kr} - 1$. Если r нечетное, то простые делители $z^k + 1$ появляются в факторизации $z^{kr} + 1$ и $z^{2k} - 1$ равно $(z^k - 1)(z^k + 1)$.

8.
```
JOV  *+1   (Убедитесь, что переполнение выключено.)
LDA  X
MUL  A
STX  TEMP
ADD  TEMP  Добавить младшие разряды к старшим.
JNOV *+2   Если ≥ w, вычесть w − 1.
INCA 1     (Переполнение невозможно на этом шаге.)
```
∎

Замечание. Так как суммирование на e-разрядном компьютере с единичным дополнением производится по $\bmod (2^e - 1)$, то можно комбинировать технику из упр. 4 и 8, выполняя операцию $(yz) \bmod (2^e - 1)$ путем суммирования двух e-разрядных половин произведения yz для всех единично дополненных чисел y и z, не обращая внимания на знак.

9. (a) Обе части равенства совпадают с выражением $aq\lfloor x/q\rfloor$.

(b) Положим $t \leftarrow a(x \bmod q) - r\lfloor x/q\rfloor$, где $r = m \bmod a$; константы q и r могут быть вычислены заранее. Тогда $ax \bmod m = t + [t<0]m$, так как можно доказать, что $t > -m$. Ясно, что $a(x \bmod q) \le a(q-1) < m$. Также $r\lfloor x/q\rfloor \le r\lfloor (m-1)/q\rfloor = r\lfloor a + (r-1)/q\rfloor = ra \le qa < m$, если $0 < r \le q$; и $a^2 \le m$ влечет $r < a \le q$. [Эта методика в неявном виде использована в программе, опубликованной Б. А. Вихманом (B. A. Wichmann) и Я. Д. Хиллом (I. D. Hill): *Applied Stat.* **31** (1982), 190.]

10. Если $r > q$ и $x = m-1$, то $r\lfloor x/q\rfloor \ge (q+1)(a+1) > m$. Итак, условие $r \le q$ необходимо и достаточно для применения метода из упр. 9, (b). Подразумевается, что $\frac{m}{q} - 1 \le a \le \frac{m}{q}$. Пусть $t = \lfloor\sqrt{m}\,\rfloor$. Интервалы $[\frac{m}{q} - 1\,..\,\frac{m}{q}]$ не пересекаются при $1 \le q \le t$ и обязательно включают 1 или 2 целых числа в зависимости от того, будет ли q делителем m. Эти интервалы дают все решения с $a > \sqrt{m}$; они также включают случаи для $a = t$, если $(\sqrt{m} \bmod 1) < \frac{1}{2}$, и $a = t-1$, если $m = t^2$. Таким образом, общее число "удачных" множителей точно равно $2\lfloor\sqrt{m}\,\rfloor + \lfloor d(m)/2\rfloor - [(\sqrt{m} \bmod 1) < \frac{1}{2}] - 1$, где $d(m)$ — число делителей m.

11. Можно предположить, что $a \le \frac{1}{2}m$; иначе можно получить $ax \bmod m$ из $(m-a)x \bmod m$. Тогда можно представить $a = a'a'' - a'''$, где все числа a', a'' и a''' меньше $\sqrt{m}$; например, можно взять $a' \approx \sqrt{m} - 1$ и $a'' = \lceil a/a'\rceil$. Следовательно, $ax \bmod m$ равно $(a'(a''x \bmod m) \bmod m - (a'''x \bmod m)) \bmod m$ и все три внутренние операции могут быть заимствованы из упр. 9.

Когда $m = 2^{31} - 1$, можно воспользоваться тем, что $m-1$ имеет 192 делителя, для определения случаев, в которых $m = q'a' + 1$. При этом упрощается общий метод, так как $r' = 1$. Кроме того, 86 из этих делителей ведут к удачным a'' и a''', когда $a = 62089911$. Наилучшим из этих случаев будет, вероятно, случай, когда $a' = 3641$, $a'' = 17053$, $a''' = 62$, потому что $m-1$ делится как на 3641, так и на 62. Это разложение осуществляется по схеме

$$t \leftarrow 17053(x \bmod 125929) - 16410\lfloor x/125929\rfloor\,,$$
$$t \leftarrow 3641(t \bmod 589806) - \lfloor t/589806\rfloor\,,$$
$$t \leftarrow t - (62(x \bmod 34636833) - \lfloor x/34636833\rfloor)\,,$$

где "$-$" обозначает вычитание по модулю m. Операция сравнения по модулю рассматривается как две операции — умножения и вычитания. Поскольку $x \bmod q = x - q\lfloor x/q\rfloor$ и операция $\lfloor x/q\rfloor$ уже выполнена, то совершено семь умножений, три деления и семь вычитаний. Заметим, что число 62089911 имеет 24 делителя; они позволяют получить пять подходящих факторизаций с $a''' = 0$. Например, когда $a' = 883$ и $a'' = 70317$, достаточно только шести умножений, двух делений, четырех вычитаний:

$$t \leftarrow 883(x \bmod 2432031) - 274\lfloor x/2432031\rfloor\,,$$
$$t \leftarrow 70317(t \bmod 30540) - 2467\lfloor t/30540\rfloor\,.$$

[Может ли наихудшее число умножений и делений быть сведено максимум к 11 для всех a и m, либо 12 является наилучшей верхней границей? Другой способ достичь значения 12 приведен в упр. 4.3.3–19.]

12. (a) Пусть $m = 9999998999 = 10^{10} - 10^3 - 1$. Чтобы умножить $(x_9x_8\ldots x_0)_{10}$ на 10 по модулю m, используем тот факт, что $10^{10}x_9 \equiv 10^3x_9 + x_9$: добавим $(x_9000)_{10}$ к $(x_8x_7\ldots x_0x_9)_{10}$. Чтобы избежать циклических сдвигов, представим, что цифры упорядочены на круге. Добавим цифру высшего порядка x_9 к цифре x_2, переместим на три позиции влево и рассмотрим x_8 как новую цифру высшего порядка. Если $x_9 + x_2 \ge 10$, то перенос совершается влево. И если этот перенос покрывает весь путь влево от x_8, он совершается не только к позиции x_9, но и к позиции x_2. Он может распространяться

от x_9 и x_2 до тех пор, пока процедура не остановится. (Числа также могут не намного превышать m. Например, 0999999900 переходит в $9999999000 = m + 1$, которая переходит в $9999999009 = m + 10$. Но избыточное представление не является обязательно пагубным.)

(b) Это операция *деления* на 10. Выполняем процедуру, обратную процедуре (a): перемещаем цифру высшего порядка *влево* и *вычитаем* новую цифру высшего порядка от третьей цифры слева. Если результат вычитания отрицателен, выполняем "заимствование" обычным способом (алгоритм 4.3.1S), т. е. уменьшаем предыдущую цифру на 1. Заимствование может распространяться, как в (a), но не за позицию цифры высшего порядка. В результате этой операции числа сохраняются неотрицательными и меньшими, чем m. (Таким образом, деление на 10 выполняется проще умножения на 10.)

(c) Можно *запомнить* заимствованный бит вместо того, чтобы распространять его, потому что он может быть включен в процесс вычитания на следующем шаге. Таким образом, если определить цифру x_n и заимствованные биты b_n рекуррентной формулой

$$x_n = (x_{n-10} - x_{n-3} - b_n) \bmod 10 = x_{n-10} - x_{n-3} - b_n + 10b_{n+1},$$

то, используя индукцию по n, можно получить $9999999000^n \bmod 9999998999 = X_n$, где

$$\begin{aligned} X_n &= (x_{n-1}x_{n-2}x_{n-3}x_{n-4}x_{n-5}x_{n-6}x_{n-7}x_{n+2}x_{n+1}x_n)_{10} - 1000b_{n+3} \\ &= (x_{n-1}x_{n-2}\ldots x_{n-10})_{10} - (x_{n-1}x_{n-2}x_{n-3})_{10} - b_n \end{aligned}$$

при начальных условиях $X_0 = 1$. Заметим, что

$$10X_{n+1} = (x_nx_{n-1}x_{n-2}x_{n-3}x_{n-4}x_{n-5}x_{n-6}x_{n+3}x_{n+2}x_{n+1}0)_{10} - 10000b_{n+4} = mx_n + X_n;$$

следовательно, $0 \le X_n < m$ для всех $n \ge 0$.

(d) Если $0 \le U < m$, первая цифра десятичного представления U/m равна $\lfloor 10U/m \rfloor$ и последующие цифры являются десятичным представлением $(10U \bmod m)/m$ (см., например, метод 2, а в разделе 4.4). Таким образом, $U/m = (.u_1u_2\ldots)_{10}$, если положить $U_0 = U$ и $U_n = 10U_{n-1} \bmod m = 10U_{n-1} - mu_n$. Неформально цифры $1/m$ являются начальными цифрами $10^n \bmod m$ при $n = 1, 2, \ldots$. Последовательность, в конце концов, периодическая; она совпадает с начальными цифрами $10^{-n} \bmod m$, взятыми в обратном порядке. Таким образом, можно вычислять их, как в (c).

Точное доказательство, конечно, предпочтительнее неформального. Пусть λ — наименьшее положительное число с $10^\lambda \equiv 1$ (по модулю m). Определим $x_n = x_{n \bmod \lambda}$, $b_n = b_{n \bmod \lambda}$, $X_n = X_{n \bmod \lambda}$ для всех $n < 0$. Тогда рекуррентное соотношение для x_n, b_n и X_n в (c) справедливо для всех целых n. Если $U_0 = 1$, то $U_n = X_{-n}$ и $u_n = x_{-n}$; следовательно,

$$\frac{999999900^n \bmod 9999998999}{9999998999} = (.x_{n-1}x_{n-2}x_{n-3}\ldots)_{10}.$$

(e) Пусть w — длина компьютерного слова w. Используем рекуррентное соотношение

$$x_n = (x_{n-k} - x_{n-l} - b_n) \bmod w = x_{n-k} - x_{n-l} - b_n + wb_{n+1},$$

где $0 < l < k$ и k большое. Тогда $(.x_{n-1}x_{n-2}x_{n-3}\ldots)_w = X_n/m$, где $m = w^k - w^l - 1$ и $X_{n+1} = (w^{k-1} - w^{l-1})X_n \bmod m$. Соотношение

$$X_n = (x_{n-1}\ldots x_{n-k})_w - (x_{n-1}\ldots x_{n-l})_w + b_n$$

справедливо для $n \ge 0$; величины $x_{-1}, \ldots, x_{-k}$ и b_0 должны быть такими, что $0 \le X_0 < m$.

Такие генераторы случайных чисел и подобные им (см. следующие упражнения) были введены в работе G. Marsaglia and A. Zaman, *Annals of Applied Probability* **1** (1991), 462–480. Авторы назвали свой метод *вычитанием с заимствованием*. Они исходили из представления с основанием w дробей со знаменателем m. Связь с линейной конгруэнтной

последовательностью была замечена Шу Тезука (Shu Tezuka) и детально проанализирована в работе Tezuka, L'Ecuyer and Couture, *ACM Trans. Modeling and Computer Simulation* **3** (1993), 315–331. Длина периода обсуждалась в упр. 3.2.1.2–22.

13. Для умножения на 10 сейчас требуется представление добавленных цифр в виде *отрицания* их дополнения. Для этого удобно представить число так, чтобы последние три цифры заменялись отрицанием их дополнений, например $9876543210 = (987654\bar{7}\bar{9}\bar{0})_{10}$. Тогда на 10-м шаге $(x_9 \ldots x_3\bar{x}_2\bar{x}_1\bar{x}_0)_{10}$ равно $(x_8 \ldots x_3x'\bar{x}_1\bar{x}_0\bar{x}_9)_{10}$, где $x' = x_9 - x_2$. Аналогично $(x_9 \ldots x_3\bar{x}_2\bar{x}_1\bar{x}_0)_{10}$, деленное на 10, равно $(x_0x_9 \ldots x_4\bar{x}''\bar{x}_2\bar{x}_1)_{10}$, где $x'' = x_0 - x_3$. Из рекуррентного соотношения

$$x_n = (x_{n-3} - x_{n-10} - b_{n-1}) \bmod 10 = x_{n-3} - x_{n-10} - b_{n-1} + 10b_n$$

следует $8999999101^n \bmod 9999999001 = X_n$, где

$$\begin{aligned} X_n &= (x_{n-1}x_{n-2}x_{n-3}x_{n-4}x_{n-5}x_{n-6}x_{n-7}\bar{x}_{n+2}\bar{x}_{n+1}\bar{x}_n)_{10} + 1000b_{n+3} \\ &= (x_{n-1}x_{n-2} \ldots x_{n-10})_{10} - (x_{n-1}x_{n-2}x_{n-3})_{10} + b_n. \end{aligned}$$

Когда за основание системы счисления вместо 10 принимается w, находим, что обратная степень w по модулю $w^k - w^l + 1$ порождена рекуррентным соотношением

$$x_n = (x_{n-l} - x_{n-k} - b_n) \bmod w = x_{n-l} - x_{n-k} - b_n + wb_{n+1}$$

(таким же, как в упр. 12, но k и l меняются местами).

14. Небольшое обобщение. Для любого b, меньшего или равного длине слова w, можно эффективно осуществлять деление на b по модулю $b^k - b^l \pm 1$. Таким образом, рекуррентное соотношение для x_n почти так же эффективно при $b < w$, как и при $b = w$.

Сильное обобщение. Рекуррентное соотношение

$$x_n = (a_1x_{n-1} + \cdots + a_kx_{n-k} + c_n) \bmod b, \quad c_{n+1} = \left\lfloor \frac{a_1x_{n-1} + \cdots + a_kx_{n-k} + c_n}{b} \right\rfloor$$

эквивалентно $X_n = b^{-1}X_{n-1} \bmod |m|$ в том смысле, что $X_n/|m| = (.x_{n-1}x_{n-2} \ldots)_b$, если определить m и X_n следующим образом:

$$m = a_kb^k + \cdots + a_1b - 1, \qquad X_n = \left(\sum_{j=1}^{k} a_j(x_{n-1} \ldots x_{n-k})_b + c_n \right) (\operatorname{sign} m).$$

Начальные значения $x_{-1} \ldots x_{-k}$ и c_0 должны быть выбраны так, что $0 \le X_0 < |m|$; тогда получим $x_n = (bX_{n+1} - X_n)/|m|$ для $n \ge 0$. Значения x_j для $j < 0$, которые появляются в формуле $X_n/|m| = (.x_{n-1}x_{n-2} \ldots)_b$, можно просто рассматривать, как $x_{j \bmod \lambda}$, где $b^\lambda \equiv 1$ (по модулю m). Эти величины могут отличаться от заданных вначале чисел $x_{-1}, \ldots, x_{-k}$. Цифра переноса c_n удовлетворяет неравенствам

$$\sum_{j=1}^{k} \min(0, a_j) \le c_n < \sum_{j=1}^{k} \max(0, a_j),$$

если начальный перенос c_0 лежит в тех же пределах.

Представляет особый интерес случай, когда $m = b^k + b^l - 1$, для которого $a_j = \delta_{jl} + \delta_{jk}$, поскольку он легко вычисляется. Марсалья и Заман назвали его генератором *суммирования с переносом*:

$$x_n = (x_{n-l} + x_{n-k} + c_n) \bmod b = x_{n-l} + x_{n-k} + c_n - b\,c_{n+1}.$$

Другой привлекательной потенциальной возможностью является использование $k = 2$ в генераторе с, скажем, $b = 2^{31}$ и $m = 65430b^2 + b - 1$. Этот модуль m является простым

числом, и длина периода оказывается равной $(m-1)/2$. Спектральный тест из раздела 3.3.4 показывает, что интервал между уровнями хороший (большие значения ν), хотя, конечно, множитель b^{-1} плохой по сравнению с другими множителями для этого значения модуля m.

В упр. 3.2.1.2–22 содержится дополнительная информация о модулях, позволяющих получить чрезвычайно длинные периоды в методах "вычитание с заимствованием" и "суммирование с переносом".

РАЗДЕЛ 3.2.1.2

1. Согласно теореме A длина периода равна m (см. упр. 3).

2. Да, из этих условий следует, что выполняются условия теоремы A, так как единственным простым делителем 2^e является 2 и любое нечетное число является относительно простым с 2^e. (На самом деле условия упражнения являются *необходимыми* и достаточными.)

3. Согласно теореме A требуется, чтобы $a \equiv 1$ (по модулю 4) и $a \equiv 1$ (по модулю 5). По закону D из раздела 1.2.4 это эквивалентно тому, что $a \equiv 1$ (по модулю 20).

4. Из теоремы A следует, что $X_{2^{e-1}} \equiv 0$ (по модулю 2^{e-1}) (случай, когда $m = 2^{e-1}$). Используя также теорему A при $m = 2^e$, получим, что $X_{2^{e-1}} \not\equiv 0$ (по модулю 2^e). Отсюда следует равенство $X_{2^{e-1}} = 2^{e-1}$. Вообще говоря, можно использовать формулы 3.2.1–(6) для доказательства того, что вторая половина периода, по существу, подобна первой половине, так как $X_{n+2^{e-1}} = (X_n + 2^{e-1}) \bmod 2^e$. (Четверти также подобны; см. упр. 21.)

5. Необходимо, чтобы выполнялось следующее соотношение $a \equiv 1$ (по модулю p) для $p = 3, 11, 43, 281, 86171$. По закону D из раздела 1.2.4 это эквивалентно тому, что $a \equiv 1$ (по модулю $3 \cdot 11 \cdot 43 \cdot 281 \cdot 86171$). Итак, *единственным* решением будет ужасный множитель $a = 1$.

6. (См. предыдущее упражнение.) Из подобия $a \equiv 1$ (по модулю $3 \cdot 7 \cdot 11 \cdot 13 \cdot 37$) следует, что решением будет число $a = 1 + 111111k$ для $0 \le k \le 8$.

7. Воспользуемся обозначениями из доказательства леммы Q. μ является наименьшим значением, при котором $X_{\mu+\lambda} = X_\mu$; также оно является наименьшим значением, при котором $Y_{\mu+\lambda} = Y_\mu$ и $Z_{\mu+\lambda} = Z_\mu$. Таким образом, показано, что $\mu = \max(\mu_1, \ldots, \mu_t)$. Наибольшим из возможных значений μ есть $\max(e_1, \ldots, e_t)$, но никто не пытается достичь его.

8. Легко видеть, что $a^2 \equiv 1$ (по модулю 8) и $a^4 \equiv 1$ (по модулю 16), $a^8 \equiv 1$ (по модулю 32) и т. д. Если $a \bmod 4 = 3$, то $a - 1$ — удвоенное нечетное число. Таким образом, $(a^{2^{e-1}} - 1)/(a - 1) \equiv 0$ (по модулю 2^e) тогда и только тогда, когда $(a^{2^{e-1}} - 1)/2 \equiv 0$ (по модулю $2^{e+1}/2$), что справедливо.

9. Представить выражение для X_n в терминах Y_n и упростить его. Если X_0 по модулю (mod) 4 = 3, формулы из упражнения неприменимы; однако, они применимы к последовательности $Z_n = (-X_n) \bmod 2^e$, которая, по существу, ведет себя так же.

10. Только для $m = 1$, 2, 4, p^e и $2p^e$, для нечетных простых p. Во всех других случаях результат теоремы B является усовершенствованным вариантом теоремы Эйлера (упр. 1.2.4–28).

11. (a) Каждое число $x + 1$ или $x - 1$ (но не оба) кратно 4. Таким образом, $x \mp 1 = q2^f$, где q — нечетное число и $f \ge 1$. (b) При данных обстоятельствах $f < e$ и, значит, $e \ge 3$. Получим $\pm x \equiv 1$ (по модулю 2^f) и $\pm x \not\equiv 1$ (по модулю 2^{f+1}) и $f > 1$. Отсюда, применяя лемму P, находим, что $(\pm x)^{2^{e-f-1}} \not\equiv 1$ (по модулю 2^e), тогда как $x^{2^{e-f}} = (\pm x)^{2^{e-f}} \equiv 1$ (по модулю 2^e). Таким образом, порядок является делителем 2^{e-f}, но не

является делителем 2^{e-f-1}. (c) 1 имеет порядок 1; $2^e - 1$ — порядок 2; максимальный период, где $e \geq 3$, равен, следовательно, 2^{e-2}, и для $e \geq 4$ необходимо, чтобы $f = 2$, т. е. $x \equiv 4 \pm 1$ (по модулю 8).

12. Если k — делитель $p-1$ и если $a^k \equiv 1$ (по модулю p), то по лемме P $a^{kp^{e-1}} \equiv 1$ (по модулю p^e). Аналогично, если $a^{p-1} \equiv 1$ (по модулю p^2), находим, что $a^{(p-1)p^{e-2}} \equiv 1$ (по модулю p^e). Таким образом, в данных случаях a *не* является первообразным элементом. Обратно, если $a^{p-1} \not\equiv 1$ (по модулю p^2), то по теореме 1.2.4F и лемме P имеем, что $a^{(p-1)p^{e-2}} \not\equiv 1$ (по модулю p^e), но $a^{(p-1)p^{e-1}} \equiv 1$ (по модулю p^e). Поэтому порядок является делителем $(p-1)p^{e-1}$, но не $(p-1)p^{e-2}$; следовательно, он имеет вид kp^{e-1}, где k делит $p-1$. Но, если a является первообразным элементом по модулю p, конгруэнтность $a^{kp^{e-1}} \equiv a^k \equiv 1$ (по модулю p) влечет $k = p-1$.

13. Предположим, $a \bmod p \neq 0$, и пусть λ — порядок a по модулю p. По теореме 1.2.4F λ является делителем $p-1$. Если $\lambda < p-1$, то q имеет простой множитель $(p-1)/\lambda$.

14. Пусть $0 < k < p$. Если $a^{p-1} \equiv 1$ (по модулю p^2), то $(a+kp)^{p-1} \equiv a^{p-1} + (p-1)a^{p-2}kp$ (по модулю p^2) и это выражение $\not\equiv 1$, так как $(p-1)a^{p-2}k$ не кратно p. Из упр. 12 следует, что $a+kp$ — первообразный элемент по модулю p^e.

15. (a) Если $\lambda_1 = p_1^{e_1} \dots p_t^{e_t}$ и $\lambda_2 = p_1^{f_1} \dots p_t^{f_t}$, положим $\kappa_1 = p_1^{g_1} \dots p_t^{g_t}$ и $\kappa_2 = p_1^{h_1} \dots p_t^{h_t}$, где

$$\begin{aligned} g_j = e_j \quad &\text{и} \quad h_j = 0, \qquad \text{если} \quad e_j < f_j, \\ g_j = 0 \quad &\text{и} \quad h_j = f_j, \qquad \text{если} \quad e_j \geq f_j. \end{aligned}$$

Тогда $a_1^{\kappa_1}$ и $a_2^{\kappa_2}$ имеют взаимно простые периоды λ_1/κ_1 и λ_2/κ_2 соответственно. К тому же $(\lambda_1/\kappa_1)(\lambda_2/\kappa_2) = \lambda$. Таким образом, достаточно рассмотреть случай, когда λ_1 и λ_2 — взаимно простые числа, т. е. когда $\lambda = \lambda_1\lambda_2$. Теперь, поскольку $(a_1a_2)^\lambda \equiv 1$, получаем $1 \equiv (a_1a_2)^{\lambda\lambda_1} \equiv a_2^{\lambda\lambda_1}$; отсюда следует, что $\lambda\lambda_1$ кратно λ_2. Это влечет, что λ кратно λ_2, так как λ_1 и λ_2 — взаимно простые числа. Аналогично λ кратно λ_1; значит, λ кратно $\lambda_1\lambda_2$. Очевидно, что $(a_1a_2)^{\lambda_1\lambda_2} \equiv 1$; таким образом, $\lambda = \lambda_1\lambda_2$.

(b) Если a_1 имеет порядок $\lambda(m)$, a_2 — порядок λ, из (a) следует, что $\lambda(m)$ кратно λ. С другой стороны, можно найти элемент более высокого порядка, а именно — порядка $\operatorname{lcm}(\lambda, \lambda(m))$.

16. (a) $f(x) = (x-a)(x^{n-1} + (a+c_1)x^{n-2} + \cdots + (a^{n-1} + \cdots + c_{n-1})) + f(a)$.

(b) Утверждение очевидно, когда $n = 0$. Если a является корнем, то $f(x) \equiv (x-a)q(x)$; поэтому, если a' — какой-нибудь другой корень, то

$$0 \equiv f(a') \equiv (a'-a)q(a').$$

Поскольку $a'-a$ не кратно p, то a' должно быть корнем $q(x)$. Итак, если $f(x)$ имеет более n различных корней, то $q(x)$ имеет более $n-1$ различных корней. (c) $\lambda(p) \geq p-1$, так как $f(x)$ должен иметь степень $\geq p-1$ для того, чтобы иметь так много корней. Но по теореме 1.2.4F $\lambda(p) \leq p-1$.

17. Согласно лемме P $11^5 \equiv 1$ (по модулю 25), $11^5 \not\equiv 1$ (по модулю 125), и т. д.; таким образом, порядок 11 есть 5^{e-1} (по модулю 5^e). Однако максимальное значение $\lambda(5^e) = 4 \cdot 5^{e-1}$. Но согласно лемме Q общая длина периода является наименьшим общим кратным периода по модулю 2^e (а именно — 2^{e-2}), периода по модулю 5^e (а именно — 5^{e-1}) и равна $2^{e-2}5^{e-1} = \lambda(10^e)$. Период по модулю 5^e может быть равен 5^{e-1} или $2 \cdot 5^{e-1}$, или $4 \cdot 5^{e-1}$, не влияя на длину периода по модулю 10^e, так как найдено наименьшее общее кратное. Значения первообразных элементов по модулю 5^e сравнимы с 2, 3, 8, 12, 13, 17, 22, 23 по модулю 25 (см. упр. 12), а именно — 3, 13, 27, 37, 53, 67, 77, 83, 117, 123, 133, 147, 163, 173, 187, 197.

18. В соответствии с теоремой C $a \bmod 8$ должно быть равно 3 или 5. Знание периода a по модулю 5 и модулю 25 позволяет применить лемму P, чтобы определить допустимые значения $a \bmod 25$. Период $= 4 \cdot 5^{e-1}$: 2, 3, 8, 12, 13, 17, 22, 23; период $= 2 \cdot 5^{e-1}$: 4, 9, 14, 19; период $= 5^{e-1}$: 6, 11, 16, 21. Каждое из этих 16 значений дает одно значение a, $0 \le a < 200$, такое, что $a \bmod 8 = 3$, и другое значение a, такое, что $a \bmod 8 = 5$.

19. Некоторые примеры можно найти в строках 17–20 табл. 3.3.4–1.

20. (a) $AY_n + X_0 \equiv AY_{n+k} + X_0$ (по модулю m) тогда и только тогда, когда $Y_n \equiv Y_{n+k}$ (по модулю m'). (b) (i) Очевидно. (ii) Теорема A. (iii) $(a^n - 1)/(a - 1) \equiv 0$ (по модулю 2^e) тогда и только тогда, когда $a^n \equiv 1$ (по модулю 2^{e+1}); если $a \not\equiv -1$, порядок a по модулю 2^{e+1} равен удвоенному порядку по модулю 2^e. (iv) $(a^n - 1)/(a - 1) \equiv 0$ (по модулю p^e) тогда и только тогда, когда $a^n \equiv 1$.

21. $X_{n+s} \equiv X_n + X_s$ согласно равенству 3.2.1–(6), и s является делителем m, так как s — степень p, когда m — степень p. Следовательно, заданное целое число q кратно m/s тогда и только тогда, когда $X_{qs} \equiv 0$, а q кратно $m/\gcd(X_s, m)$.

22. Алгоритм 4.5.4P позволяет проверять за приемлемое время, будут ли числа вида $m = b^k \pm b^l \pm 1$ простыми, когда, скажем, $b \approx 2^{32}$ и $l < k \approx 100$. Вычисления могут производиться в b-ичной системе счисления, так как особый вид m содействует ускорению операции возведения в квадрат $\bmod m$. (Рассмотрим, например, возведение в квадрат mod 9999998999 в десятичной системе счисления.) Алгоритм 4.5.4P следует, конечно, использовать только тогда, когда известно, что m не имеет малых делителей.

Марсалья и Заман [*Annals of Applied Probability* **1** (1991), 474–475] показали, что $m = b^{43} - b^{22} + 1$ является простым числом с первообразным корнем b, когда b равно простому числу $2^{32}-5$. Разложение на множители $m-1 = b^{22}(b-1)(b^6+b^5+b^4+b^3+b^2+b+1)(b^{14}+b^7+1)$ требуется для того, чтобы установить первообразность b; один из 17 простых множителей $m - 1$ имеет 99 десятичных цифр. В результате можно быть уверенным, что последовательность $x_n = (x_{n-22} - x_{n-43} - c_n) \bmod b = x_{n-22} - x_{n-43} - c_n + bc_{n+1}$ имеет период длиной $m - 1 \approx 10^{414}$ для каждого ненулевого выбора начальных значений $0 \le x_{-1}, \ldots, x_{-43} < b$, когда $c_0 = 0$.

Тем не менее 43 является, скорее всего, малым значением k с точки зрения шагового критерия "день рождения" (см. раздел 3.3.2J) и 22 примерно равно 43/2. Рассмотрев "смесь", можно прийти к выводу, что мы предпочитаем значения k и l, для которых несколько первых частичных отношений в цепной дроби l/k малы. Чтобы избежать возможных проблем с этим генератором, Люшером (Lüscher) была предложена хорошая идея — отбросить несколько чисел (см. раздел 3.2.2).

Здесь приведено несколько простых чисел вида $b^k \pm b^l \pm 1$, удовлетворяющих ограничению $b = 2^{32}$ и $50 < k \le 100$. Для вычитания с заимствованием: $b^{57} - b^{17} - 1$, $b^{73} - b^{17} - 1$, $b^{86} - b^{62} - 1$, $b^{88} - b^{52} - 1$, $b^{95} - b^{61} - 1$; $b^{58} - b^{33} + 1$, $b^{62} - b^{17} + 1$, $b^{69} - b^{24} + 1$, $b^{70} - b^{57} + 1$, $b^{87} - b^{24} + 1$. Для суммирования с переносом: $b^{56} + b^{22} - 1$, $b^{61} + b^{44} - 1$, $b^{74} + b^{27} - 1$, $b^{90} + b^{65} - 1$. (Неподходящие с точки зрения "смеси" простые числа: $b^{56} - b^5 - 1$, $b^{56} - b^{32} - 1$, $b^{66} - b^{57} - 1$, $b^{76} - b^{15} - 1$, $b^{84} - b^{26} - 1$, $b^{90} - b^{42} - 1$, $b^{93} - b^{18} - 1$; $b^{52} - b^8 + 1$, $b^{60} - b^{12} + 1$, $b^{67} - b^8 + 1$, $b^{67} - b^{63} + 1$, $b^{83} - b^{14} + 1$; $b^{65} + b^2 - 1$, $b^{76} + b^{11} - 1$, $b^{88} + b^{30} - 1$, $b^{92} + b^{48} - 1$.)

Для вычисления периода полученной последовательности необходимо знать множители $m - 1$, но это неосуществимо для таких больших чисел (разве что нам крайне повезет). Предположим, что удалось найти простые множители $q_1, \ldots, q_t$; тогда вероятность, что $b^{(m-1)/q} \bmod m = 1$, является крайне малой, только $1/q$, за исключением очень малых простых q. Следовательно, можно быть совершенно уверенным, что период $b^n \bmod m$ является очень длинным, несмотря на то что множитель $m - 1$ неизвестен.

Действительно, период является почти безусловно очень длинным, даже если m — не простое число. Рассмотрим, например, случай для $k = 10$, $l = 3$, $b = 10$ (кото-

рый мало подходит для генерирования случайных чисел, но значения k, l и b настолько малы, что можно получить точные результаты). $\langle 10^n \bmod m\rangle$ имеет период длиной $\operatorname{lcm}(219, 11389520) = 2494304880$, где $m = 9999998999 = 439 \cdot 22779041$; 4999999500, где $m = 9999999001$; 5000000499, где $m = 10000000999$; $\operatorname{lcm}(1, 16, 2686, 12162) = 130668528$, где $m = 10000001001 = 3 \cdot 17 \cdot 2687 \cdot 72973$. Некоторые редко встречающиеся наборы начальных значений могут сократить период, когда m — не простое число. Но можно быть твердо уверенным в получении хорошего результата, если выбрать, скажем, $k = 1000$, $l = 619$ и $b = 2^{16}$.

РАЗДЕЛ 3.2.1.3

1. $c = 1$ и B^5 — всегда взаимно простые числа, и каждый простой делитель $m = B^5$ является делителем B. Таким образом, по крайней мере квадрат этого делителя делит число $b = B^2$.

2. Только 3. Таким образом, генератор не рекомендуется, несмотря на его длинный период.

3. Потенциал равен 18 в обоих случаях (см. следующее упражнение).

4. Так как из того, что $a \bmod 4 = 1$, следует, что $a \bmod 8 = 1$ или 5, получаем $b \bmod 8 = 0$ или 4. Если b кратно 4, но не кратно 8, а b_1 кратно 8, ясно, что $b^s \equiv 0$ (по модулю $2)^e$ влечет $b_1^s \equiv 0$ (по модулю $2)^e$. Таким образом, b_1 не может иметь потенциал, выше b.

5. Потенциал равен наименьшему из значений s, таких, что $f_j s \geq e_j$ для всех j.

6. Модуль должен делиться на 2^7 или на p^4 (для нечетных простых p) для того, чтобы потенциал был равен 4. Такими будут только величины $m = 2^{27} + 1$ и $10^9 - 1$.

7. $a' = (1 - b + b^2 - \cdots) \bmod m$, где члены b^s, b^{s+1} и т. д. опускаются (если s — потенциал).

8. Так как X_n всегда нечетно,

$$X_{n+2} = (2^{34} + 3 \cdot 2^{18} + 9)X_n \bmod 2^{35} = (2^{34} + 6X_{n+1} - 9X_n) \bmod 2^{35}.$$

Если даны Y_n и Y_{n+1}, то возможности для

$$Y_{n+2} \approx (5 + 6(Y_{n+1} + \epsilon_1) - 9(Y_n + \epsilon_2)) \bmod 10,$$

$0 \leq \epsilon_1 < 1$ и $0 \leq \epsilon_2 < 1$, ограничены и неслучайны.

Замечание. Если множители, предложенные в упр. 3, были бы, скажем, равны $2^{33} + 2^{18} + 2^2 + 1$, а не $2^{23} + 2^{13} + 2^2 + 1$, можно было бы аналогично показать, что $X_{n+2} - 10X_{n+1} + 25X_n \equiv$ constant (по модулю 2^{35}). Вообще говоря, $a \pm \delta$ не должно делиться на высокие степени 2, когда δ мало́. В противном случае получится "несостоятельность второго порядка". Более подробно этот вопрос обсуждается в разделе 3.3.4.

Генератор из этого упражнения рассматривался в работе Мак-Ларена и Марсалья (MacLaren, Marsaglia, *JACM* **12** (1965), 83–89). Недостатки таких генераторов , впервые были продемонстрированы М. Гринбергером (M. Greenberger), *CACM* **8** (1965), 177–179). Даже через десять лет после появления этой работы подобные генераторы широко использовались (см. обсуждение `RANDU` в разделе 3.3.4).

РАЗДЕЛ 3.2.2

1. Метод следует применять с большой осторожностью. Прежде всего, aU_n, вероятно, будет настолько большим, что добавлять c/m не имеет смысла, и операции по "mod 1" почти уничтожат любые следы оставшихся от добавления значений. Мы заключаем, что необходимо использовать арифметические операции с плавающей запятой с двойной точностью. Даже с двойной точностью нужна уверенность, что нет округления и т. д., иначе последовательность чисел будет вести себя совершенно по-другому, так как нарушатся теоретические основы хорошего поведения последовательности (но см. упр. 23).

2. X_{n+1} равно либо $X_{n-1}+X_n$, либо $X_{n-1}+X_n-m$. Если $X_{n+1}<X_n$, то должно быть $X_{n+1}=X_{n-1}+X_n-m$; отсюда $X_{n+1}<X_{n-1}$.

3. (a) Подчеркнутые числа — это $V[j]$ после шага M3.

Выход:	исходный	0 4 5 6 2 0 3(2 7 4 1 6 3 0 5) и повторения
$V[0]$:	0	4 7 7 7 7 7 7 7 4 7 7 7 7 7 7 7 4 7 ...
$V[1]$:	3	3 3 3 3 3 3 2 5 5 5 5 5 5 5 2 5 5 5 ...
$V[2]$:	2	2 2 2 2 0 3 3 3 3 3 3 3 0 3 3 3 3 3 ...
$V[3]$:	5	5 5 6 1 1 1 1 1 1 1 6 1 1 1 1 1 1 1 ...
X:		4 7 6 1 0 3 2 5 4 7 6 1 0 3 2 5 4 7 ...
Y:		0 1 6 7 4 5 2 3 0 1 6 7 4 5 2 3 0 1 ...

Таким образом, потенциал может быть сведен к 1! (См. комментарии, приведенные в ответе к упр. 15.)

(b) Подчеркнутые числа — это $V[j]$ после шага B2.

Выход:	исходный	2 3 6 5 7 0 0 5 3 ... 4 6(3 0 ... 4 7)...
$V[0]$:	0	0 0 0 0 0 0 5 4 4 ... 1 1 1 1 ... 1 1 ...
$V[1]$:	3	3 6 1 1 1 1 1 1 1 ... 0 0 0 4 ... 0 0 ...
$V[2]$:	2	7 7 7 7 3 3 3 3 7 ... 6 2 2 2 ... 7 2 ...
$V[3]$:	5	5 5 5 0 0 2 2 2 ... 3 3 5 5 ... 3 3 ...
X:	4	7 6 1 0 3 2 5 4 7 ... 3 2 5 4 ... 3 2 ...

В этом случае выход значительно лучше входа; он привносит повторяющийся цикл длиной 40 после 46 шагов: 236570 05314 72632 40110 37564 76025 12541 73625 03746 (30175 24061 52317 46203 74531 60425 16753 02647). Можно легко найти цикл, применяя методы из упр. 3.1–7 к приведенной выше таблице до повторяющихся столбцов.

4. Байты младшего порядка многих случайных последовательностей (например, линейная конгруэнтная последовательность с m = длина слова) намного менее случайны, чем байты высокого порядка (см. раздел 3.2.1.1).

5. Эффект рандомизации будет сведен к минимуму, потому что $V[j]$ всегда будет содержать числа из определенного интервала, а именно — $j/k \le V[j]/m < (j+1)/k$. Однако использование подобных подходов вполне допустимо. Можно положить $Y_n = X_{n-1}$ или выбрать j из X_n, извлекая некоторые цифры из середины числа, а не из его левой части. Никакое из этих предложений не будет удлинять период, как это делает алгоритм B. (В упр. 27 показано, однако, что алгоритм B необязательно увеличивает длину периода.)

6. Например, если $\texttt{X}_n/m < \frac{1}{2}$, то $\texttt{X}_{n+1} = 2\texttt{X}_n$.

7. (Мантель B. [W. Mantel, *Nieuw Archief voor Wiskunde* (2) **1** (1897), 172–184.]

Последовательность значений X

00...01
00...10
...
10...00
CONTENTS(A)

становится такой:

00...01
00...10
...
10...00
00...00
CONTENTS(A)

8. Можно предположить, что $X_0 = 0$ и $m = p^e$, как и при доказательстве теоремы 3.2.1.2A. Сначала предположим, что последовательность имеет длину периода p^e. Отсюда следует, что период последовательности по модулю p^f имеет длину p^f для $1 \le f \le e$,

иначе некоторые вычеты по модулю p^f никогда не будут встречаться. Ясно, что c не кратно p; с другой стороны, X_n должно быть кратно p. Если $p \le 3$, то легко доказать необходимость условий (iii) и (iv) методом проб и ошибок, поэтому можно предположить, что $p \ge 5$. Если $d \not\equiv 0$ (по модулю p), то $dx^2 + ax + c \equiv d(x + a_1)^2 + c_1$ (по модулю p^e) для некоторых целых a_1 и c_1 и для всех целых x. Эта квадратичная форма принимает одни и те же значения в точках x и $-x - 2a_1$, поэтому она не может принимать все значения по модулю p^e. Следовательно, $d \equiv 0$ (по модулю p) и, если $a \not\equiv 1$, выполнялось бы равенство $dx^2 + ax + c \equiv x$ (по модулю p) для некоторых x, а это противоречит тому факту, что последовательность по mod p имеет период длиной p.

Чтобы показать достаточность условий, можно воспользоваться теоремой 3.2.1.2A и рассмотреть несколько тривиальных случаев, когда $m = p^e$, где $e \ge 2$. Если $p = 2$, то ясно, что $X_{n+2} \equiv X_n + 2$ (по модулю 4); если $p = 3$, получим $X_{n+3} \equiv X_n - d + 3c$ (по модулю 9), используя (i) и (ii). Для $p \ge 5$ можно следующим образом доказать, что $X_{n+p} \equiv X_n + pc$ (по модулю p^2). Пусть $d = pr$, $a = 1 + ps$. Тогда, если $X_n \equiv cn + pY_n$ (по модулю p^2), необходимо получить $Y_{n+1} \equiv n^2c^2r + ncs + Y_n$ (по модулю p); поэтому $Y_n \equiv \binom{n}{3}2c^2r + \binom{n}{2}(c^2r + cs)$ (по модулю p). Таким образом, $Y_p \bmod p = 0$, и соотношение доказано.

Сейчас можно доказать, что последовательность $\langle X_n \rangle$ целых чисел, определенная в "указании", удовлетворяет соотношению

$$X_{n+p^f} \equiv X_n + tp^f \ \text{(по модулю } p^{f+1}), \qquad n \ge 0,$$

для некоторых t с $t \bmod p \ne 0$ и для всех $f \ge 1$. Этого достаточно для доказательства, что последовательность $\langle X_n \bmod p^e \rangle$ имеет период длиной p^e, если длина периода является делителем p^e, но не делителем p^{e-1}. Соотношение, приведенное выше, уже было установлено для $f = 1$, и для $f > 1$ оно может быть получено по индукции следующим образом. Пусть

$$X_{n+p^f} \equiv X_n + tp^f + Z_n p^{f+1} \ \text{(по модулю } p^{f+2});$$

тогда из квадратичного закона для генерирования последовательности с $d = pr$, $a = 1 + ps$ следует, что $Z_{n+1} \equiv 2rtnc + st + Z_n$ (по модулю p). Поэтому $Z_{n+p} \equiv Z_n$ (по модулю p). Следовательно,

$$X_{n+kp^f} \equiv X_n + k(tp^f + Z_n p^{f+1}) \ \text{(по модулю } p^{f+2})$$

для $k = 1, 2, 3, \ldots$. Если положить теперь $k = p$, то на этом доказательство будет завершено.

Замечание. Если $f(x)$ — полином степени, более высокой, чем 2, и $X_{n+1} = f(X_n)$, анализ будет несколько сложнее, хотя можно использовать тот факт, что $f(m + p^k) = f(m) + p^k f'(m) + p^{2k} f''(m)/2! + \cdots$, для доказательства того, что многие полиномиальные последовательности дают максимальный период. Например, Ковэю (Coveyou) доказал, что период равен $m = 2^e$, если $f(0)$ нечетно, $f'(j) \equiv 1$, $f''(j) \equiv 0$ и $f(j + 1) \equiv f(j) + 1$ (по модулю 4) для $j = 0, 1, 2, 3$. [*Studies in Applied Math.* **3** (Philadelphia: SIAM, 1969), 70–111.]

9. Пусть $X_n = 4Y_n + 2$; тогда последовательность Y_n удовлетворяет квадратичному рекуррентному соотношению $Y_{n+1} = (4Y_n^2 + 5Y_n + 1) \bmod 2^{e-2}$.

10. *Случай 1:* $X_0 = 0$, $X_1 = 1$; следовательно, $X_n \equiv F_n$. Ищем наименьшее n, для которого $F_n \equiv 0$ и $F_{n+1} \equiv 1$ (по модулю 2^e). Так как $F_{2n} = F_n(F_{n-1} + F_{n+1})$, $F_{2n+1} = F_n^2 + F_{n+1}^2$, найдем индукцией по e, что для $e > 1$ $F_{3\cdot 2^{e-1}} \equiv 0$ и $F_{3\cdot 2^{e-1}+1} \equiv 2^e + 1$ (по модулю 2^{e+1}). Это означает, что период является делителем $3 \cdot 2^{e-1}$, но не $3 \cdot 2^{e-2}$. Следовательно, период

равен либо $3 \cdot 2^{e-1}$, либо 2^{e-1}. Но $F_{2^{e-1}}$ — всегда нечетное число (так как только F_{3n} — четное).

Случай 2: $X_0 = a$, $X_1 = b$. Тогда $X_n \equiv aF_{n-1} + bF_n$. Нужно найти наименьшее положительное n, такое, что $a(F_{n+1} - F_n) + bF_n \equiv a$ и $aF_n + bF_{n+1} \equiv b$. Это влечет, что $(b^2 - ab - a^2)F_n \equiv 0$, $(b^2 - ab - a^2)(F_{n+1} - 1) \equiv 0$. И $b^2 - ab - a^2$ нечетно (т. е. оно простое относительно m). Итак, условие эквивалентно тому, что $F_n \equiv 0$, $F_{n+1} \equiv 1$.

Метод определения периода F_n для любого модуля впервые был приведен в статье Д. Д. Волла (D. D. Wall, *AMM* **67** (1960), 525–532). Другие факты относительно последовательности Фибоначчи по модулю 2^e найдены Б. Йенссеном (B. Jansson, *Random Number Generators* (Stockholm: Almqvist & Wiksell, 1966), Section 3C1).

11. (a) Легко видеть, что $z^\lambda = 1 + f(z)u(z) + p^e v(z)$ для некоторых $u(z)$ и $v(z)$, где $v(z) \not\equiv 0$ (по модулю f)(z) и p. По биномиальной теореме

$$z^{\lambda p} = 1 + p^{e+1}v(z) + p^{2e+1}v(z)^2(p-1)/2$$

плюс следующие члены, конгруэнтные нулю по модулям $f(z)$ и p^{e+2}. Так как $p^e > 2$, то $z^{\lambda p} \equiv 1 + p^{e+1}v(z)$ по модулям $f(z)$ и p^{e+2}. Если $p^{e+1}v(z) \equiv 0$ по модулям $f(z)$ и p^{e+2}, должны существовать полиномы $a(z)$ и $b(z)$, такие, что $p^{e+1}(v(z) + pa(z)) = f(z)b(z)$. Поскольку $f(0) = 1$, получаем, что $b(z)$ кратно p^{e+1} (по лемме Гаусса 4.6.1G). Отсюда $v(z) \equiv 0$ по модулям $f(z)$ и p. В итоге получено противоречие.

(b) Если $z^\lambda - 1 = f(z)u(z) + p^e v(z)$, то

$$G(z) = u(z)/(z^\lambda - 1) + p^e v(z)/f(z)(z^\lambda - 1);$$

следовательно, $A_{n+\lambda} \equiv A_n$ (по модулю p^e) для больших n. Обратно, если $\langle A_n \rangle$ обладает последним свойством, то $G(z) = u(z) + v(z)/(1 - z^\lambda) + p^e H(z)$ для некоторых полиномов $u(z)$ и $v(z)$ и некоторого степенного ряда $H(z)$, таких, что ряд и полиномы имеют целые коэффициенты. Отсюда следует равенство $1 - z^\lambda = u(z)f(z)(1 - z^\lambda) + v(z)f(z) + p^e H(z)f(z)(1 - z^\lambda)$, и $H(z)f(z)(1 - z^\lambda)$ является полиномом, так как другой член равенства — полином.

(c) Достаточно доказать, что неравенство $\lambda(p^e) \neq \lambda(p^{e+1})$ влечет такие соотношения $\lambda(p^{e+1}) = p\lambda(p^e) \neq \lambda(p^{e+2})$. Из (a) и (b) следует, что $\lambda(p^{e+2}) \neq p\lambda(p^e)$ и что $\lambda(p^{e+1})$ является делителем $p\lambda(p^e)$, но не $\lambda(p^e)$. Следовательно, если $\lambda(p^e) = p^f q$, где $q \bmod p \neq 0$, то $\lambda(p^{e+1})$ должно равняться $p^{f+1}d$, где d — делитель q. Но сейчас $X_{n+p^f\lambda} \equiv X_n$ (по модулю p^e); значит, $p^{f+1}d$ кратно $p^f q$ и $d = q$. [*Замечание.* Предположение о том, что $p^e > 2$, необходимо; например, если $a_1 = 4$, $a_2 = -1$, $k = 2$, то $\langle A_n \rangle = 1, 4, 15, 56, 209, 780, \ldots$; $\lambda(2) = 2$, $\lambda(4) = 4$, $\lambda(8) = 4$.]

(d) $g(z) = X_0 + (X_1 - a_1X_0)z + \cdots + (X_{k-1} - a_1X_{k-2} - a_2X_{k-3} - \cdots - a_{k-1}X_0)z^{k-1}$.

(e) Утверждение (b) можно обобщить для $G(z) = g(z)/f(z)$; следовательно, предположение, что длина периода равна λ, влечет то, что $g(z)(1 - z^\lambda) \equiv 0$ по модулям $f(z)$ и p^e. Выше был рассмотрен только частный случай для $g(z) = 1$. Но обе части этого сравнения можно умножить на полином Хенселя $b(z)$ и получить, что $1 - z^\lambda \equiv 0$ по модулям $f(z)$ и p^e.

Замечание. Более "элементарное" доказательство результата в (c) можно получить без помощи производящих функций, если использовать метод, аналогичный примененному в ответе к упр. 8. Если $A_{\lambda+n} = A_n + p^e B_n$ для $n = r, r+1, \ldots, r+k-1$ и некоторых целых чисел B_n, то такое же соотношение имеет место для *всех* $n \geq r$, если определить $B_{r+k}, B_{r+k+1}, \ldots$ заданным рекуррентным соотношением. Так как полученная последовательность для B_n является линейной комбинацией смещений последовательности A_n, получим $B_{\lambda+n} \equiv B_n$ (по модулю p^e) для всех достаточно больших значений n. Теперь $\lambda(p^{e+1})$ должно быть кратным $\lambda = \lambda(p^e)$. Для всех достаточно больших n справедли-

вы соотношения $A_{n+j\lambda} = A_n + p^e(B_n + B_{n+\lambda} + B_{n+2\lambda} + \cdots + B_{n+(j-1)\lambda}) \equiv A_n + jp^eB_n$ (по модулю p^{2e}) при $j = 1, 2, 3, \ldots$. Никакие k последовательных B_n не кратны p; отсюда немедленно следуют соотношения $\lambda(p^{e+1}) = p\lambda(p^e) \neq \lambda(p^{e+2})$ при $e \geq 2$. Необходимо еще доказать, что $\lambda(p^{e+2}) \neq p\lambda(p^e)$, когда p нечетно и $e = 1$. Положим $B_{\lambda+n} = B_n + pC_n$ и заметим, что $C_{n+\lambda} \equiv C_n$ (по модулю p), когда n достаточно велико. Тогда $A_{n+p} \equiv A_n + p^2\left(B_n + \binom{p}{2}C_n\right)$ (по модулю p^3), и доказательство окончено.

Историю решения этой задачи можно найти в работе Моргана Варда (Morgan Ward, *Trans. Amer. Math. Soc.* **35** (1933), 600–628); см. также работу Д. В. Робинсона (D. W. Robinson, *AMM* **73** (1966), 619–621).

12. Длина периода по модулю 2 не может быть больше 4. Длина периода по модулю 2^{e+1} не может быть больше удвоенной длины максимального периода mod 2^e. Это следует из предыдущего упражнения. Поэтому длина максимального возможного периода равна 2^{e+1}. Она достигается, например, в тривиальном случае при $a = 0$, $b = c = 1$.

13, 14. Очевидно, что $Z_{n+\lambda} = Z_n$, поэтому λ' является делителем λ. Пусть наименьшее общее кратное λ' и λ_1 равно λ_1', а λ_2' определяется аналогично. Справедливы соотношения $X_n + Y_n \equiv Z_n \equiv Z_{n+\lambda_1'} \equiv X_n + Y_{n+\lambda_1'}$, поэтому λ_1' кратно λ_2. Аналогично можно показать, что λ_2' кратно λ_1. Это дает желаемый результат. (Он является "наилучшим из возможных" в том смысле, что последовательности, для которых $\lambda' = \lambda_0$, могут быть построены так же, как последовательности, для которых $\lambda' = \lambda$.)

15. Алгоритм M генерирует (X_{n+k}, Y_n) на шаге M1 и дает на выходе $Z_n = X_{n+k-q_n}$ на шаге M3 для всех достаточно больших n. Таким образом, $\langle Z_n\rangle$ имеет период длиной λ', где λ' — наименьшее положительное целое число, такое, что $X_{n+k-q_n} = X_{n+\lambda'+k-q_{n+\lambda'}}$ для всех больших n. Так как λ кратно λ_1 и λ_2, отсюда следует, что λ' является делителем λ. (Это заметил Алан Дж. Вотерман (Alan G. Waterman).)

Также справедливы соотношения $n + k - q_n \equiv n + \lambda' + k - q_{n+\lambda'}$ (по модулю λ_1) для всех больших n из-за различия X_n. Предположение об ограниченности $\langle q_n\rangle$ обеспечивает равенство $q_{n+\lambda'} = q_n + c$ для всех больших n, где $c \equiv \lambda'$ (по модулю λ_1) и $|c| < \frac{1}{2}\lambda_1$. Но c должно равняться 0, так как $\langle q_n\rangle$ ограничено. Значит, $\lambda' \equiv 0$ (по модулю λ_1) и $q_{n+\lambda'} = q_n$ для всех больших n. Отсюда следует, что λ' кратно λ_2 и λ_1, поэтому $\lambda' = \lambda$.

Замечание. Из ответа к упр. 3.2.1.2–4 следует, что, если $\langle Y_n\rangle$ — линейная конгруэнтная последовательность максимального периода по модулю $m = 2^e$, длина периода λ_2 будет не больше 2^{e-2}, когда k — степень 2.

16. Существует несколько методов доказательства.

(1) Использование теории конечных полей. Рассмотрим поле, содержащее 2^k элементов. Пусть ξ удовлетворяет соотношению $\xi^k = a_1\xi^{k-1} + \cdots + a_k$. Пусть $f(b_1\xi^{k-1} + \cdots + b_k) = b_k$, где каждое b_j равно нулю или единице, — линейная функция. Если слово X в порождающем алгоритме имеет вид $(b_1b_2\ldots b_k)_2$ до выполнения (10) и если $b_1\xi^{k-1} + \cdots + b_k\xi^0 = \xi^n$, то слово X после выполнения (10) будет иметь вид ξ^{n+1}. Значит, последовательность будет иметь вид $f(\xi^n)$, $f(\xi^{n+1})$, $f(\xi^{n+2})$, ...; и $f(\xi^{n+k}) = f(\xi^n\xi^k) = f(a_1\xi^{n+k-1} + \cdots + a_k\xi^n) = a_1f(\xi^{n+k-1}) + \cdots + a_kf(\xi^n)$.

(2) Использование грубой силы или элементарной изобретательности. Рассмотрим последовательность X_{nj}, $n \geq 0$, $1 \leq j \leq k$, удовлетворяющую соотношениям

$$X_{(n+1)j} \equiv X_{n(j+1)} + a_jX_{n1}, \quad 1 \leq j < k; \qquad X_{(n+1)k} \equiv a_kX_{n1} \text{ (по модулю 2).}$$

Необходимо показать, что $X_{nk} \equiv a_1X_{(n-1)k} + \cdots + a_kX_{(n-k)k}$ для $n \geq k$. В самом деле, получаем, что $X_{nj} \equiv a_1X_{(n-1)j} + \cdots + a_kX_{(n-k)j}$, когда $1 \leq j \leq k \leq n$. Это очевидно для $j = 1$, так как $X_{n1} \equiv a_1X_{(n-1)1} + X_{(n-1)2} \equiv a_1X_{(n-1)1} + a_2X_{(n-2)2} + X_{(n-2)3}$ и т. д. Для

$j > 1$ по индукции получим

$$\begin{aligned} X_{nj} &\equiv X_{(n+1)(j-1)} - a_{j-1}X_{n1} \\ &\equiv \sum_{1\le i\le k} a_i X_{(n+1-i)(j-1)} - a_{j-1}\sum_{1\le i\le k} a_i X_{(n-i)1} \\ &\equiv \sum_{1\le i\le k} a_i (X_{(n+1-i)(j-1)} - a_{j-1}X_{(n-i)1}) \\ &\equiv a_1 X_{(n-1)j} + \cdots + a_k X_{(n-k)j}. \end{aligned}$$

Доказательство *не* зависит от того, как будут рассматриваться операции: по модулю 2 либо по модулю, равному любому простому числу.

17. (a) Когда последовательность закончится, $(k-1)$-мерная строка $(X_{n+1},\ldots,X_{n+k-1})$ появится $(m+1)$-й раз. Для данной $(k-1)$-мерной строки $(X_{r+1},\ldots,X_{r+k-1})$ существует только m предшественников X_r, поэтому один из них должен появиться при $r=0$. (b) Так как строка размерности $(k-1)$ $(0,\ldots,0)$ встречается $(m+1)$ раз, каждый возможный предшественник появится обязательно. Поэтому строка размерности k $(a_1,0,\ldots,0)$ появляется для всех a_1, $0\le a_1<m$. Пусть $1\le s<k$, и предположим, что доказано, что все строки размерности k $(a_1,\ldots,a_s,0,\ldots,0)$ появились в последовательности при $a_s\neq 0$. По построению эта строка размерности k $(a_1,\ldots,a_s,0,\ldots,0,y)$ не может появиться раньше строки $(a_1,\ldots,a_s,0,\ldots,0,y)$ для $1\le y<m$. Следовательно, строка размерности $(k-1)$ $(a_1,\ldots,a_s,0,\ldots,0)$ появилась m раз, и все m возможных предшественников также появились. Это означает, что строка $(a,a_1,\ldots,a_s,0,\ldots,0)$ появилась для $0\le a<m$. Доказательство завершается по индукции.

Результат также следует из теоремы 2.3.4.2D, если использовать ориентированные графы из упр. 2.3.4.2–23. Множество дуг из $(x_1,\ldots,x_j,0,\ldots,0)$ в $(x_2,\ldots,x_j,0,0,\ldots,0)$, где $x_j\neq 0$ и $1\le j\le k$, образуют ориентированное поддерево, четко связанное с десятичными обозначениями Девея.

18. Третий из старших двоичных разрядов U_{n+1} полностью определяется первым и третьим двоичными разрядами U_n. Поэтому появляются только 32 из 64 возможных пар $(\lfloor 8U_n\rfloor,\lfloor 8U_{n+1}\rfloor)$. [*Замечание.* Если бы использовались, скажем, 11-разрядные числа $U_n=(.X_{11n}X_{11n+1}\ldots X_{11n+10})_2$, то последовательность *была бы* удовлетворительной для многих случаев применения. Если другие, имеющие более одного двоичного разряда, постоянные появятся в регистре A, то последовательность будет удовлетворять обобщенному спектральному критерию. (См. упр. 3.3.4–24; необходимо проверять ν_t при $t=36,37,38,\ldots$.]

21. [*J. London Math. Soc.* **21** (1946), 169–172.] Любая последовательность с длиной периода m^k-1 без k последовательных нулей приводит к образованию последовательности с длиной периода m^k, если вставить нули в подходящие места, как в упр. 7. Обратно, можно начинать с последовательности с длиной периода m^k и удалять подходящие нули из периода, чтобы сформировать последовательность другого типа. Назовем эти (m,k)-последовательности последовательностями типа A и B. Предположения обеспечивают существование (p,k)-последовательностей типа A для всех простых чисел p и всех $k\ge 1$. Таким образом, существуют последовательности (p,k) типа B для всех таких p и k.

Чтобы получить последовательность (p^e,k) типа B, возьмем $e=qr$, где q — степень p, а r не кратно p. Начнем с (p,qrk)-последовательности типа A, а именно — с $X_0,X_1,X_2,\ldots$; тогда с помощью системы счисления с основанием p сгруппированные цифры $(X_0\ldots X_{q-1})_p,(X_q\ldots X_{2q-1})_p,\ldots$ образуют последовательность (p^q,rk) типа A, так как q и $p^{qrk}-1$ — взаимно простые числа и последовательность имеет период длиной $p^{qrk}-1$. Это приводит к получению (p^q,rk)-последовательности $\langle Y_n\rangle$ типа B, а $(Y_0Y_1\ldots Y_{r-1})_{p^q}$, $(Y_rY_{r+1}\ldots Y_{2r-1})_{p^q},\ldots$ являются (p^{qr},k)-последовательностями типа B. Доказывается аналогично, так как r и p^{qk} — взаимно простые числа.

Чтобы получить (m,k)-последовательность типа B для произвольного m, можно объединить (p^e,k)-последовательности для каждой степени простого множителя m, используя китайскую теорему об остатках, но существует более простой метод. Пусть $\langle X_n\rangle$ — это (r,k)-последовательность типа B и пусть $\langle Y_n\rangle$ — (s,k)-последовательность типа B, где r и s — взаимно простые числа. Тогда $\langle sX_n + Y_n\rangle$ является (rs,k)-последовательностью типа B.

Простая равномерная конструкция, которая производит $(2,k)$-последовательности для произвольного k, была открыта A. Лемпелем (A. Lempel, *IEEE Trans.* **C-19** (1970), 1204–1209).

22. С помощью китайской теоремы об остатках можно найти константы $a_1,\ldots,a_k$, имеющие желаемые вычеты по модулю, равному каждому простому делителю m. Если $m = p_1p_2\ldots p_t$, длина периода равна $\operatorname{lcm}(p_1^k-1,\ldots,p_t^k-1)$. В самом деле, можно получить приемлемо длинный период для произвольного m (необязательно свободного от квадратов), как показано в упр. 11.

23. Вычитание может происходить быстрее суммирования (см. упр. 3.2.1.1–5); длина периода все еще равна $2^{e-1}(2^{55}-1)$ согласно упр. 30. P. Брент (R. Brent) отметил, что при использовании чисел с плавающей точкой вычисления можно произвести точно в $[0\,..\,1)$ (см. упр. 3.6–11).

24. Обратите последовательность. Другими словами, если $Z_n = Y_{-n}$, то $Z_n = (Z_{n-k+l} - Z_{n-k}) \bmod 2 = (Z_{n-k+l} + Z_{n-k}) \bmod 2$.

25. Эта идея поможет избежать большинства затрат времени на обращение к программам. Например, предположим, что программа A вызывается командой `JMP RANDM`, где

```
RANDM STJ  1F
      LDA  Y,6
      :          }
      ENT6 55    } Программа A.
1H    JMP  *     ▮
```

Цена использования таких случайных чисел составляет $14+\frac{2}{55}$ единиц времени. Но предположим, что мы обращаемся не к этой программе, а к программе получения случайных чисел `RNGEN: DEC6 1; J6Z RNGEN;LDA Y,6`.

```
RNGEN STJ  1F                ENT6 31
      ENT6 24                LDA  Y,6
      LDA  Y+31,6            ADD  Y+24,6
      ADD  Y,6               STA  Y,6
      STA  Y+31,6            DEC6 1
      DEC6 1                 J6P  *-4
      J6P  *-4               ENT6 55
                          1H JMP  *        ▮
```

Сейчас цена равна всего $(12+\frac{6}{55})u$. [Подобное обращение на языке C используется в *The Stanford GraphBase* (New York: ACM Press, 1994), GB_FLIP.] Действительно, во многих случаях предпочтительней генерировать весь массив случайных чисел одновременно. Более того, приведенный выше подход обязателен, когда случайность усиливается методом Люшера (Lüscher) (см. программы на алгоритмических языках C и FORTRAN в разделе 3.6).

27. Пусть $J_n = \lfloor kX_n/m\rfloor$.

Лемма. *После того как* $(k^2+7k-2)/2$ *последовательных значений*

$$0^{k+2}\ 1\ 0^{k+1}\ 2\ 0^k\ \ldots\ (k-1)\ 0^3$$

появятся в последовательности $\langle J_n \rangle$, для алгоритма B будут выполняться неравенства $V[i] < m/k$, $0 \le j < k$, а также $Y < m/k$.

Доказательство. Пусть S_n — множество позиций i, таких, что $V[i] < m/k$ точно перед тем, как генерируется X_n, и пусть j_n — такой индекс, что $V[j_n] \leftarrow X_n$. Если $j_n \notin S_n$ и $J_n = 0$, то $S_{n+1} = S_n \cup \{j_n\}$ и $j_{n+1} > 0$; если же $j_n \in S_n$ и $J_n = 0$, то $S_{n+1} = S_n$ и $j_{n+1} = 0$. После $k+2$ последовательных нулей должно выполняться $0 \in S_n$ и $j_{n+1} = 0$. Тогда после "$1\ 0^{k+1}$" должно выполняться $\{0,1\} \subseteq S_n$ и $j_{n+1} = 0$; после "$2\ 0^k$" выполняется $\{0,1,2\} \subseteq S_n$ и $j_{n+1} = 0$; и т. д.

Следствие. *Пусть $l = (k^2+7k-2)/2$. Если $\lambda \ge lk^l$, то либо алгоритм B обеспечивает длину периода λ, либо последовательность $\langle X_n \rangle$ плохо распределена.*

Доказательство. Вероятность того, что любая данная схема J длиной l не появляется в случайной последовательности длиной λ, меньше, чем $(1-k^{-l})^{\lambda/l} < \exp(-k^{-l}\lambda/l) \le e^{-1}$; следовательно, фиксированные схемы могут появляться. После того как это произойдет, дальнейшее поведение алгоритма B будет таким же каждый раз, когда он достигнет этой части периода. (Когда $k > 4$, то требуется $\lambda > 10^{21}$, поэтому данный результат чисто академический, но возможны также меньшие границы.)

29. Следующий алгоритм в худшем случае осуществляет около k^2 операций. Но среднее время выполнения операций намного меньше; возможно, $O(\log k)$ или даже $O(1)$.

X1. Присвоить $(a_0, a_1, \ldots, a_k) \leftarrow (x_1, \ldots, x_k, m-1)$.

X2. Пусть i — минимум для $a_i > 0$ и $i > 0$. Реализуйте программу Y для $j = i+1$, $\ldots$, k, пока $a_k > 0$.

X3. Если $a_0 > a_k$, $f(x_1, \ldots, x_k) = a_0$; иначе, если $a_0 > 0$, $f(x_1, \ldots, x_k) = a_0 - 1$; иначе $f(x_1, \ldots, x_k) = a_k$. ▮

Y1. Присвоить $l \leftarrow 0$. (Программа на шагах Y1–Y3 проверяет лексикографическое отношение $(a_i, \ldots, a_{i+k-1}) \ge (a_j, \ldots, a_{j+k-1})$. Убывание a_k необходимо, чтобы сделать это неравенство верным. Предполагается, что $a_{k+1} = a_1$, $a_{k+2} = a_2$ и т. д.)

Y2. Если $a_{i+l} > a_{j+l}$, выйти из программы. Иначе, если $j+l = k$, присвоить $a_k \leftarrow a_{i+l}$. Иначе, если $a_{i+l} = a_{j+l}$, перейти к шагу Y3. Иначе, если $j+l > k$, уменьшить a_k на 1 и выйти из программы. Иначе присвоить $a_k \leftarrow 0$ и выйти из программы.

Y3. Увеличить l на 1 и возвратиться к шагу Y2, если $l < k$. ▮

Эта проблема впервые была решена Г. Фредриксеном (H. Fredricksen) для $m = 2$ [*J. Combinatorial Theory* **9** (1970), 1–5; **A12** (1972), 153–154]. В этом частном случае алгоритм проще и может быть задан с k-разрядным регистром. См. также работу S. Xie, *Discrete Applied Math.* **16** (1987), 157–177.

30. Из упр. 11 следует, что достаточно показать, что длина периода по модулям 8 равна $4(2^k-1)$. Это будет выполняться тогда и только тогда, когда $x^{2(2^k-1)} \not\equiv 1$ (по модулям 8 и $f(x)$), а также тогда и только тогда, когда $x^{2^k-1} \not\equiv 1$ (по модулям 4 и $f(x)$). Запишем $f(x) = f_e(x^2) + xf_o(x^2)$, где $f_e(x^2) = \frac{1}{2}(f(x)+f(-x))$. Тогда $f(x)^2 + f(-x)^2 \equiv 2f(x^2)$ (по модулю 8) тогда и только тогда, когда $f_e(x)^2 + xf_o(x)^2 \equiv f(x)$ (по модулю 4); и последнее условие имеет место тогда и только тогда, когда $f_e(x)^2 \equiv -xf_o(x)^2$ (по модулям 4 и $f(x)$), так как $f_e(x)^2 + xf_o(x)^2 = f(x) + O(x^{k-1})$. Более того, работая по модулю 2 и $f(x)$, мы получим $f_e(x)^2 \equiv f_e(x^2) \equiv xf_o(x^2) \equiv x^{2^k} f_o(x)^2$. Следовательно, $f_e(x) \equiv x^{2^{k-1}} f_o(x)$. Поэтому $f_e(x)^2 \equiv x^{2^k} f_o(x)^2$ (по модулям 4 и $f(x)$), отсюда следует утверждение указания. Рассуждая подобным образом, можно доказать, что $x^{2^k} \equiv x$ (по модулю 4 и $f(x)$) тогда и только тогда, когда $f(x)^2 + f(-x)^2 \equiv 2(-1)^k f(-x^2)$ (по модулю 8).

(b) Условие может выполняться только тогда, когда l нечетно и $k = 2l$. Но в таком случае $f(x)$ является первообразным полиномом по модулю 2, когда $k = 2$. [*Math. Comp.* **63** (1994), 389–401.]

31. Соотношение $X_n \equiv (-1)^{Y_n} 3^{Z_n} \bmod 2^e$ выполняется для некоторых Y_n и Z_n по теореме 3.2.1.2C. Следовательно, $Y_n = (Y_{n-24} + Y_{n-55}) \bmod 2$ и $Z_n = (Z_{n-24} + Z_{n-55}) \bmod 2^{e-2}$. Так как Z_k нечетно тогда и только тогда, когда $X_k \bmod 8 = 3$ или 5, из предыдущего упражнения следует, что длина периода равна $2^{e-3}(2^{55} - 1)$.

32. Можно не обращать внимания на $\bmod m$ и вернуться к этому позже. Производящая функция $g(z) = \sum_n X_n z^n$ — это полином, кратный $1/(1 - z^{24} - z^{55})$; поэтому $\sum_n X_{2n} z^{2n} = \frac{1}{2}(g(z)+g(-z))$ — это полином, деленный на $(1-z^{24}-z^{55})(1-z^{24}+z^{55}) = 1-2z^{24}+z^{48} - z^{110}$. Первое требуемое рекуррентное соотношение поэтому имеет вид $X_{2n} = (2X_{2(n-12)} - X_{2(n-24)} + X_{2(n-55)}) \bmod m$. Аналогично $\sum_n X_{3n} z^{3n} = \frac{1}{3}(g(z) + g(\omega z) + g(\omega^2 z))$, где $\omega = e^{2\pi i/3}$, и можно найти, что $X_{3n} = (3X_{3(n-8)} - 3X_{3(n-16)} + X_{3(n-24)} + X_{3(n-55)}) \bmod m$.

33. (a) $g_{n+t}(z) \equiv z^t g_n(z)$ (по модулю m и $1 + z^{31} - z^{55}$) индукцией по t. (b) Так как $z^{500} \bmod (1 + z^{31} - z^{55}) = 792z^2 + z^5 + 17z^6 + 715z^9 + 36z^{12} + z^{13} + 364z^{16} + 210z^{19} + 105z^{23} + 462z^{26} + 16z^{30} + 1287z^{33} + 9z^{36} + 18z^{37} + 1001z^{40} + 120z^{43} + z^{44} + 455z^{47} + 462z^{50} + 120z^{54}$ (см. алгоритм 4.6.1D), то получим $X_{500} = (792X_2 + X_5 + \cdots + 120X_{54}) \bmod m$.

[Интересно сравнить подобную формулу $X_{165} = (X_0 + 3X_7 + X_{14} + 3X_{31} + 4X_{38} + X_{45}) \bmod m$ с "прореженным" рекуррентным соотношением для $\langle X_{3n} \rangle$ в предыдущем упражнении. Метод Люшера для генерирования 165 чисел, в котором используются только первые 55 чисел, очевидно, не лучше идеи генерирования 165 чисел с помощью только X_3, $X_6, \ldots, X_{165}$.]

34. Пусть $q_0 = 0$, $q_1 = 1$, $q_{n+1} = cq_n + aq_{n-1}$. Тогда выполняется равенство $\left(\begin{smallmatrix} 0 & 1 \\ a & c \end{smallmatrix}\right)^n = \left(\begin{smallmatrix} aq_{n-1} & q_n \\ aq_n & q_{n+1} \end{smallmatrix}\right)$, $X_n = (q_{n+1}X_0 + aq_n)/(q_nX_0 + aq_{n-1})$ и $x^n \bmod f(x) \equiv q_n x + aq_{n-1}$ для $n \geq 1$. Таким образом, если $X_0 = 0$, то $X_n = 0$ тогда и только тогда, когда $x^n \bmod f(x)$ — не равная нулю константа.

35. Из условий (i) и (ii) получаем, что $f(x)$ неприводима. Если $f(x) = (x - r_1)(x - r_2)$ и $r_1 r_2 \neq 0$, то $x^{p-1} \equiv 1$, если $r_1 \neq r_2$, и $x^p \equiv r_1$, если $r_1 = r_2$.

Пусть ξ — первообразный корень поля, имеющего p^2 элементов, и предположим, что $\xi^{2k} = c_k \xi^k + a_k$. Квадратичные полиномы, как было установлено, — это точно полиномы $f_k(x) = x^2 - c_k x - a_k$, где $1 \leq k < p^2 - 1$ и $k \perp p + 1$. (См. упр. 4.6.2–16.) Каждый полином появляется для двух значений k; следовательно, число решений равно $\frac{1}{2}(p^2 - 1) \prod_{q \setminus p+1,\, q \text{ простое}} (1 - 1/q)$.

36. В данном случае X_n всегда нечетно, поэтому X_n^{-1} существует по модулю 2^e. Последовательность $\langle q_n \rangle$, определенная в ответе 34, имеет вид 0, 1, 2, 1, 0, 1, 2, 1, ... по модулю 4. Справедливы также $q_{2n} = q_n(q_{n+1} + aq_{n-1})$ и $q_{2n-1} = aq_{n-1}^2 + q_n^2$, поэтому $q_{2n+1} - aq_{2n-1} = (q_{n+1} - aq_{n-1})(q_{n+1} + aq_{n+1})$. Следовательно, $q_{n+1} + aq_{n+1} \equiv 2$ (по модулю 4), когда n четно. Получим, что q_{2^e} — это нечетное кратное 2^e и $q_{2^e+1} - aq_{2^e-1}$ — это нечетное кратное 2^{e+1} для всех $e \geq 0$. Поэтому

$$q_{2^e} + aq_{2^e-1} \equiv q_{2^e+1} + aq_{2^e} + 2^{e+1} \quad (\text{по модулю } 2^{e+2}).$$

И $X_{2^e-2} \equiv (q_{2^e-2+1} + aq_{2^e-2})/(q_{2^e-2} + aq_{2^e-2-1}) \not\equiv 1$ (по модулю 2^e), в то время как $X_{2^e-1} \equiv 1$. И обратно: необходимо, чтобы $a \bmod 4 = 1$ и $c \bmod 4 = 2$, иначе $X_{2n} \equiv 1$ (по модулю 8). [См. работу Эйхенауэра, Лехна и Топазогли (Eichenauer, Lehn, and Topuzoğlu, *Math. Comp.* **51** (1988), 757–759).] Младший двоичный разряд этой последовательности имеет короткий период, поэтому предпочтителен обратный генератор с простым модулем.

37. Можно предположить, что $b_1 = 0$. Из упр. 34 следует, что типичным вектором в V будет вектор

$$(x, (s_2'x + as_2)/(s_2x + as_2''), \ldots, (s_d'x + as_d)/(s_dx + as_d'')),$$

где $s_j = q_{b_j}$, $s_j' = q_{b_j+1}$, $s_j'' = q_{b_j-1}$. Он принадлежит гиперплоскости H тогда и только тогда, когда

$$r_1x + \frac{r_2t_2}{x - u_2} + \cdots + \frac{r_dt_d}{x - u_d} \equiv r_0 - r_2s_2's_2^{-1} - \cdots - r_ds_d's_d^{-1} \text{ (по модулю } p),$$

где $t_j = a - as_j's_j''s_j^{-2} = -(-a)^{b_j}s_j^{-2}$ и $u_j = as_j''s_j^{-1}$. Но это соотношение эквивалентно полиномиальной сравнимости степени $\le d$, поэтому для $d + 1$ нельзя найти значения x, если это соотношение не выполняется для всех x, включая различные точки $x = u_2$, ..., $x = u_d$. Следовательно, $r_2 = \cdots = r_d \equiv 0$ и $r_1 \equiv 0$. [См. работу Ю. Эйхенауэр-Геррмана (J. Eichenauer-Herrmann, *Math. Comp.* **56** (1991), 297–301).]

Замечание. Если рассмотреть матрицу M размера $(p + 1 - d) \times (d + 1)$ со строками $\{(1, v_1, \ldots, v_d) \mid (v_1, \ldots, v_d) \in V\}$, то это упражнение будет эквивалентно утверждению, что любые $d + 1$ строк матрицы M линейно независимы по модулю p. Интересно было бы построить график точек (X_n, X_{n+1}) для $p \approx 1000$ и $0 \le n \le p$; тогда можно увидеть больше следов от окружностей, чем от прямых линий.

РАЗДЕЛ 3.3.1

1. Есть $k = 11$ категорий, поэтому следует использовать строку $\nu = 10$.

2. $\frac{2}{49}, \frac{3}{49}, \frac{4}{49}, \frac{5}{49}, \frac{6}{49}, \frac{9}{49}, \frac{6}{49}, \frac{5}{49}, \frac{4}{49}, \frac{3}{49}, \frac{2}{49}$.

3. $V = 7\frac{173}{240}$ только немногим больше, чем значение, полученное при использовании хороших игральных костей! Существует два объяснения, почему мы не определили, что игральная кость поддельная. (a) Новые вероятности (см. упр. 2) в действительности не очень далеки от тех, которые заданы в (1). Сумма показаний двух игральных костей стремится к сглаживанию вероятностей. Если подсчитать вместо сумм каждую из 36 возможных пар значений, скорее всего, можно будет определить разницу быстрее (предполагая, что игральные кости различимы). (b) Намного более важное объяснение состоит в том, что n является слишком малым для того, чтобы фиксировать значимое различие. Если бы этот же эксперимент был проведен для достаточно большого n, то подделка была бы раскрыта (см. упр. 12).

4. $p_s = \frac{1}{12}$ для $2 \le s \le 12$ и $s \ne 7$; $p_7 = \frac{1}{6}$. Значение V равно $16\frac{1}{2}$. Оно попадает между значениями 75% и 95% табл. 1, так что критерий подтверждает гипотезу, хотя выпало слишком мало семерок.

5. $K_{20}^+ = 1.15$; $K_{20}^- = 0.215$. Значения находятся приблизительно между уровнями 94% и 86% и поэтому не противоречат предположению, но они очень близки. (Данные для этого упражнения взяты из приложения A, табл. 1.)

6. Вероятность, что $X_j \le x$, равна $F(x)$, поэтому имеем биномиальное распределение, обсуждавшееся в разделе 1.2.10. $F_n(x) = s/n$ с вероятностью $\binom{n}{s} F(x)^s(1 - F(x))^{n-s}$; среднее равно $F(x)$; стандартное отклонение — $\sqrt{F(x)(1 - F(x))/n}$ [см. равенство 1.2.10–(19)]. Это указывает на то, что немного лучшей статистикой была бы

$$K_n^+ = \sqrt{n} \max_{-\infty < x < \infty} (F_n(x) - F(x))/\sqrt{F(x)(1 - F(x))};$$

см. упр. 22. Можно вычислить среднее и стандартное отклонения для $F_n(y) - F_n(x)$ при $x < y$ и получить ковариацию между $F_n(x)$ и $F_n(y)$. Используя данные факты, можно показать, что при больших n функция $F_n(x)$ ведет себя, как броуновское движение, и методы из этого раздела теории вероятностей можно использовать для ее изучения.

[См. статьи Дж. Л. Дуба и М. Д. Донскера (J. L. Doob and M. D. Donsker, *Annals Math. Stat.* **20** (1949), 393–403 и **23** (1952), 277–281). Их подход, вообще говоря, рассматривается как наиболее перспективный для изучения КС-критериев.]

7. Положим $j = n$ в (13), чтобы увидеть, что K_n^+ никогда не может быть отрицательной и что максимальное значение, которое она может принимать, равно $\sqrt{n}$. Аналогично, положив $j = 1$, можно произвести подобные наблюдения над K_n^-.

8. Новая КС-статистика была подсчитана для 20 наблюдений. Распределение K_{10}^+ использовалось как $F(x)$, когда КС-статистика была подсчитана.

9. Идея ошибочна, так как все наблюдения должны быть *независимыми*. Между статистиками K_n^+ и K_n^-, полученными из тех же данных, существует связь, поэтому каждую проверку нужно выполнять отдельно. (Большое значение одной из статистик приводит к малому значению другой.) Подобным образом данные на рис. 2 и 5, на которых показаны 15 результатов проверки, не являются 15-ю независимыми наблюдениями, поскольку критерий "максимум-5" не зависит от критерия "максимум-4". Три критерия каждого горизонтального ряда являются независимыми (так как они применялись к разным частям последовательности), но пять критериев в столбцах являются в некоторой степени коррелированными. В результате 95-процентные и другие вероятностные уровни, которые используются в одном критерии, нельзя применять к целой группе критериев, основанных на тех же данных. Мораль такова: генератор случайных чисел можно "проверять" с помощью нескольких критериев, таких как частотный критерий, критерий по максимуму и критерий монотонности, но набор данных, полученных после подсчета различных критериев, нельзя рассматривать как материал для новой проверки, поскольку сами критерии могут быть зависимыми. Статистики K_n^+ и K_n^- следует рассматривать как два различных критерия. Хороший датчик случайных чисел выдержит проверку в обоих случаях.

10. Каждое Y_s и np_s дублируется, поэтому числитель (6) умножается на 4, а знаменатель — только на 2. В итоге новое значение V вдвое больше старого.

11. Эмпирическая функция распределения остается той же; значения K_n^+ и K_n^- умножаются на $\sqrt{2}$.

12. Пусть $Z_s = (Y_s - nq_s)/\sqrt{nq_s}$. Значение V равно n, умноженному на

$$\sum_{s=1}^{k} (q_s - p_s + \sqrt{q_s/n}Z_s)^2/p_s.$$

Это последнее выражение отделено от 0, когда n возрастает (так как $Z_s n^{-1/4}$ ограничено с вероятностью 1). Следовательно, V будет возрастать и принимать совершенно невероятные значения при предположении, что вероятности равны p_s.

Для КС-критерия предположим, что $F(x)$ — предполагаемое распределение, а $G(x)$ — истинное распределение, и пусть $h = \max|G(x) - F(x)|$. Возьмем n достаточно большим, чтобы неравенство $|F_n(x) - G(x)| > h/2$ выполнялось с очень малой вероятностью; тогда $|F_n(x) - F(x)|$ будет невероятно большим для гипотетического распределения $F(x)$.

13. ("max" на самом деле следовало бы заменить обозначением "sup", так как здесь имеется в виду наименьшая верхняя граница. Однако обозначение "max" использовалось, чтобы не смущать читателей, которые не знакомы с обозначением "sup".) Предположим для удобства, что $X_0 = -\infty$, $X_{n+1} = +\infty$. Если $X_j \le x < X_{j+1}$, то $F_n(x) = j/n$, поэтому $\max(F_n(x) - F(x)) = j/n - F(X_j)$ и $\max(F(x) - F_n(x)) = F(X_{j+1}) - j/n$ на данном интервале. Когда j изменяется от 0 до n, рассматриваем все действительные значения x. Это

доказывает равенства

$$K_n^+ = \sqrt{n} \max_{0\le j\le n} \Big(\frac{j}{n} - F(X_j)\Big);$$

$$K_n^- = \sqrt{n} \max_{1\le j\le n+1} \Big(F(X_j) - \frac{j-1}{n}\Big).$$

Данные равенства эквивалентны (13), так как добавочный член под знаком максимума не положителен и изменен в соответствии с упр. 7.

14. Логарифм левой части требуемого равенства можно записать как

$$-\sum_{s=1}^{k} Y_s \ln\Big(1+\frac{Z_s}{\sqrt{np_s}}\Big) + \frac{1-k}{2}\ln(2\pi n) - \frac{1}{2}\sum_{s=1}^{k}\ln p_s - \frac{1}{2}\sum_{s=1}^{k}\ln\Big(1+\frac{Z_s}{\sqrt{np_s}}\Big) + O\Big(\frac{1}{n}\Big),$$

и эта величина (с использованием разложения $\ln(1 + Z_s/\sqrt{np_s})$ и того, что $\sum_{s=1}^{k} Z_s\sqrt{np_s} = 0$), примет вид

$$-\frac{1}{2}\sum_{s=1}^{k} Z_s^2 + \frac{1-k}{2}\ln(2\pi n) - \frac{1}{2}\ln(p_1\dots p_k) + O\left(\frac{1}{\sqrt{n}}\right).$$

15. Соответствующий якобиан легко вычислить: (i) вынеся множитель r^{n-1} из определителя, (ii) разложив полученный определитель по алгебраическим дополнениям строки, содержащей "$\cos\theta_1 \ \ -\sin\theta_1 \ \ 0 \ \dots \ 0$" (каждый детерминант алгебраического дополнения может быть вычислен по индукции) и (iii) вспомнив, что $\sin^2\theta_1 + \cos^2\theta_1 = 1$.

16. $$\int_0^{z\sqrt{2x}+y} \exp\Big(-\frac{u^2}{2x} + \cdots\Big)\,du = ye^{-z^2} + O\left(\frac{1}{\sqrt{x}}\right) + \int_0^{z\sqrt{2x}} \exp\Big(-\frac{u^2}{2x} + \cdots\Big)\,du.$$

Последний интеграл равен

$$\int_0^{z\sqrt{2x}} e^{-u^2/2x}\,du + \frac{1}{3x^2}\int_0^{z\sqrt{2x}} e^{-u^2/2x}u^3\,du + O\left(\frac{1}{\sqrt{x}}\right).$$

Если все это собрать вместе, то будет получено

$$\frac{\gamma(x+1,\, x+z\sqrt{2x}+y)}{\Gamma(x+1)} = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{z\sqrt{2}} e^{-u^2/2}\,du + \frac{e^{-z^2}}{\sqrt{2\pi x}}(y - \tfrac{2}{3} - \tfrac{2}{3}z^2) + O\Big(\frac{1}{x}\Big).$$

Если положить $z\sqrt{2} = x_p$ и записать

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{z\sqrt{2}} e^{-u^2/2}\,du = p, \qquad x+1 = \frac{\nu}{2}, \qquad \gamma\Big(\frac{\nu}{2}, \frac{t}{2}\Big)\Big/\Gamma\Big(\frac{\nu}{2}\Big) = p,$$

где $t/2 = x + z\sqrt{2x} + y$, можно найти y, а именно — $y = \frac{2}{3}(1+z^2) + O(1/\sqrt{x})$, что согласуется с анализом, приведенным выше. Поэтому решение имеет вид $t = \nu + 2\sqrt{\nu}z + \frac{4}{3}z^2 - \frac{2}{3} + O(1/\sqrt{\nu})$.

17. (a) Сделать замену переменных $x_j \leftarrow x_j + t$.

(b) Индукция по n; по определению $P_{n0}(x-t) = \int_n^x P_{(n-1)0}(x_n - t)\,dx_n$.

(c) Левая часть равна

$$\int_n^{x+t} dx_n \dots \int_{k+1}^{x_{k+2}} dx_{k+1}, \qquad \text{умноженному на} \qquad \int_t^k dx_k \int_t^{x_k} dx_{k-1} \dots \int_t^{x_2} dx_1.$$

(d) Из (b) и (c) получим $P_{nk}(x) = \sum_{r=0}^{k} \dfrac{(r-t)^r}{r!}\dfrac{(x+t-r)^{n-r-1}}{(n-r)!}(x+t-n)$. Числитель в (24) равен $P_{n\lfloor t\rfloor}(n)$.

18. Можно предположить, что $F(x) = x$ для $0 \le x \le 1$, как отмечено в выражении (24) раздела. Если $0 \le X_1 \le \cdots \le X_n \le 1$, то пусть $Z_j = 1 - X_{n+1-j}$. Справедливы неравенства $0 \le Z_1 \le \cdots \le Z_n \le 1$, и K_n^+, определенное для $X_1, \ldots, X_n$, равно K_n^-, определенному для $Z_1, \ldots, Z_n$. Эти симметричные соотношения дают взаимно однозначное соответствие между множествами с одним и тем же числом элементов, для которых K_n^+ и K_n^- попадают в одну и ту же область.

20. Например, член порядка $O(1/n)$ равен $-(\frac{4}{9}s^4 - \frac{2}{3}s^2)/n + O(n^{-3/2})$. Полное разложение было получено Х. А. Ловерье (H. A. Lauwerier, *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* **2** (1963), 61–68).

23. Пусть m — некоторое число $\ge n$. (a) Если $\lfloor mF(X_i) \rfloor = \lfloor mF(X_j) \rfloor$ и $i > j$, то $i/n - F(X_i) > j/n - F(X_j)$. (b) Начните с $a_k = 1.0$, $b_k = 0.0$ и $c_k = 0$ для $0 \le k < m$. Затем для каждого наблюдения X_j выполните такие операции: присвойте $Y \leftarrow F(X_j)$, $k \leftarrow \lfloor mY \rfloor$, $a_k \leftarrow \min(a_k, Y)$, $b_k \leftarrow \max(b_k, Y)$, $c_k \leftarrow c_k + 1$. (Предположим, что $F(X_j) < 1$, поэтому $k < m$.) Затем присвойте $j \leftarrow 0$, $r^+ \leftarrow r^- \leftarrow 0$ и для $k = 0, 1, \ldots, m-1$ (в таком порядке) поступайте следующим образом, каким бы ни было $c_k > 0$: присвойте $r^- \leftarrow \max(r^-, a_k - j/n)$, $j \leftarrow j + c_k$, $r^+ \leftarrow \max(r^+, j/n - b_k)$. Окончательно присвойте $K_n^+ \leftarrow \sqrt{n}\, r^+$, $K_n^- \leftarrow \sqrt{n}\, r^-$. Требуемое время равно $O(m+n)$, и точное значение n должно быть известно заранее. (Если оценка $(k + \frac{1}{2})/m$ использована для a_k и b_k так, что только значения c_k действительно вычислены для каждого k, получим оценки для K_n^+ и K_n^- в пределах $\frac{1}{2}\sqrt{n}/m$, даже когда $m < n$.) [*ACM Trans. Math. Software* **3** (1977), 60–64.]

25. (a) Так как $c_{ij} = \mathrm{E}(\sum_{k=1}^n a_{ik}X_k \sum_{l=1}^n a_{jl}X_l) = \sum_{k=1}^n a_{ik}a_{jk}$, то $C = AA^T$.

(b) Рассмотрим сингулярное разложение $A = UDV^T$, где U и V — ортогональные матрицы размера $m \times m$ и $n \times n$ и D — матрица размера $m \times n$ с членами $d_{ij} = [i=j]\sigma_j$; сингулярные значения σ_j все положительны. [См., например, работу Голуба и Ван Лоана (Golub and Van Loan, *Matrix Computations* (1996), §2.5.3.] Если $C\bar{C}C = C$, то $SBS = S$, где $S = DD^T$ и $B = U^T\bar{C}U$. Таким образом, $s_{ij} = [i=j]\sigma_j^2$, где мы считаем, что $\sigma_{n+1} = \cdots = \sigma_m = 0$ и $s_{ij} = \sum_{k,l} s_{ik}b_{kl}s_{lj} = \sigma_i^2\sigma_j^2 b_{ij}$. Следовательно, $b_{ij} = [i=j]/\sigma_j^2$, если $i, j \le n$, и получим, что D^TBD — это единичная матрица размера $n \times n$. Пусть $Y = (Y_1 - \mu_1, \ldots, Y_m - \mu_m)^T$ и $X = (X_1, \ldots, X_n)^T$. Из этого следует, что $W = Y^T\bar{C}Y = X^TA^T\bar{C}AX = X^TVD^TBDV^TX = X^TX$.

РАЗДЕЛ 3.3.2

1. Наблюдения в χ^2-критерии должны быть независимыми. Во второй последовательности соседние наблюдения, очевидно, зависимы, так как вторая компонента одной пары равна первой компоненте следующей пары.

2. Образуйте t-мерную строку $(Y_{jt}, \ldots, Y_{jt+t-1})$ для $0 \le j < n$ и подсчитайте, сколько раз эти строки совпадают с любыми заданными. Примените χ^2-критерий с $k = d^t$ и вероятностью причисления к каждой категории $1/d^t$. Количество наблюдений n должно быть равно по крайней мере $5d^t$.

3. Вероятность того, что точно j значений проверены, а именно — вероятность того, что U_{j-1} является n-м элементом, лежащим в области $\alpha \le U_{j-1} < \beta$, как легко видеть, равна

$$\binom{j-1}{n-1} p^n (1-p)^{j-n}.$$

Ее можно вычислить, подсчитав возможные комбинации из остальных $n-1$ элементов и определив вероятность такой схемы. Производящая функция равна $G(z) = (pz/(1-(1-p)z))^n$, что имеет смысл, так как данное распределение является n-кратной

сверткой того же распределения при $n = 1$. Следовательно, среднее и дисперсия пропорциональны n. Сейчас можно легко найти достаточное для проверки число U_j и его характеристики (min n, ave (среднее) n/p, max ∞, dev (стандартное отклонение) $\sqrt{n(1-p)/p}$). Более детальное обсуждение этого вероятностного распределения, когда $n = 1$, приводится в ответе к упр. 3.4.1–17 (см. также значительно более общие результаты в упр. 2.3.4.2–26).

4. Вероятность интервала длиной $\geq r$ равна вероятности того, что r последовательных U_j лежат вне заданной области, а именно $(1-p)^r$. Вероятность того, что длина интервала r точно равна вероятности того, что длина интервала $\geq r$ минус вероятность, что длина интервала $\geq (r+1)$.

5. Поскольку N стремится к бесконечности, n (с вероятностью 1) также стремится к бесконечности, поэтому данный критерий точно такой же, как критерий интервалов, описанный в разделе, если исключить длины очень больших интервалов. И критерий интервалов данного раздела, определенно, имеет асимптотически χ^2-распределение, поскольку длина каждого интервала не зависит от длин других интервалов. [*Замечание.* Вполне завершенное доказательство этого результата приведено в работе Е. Бофингер и В. Дж. Бофингер (E. Bofinger and V. J. Bofinger *Annals Math. Stat.* **32** (1961), 524–534). Эта работа заслуживает внимания, потому что в ней обсуждается несколько интересных вариантов критерия интервалов. Авторы показали, например, что величина

$$\sum_{0\leq r\leq t} \frac{(Y_r - (Np)p_r)^2}{(Np)p_r}$$

не приближается к χ^2-распределению, хотя эта статистика (другие авторы относили ее к "строгим" критериям) ранее считалась более "сильным" критерием, поскольку Np равно ожидаемому значению n.]

7. 5, 3, 5, 6, 5, 5, 4.

8. См. упр. 10 с $w = d$.

9. (Замените d на w на шагах C1 и C4.) Тогда получим

$$p_r = \frac{d(d-1)\ldots(d-w+1)}{d^r} \left\{ {r-1 \atop w-1} \right\} \qquad \text{для } w \leq r < t;$$

$$p_t = 1 - \frac{d!}{d^{t-1}} \left(\frac{1}{0!} \left\{ {t-1 \atop d} \right\} + \cdots + \frac{1}{(d-w)!} \left\{ {t-1 \atop w} \right\} \right).$$

10. Как и в упр. 3, на самом деле нужно рассмотреть только случай для $n = 1$. Производящая функция для вероятности того, что множество купонов имеет длину r, выглядит следующим образом:

$$G(z) = \frac{d!}{(d-w)!} \sum_{r>0} \left\{ {r-1 \atop w-1} \right\} \left(\frac{z}{d}\right)^r = z^w \left(\frac{d-1}{d-z}\right) \cdots \left(\frac{d-w+1}{d-(w-1)z}\right).$$

Это следует из предыдущего упражнения и формулы 1.2.9–(28). Среднее и дисперсию легко подсчитать, используя теорему 1.2.10A и упр. 3.4.1–17. Найдем, что

$$\text{среднее}(G) = w + \left(\frac{d}{d-1} - 1\right) + \cdots + \left(\frac{d}{d-w+1} - 1\right) = d(H_d - H_{d-w}) = \mu;$$

$$\text{дисперсия}(G) = d^2(H_d^{(2)} - H_{d-w}^{(2)}) - d(H_d - H_{d-w}) = \sigma^2.$$

Число проверенных U_j для набора множеств купонов повторяется n раз, поэтому его характеристики равны (min wn, ave (среднее) μn, max ∞, dev (стандартное отклонение) $\sigma\sqrt{n}$).

11. $|1|2|9\ 8\ 5\ 3|6|7\ 0|4|$.

12. Алгоритм R (*Данные для критерия монотонности*).

R1. [Инициализация.] Присвоить $j \leftarrow -1$ и присвоить $\texttt{COUNT}[1] \leftarrow \texttt{COUNT}[2] \leftarrow \cdots \leftarrow \texttt{COUNT}[6] \leftarrow 0$. Также присвоить $U_n \leftarrow U_{n-1}$ для удобства по завершении работы алгоритма.

R2. [Присвоить r значение 0.] Присвоить $r \leftarrow 0$.

R3. [Будет ли $U_j < U_{j+1}$?] Увеличить r и j на 1. Если $U_j < U_{j+1}$, повторить этот шаг.

R4. [Регистрация длины.] Если $r \geq 6$, увеличить $\texttt{COUNT}[6]$ на 1, иначе увеличить $\texttt{COUNT}[r]$ на 1.

R5. [Конец?] Если $j < n-1$, вернуться к шагу R2. ▮

13. Существует $(p+q+1)\binom{p+q}{p}$ способов получения $U_{i-1} \gtrdot U_i < \cdots < U_{i+p-1} \gtrdot U_{i+p} < \cdots < U_{i+p+q-1}$. Вычтите $\binom{p+q+1}{p+1}$ способов, в которых $U_{i-1} < U_i$, и вычтите $\binom{p+q+1}{1}$ способов, в которых $U_{i+p-1} < U_{i+p}$. Затем добавьте 1 для способа, когда выполняются оба неравенства $U_{i-1} < U_i$ и $U_{i+p-1} < U_{i+p}$, так как этот способ вычитался дважды. (Это частный случай принципа включения-исключения, который подробно рассматривается в разделе 1.3.3.)

14. Серия длиной r встречается с вероятностью $1/r! - 1/(r+1)!$, если предположить, что U_j различны. Поэтому используем $p_r = 1/r! - 1/(r+1)!$ для $r < t$ и $p_t = 1/t!$ — для серий длиной $\geq t$.

15. Это всегда верно для $F(X)$, когда F непрерывна и X имеет распределение F (см. замечание, следующее за формулой 3.3.1–(23)).

16. (a) $Z_{jt} = \max(Z_{j(t-1)}, Z_{(j+1)(t-1)})$. Если $Z_{j(t-1)}$ занесено в память, то достаточно просто преобразовать этот массив в множество Z_{jt} без использования дополнительной памяти. (b) В его "улучшении" каждое V_k должно на самом деле иметь требуемое распределение, но наблюдения не являются независимыми при больших значениях V_k. Действительно, когда U_j — относительно большая величина, то все Z_{jt}, $Z_{(j-1)t}$, $\ldots$, $Z_{(j-t+1)t}$ будут равны U_j, так что возможен эффект повторения тех же данных t раз (V будет умножаться на t, как в упр. 3.3.1–10).

17. (b) Согласно тождеству Бине (Binet) разность равна $\sum_{0\leq k<j<n}(U'_kV'_j - U'_jV'_k)^2$, и поэтому она неотрицательна. (c) Поэтому, если $D^2 = N^2$, получим $U'_kV'_j - U'_jV'_k = 0$ для всех пар j, k. Отсюда следует, что матрица

$$\begin{pmatrix} U'_0 & U'_1 & \ldots & U'_{n-1} \\ V'_0 & V'_1 & \ldots & V'_{n-1} \end{pmatrix}$$

имеет ранг < 2, поэтому ее строки линейно зависимы. (Более элементарное доказательство можно получить, если учесть тот факт, что равенство $U'_0V'_j - U'_jV'_0 = 0$ для $1 \leq j < n$ влечет существование постоянных α, β, таких, что $\alpha U'_j + \beta V'_j = 0$ для всех j при условии, что U'_0 и V'_0 — не нули; последний случай может быть исключен в результате перенумерации.)

18. (a) Числитель равен $-(U_0 - U_1)^2$, знаменатель равен $(U_0 - U_1)^2$. (b) Числитель в этом случае равен $-(U_0^2 + U_1^2 + U_2^2 - U_0U_1 - U_1U_2 - U_2U_0)$, а знаменатель — $2(U_0^2 + \cdots - U_2U_0)$. (c) Знаменатель всегда равен $\sum_{0\leq j<k<n}(U_j - U_k)^2$, что следует из упр. 1.2.3–30 или 1.2.3–31.

19. Сформулированный результат на самом деле имеет место для любого симметричного совместного распределения $U_0, \ldots, U_{n-1}$ (распределение не меняется при перестановках). Пусть $S_1 = U_0 + \cdots + U_{n-1}$, $S_2 = U_0^2 + \cdots + U_{n-1}^2$, $X = U_0U_1 + \cdots + U_{n-2}U_{n-1} + U_{n-1}U_0$ и $D = nS_2 - S_1^2$. Также пусть $\mathrm{E}\, f(U_0, \ldots, U_{n-1})$ обозначает ожидаемое значение $f(U_0, \ldots, U_{n-1})$ при условии, что $D \neq 0$. Так как D — симметричная функция, $\mathrm{E}\, f(U_0, \ldots, U_{n-1}) =$

$\mathrm{E}\, f(U_{p(0)}, \ldots, U_{p(n-1)})$ для всех перестановок p из $\{0, \ldots, n-1\}$. Следовательно, $\mathrm{E}\, S_2/D = n\,\mathrm{E}\, U_0^2/D$, $\mathrm{E}\, S_1^2/D = n(n-1)\,\mathrm{E}(U_0U_1/D) + n\,\mathrm{E}\, U_0^2/D$ и $\mathrm{E}\, X/D = n\,\mathrm{E}(U_0U_1/D)$. Это влечет равенство $1 = \mathrm{E}\,(nS_2 - S_1^2)/D = -(n-1)\,\mathrm{E}\,(nX - S_1^2)/D$. (Строго говоря, $\mathrm{E}\, S_2/D$ и $\mathrm{E}\, S_1^2/D$ могут быть бесконечными, поэтому необходимо позаботиться о том, чтобы можно было работать только с линейными комбинациями ожидаемых величин, которые, как известно, существуют.)

20. Пусть E_{1111}, E_{211}, E_{22}, E_{31} и E_4 означают соответственно величины $\mathrm{E}(U_0U_1U_2U_3/D^2)$, $\mathrm{E}(U_0^2U_1U_2/D^2)$, $\mathrm{E}(U_0^2U_1^2/D^2)$, $\mathrm{E}(U_0^3U_1/D^2)$, $\mathrm{E}(U_0^4/D^2)$. Тогда выполняется $\mathrm{E}\, S_2^2/D^2 = n(n-1)E_{22} + nE_4$, $\mathrm{E}(S_2S_1^2/D^2) = n(n-1)(n-2)E_{211} + n(n-1)E_{22} + 2n(n-1)E_{31} + nE_4$, $\mathrm{E}\, S_1^4/D^2 = n(n-1)(n-2)(n-3)E_{1111} + 6n(n-1)(n-2)E_{211} + 3n(n-1)E_{22} + 4n(n-1)E_{31} + nE_4$, $\mathrm{E}\, X^2/D^2 = n(n-3)E_{1111} + 2nE_{211} + nE_{22}$, $\mathrm{E}(XS_1^2/D^2) = n(n-2)(n-3)E_{1111} + 5n(n-2)E_{211} + 2nE_{22} + 2nE_{31}$, $\mathrm{E}\big((U_0 - U_1)^4/D^2\big) = 6E_{22} - 8E_{31} + 2E_4$, откуда следует первый результат.

Пусть $\delta = \alpha((\ln n)/n)^{1/3}$, $M = \alpha^3/2 + 1/3$ и $m = \lceil 1/\delta \rceil$. Если разделить область распределения на m равновероятных частей, то можно показать, что в каждой части содержится число точек, равное числу, лежащему между $n\delta(1-\delta)$ и $n\delta(1+\delta)$ с вероятностью $\geq 1 - O(n^{-M})$. Для этого нужно использовать неравенства для хвостов 1.2.10–(24) и (25). Следовательно, если распределение равномерно, $D = \frac{1}{12}n^2(1 + O(\delta))$, по крайней мере для этих вероятностей. Если D не из этой области, то $0 \leq (U_0 - U_1)^4/D^2 \leq 1$. Так как $\mathrm{E}\big((U_0 - U_1)^4\big) = \int_0^1 \int_0^1 (x-y)^4\, dx\, dy = \frac{1}{15}$, можно получить, что $\mathrm{E}\big((U_0 - U_1)^4/D^2\big) = \frac{48}{5}n^{-4}(1 + O(\delta)) + O(n^{-M})$.

Замечание. Пусть N — числитель в (23). В. Дж. Диксон (W. J. Dixon) доказал, что когда все переменные имеют нормальное распределение, ожидаемое значение величины $e^{(wN+zD)/n}$ равно

$$(1 - 2z - 2w)^{1/2}\big(1 - 2z + \sqrt{(1-2z)^2 - 4w^2}\,\big)^{-n/2} + O(w^n).$$

Дифференцируя по w и интегрируя по z, он нашел моменты $\mathrm{E}(N/D)^{2k-1} = (-\frac{1}{2})^{\overline{k}}/(n - \frac{1}{2})^{\overline{k}}$, $\mathrm{E}(N/D)^{2k} = (+\frac{1}{2})^{\overline{k}}/(n + \frac{1}{2})^{\overline{k}}$, когда $n > 2k$. В частности, дисперсия в этом случае точно равна $1/(n+1) - 1/(n-1)^2$. [*Annals of Math. Stat.* **15** (1944), 119–144.]

21. Последовательные значения $c_{r-1} = s - 1$ на шаге P2 равны 2, 3, 7, 6, 4, 2, 2, 1, 0; следовательно, $f = 886862$.

22. $1024 = 6! + 2 \cdot 5! + 2 \cdot 4! + 2 \cdot 3! + 2 \cdot 2! + 0 \cdot 1!$, поэтому нужно, чтобы последовательные значения $s - 1$ на шаге P2 были равны 0, 0, 0, 1, 2, 2, 2, 2, 0. Если теперь идти в обратном направлении, то получится перестановка $(9, 6, 5, 2, 3, 4, 0, 1, 7, 8)$.

23. Пусть $P'(x_1, \ldots, x_t) = \frac{1}{\lambda'}\sum_{n=0}^{\lambda'-1}[(Y'_n, \ldots, Y'_{n+t-1}) = (x_1, \ldots, x_t)]$. Тогда

$$Q(x_1, \ldots, x_t) = \sum_{(y_1, \ldots, y_t)} P'(y_1, \ldots, y_t)P((x_1 - y_1) \bmod d,\ \ldots,\ (x_t - y_t) \bmod d)$$

или, более компактно, $Q(x) = \sum_y P'(y)P(x-y)$. Следовательно, используя общее неравенство $(\mathrm{E}\, X)^2 \leq \mathrm{E}\, X^2$, получим $\sum_x (Q(x) - d^{-t})^2 = \sum_x (\sum_y P'(y)(P(x-y) - d^{-t}))^2 \leq \sum_x \sum_y P'(y)(P(x-y) - d^{-t})^2 = \sum_y P'(y) \sum_x (P(x) - d^{-t})^2 = \sum_x (P(x) - d^{-t})^2$. [См. работу G. Marsaglia, *Comp. Sci. and Statistics: Symp. on the Interface* **16** (1984), 5–6. Результат интересен только тогда, когда $d^t \leq 2\lambda$, так как каждое $P(x)$ кратно $1/\lambda$.]

24. Запишите $k : \alpha$ и $\alpha : k$ для первых k и последних k элементов цепочки α. Пусть $K(\alpha, \beta) = [\alpha = \beta]/P(\alpha)$ и пусть $\bar{C}$ — это матрица $d^t \times d^t$ с элементами $\bar{c}_{\alpha\beta} = K(\alpha, \beta) - K(t-1 : \alpha, t-1 : \beta)$. Пусть C — ковариационная матрица случайных величин $N(\alpha)$ для $|\alpha| = t$, деленная на n. Эти величины подчинены ограничению $\sum_{a=0}^{d-1} N(\alpha a) = \sum_{a=0}^{d-1} N(a\alpha)$ для каждой из d^{t-1} цепочек α, но все другие линейные ограничения вытекают из этого

(см. теорему 2.3.4.2G). Поэтому C имеет ранг $d^t - d^{t-1}$ и согласно упр. 3.3.1–25 достаточно показать, что $C\bar{C}C = C$.

Нетрудно проверить, что $c_{\alpha\beta} = P(\alpha\beta)\sum_{|k|<t} T_k(\alpha,\beta)$, где $T_k(\alpha,\beta)$ — член, соответствующий пересечению, которое может произойти, если наложить β на α и сдвинуть ее на k позиций вправо:

$$T_k(\alpha,\beta) = \begin{cases} K(t+k:\alpha,\ \beta:t+k)-1, & \text{если } k\le 0;\\ K(\alpha:t-k,\ t-k:\beta)-1, & \text{если } k\ge 0.\end{cases}$$

Например, если $d=2$, $t=5$, $\alpha=01101$ и $\beta=10101$, то выполняется $c_{\alpha\beta} = P(0)^4P(1)^6\times (P(01)^{-1}+P(101)^{-1}+P(1)^{-1}-9)$. Элемент $\alpha\beta\ C\bar{C}C$ поэтому равен $P(\alpha\beta)$, умноженному на

$$\sum_{|\gamma|=t-1}\ \sum_{a,b=0}^{d-1} P(\gamma ab)\sum_{|k|<t}\sum_{|l|<t} T_k(\alpha,\gamma a)(K(a,b)-1)T_l(\gamma b,\beta)\,.$$

Если заданы k и l, то произведение $T_k(\alpha,\gamma a)(K(a,b)-1)T_l(\gamma b,\beta)$ разлагается на восемь членов, к каждому из которых добавляется ±1, до умножения на $P(\gamma ab)$, а затем выполняется суммирование по всем γab. Например, сумма $P(\gamma ab)K(2:\alpha,\ \gamma a:2)K(a,b)\times K(3:\gamma b,\ \beta:3)$, когда $\alpha=a_1\dots a_t$, $\beta=b_1\dots b_t$, $\gamma=c_1\dots c_{t-1}$ и $t\ge5$, равна сумме $P(c_4\dots c_{t-2})$, которая равна 1. Если $t=4$, та же сумма должна равняться $K(a_1,b_4)$, но не должна входить в сумму $P(\gamma ab)K(2:\alpha,\ \gamma a:2)(-1)K(3:\gamma b,\ \beta:3)$. Поэтому результат равен нулю, если только не выполняется неравенство $k\le0\le l$; в противном случае исключаются $K(i:(j:\alpha),\ (\beta:l)-1:i)-K(i-1:(j:\alpha),\ (\beta:l+1):i-1)$, где $i=\min(t+k,\ t-l)$ и $j=\max(0,\ k+l)$. Сумма по k и l прибавляется к $c_{\alpha\beta}$.

25. Эмпирическая проверка показывает, что на самом деле, когда (22) обобщено для произвольного t, отношение соответствующих элементов C_1^{-1} и $C_1^{-1}C_2C_1^{-1}$ очень близко к $-t$ при $t\ge5$. Например, когда $t=6$, все эти соотношения лежат между -6.039 и -6.111; когда $t=20$, то все они лежат между -20.039 и -20.045. Этот феномен требует объяснения.

26. (a) Вектор $(S_1,\dots,S_n)$ является равномерно распределенной точкой в $(n-1)$-мерном многограннике, определенном неравенствами $S_1\ge0$, …, $S_n\ge0$ на гиперплоскости $S_1+\cdots+S_n=1$. Легко доказать по индукции, что

$$\int_{s_1}^{\infty} dt_1\int_{s_2}^{\infty} dt_2\cdots\int_{s_{n-1}}^{\infty} dt_{n-1}\,[1-t_1-\cdots-t_{n-1}\ge s_n] = \frac{(1-s_1-s_2-\cdots-s_n)_+^{n-1}}{(n-1)!}.$$

Чтобы получить вероятности, разделим этот интеграл на его значение в частном случае, когда $s_1=\cdots=s_n=0$ [см. работу Бруно де Финетти (Bruno de Finetti, *Giornale Istituto Italiano degli Attuari* **27** (1964), 151–173)].

(b) Вероятность того, что $S_{(1)}\ge s$, равна вероятности $S_1\ge s$, …, $S_n\ge s$.

(c) Вероятность того, что $S_{(k)}\ge s$, равна вероятности, что по крайней мере $k-1$ из S_j будет $<s$. Следовательно, $1-F_k(s)=G_1(s)+\cdots+G_{k-1}(s)$, где $G_j(s)$ — это вероятность, что точно j разностей $<s$. Из соображений симметрии $G_j(s)$ равно $\binom{n}{j}$, умноженному на вероятность того, что $S_1<s$, …, $S_j<s$, $S_{j+1}\ge s$, …, $S_n\ge s$, и эта вероятность равна $\Pr(S_1<s,\dots,S_{j-1}<s,S_j\ge0,S_{j+1}\ge s,\dots,S_n\ge s)-\Pr(S_1<s,\dots,S_{j-1}<s,S_j\ge s,\dots,S_n\ge s)$. Повторное применение (a) показывает, что $G_j(s)=\binom{n}{j}\sum_l\binom{j}{l}(-1)^{j-l}(1-(n-l)s)_+^{n-1}$; следовательно,

$$1-F_k(s)=\sum_l\binom{n}{l}\binom{n-l-1}{k-l-1}(-1)^{k-l-1}(1-(n-l)s)_+^{n-1}.$$

В частности, наибольшая разность $S_{(n)}$ имеет распределение

$$F_n(s) = 1 - \sum_l \binom{n}{l}\binom{n-l-1}{n-l-1}(-1)^{n-l-1}(1-(n-l)s)_+^{n-1} = \sum_l \binom{n}{l}(-1)^l(1-ls)_+^{n-1}.$$

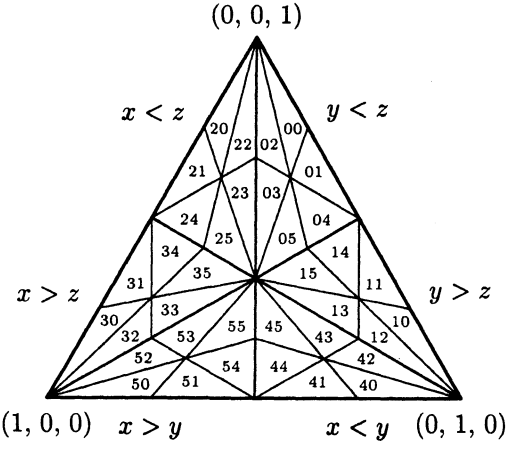
[Между прочим, подобная величина $x^{n-1}(n-1)!^{-1}F_n(x^{-1})$, оказывается, является *плотностью* распределения *суммы* $U_1+\cdots+U_n$ равномерно распределенных случайных величин.]

(d) Из формул $\mathrm{E}\, s^r = r\int_0^1(1-F(s))s^{r-1}\,ds$ и $\int_0^1 s^r(1-ks)_+^{n-1}\,ds = k^{-r-1}n^{-1}\binom{n+r}{r}^{-1}$ находим $\mathrm{E}\, S_{(k)} = n^{-1}(H_n - H_{n-k})$ и с помощью алгебры получаем $\mathrm{E}\, S_{(k)}^2 = n^{-1}(n+1)\times(H_n^{(2)} - H_{n-k}^{(2)} + (H_n - H_{n-k})^2)$. Поэтому дисперсия $S_{(k)}$ равна $n^{-1}(n+1)^{-1}(H_n^{(2)} - H_{n-k}^{(2)} - (H_n - H_{n-k})^2/n)$.

[Распределение $F_k(s)$ впервые было найдено В. А. Витвортом (W. A. Whitworth), который сформулировал вопрос о распределении $S_{(k)}$ как проблему 667 в *Choice and Chance* (Cambridge, 1867) и дал ее решение в *DCC Exercises in Choice and Chance* (Cambridge, 1897). Витворт также нашел элегантный метод подсчета математического ожидания любого полинома от функции $G_k(s) = F_k(s) - F_{k+1}(s)$. Этот результат был опубликован в буклете, озаглавленном *The Expectation of Parts* (Cambridge, 1898), и включен в пятое издание *Choice and Chance* (1901). Упрощенные выражения для среднего, дисперсии и некоторых более общих статистик разностей были найдены Бартоном и Дэвидом (см. работу Barton and David, *J. Royal Stat. Soc.* **B18** (1956), 79–94). Обзор методов, которыми статистики традиционно анализируют разности как ключи к возможным смещениям данных, приведен в работе Р. Пайка (R. Pyke, *J. Royal Stat. Soc.* **B27** (1965), 395–449).]

27. Рассмотрим многогранник в гиперплоскости $S_1 + \cdots + S_n = 1$, определенный неравенствами $S_1 \ge 0, \ldots, S_n \ge 0$. Он состоит из $n!$ конгруэнтных подмногогранников, определенных упорядочением S_j (предполагается, что S_j различны), и операцией сортировки является от $n!$ к одному складывание больших многогранников из подмногогранников, в которых $S_1 \le \cdots \le S_n$. Преобразование, которое переводит $(S_{(1)}, \ldots, S_{(n)})$ в $(S_1', \ldots, S_n')$, является взаимно однозначным отображением, разлагающим дифференциальные объемы на $n!$ частей. Оно образует вершины $(\frac{1}{n}, \ldots, \frac{1}{n})$, $(0, \frac{1}{n-1}, \ldots, \frac{1}{n-1})$, $\ldots$, $(0, \ldots, 0, 1)$ подмногогранников в соответствующих вершинах $(1, 0, \ldots, 0)$, $(0, 1, 0, \ldots, 0)$, $\ldots$, $(0, \ldots, 0, 1)$; линейное растяжение и искажение полностью приспособлены к процессу. (Евклидово расстояние между вершинами $(0, \ldots, 0, \frac{1}{j}, \ldots, \frac{1}{j})$ и $(0, \ldots, 0, \frac{1}{k}, \ldots, \frac{1}{k})$ в подмногограннике равно $|j^{-1} - k^{-1}|^{1/2}$; преобразование образует регулярный симплекс, в котором все n вершин находятся на расстоянии $\sqrt{2}$.)

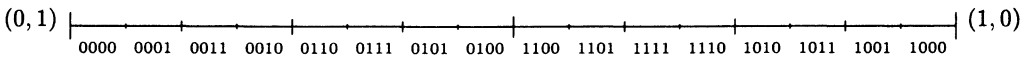
Поведение итерированных разностей легче понять, если проверить детали графически при $n = 3$. В этом случае многогранник является простым равносторонним треугольником, точки которого выражены в барицентрических координатах (x, y, z), $x + y + z = 1$. На приведенном рисунке иллюстрируются два первых уровня рекурсивного разбиения этого треугольника. Каждый из 6^2 подтреугольников помечен двузначным кодом pq, где p соответствует подходящей перестановке, когда $(x, y, z) = (S_1, S_2, S_3)$ рассортированы в виде $(S_{(1)}, S_{(2)}, S_{(3)})$, а q соответствует перестановке следующей стадии, когда S_1',



S_2' и S_3' рассортированы в соответствии со следующим кодом:

$$0\colon x<y<z,\quad 1\colon x<z<y,\quad 2\colon y<x<z,\quad 3\colon y<z<x,\quad 4\colon z<x<y,\quad 5\colon z<y<x.$$

Например, для точек подтреугольника 34 выполняются неравенства $S_2 < S_3 < S_1$ и $S_3' < S_1' < S_2'$. Этот процесс можно продолжить до получения бесконечного числа уровней; все точки треугольника с иррациональными барицентрическими координатами вследствие этого получат единственное представление в системе счисления с основанием 6. Четырехгранник аналогично может быть разделен на 24, 24^2, 24^3, ... подчетырехгранников; в общем случае эта процедура образует представление с основанием $n!$ для любого $(n-1)$-мерного симплекса. Когда $n = 2$, процесс особенно прост: если $x \notin \{0, \frac{1}{2}, 1\}$, при преобразовании образуется разность $(x, 1-x) = (x, y)$ в $(2x \bmod 1, 2y \bmod 1)$ либо $(2y \bmod 1, 2x \bmod 1)$ в зависимости от того, будет ли $x < y$ или $x > y$. Повторные испытания — это, по существу, сдвиг бинарного представления влево на один разряд, который, возможно, дополняет результат. После не более чем $e+1$ итераций на e-разрядные числа процесс должен сойтись к фиксированной точке $(0,1)$. Перестановочное кодирование для $n = 2$ соответствует просто свертке и растягиванию линии; первые четыре уровня разбиений имеют следующие четырехразрядные коды.



Эта последовательность точно совпадает с двоичным кодом Грея, рассматриваемым в разделе 7.2.1. В общем случае в перестановочном коде в системе счисления с основанием $n!$ для n-симплекса смежные области имеют идентичные коды, за исключением одной цифры. После каждой итерации разностного преобразования в представлении каждой точки удаляется крайняя слева цифра. Заметим, что равные разности дней рождения — это точки около границы разложения на первом уровне.

Фундаментальное преобразование из $(S_1, \ldots, S_n)$ в $(S_1', \ldots, S_n')$ подразумевается в доказательстве Витворта предложения LVI в пятом издании *Choice and Chance* (см. ссылку на литературу в ответе к упр. 26). Сначала оно было подробно изучено Дж. Дурбиным (J. Durbin, *Biometrika* **48** (1961), 41–55), которого вдохновил подобными построениями П. В. Сухатме (P. V. Sukhatme, *Annals of Eugenics* **8** (1937), 52–56.) Перестановочное кодирование для повторных разностей было введено Г. Э. Даниелем (H. E. Daniels, *Biometrika* **49** (1962), 139–149).

28. (a) Число разбиений m на n различных положительных частей равно $p_n\big(m - \binom{n+1}{2}\big)$, что следует из упр. 5.1.1–16. Эти разбиения могут быть переставлены $n!$ способами, чтобы образовать n-мерные строки $(y_1, \ldots, y_n)$ с $0 = y_1 < y_2 < \cdots < y_n < m$, и каждая из этих n-мерных строк приводит к появлению $(n-1)!$ n-мерных строк, таких, что $y_1 = 0$ и $0 < y_2, \ldots, y_n < m$. Добавим константу по модулю m к каждому y_j, что позволит сохранить разность. Отсюда $b_{n00}(m) = mn!\,(n-1)!\,p_n\big(m - \binom{n+1}{2}\big)$.

(b) Нулевые разности соответствуют шарам в одной и той же урне, и $s-1$ — их вклад в число равных разностей. Поэтому $b_{nrs}(m) = \left\{ {n \atop n-s} \right\} b_{(n-s)(r+1-s)0}(m)$.

(c) Так как $\left\{ {n \atop n-1} \right\} = \binom{n}{2}$, вероятности равны

$$n!\,(n-1)!\,m^{1-n}\left(p_n\Big(m - \binom{n+1}{2}\Big) - \frac{1}{2}p_{n-1}\Big(m - \binom{n}{2}\Big)\right).$$

29. Из предыдущего ответа и упр. 5.1.1–15 получаем $b_{n0}(z) = n!\,(n-1)!\,z^{\binom{n+1}{2}}/((1-z) \times \ldots \times (1-z^n))$. Когда $r = 1$, $n!$ в наших предыдущих рассуждениях преобразуется в $n!/2$ и число решений до $0 < s_1 < \cdots < s_k \le s_{k+1} < \cdots < s_n$ с $s_1 + \cdots + s_n = m$ равно числу решений до $0 \le s_1 - 1 \le \cdots \le s_k - k \le s_{k+1} - k \le \cdots \le s_n - n + 1$ с $(s_1 - 1) +$

$\cdots + (s_k - k) + (s_{k+1} - k) + \cdots + (s_n - n + 1) = m - \binom{n}{2} - k$. Следовательно, $b_{n1}(z) = \frac{1}{2} n!\,(n-1)! \sum_{k=1}^{n} (z^k - z^n)\, z^{\binom{n}{2}} / (1-z)\ldots(1-z^n)$. Аналогично можно показать, что

$$\frac{b_{n2}(z)}{n!\,(n-1)!} = \left(\frac{1}{2!\,2!} \sum_{1 \le j < k < n} (z^j - z^n)(z^k - z^{n-1}) + \frac{1}{3!} \sum_{1 \le k < n} (z^k - z^n)(z^k - z^{n-1}) \right) \times \frac{z^{\binom{n-1}{2}}}{(1-z)\ldots(1-z^n)}.$$

Можно получить $b_{nr}(z)$ для произвольных r из формулы

$$\frac{\sum_r b_{nr}(z) w^r}{n!\,(n-1)!\,z^n} = \sum_{0 \le b_1, \ldots, b_{n-1} \le 1} \frac{(z - b_1 z^n)\ldots(z^{n-1} - b_{n-1} z^n)}{c_1 \ldots c_{n-1} (1-z)\ldots(1-z^n)} \left(\frac{w}{z^{n-1}}\right)^{b_1} \cdots \left(\frac{w}{z^1}\right)^{b_{n-1}},$$

где $c_k = 1 + b_k + b_k b_{k-1} + \cdots + b_k \ldots b_2 b_1 = 1 + b_k c_{k-1}$. (Частный случай для $w = 1$ интересен, поскольку левая часть суммируется до $(1-z)^{-n}/n!$.)

30. Это хорошая задача для метода перевала [см. работу Н. Г. де Брейна (N. G. de Bruijn, *Asymptotic Methods in Analysis* (North-Holland, 1961), Chapter 5)]. Справедливо равенство $p_n(m) = \frac{1}{2\pi i} \oint e^{f(z)} \frac{dz}{z}$, где $f(z) = -m \ln z - \sum_{k=1}^{n} \ln(1 - z^k)$. Пусть $\rho = n/m$ и $\delta = \sqrt{n}/m$; интегрирование по пути $z = e^{-\rho + it\delta}$ дает $p_n(m) = \frac{\delta}{2\pi} \int_{-\pi/\delta}^{\pi/\delta} \exp(f(e^{-\rho+it\delta}))\, dt$. Удобно использовать равенство

$$g(se^t) = \sum_{j=0}^{n} \frac{t^j}{j!} \vartheta^j g(s) + \int_0^t \frac{u^n}{n!} \vartheta^{n+1} g(se^{t-u})\, du,$$

где $g = g(z)$ — любая аналитическая функция и ϑ — оператор $z\frac{d}{dz}$. Когда функция $\vartheta^j g$ вычислена в точке e^z, результат будет таким же, как и тогда, когда $g(e^z)$ дифференцируется j раз по z. Этот принцип приводит к получению формулы

$$\vartheta^j f(e^{-\rho}) = -m[j=1] + \frac{j!\,n}{\rho^j} + (-1)^j \sum_{k=1}^{n} \sum_{l \ge j} \frac{l^{\underline{j}} B_l}{l \cdot l!} k^l \rho^{l-j}$$

из другого удобного равенства,

$$\ln\left(\frac{1 - e^{-z}}{z}\right) = \sum_{n \ge 1} \frac{B_n z^n}{n \cdot n!}.$$

Таким образом, получена асимптотическая формула для подынтегрального выражения

$$\exp f(e^{-\rho+it\delta}) = \exp\left(\sum_{j \ge 0} \frac{i^j \delta^j t^j}{j!} \vartheta^j f(e^{-\rho}) \right) = e^{-t^2/2 + f(e^{-\rho})} \exp(ic_1 t - c_2 t^2 - ic_3 t^3 + \cdots),$$

где $c_1 = (\frac{n(n+1)}{2} B_1 + \frac{n(n+1/2)(n+1)}{6} B_2 \rho)\delta + O(n^{-3})$ и т. д., и оказывается, что $c_j = O(n^{-3})$ для $j \ge 8$. Вынеся за скобки константу, получим

$$\frac{\delta}{2\pi} e^{f(e^{-\rho})} = \frac{\delta}{2\pi\, n!\, \rho^n e^{-m\rho}} \exp\left(- \sum_{k=1}^{n} \sum_{l \ge 1} \frac{B_l}{l \cdot l!} k^l \rho^l \right)$$
$$= \frac{\sqrt{n}\, m^{n-1} e^{n+\alpha/4}}{2\pi\, n!\, n^n} \left(1 + \frac{18\alpha - \alpha^2}{72n} + \frac{108\alpha^2 - 36\alpha^3 + \alpha^4}{10368 n^2} + O(n^{-3}) \right),$$

что приводит к интегралу, подынтегральное выражение которого является экспоненциально малым, когда $|t| \ge n^{\epsilon}$. Можно пренебречь большими значениями t, поскольку разложение на элементарные дроби показывает, что подынтегральное выражение имеет

порядок $O((m/n)^{n/2})$. Ни один из других корней единицы не появляется более $n/2$ раз как полюс знаменателя. Следовательно, было допущено существование "тяжелых хвостов" [см. *CMath*, §9.4] и выполнено интегрирование по всем t. Формулы $\int_{-\infty}^{\infty} e^{-t^2/2} t^j\,dt = (j-1)(j-3)\ldots(1)\sqrt{2\pi}$ [j — четное] и $n! = (n/e)^n\sqrt{2\pi n}\exp(\frac{1}{12}n^{-1} + O(n^{-3}))$ достаточны для завершения вычислений.

С $q_n(m) = p_n(m - \binom{n+1}{2})$ вместо $p_n(m)$ вычисления производятся таким же образом, но с c_1, возрастающим, как $\frac{1}{2}\alpha(n^{1/2} - n^{-1/2})$, и с дополнительным множителем $\exp(-\rho\binom{n+1}{2})$. Получаем

$$q_n(m) = \frac{m^{n-1}e^{-\alpha/4}}{n!\,(n-1)!}\left(1 - \frac{13\alpha^2}{288n} + \frac{169\alpha^4 - 2016\alpha^3 - 1728\alpha^2 + 41472\alpha}{165888n^2} + O(n^{-3})\right),$$

что совпадает с формулой для $p_n(m)$, но α заменено на $-\alpha$. Действительно, если определить $p_n(m) = r_n(2m + \binom{n+1}{2})$ и $q_n(m) = r_n(2m - \binom{n+1}{2})$, производящая функция $R_n(z) = \sum_m r_n(z^m) = \prod_{k=1}^{n}(z^{-k} - z^k)^{-1}$ удовлетворит равенству $R_n(1/z) = (-1)^n R_n(z)$. Отсюда следует формула двойственности $r_n(-m) = (-1)^{n-1} r_n(m)$ в том смысле, что уравнение превратится в равенство, когда $r_n(m)$ выражается через полином по m и корни единицы. Поэтому можно сказать, что $q_n(m) = p_n(-m)$. В общем случае интерпретацию такой двойственности можно найти в работе Дж. Пойа (см. G. Pólya, *Math. Zeitschrift* **29** (1928), 549–640, §44). (Дополнительно см. работу Дж. Шекерса (G. Szekeres, *Quarterly J. Math. Oxford* **2** (1951), 85–108; **4** (1953), 96–111.)

Точное значение $q_n(m)$, когда $m = 2^{25}$ и $n = 512$, равно $7.08069\,34695\,90264\,094\ldots \times 10^{1514}$; наша аппроксимация дает оценку $7.080693501 \times 10^{1514}$.

Вероятность того, что критерий дней рождения обнаружит $R = 0$ разностей, равна $b_{n00}(m)/m^n = n!\,(n-1)!\,m^{1-n}q_n(m) = e^{-\alpha/4} + O(n^{-1})$ согласно упр. 28, поскольку вклад от $b_{n01}(m)$ равен $\approx \frac{\alpha}{2n}e^{-\alpha/4} = O(n^{-1})$. Включение множителя $g_n(z) = \sum_{k=1}^{n-1}(z^{-k} - 1)$ в подынтегральное выражение $q_n(m)$ дает тот же эффект, что и умножение результата на $\frac{\alpha}{2} + O(n^{-1})$, поскольку $g_n(e^{-\rho+it\delta}) = \binom{n}{2}\rho + O(n^3\rho^2) + itO(n^2\delta) - \frac{1}{2}t^2O(n^3\delta^2) + \cdots$. Подобным же образом экстрамножитель $\sum_{1\le j<k<n}(z^{-j}-1)(z^{-k}-1)$, по существу, умножает на $\frac{1}{8}n^4\rho^2 = \frac{1}{8}\alpha^2$ и добавляет $O(n^{-1})$. Другие вклады в вероятность того, что $R = 2$, имеют порядок $O(n^{-1})$. Таким же образом найдем, что вероятность получения r равных разностей равна $e^{-\alpha/4}(\alpha/4)^r/r! + O(n^{-1})$, т. е. число равных разностей приближенно имеет распределение Пуассона. Более сложные члены возникают, если осуществить разложение до порядка $O(n^{-2})$.

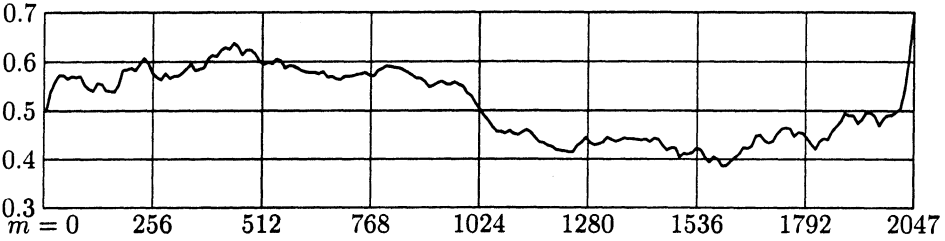
31. 79 двоичных разрядов содержат 24 множества троек $\{Y_n, Y_{n+31}, Y_{n+55}\}$, $\{Y_{n+1}, Y_{n+32}, Y_{n+56}\}$, $\ldots$, $\{Y_{n+23}, Y_{n+54}, Y_{n+78}\}$, плюс 7 дополнительных разрядов $Y_{n+24}, \ldots, Y_{n+30}$. Последний двоичный разряд с равной вероятностью является 0 или 1, но в каждой группе троек с вероятностью $\frac{1}{4}$ двоичные разряды имеют вид $\{0,0,0\}$ и с вероятностью $\frac{3}{4}$ они будут иметь вид $\{0,1,1\}$. Поэтому производящая функция для суммы двоичных разрядов равна $f(z) = (\frac{1+z}{2})^7(\frac{1+3z^2}{4})^{24}$ (это полином 55-й степени). (Данная формула не точна; строго говоря, она имеет вид $(2^{55}f(z) - 1)/(2^{55} - 1)$, поскольку все 0-случаи исключены.) Коэффициенты $2^{55}f(z)$ легко подсчитать на вычислительной машине, и, следовательно, можно найти, что вероятность того, что единиц больше, чем нулей, равна $18509401282464000/(2^{55} - 1) \approx 0.51374$.

Замечание. Это упражнение основано на открытии Ваттулайнена, Ала-Ниссила и Канкаала (см. работу Vattulainen, Ala-Nissila, and Kankaala, *Physical Review Letters* **73** (1994), 2513–2516), состоящего в том, что генератор Фибоначчи с запаздыванием не удовлетворяет более сложному критерию двумерного случайного блуждания. Заметим, что и последовательность $Y_{2n}, Y_{2n+2}, \ldots$ не удовлетворяет этому критерию, поскольку она описывается тем же рекуррентным соотношением. Смещение в сторону единиц распростра-

няется на подпоследовательности, которые состоят из четных элементов, генерируемых последовательностью $X_n = (X_{n-55} \pm X_{n-24}) \bmod 2^e$; ей присуща тенденция иметь больше случаев $(\ldots 10)_2$, чем $(\ldots 00)_2$ в двоичной записи.

Нет ничего магического в числе 79 в этом критерии. Эксперименты показывают, что значимые отклонения в сторону единиц присущи также случайным блужданиям длиной 101, 1001 или 10001. Но формальное доказательство этого, вероятно, будет трудным. После 86 шагов производящая функция равна $(\frac{1+3z^2}{4})^{17}(\frac{1+2z^2+4z^3+z^4}{8})^7$, затем получим множители $(1+2z^2+5z^3+5z^4+10z^5+8z^6+z^7)/32$, $(1+2z^2+7z^3+7z^4+15z^5+25z^6+29z^7+28z^8+13z^9+z^{10})/128$ и т. д. Анализ становится все более и более сложным с удлинением блужданий.

Интуитивно ясно, что преобладание единиц на первых 79 шагах будет продолжаться столько, сколько числовые подпоследовательности умеренно колеблются между 0 и 1. Сопутствующая диаграмма показывает результаты более простого случая, а именно — использования генератора $Y_n = (Y_{n-2} + Y_{n-11}) \bmod 2$, который можно легко проанализировать. Для него случайные блуждания длиной 445 имеют 64% шансов закончиться справа от начальной точки. Это смещение пропадет только тогда, когда длина блужданий возрастет до половины периода (после чего, конечно, нули станут более вероятны, хотя в полном периоде будет недоставать одного нуля).



Вероятности, что число 1 превосходит число 0 в случайных наборах из m чисел, когда $Y_n = Y_{n-2} \oplus Y_{n-11}$.

Техника отбрасывания Люшера (Lüscher) может использоваться для того, чтобы избежать смещения в сторону единиц (см. окончание раздела 3.2.2). Например, со смещениями 55 и 24 отклонений от случайности в наблюдениях за случайными блужданиями длиной 1001 нет, если генерировать числа группами по 165 и использовать только первые 55 чисел.

32. Нет, если они принимают значения $(-1-2\epsilon^2,\ \epsilon-2\epsilon^2,\ 1-2\epsilon)$ с соответствующими вероятностями $(\frac{1}{2}-\epsilon,\ \epsilon,\ \frac{1}{2})$. Тогда $X+Y>0$ с вероятностью $(\frac{1}{2}+\epsilon)^2 < \frac{1}{2}$, если ϵ достаточно мало́. [Таким образом, два игрока в гольф могут быть в среднем одинаково сильными, но один из них может с большей вероятностью выиграть один раунд, тогда как другой с большей вероятностью выиграет два раунда. См. работу T. M. Кавера (T. M. Cover, *Amer. Statistician* **43** (1989), 277–278), в которой обсуждаются подобные феномены.]

33. По существу, нужно рассмотреть $[z^{(k+l-1)/2}](\frac{1+z}{2})^{k-2l}(\frac{1+3z^2}{4})^l/(1-z)$. Пусть $m = k-2l$, $n=l$ и требуемые коэффициенты равны $\frac{1}{2\pi i}\oint e^{g(z)}\frac{dz}{z(1-z)}$, где $g(z) = m\ln(\frac{1+z}{2}) + n\ln(\frac{1+3z^2}{4}) - (\frac{m+3n-1}{2})\ln z$. Удобно (и разумно) интегрировать вдоль пути $z = e^{\epsilon u}$, где $\epsilon^2 = 4/(m+3n)$ и $u = -1+it$ для $-\infty < t < \infty$. Получим $g(e^{\epsilon u}) = -\epsilon u/2 + u^2/2 + c_3\epsilon u^3 + c_4\epsilon^2 u^4 + \cdots$, где $c_k = \epsilon^2\vartheta^k g(1)/k! = O(1)$, а также $1/(1-e^{\epsilon u}) = \frac{-1}{\epsilon u} + \frac{1}{2} - B_2\epsilon u/2! - \cdots$. Вынеся множители из подынтегрального выражения и используя тот факт, что $\frac{1}{2\pi i}\int_{1-i\infty}^{1+i\infty} e^{u^2/2}\frac{du}{u} = \frac{1}{2}$ и $\frac{1}{2\pi i}\int_{a-i\infty}^{a+i\infty} e^{u^2/2}u^{2k}\,du = (-1)^k(2k-1)(2k-3)\ldots(1)\sqrt{2\pi}$, получаем асимптотическую формулу $\frac{1}{2} + (2\pi)^{-1/2}n(m+3n)^{-3/2} + O((m+3n)^{-3/2})$. Если $m+3n$ четно, то справедлива та же асимптотическая формула, поскольку мы даем одну половину

коэффициента при $z^{(m+3n)/2}$ единицам, а вторую — нулям. (Этот коэффициент равен $(\frac{2}{\pi(m+3n)})^{1/2} + O((m-3n)^{-3/2})$.)

34. Число строк длиной n, не содержащих заданные двухбуквенные подстроки или пары подстрок, — это коэффициенты при z^n в соответствующей производящей функции. Они могут быть записаны в виде $ce^{n\tau}m^n + O(1)$, где c и τ можно разложить по степеням $\epsilon = 1/m$.

Случай	Исключение	Производящая функция	c	τ
1	aa	$(1+z)/p(z)$	$1+\epsilon^2-2\epsilon^3+\cdots$	$-\epsilon^2+\epsilon^3-\frac{5}{2}\epsilon^4+\cdots$
2	ab	$1/(1-mz+z^2)$	$1+\epsilon^2+3\epsilon^4+\cdots$	$-\epsilon^2-\frac{3}{2}\epsilon^4+\cdots$
3	aa, bb	$(1+z)/(p(z)+z^2)$	$1+2\epsilon^2-4\epsilon^3+\cdots$	$-2\epsilon^2+2\epsilon^3-8\epsilon^4+\cdots$
4	aa, bc	$(1+z)/(p(z)+z^2+z^3)$	$1+2\epsilon^2-2\epsilon^3+\cdots$	$-2\epsilon^2+\epsilon^3-7\epsilon^4+\cdots$
5	ab, bc	$(1+z)/(1-mz+2z^2-z^3)$	$1+2\epsilon^2-2\epsilon^3+\cdots$	$-2\epsilon^2+\epsilon^3-6\epsilon^4+\cdots$
6	ab, cd	$1/(1-mz+2z^2)$	$1+2\epsilon^2+12\epsilon^4+\cdots$	$-2\epsilon^2-6\epsilon^4+\cdots$

(Здесь a, b, c, d обозначают буквы и $p(z) = 1-(m-1)(z+z^2)$. Оказывается, что эффект исключения $\{ab, ba\}$ или $\{aa, ab\}$ эквивалентен исключению $\{aa, bb\}$; исключение $\{ab, ac\}$ эквивалентно исключению $\{ab, cd\}$.) Пусть $S_n^{(j)}$ — коэффициент при z^n в случае j и пусть X — общее число двухбуквенных комбинаций, которые не появляются. Тогда $\mathrm{E}\,X = (mS_n^{(1)} + m^2S_n^{(2)})/m^n$ и

$$\mathrm{E}\,X^2 = (mS_n^{(1)} + m^2(S_n^{(2)} + 6S_n^{(3)}) + 2m^3(S_n^{(4)} + S_n^{(5)} + S_n^{(6)}) + m^4S_n^{(6)})/m^n.$$

35. (a) $\mathrm{E}\,S_m = N^{-1}\sum_{n=0}^{N-1}\sum_{j=0}^{m-1} Z_{n+j} = N^{-1}\sum_{j=0}^{m-1}\sum_{n=0}^{N-1} Z_{n+j} = N^{-1}\sum_{j=0}^{m-1} 1 = m/N$, так как $\sum_{n=0}^{N-1} Z_{n+j} = 2^{k-1} - (2^{k-1}-1) = 1$.

(b) Пусть $\xi^k = a_1\xi^{k-1} + \cdots + a_k$. Определим линейную функцию f, как и в первом ответе к упр. 3.2.2–16. Тогда $Y_n = f(\xi^n)$, а отсюда следует, что $Y_{n+i} + Y_{n+j} = f(\xi^{n+i}) + f(\xi^{n+j}) \equiv f(\xi^{n+i} + \xi^{n+j}) = f(\xi^n\alpha)$ (по модулю 2), где α не равно нулю, когда $i \not\equiv j$ (по модулю N). Отсюда $\mathrm{E}\,S_m^2 = N^{-1}\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\sum_{n=0}^{N-1} Z_{n+i}Z_{n+j} = N^{-1}(\sum_{i=0}^{m-1}\sum_{n=0}^{N-1} Z_{n+i}^2 - 2\sum_{0\le i<j<m}\sum_{n=0}^{N-1} Z_n) = m - m(m-1)/N$.

(c) $\mathrm{E}\sum_{j=0}^{m-1} Z_{n+j} = \sum_{j=0}^{m-1}\mathrm{E}\,Z_{n+j} = 0$ и $\mathrm{E}(\sum_{j=0}^{m-1} Z_{n+j})^2 = \sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\mathrm{E}\,Z_{n+i}Z_{n+j} = \sum_{j=0}^{m-1}\mathrm{E}\,Z_{n+j}^2 + \sum_{0\le i<j<m}(\mathrm{E}\,Z_{n+i})(\mathrm{E}\,Z_{n+j}) = m$, когда каждое Z_n действительно случайно. Таким образом, среднее и дисперсия S_m очень близки к истинным значениям, когда $m \ll N$.

(d) $\mathrm{E}\,S_m^3 = N^{-1}\sum_{h=0}^{m-1}\sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\sum_{n=0}^{N-1} Z_{n+h}Z_{n+i}Z_{n+j}$. Если любые h, i и j равны, сумма по n равна 1; следовательно,

$$\mathrm{E}\,S_m^3 = \frac{1}{N}\Bigg(m^3 - m^3 + 6\sum_{0\le h<i<j<m}\sum_{n=0}^{N-1} Z_{n+h}Z_{n+i}Z_{n+j}\Bigg).$$

Так же, как и в (b), покажем, что сумма по n равна 1, если $\xi^h + \xi^i + \xi^j \neq 0$; иначе она равна $-N$. Таким образом, $\mathrm{E}\,S_m^3 = m^3 - 6B(N+1)/N$, где $B = \sum_{0\le h<i<j<m}[\xi^h + \xi^i + \xi^j = 0] = \sum_{0<i<j<m}[1 + \xi^i + \xi^j = 0]\,(m-j)$. Наконец заметим, что $1 + \xi^i = \xi^j$ тогда и только тогда, когда $f(\xi^{i+l}) = f(\xi^{j+l})$ для $0 < l < k$. Предполагаем, что $0 < i < j < N$.

(e) Для $i = 31$ и $j = 55$ появляются только ненулевые члены; значит, $B = 79 - 55 = 24$. (Следующий ненулевой член появится, когда $i = 62$ и $j = 110$.) В настоящей случайной ситуации $\mathrm{E}\,S_m^3$ должно быть равно 0, так что значение $\mathrm{E}\,S_{79}^3 \approx -144$ определенно говорит о неслучайности. Любопытно, что это значение отрицательное, хотя в упр. 31 показано, что S_{79} обычно *положительное*. Значение S_{79} стремится быть основательно отрицательным, когда оно оказывается ниже 0.

Литература. *IEEE Trans.* **IT-14** (1968), 569–576. M. Matsumoto and Y. Kurita, *ACM Trans. Modeling and Comp. Simul.* **2** (1992), 179–194; **4** (1994), 254–266. М. Мацумото и Й. Курита утверждают, что генераторы с троичным основанием не удовлетворяют таким критериям, проверяющим распределения, даже когда смещение достаточно большое. См. также работу *ACM Trans. Modeling and Comp. Simul.* **6** (1996), 99–106, в которой они демонстрируют длинную подпоследовательность низкой плотности.

РАЗДЕЛ 3.3.3

1. $y((x/y)) + \frac{1}{2}y - \frac{1}{2}y\delta(x/y)$.

2. См. упр. 1.2.4–38 и 1.2.4–39(a, b, g).

3. $((x)) = -\sum_{n\geq 1}\frac{1}{n\pi}\sin 2\pi nx$, ряд сходится для всех x. (Представление в (24) можно рассматривать как "конечный" ряд Фурье в случае, когда x рационально.)

4. $d_{\max} = 2^{10}\cdot 5$. Заметим, что неравенство $X_{n+1} < X_n$ имеет вероятность $\frac{1}{2}+\epsilon$, где

$$|\epsilon| < d/(2\cdot 10^{10}) \leq 1/(2\cdot 5^9).$$

Следовательно, *каждый* генератор с потенциалом 10 приемлем с точки зрения теоремы P.

5. Промежуточный результат:

$$\sum_{0\leq x<m}\frac{x}{m}\frac{s(x)}{m} = \frac{1}{12}\sigma(a,m,c) + \frac{m}{4} - \frac{c}{2m} - \frac{x'}{2m}.$$

6. (a) Используйте индукцию и формулу

$$\left(\!\!\left(\frac{hj+c}{k}\right)\!\!\right) - \left(\!\!\left(\frac{hj+c-1}{k}\right)\!\!\right) = \frac{1}{k} - \frac{1}{2}\delta\left(\frac{hj+c}{k}\right) - \frac{1}{2}\delta\left(\frac{hj+c-1}{k}\right).$$

(b) Используйте тот факт, что $-\left(\!\!\left(\frac{h'j}{k}\right)\!\!\right) = -\left(\!\!\left(\frac{j}{hk} - \frac{k'j}{h}\right)\!\!\right) = \left(\!\!\left(\frac{k'j}{h}\right)\!\!\right) - \frac{j}{hk} + \frac{1}{2}\delta\left(\frac{k'j}{h}\right)$.

7. Положите $m = h$, $n = k$, $k = 2$ во второй формуле упр. 1.2.4–45:

$$\sum_{0<j<k}\left(\frac{hj}{k} - \left(\!\!\left(\frac{hj}{k}\right)\!\!\right) + \frac{1}{2}\right)\left(\frac{hj}{k} - \left(\!\!\left(\frac{hj}{k}\right)\!\!\right) - \frac{1}{2}\right) + 2\sum_{0<j<h}\left(\frac{kj}{h} - \left(\!\!\left(\frac{kj}{h}\right)\!\!\right) + \frac{1}{2}\right) j = kh(h-1).$$

Суммы в левой части упрощаются, а после стандартных преобразований получим

$$h^2k - hk - \frac{h}{2} + \frac{h^2}{6k} + \frac{k}{12} + \frac{1}{4} - \frac{h}{6}\sigma(h,k,0) - \frac{h}{6}\sigma(k,h,0) + \frac{1}{12}\sigma(1,k,0) = h^2k - hk.$$

Так как $\sigma(1,k,0) = (k-1)(k-2)/k$, это приводит к закону взаимности.

8. См. работу *Duke Math. J.* **21** (1954), 391–397.

9. Начните с интересного тождества

$$\sum_{k=0}^{r-1}\lfloor kp/r\rfloor\lfloor kq/r\rfloor + \sum_{k=0}^{p-1}\lfloor kq/p\rfloor\lfloor kr/p\rfloor + \sum_{k=0}^{q-1}\lfloor kr/q\rfloor\lfloor kp/q\rfloor = (p-1)(q-1)(r-1),$$

для доказательства которого можно применить простой геометрический метод, предполагая, что $p \perp q$, $q \perp r$ и $r \perp p$. См. работу У. Дитера (U. Dieter, *Abh. math. Sem. Univ. Hamburg* **21** (1957), 109–125).

10. Очевидно, что из (8) следует равенство $\sigma(k-h,k,c) = -\sigma(h,k,-c)$. Заменим j на $k-j$ в определении (16), чтобы доказать равенство $\sigma(h,k,c) = \sigma(h,k,-c)$.

11. (a) $\sum_{0\le j<dk}\big(\!\big(\frac{j}{dk}\big)\!\big)\big(\!\big(\frac{hj+c}{k}\big)\!\big) = \sum_{\substack{0\le i<d\\ 0\le j<k}}\big(\!\big(\frac{ik+j}{dk}\big)\!\big)\big(\!\big(\frac{hj+c}{k}\big)\!\big)$; используем (10) для суммирования по i.

(b) $\big(\!\big(\frac{hj+c+\theta}{k}\big)\!\big) = \big(\!\big(\frac{hj+c}{k}\big)\!\big) + \frac{\theta}{k} - \frac{1}{2}\delta\big(\frac{hj+c}{k}\big)$; сейчас суммируем.

12. Так как $(\!(\frac{hj+c}{k})\!)$ принимает те же значения, что и $(\!(\frac{j}{k})\!)$ в другом порядке, неравенство Коши влечет неравенство $\sigma(h,k,c)^2 \le \sigma(h,k,0)^2$ и $\sigma(1,k,0)$ может быть легко просуммирована непосредственно (см. упр. 7).

13. $\sigma(h,k,c) + \dfrac{3(k-1)}{k} = \dfrac{12}{k}\displaystyle\sum_{0<j<k}\frac{\omega^{-cj}}{(\omega^{-hj}-1)(\omega^{j}-1)} + \frac{6}{k}(c \bmod k) - 6\big(\!\big(\frac{h'c}{k}\big)\!\big)$,

если $hh' \equiv 1$ (по модулю k).

14. $(2^{38} - 3\cdot 2^{20} + 5)/(2^{70}-1) \approx 2^{-32}$. Весьма удовлетворительное глобальное значение вопреки локальной неслучайности!

15. Замените c^2 в (19) на $\lfloor c\rfloor\lceil c\rceil$.

16. По индукции можно доказать, что тождество, приведенное в указании, эквивалентно $m_1 = p_r m_{r+1} + p_{r-1}m_{r+2}$ для $1 \le r \le t$. (См. также упр. 4.5.3–32.) Теперь заменим c_j на $\sum_{j\le r\le t} b_r m_{r+1}$ и сравним коэффициенты при $b_i b_j$ в обеих частях тождества, которое необходимо доказать.

Замечание. Для всех показателей $e \ge 1$ аналогичные соображения дают

$$\sum_{1\le j\le t}(-1)^{j+1}\frac{c_j^e}{m_j m_{j+1}} = \frac{1}{m_1}\sum_{1\le j\le t}(-1)^{j+1}b_j\frac{(c_j^e - c_{j+1}^e)}{c_j - c_{j+1}}p_{j-1}.$$

17. В этом алгоритме выполняются равенства $k = m_j$, $h = m_{j+1}$, $c = c_j$, $p = p_{j-1}$, $p' = p_{j-2}$, $s = (-1)^{j+1}$ для $j = 1, 2, \ldots, t+1$.

D1. [Инициализация.] Присвоить $A \leftarrow 0$, $B \leftarrow h$, $p \leftarrow 1$, $p' \leftarrow 0$, $s \leftarrow 1$.

D2. [Деление.] Присвоить $a \leftarrow \lfloor k/h\rfloor$, $b \leftarrow \lfloor c/h\rfloor$, $r \leftarrow c \bmod h$. (Теперь $a = a_j$, $b = b_j$ и $r = c_{j+1}$.)

D3. [Накапливание.] Присвоить $A \leftarrow A + (a-6b)s$, $B \leftarrow B + 6bp(c+r)s$. Если $r \ne 0$ или $c = 0$, присвоить $A \leftarrow A - 3s$. Если $h = 1$, присвоить $B \leftarrow B + ps$. (Здесь вычитаем $3e(m_{j+1}, c_j)$ и также принимаем во внимание члены $\sum(-1)^{j+1}/m_j m_{j+1}$.)

D4. [Подготовка к следующей итерации.] Присвоить $c \leftarrow r$, $s \leftarrow -s$; присвоить $r \leftarrow k \mathbin{\dot{-}} ah$, $k \leftarrow h$, $h \leftarrow r$; присвоить $r \leftarrow ap + p'$, $p' \leftarrow p$, $p \leftarrow r$. Если $h > 0$, возвратиться к шагу D2. ▮

По окончании работы этого алгоритма p будет равно истинному значению k_0 величины k, так что требуемый ответ — $A + B/p$. Окончательное значение p' будет равно h', если $s < 0$, иначе p' будет равно $k_0 - h'$. Хорошо было бы, чтобы B благодаря подходящей корректировке A не выходило из области $0 \le B < k_0$. Поэтому используются только операции с обычной точностью (с двойной точностью выполняются умножение и деление), если k_0 — число, заданное с обычной точностью.

18. Можно моментально увидеть, что формула

$$S(h,k,c,z) = \textstyle\sum_{0\le j<k}(\lfloor j/k\rfloor - \lfloor (j-z)/k\rfloor)\,(((hj+c)/k))$$

определена для всех $z \ge 0$ не только тогда, когда $k \ge z$. Записав $\lfloor j/k\rfloor - \lfloor (j-z)/k\rfloor = \frac{z}{k} + (\!(\frac{j-z}{k})\!) - (\!(\frac{j}{k})\!) + \frac{1}{2}\delta_{j0} - \frac{1}{2}\delta(\frac{j-z}{k})$ и выполнив суммирование, получим

$$S(h,k,c,z) = \frac{zd}{k}\big(\!\big(\frac{c}{d}\big)\!\big) + \frac{1}{12}\sigma(h,k,hz+c) - \frac{1}{12}\sigma(h,k,c) + \frac{1}{2}\big(\!\big(\frac{c}{k}\big)\!\big) - \frac{1}{2}\big(\!\big(\frac{hz+c}{k}\big)\!\big),$$

где $d = \gcd(h,k)$. [Эта формула позволяет выразить вероятность того, что $X_{n+1} < X_n < \alpha$ при заданном α в терминах обобщенных сумм Дедекинда.]

19. Требуемая вероятность равна

$$m^{-1}\sum_{x=0}^{m-1}\Big(\Big\lfloor\frac{x-\alpha}{m}\Big\rfloor-\Big\lfloor\frac{x-\beta}{m}\Big\rfloor\Big)\Big(\Big\lfloor\frac{s(x)-\alpha'}{m}\Big\rfloor-\Big\lfloor\frac{s(x)-\beta'}{m}\Big\rfloor\Big)$$

$$= m^{-1}\sum_{x=0}^{m-1}\Big(\frac{\beta-\alpha}{m}+\Big(\Big(\frac{x-\beta}{m}\Big)\Big)-\Big(\Big(\frac{x-\alpha}{m}\Big)\Big)+\frac12\delta\Big(\frac{x-\alpha}{m}\Big)-\frac12\delta\Big(\frac{x-\beta}{m}\Big)\Big)$$
$$\times\Big(\frac{\beta'-\alpha'}{m}+\Big(\Big(\frac{s(x)-\beta'}{m}\Big)\Big)-\Big(\Big(\frac{s(x)-\alpha'}{m}\Big)\Big)+\frac12\delta\Big(\frac{s(x)-\alpha'}{m}\Big)-\frac12\delta\Big(\frac{s(x)-\beta'}{m}\Big)\Big)$$

$$= \frac{\beta-\alpha}{m}\frac{\beta'-\alpha'}{m}+\frac{1}{12m}\Big(\sigma(a,m,c+a\alpha-\alpha')-\sigma(a,m,c+a\alpha-\beta')$$
$$+\sigma(a,m,c+a\beta-\beta')-\sigma(a,m,c+a\beta-\alpha')\Big)+\epsilon,$$

где $|\epsilon| \le 2.5/m$.

[Этот подход предложен У. Дитером (U. Dieter). Разница между истинной вероятностью и идеальной величиной $\frac{\beta-\alpha}{m}\frac{\beta'-\alpha'}{m}$ ограничена величиной $\sum_{j=1}^{t} a_j/4m$ в соответствии с теоремой К. Обратно, выбирая α, β, α', β' приближенно, получим разницу, равную по крайней мере половине этой границы, когда существуют большие частичные отношения, если учитывать тот факт, что теорема К является "наилучшей из возможных". Заметим, что, когда $a \approx \sqrt{m}$, разница не может превышать $O(1/\sqrt{m})$, так что даже локально неслучайный генератор из упр. 14 будет хорошо выглядеть для критерия серий для полного периода. Это объясняет, почему мы должны требовать, чтобы разница была *очень* малой.]

20. $\sum_{0\le x<m}\lceil(x-s(x))/m\rceil\lceil(s(x)-s(s(x)))/m\rceil/m = \sum_{0\le x<m}((x-s(x))/m+(((bx+c)/m))+\frac12)((s(x)-s(s(x)))/m+((a(bx+c)/m))+\frac12)/m$ и $x/m = ((x/m))+\frac12-\frac12\delta(x/m)$, $s(x)/m = (((ax+c)/m))+\frac12-\frac12\delta((ax+c)/m)$, $s(s(x))/m = (((a^2x+ac+c)/m))+\frac12-\frac12\delta((a^2x+ac+c)/m)$. Положим, что $s(x') = s(s(x'')) = 0$ и $d = \gcd(b,m)$, тогда сумма сведется к виду

$$\frac14+\frac{1}{12m}(S_1-S_2+S_3-S_4+S_5-S_6+S_7-S_8+S_9)+\frac{d}{m}\Big(\Big(\frac{c}{d}\Big)\Big)$$
$$+\frac{1}{2m}\Big(\Big(\Big(\frac{x'-x''}{m}\Big)\Big)-\Big(\Big(\frac{x'}{m}\Big)\Big)+\Big(\Big(\frac{x''}{m}\Big)\Big)+\Big(\Big(\frac{ac+c}{m}\Big)\Big)-\Big(\Big(\frac{ac}{m}\Big)\Big)-\Big(\Big(\frac{c}{m}\Big)\Big)-\frac12\Big),$$

где $S_1 = \sigma(a,m,c)$, $S_2 = \sigma(a^2,m,ac+c)$, $S_3 = \sigma(ab,m,ac)$, $S_4 = \sigma(1,m,0) = (m-1)\times(m-2)/m$, $S_5 = \sigma(a,m,c)$, $S_6 = \sigma(b,m,c)$, $S_7 = -\sigma(a'-1,m,a'c)$ и $S_8 = -\sigma(a'(a'-1),m,(a')^2c)$, если $a'a \equiv 1$ (по модулю m). И окончательно

$$S_9 = 12\sum_{0\le x<m}\Big(\Big(\frac{bx+c}{m}\Big)\Big)\Big(\Big(\frac{a(bx+c)}{m}\Big)\Big)$$
$$= 12d\sum_{0\le x<m/d}\Big(\Big(\frac{x+c_0/d}{m/d}\Big)\Big)\Big(\Big(\frac{a(x+c_0/d)}{m/d}\Big)\Big)$$
$$= 12d\sum_{0\le x<m/d}\Big(\Big(\Big(\frac{x}{m/d}\Big)\Big)+\frac{c_0}{m}-\frac12\delta_{x0}\Big)\Big(\Big(\frac{a(x+c_0/d)}{m/d}\Big)\Big)$$
$$= d\Big(\sigma(ad,m,ac_0)+12\frac{c_0}{m}\Big(\Big(\frac{ac_0}{d}\Big)\Big)-6\Big(\Big(\frac{ac_0}{m}\Big)\Big)\Big),$$

где $c_0 = c \bmod d$. Общая сумма будет составлять около $\frac{1}{6}$, когда d мало́ и когда все дроби a/m, $(a^2 \bmod m)/m$, $(ab \bmod m)/m$, b/m, $(a'-1)/m$, $(a'(a'-1) \bmod m)/m$, $((ad) \bmod m)/m$ имеют малые частичные отношения. (Заметим, что $a' - 1 \equiv -b + b^2 - \cdots$, как в упр. 3.2.1.3–7.)

21. Сначала заметим, что основной интеграл точно разлагается следующим образом:

$$s_n = \int_{x_n}^{x_{n+1}} x\{ax+\theta\}\,dx = \frac{1}{a^2}\Big(\frac{1}{3} - \frac{\theta}{2} + \frac{n}{2}\Big), \qquad \text{если } x_n = \frac{n-\theta}{a};$$

$$s = \int_0^1 x\{ax+\theta\}\,dx = s_0 + s_1 + \cdots + s_{a-1} + \int_{-\theta/a}^{0} (ax+\theta)\,dx = \frac{1}{3a} - \frac{\theta}{2a} + \frac{a-1}{4a} + \frac{\theta^2}{2a}.$$

Поэтому $C = (s - (\frac{1}{2})^2)/(\frac{1}{3} - (\frac{1}{2})^2) = (1 - 6\theta + 6\theta^2)/a$.

22. Неравенство $s(x) < x$ выполняется на непересекающихся интервалах $[\frac{1-\theta}{a} \,..\, \frac{1-\theta}{a-1})$, $[\frac{2-\theta}{a} \,..\, \frac{2-\theta}{a-1})$, $\ldots$, $[\frac{a-\theta}{a} \,..\, 1)$, имеющих общую длину, равную

$$1 + \sum_{0<j\le a-1} \Big(\frac{j-\theta}{a-1}\Big) - \sum_{0<j\le a} \Big(\frac{j-\theta}{a}\Big) = 1 + \frac{a}{2} - \theta - \frac{a+1}{2} + \theta = \frac{1}{2}.$$

23. Справедливо $s(s(x)) < s(x) < x$, когда x находится в $[\frac{k-\theta}{a} \,..\, \frac{k-\theta}{a-1})$ и $ax + \theta - k$ принадлежат интервалам $[\frac{j-\theta}{a} \,..\, \frac{j-\theta}{a-1})$ для $0 < j \le k < a$ или когда x принадлежит $[\frac{a-\theta}{a} \,..\, 1)$ и $ax+\theta-a$ принадлежит либо $[\frac{j-\theta}{a} \,..\, \frac{j-\theta}{a-1})$ для $0 < j \le \lfloor a\theta \rfloor$, либо $[\frac{\lfloor a\theta\rfloor + 1 - \theta}{a} \,..\, \theta)$. Требуемая вероятность равна

$$\sum_{0<j\le k<a} \frac{j-\theta}{a^2(a-1)} + \sum_{0<j\le \lfloor a\theta\rfloor} \frac{j-\theta}{a^2(a-1)} + \frac{1}{a^2}\max(0, \{a\theta\} + \theta - 1)$$
$$= \frac{1}{6} + \frac{1}{6a} - \frac{\theta}{2a} + \frac{1}{a^2}\left(\frac{\lfloor a\theta\rfloor(\lfloor a\theta\rfloor + 1 - 2\theta)}{2(a-1)} + \max(0, \{a\theta\} + \theta - 1)\right),$$

т. е. $\frac{1}{6} + (1 - 3\theta + 3\theta^2)/6a + O(1/a^2)$ для больших a. Заметим, что $1 - 3\theta + 3\theta^2 \ge \frac{1}{4}$, поэтому θ не может быть выбрано так, чтобы сделать данную вероятность близкой к требуемой.

24. Поступите, как в предыдущем упражнении; сумма длин интервалов равна

$$\sum_{0<j_1\le\cdots\le j_{t-1}<a} \frac{j_1}{a^{t-1}(a-1)} = \frac{1}{a^{t-1}(a-1)}\binom{a+t-2}{t}.$$

Чтобы подсчитать среднюю длину, положим p_k равным вероятности того, что длина серии $\ge k$. Тогда среднее равно

$$\sum_{k\ge 1} p_k = \sum_{k\ge 1}\binom{a+k-2}{k}\frac{1}{a^{k-1}(a-1)} = \Big(\frac{a}{a-1}\Big)^a - \frac{a}{a-1}.$$

Эта величина для истинной случайной последовательности равна $e - 1$, а наше значение равно $e - 1 + (e/2 - 1)/a + O(1/a^2)$. [*Замечание.* Тот же результат справедлив для возрастающих серий, так как неравенство $U_n > U_{n+1}$ выполняется тогда и только тогда, когда $1 - U_n < 1 - U_{n+1}$. Это приводит к предположению, что серии в линейной конгруэнтной последовательности могут быть немного длиннее, чем нужно, поэтому к таким генераторам следует применять критерий монотонности.]

25. x должен принадлежать интервалу $[(k + \alpha' - \theta)/a \,..\, (k + \beta' - \theta)/a)$ для некоторых k и также интервалу $[\alpha \,..\, \beta)$. Пусть $k_0 = \lceil a\alpha + \theta - \beta' \rceil$, $k_1 = \lceil a\beta + \theta - \beta' \rceil$. С учетом граничных условий получим вероятность

$$(k_1 - k_0)(\beta' - \alpha')/a + \max(0, \beta - (k_1 + \alpha' - \theta)/a) - \max(0, \alpha - (k_0 + \alpha' - \theta)/a).$$

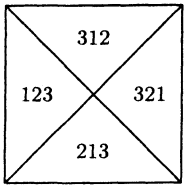


Рис. А–1. Области перестановок для генератора Фибоначчи.

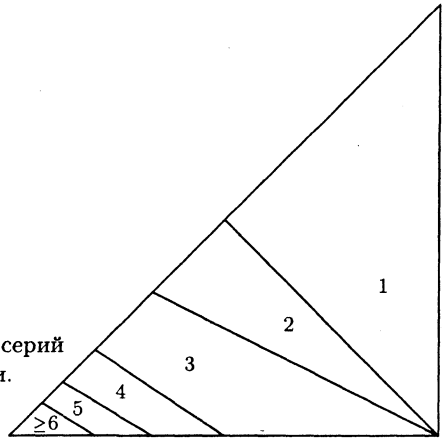


Рис. А–2. Области длин серий для генератора Фибоначчи.

Это равно $(\beta-\alpha)(\beta'-\alpha')+\epsilon$, где $|\epsilon|<2(\beta'-\alpha')/a$.

26. См. рис. А–1. Неравенства $U_1<U_3<U_2$ и $U_2<U_3<U_1$ невозможны; каждое из остальных четырех имеет вероятность $\frac{1}{4}$.

27. $U_n=\{F_{n-1}U_0+F_nU_1\}$. Необходимо, чтобы выполнялись оба следующих неравенства: $F_{k-1}U_0+F_kU_1<1$ и $F_kU_0+F_{k+1}U_1>1$. Половина единичного квадрата, в которой $U_0>U_1$, отброшена, как показано на рис. А–2, с различными отмеченными значениями k. Вероятность для серии длиной k равна $\frac{1}{2}$, если $k=1$, и равна $1/F_{k-1}F_{k+1}-1/F_kF_{k+2}$, если $k>1$. Соответствующие вероятности для случайной последовательности равны $2k/(k+1)!-2(k+1)/(k+2)!$. В приведенной ниже таблице сравниваются пять первых величин.

k:	1	2	3	4	5
Вероятности для последовательности Фибоначчи:	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{1}{10}$	$\frac{1}{24}$	$\frac{1}{65}$
Вероятности для случайной последовательности:	$\frac{1}{3}$	$\frac{5}{12}$	$\frac{11}{60}$	$\frac{19}{360}$	$\frac{29}{2520}$

28. На рис. А–3 показаны различные области для общего случая. Область 213 означает, что $U_2<U_1<U_3$, если U_1 и U_2 выбраны наудачу; область 321 означает, что $U_3<U_2<U_1$, и т. д. Вероятности для 123 и 321 равны $\frac{1}{4}-\alpha/2+\alpha^2/2$; вероятности для всех остальных случаев равны $\frac{1}{8}+\alpha/4-\alpha^2/4$. Чтобы все вероятности были равны $\frac{1}{6}$, должно выполняться равенство $1-6\alpha+6\alpha^2=0$. [Утверждение этого упражнения установлено в теореме Дж. Н. Франклина (J. N. Franklin), (см. работу J. N. Franklin, *Math. Comp.* **17** (1963), 28–59, теорема 13); другие результаты статьи Франклина имеют отношение к упр. 22 и 23.]

РАЗДЕЛ 3.3.4

1. Для генераторов с максимальным периодом 1-D точность ν_1 всегда равна m и $\mu_1=2$.

2. Пусть V — матрица со столбцами, равными $V_1,\ldots,V_t$. Минимизация $Y\cdot Y$ при условии, что $Y\neq(0,\ldots,0)$ и VY — вектор-столбец X, состоящий из целых чисел, равносильна минимизации $(V^{-1}X)\cdot(V^{-1}X)$ при условии, что X — это вектор-столбец, состоящий из целых чисел, не равных нулю. Столбцами V^{-1} являются $U_1,\ldots,U_t$.

3. $a^2\equiv 2a-1$ и $a^3\equiv 3a-2$ (по модулю m). Рассматривая все короткие решения (15), найдем, что $\nu_3^2=6$ и $\nu_4^2=4$ для соответствующих векторов $(1,-2,1)$ и $(1,-1,-1,1)$, за

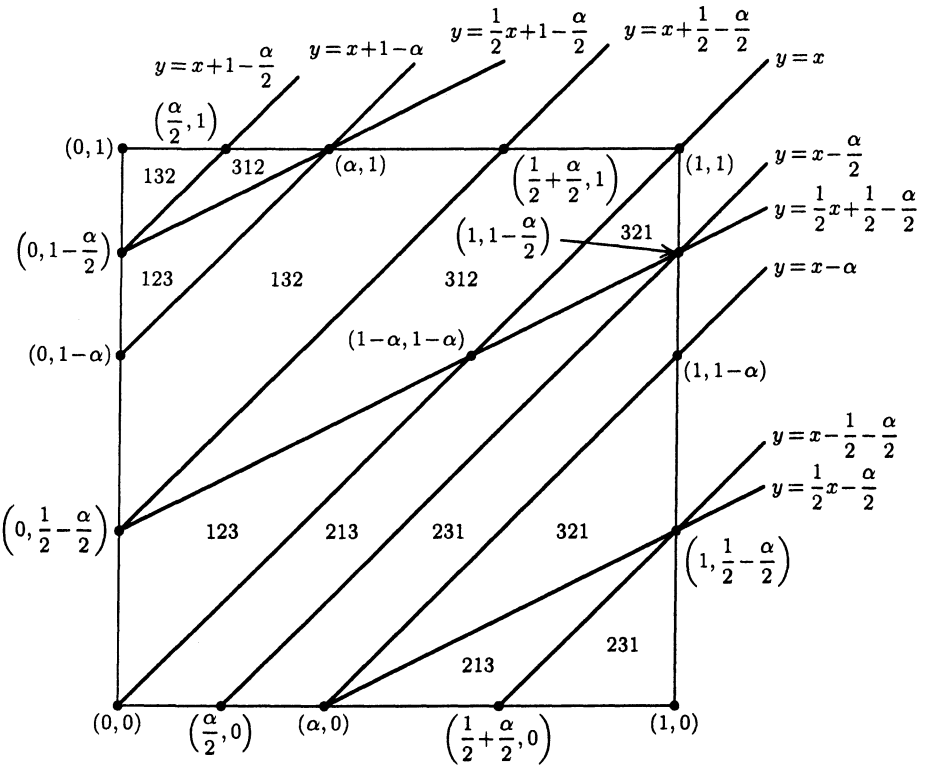


Рис. А–3. Область перестановок для генератора с потенциалом 2; $\alpha = (a-1)c/m$.

исключением следующих:

$m = 2^e q$, q — нечетное, $e \geq 3$, $a \equiv 2^{e-1}$ (по модулю 2^e), $a \equiv 1$ (по модулю q), $\nu_3^2 = \nu_4^2 = 2$;
$m = 3^e q$, $3 \perp q$, $e \geq 2$, $a \equiv 1 \pm 3^{e-1}$ (по модулю 3^e), $a \equiv 1$ (по модулю q), $\nu_4^2 = 2$;
$m = 9$, $a = 4$ или 7, $\nu_2^2 = \nu_3^2 = 5$.

4. (a) Единственным выбором для (x_1, x_2) является $\frac{1}{m}(y_1u_{22} - y_2u_{21}, -y_1u_{12} + y_2u_{11})$, и это равно $\equiv \frac{1}{m}(y_1u_{22} + y_2au_{22}, -y_1u_{12} - y_2au_{12}) \equiv (0,0)$ (по модулю 1), т. е. x_1 и x_2 являются целыми числами. (b) Когда $(x_1, x_2) \neq (0,0)$, получим $(x_1u_{11} + x_2u_{21})^2 + (x_1u_{12} + x_2u_{22})^2 = x_1^2(u_{11}^2 + u_{12}^2) + x_2^2(u_{21}^2 + u_{22}^2) + 2x_1x_2(u_{11}u_{21} + u_{12}u_{22})$, а согласно предположению это $\geq (x_1^2 + x_2^2 - |x_1x_2|)(u_{11}^2 + u_{12}^2) \geq u_{11}^2 + u_{12}^2$.

[Заметим, что полученный результат сильнее леммы A, которая утверждает только, что $x_1^2 \leq (u_{11}^2 + u_{12}^2)(u_{21}^2 + u_{22}^2)/m^2$ и $x_2^2 \leq (u_{11}^2 + u_{12}^2)^2/m^2$, где последний может быть ≥ 1. Идея, по существу, является Гауссовским методом приведения двоичной квадратичной формы (см. *Disquisitiones Arithmeticæ* (Leipzig, 1801), §171).]

5. Условия (30) остаются неизменными; следовательно, h не может быть нулем на шаге S2, когда a и m — взаимно простые числа. Так как h всегда уменьшается на этом шаге, S2 в конце концов завершится с $u^2 + v^2 \geq s$. Заметим, что $pp' \leq 0$ на протяжении всего вычисления.

6. Если $u^2+v^2 \geq s > (u-h)^2+(v-p)^2$ в предыдущем ответе, получим $(v-p)^2 > v^2$. Следовательно, $(u-h)^2 < u^2$; если $q = a_j$, то, поскольку $h' = a_j h + u$, мы должны получить $a_{j+1} = 1$. Из замечания к упр. 3.3.3–16 следует, что $\nu_2^2 = \min_{0\leq j<t}(m_j^2 + p_{j-1}^2)$.

Получим $m_0 = m_j p_j + m_{j+1}p_{j-1} = a_j m_j p_{j-1} + m_j p_{j-2} + m_{j+1}p_{j-1} < (a_j + 1 + 1/a_j) m_j p_{j-1} \leq (A + 1 + 1/A) m_j p_{j-1}$, а $m_j^2 + p_{j-1}^2 \geq 2 m_j p_{j-1}$, откуда и следует ответ.

7. Докажем, используя условие (19), что $U_j \cdot U_k = 0$ для всех $k \neq j$ тогда и только тогда, когда $V_j \cdot V_k = 0$ для всех $k \neq j$. Предположим, что $U_j \cdot U_k = 0$ для всех $k \neq j$, и пусть $U_j = \alpha_1 V_1 + \cdots + \alpha_t V_t$. Тогда $U_j \cdot U_k = \alpha_k$ для всех k. Следовательно, $U_j = \alpha_j V_j$ и $V_j \cdot V_k = \alpha_j^{-1}(U_j \cdot V_k) = 0$ для всех $k \neq j$. Аналогично доказывается обратное утверждение.

8. Ясно, что $\nu_{t+1} \leq \nu_t$ (факт безоговорочно использован в алгоритме S, так как s не изменяется при возрастании t). Для $t = 2$ это эквивалентно $(m\mu_2/\pi)^{1/2} \geq (\frac{3}{4} m \mu_3/\pi)^{1/3}$, т. е. $\mu_3 \leq \frac{4}{3}\sqrt{m/\pi}\,\mu_2^{3/2}$. Эта граница доведена до $\frac{4}{3}10^{-4}/\sqrt{\pi}$ для заданных параметров, но для больших m и фиксированного μ_2 граница (40) лучше.

9. Пусть $f(y_1, \ldots, y_t) = \theta$; $\gcd(y_1, \ldots, y_t) = 1$ (gcd — наибольший общий делитель) и W — целочисленная матрица с определителем, равным 1, первая строка которой — $(y_1, \ldots, y_t)$. (Последнее доказать по индукции по наименьшей ненулевой величине, занесенной в строку.) Если $X = (x_1, \ldots, x_t)$ — вектор-строка, получим $XW = X'$ тогда и только тогда, когда $X = X'W^{-1}$, а если W^{-1} — целочисленная матрица с определителем, равным 1, форма g, определенная WU, удовлетворяет $g(x_1, \ldots, x_t) = f(x_1', \ldots, x_t')$. Кроме того, $g(1, 0, \ldots, 0) = \theta$.

Без уменьшения общности предположим, что $f = g$. Если S — любая ортогональная матрица, матрица US определяет ту же форму, что и U, поэтому $(XUS)(XUS)^T = (XU)(XU)^T$. Выберем S так, чтобы ее первый столбец был кратным U_1^T, а все другие столбцы — любые подходящие векторы. Тогда получим

$$US = \begin{pmatrix} \alpha_1 & 0 & \ldots & 0 \\ \alpha_2 & & & \\ \vdots & & U' & \\ \alpha_t & & & \end{pmatrix}$$

для некоторых α_1, α_2, $\ldots$, α_t и некоторой матрицы U' размера $(t-1) \times (t-1)$. Следовательно, $f(x_1, \ldots, x_t) = (\alpha_1 x_1 + \cdots + \alpha_t x_t)^2 + h(x_2, \ldots, x_t)$. Из этого следует, что $\alpha_1 = \sqrt{\theta}$ [фактически $\alpha_j = (U_1 \cdot U_j)/\sqrt{\theta}$ для $1 \leq j \leq t$] и что h — положительно определенная квадратичная форма, порожденная U', где $\det U' = (\det U)/\sqrt{\theta}$. Индукцией по t можно показать, что существуют целые числа $(x_2, \ldots, x_t)$, для которых выполняется

$$h(x_2, \ldots, x_t) \leq (\tfrac{4}{3})^{(t-2)/2} |\det U|^{2/(t-1)}/\theta^{1/(t-1)},$$

и для этих целых чисел x_1 можно выбрать таким образом, что $|x_1 + (\alpha_2 x_2 + \cdots + \alpha_t x_t)/\alpha_1| \leq \frac{1}{2}$, а это эквивалентно $(\alpha_1 x_1 + \cdots + \alpha_t x_t)^2 \leq \frac{1}{4}\theta$. Значит,

$$\theta \leq f(x_1, \ldots, x_t) \leq \tfrac{1}{4}\theta + (\tfrac{4}{3})^{(t-2)/2} |\det U|^{2/(t-1)}/\theta^{1/(t-1)}$$

и желаемое неравенство получается немедленно.

[*Замечание.* Для $t = 2$ это наилучший из возможных результатов. В общем случае из теоремы Эрмита следует, что $\mu_t \leq \pi^{t/2}(4/3)^{t(t-1)/4}/(t/2)!$. Из фундаментальной теоремы Минковского ("Каждое t-мерное выпуклое множество, симметричное относительно начала координат с объемом $\geq 2^t$, содержит не равную нулю точку с целыми координатами") следует, что $\mu_t \leq 2^t$. Это более точное неравенство, чем то, которое следует из теоремы Эрмита для $t \geq 9$. Известен еще более точный результат (см. (41)).]

10. Так как y_1 и y_2 — взаимно простые числа, можно найти решение уравнения $u_1y_2 - u_2y_1 = m$. Кроме того, $(u_1 + qy_1)y_2 - (u_2 + qy_2)y_1 = m$ для всех q, поэтому, выбирая подходящее целое q, можно убедиться, что $2|u_1y_1 + u_2y_2| \le y_1^2 + y_2^2$. Сейчас $y_2(u_1 + au_2) \equiv y_2u_1 - y_1u_2 \equiv 0$ (по модулю m), y_2 и m должны быть взаимно простыми числами. Следовательно, $u_1 + au_2 \equiv 0$. Окончательно, пусть $|u_1y_1 + u_2y_2| = \alpha m$, $u_1^2 + u_2^2 = \beta m$, $y_1^2 + y_2^2 = \gamma m$, справедливо $0 \le \alpha \le \frac{1}{2}\gamma$ и остается показать, что $\alpha \le \frac{1}{2}\beta$ и $\beta\gamma \ge 1$. Тождество $(u_1y_2 - u_2y_1)^2 + (u_1y_1 + u_2y_2)^2 = (u_1^2 + u_2^2)(y_1^2 + y_2^2)$ влечет равенство $1 + \alpha^2 = \beta\gamma$. Если $\alpha > \frac{1}{2}\beta$, то справедливо $2\alpha\gamma > 1 + \alpha^2$, т. е. $\gamma - \sqrt{\gamma^2 - 1} < \alpha \le \frac{1}{2}\gamma$. Но $\frac{1}{2}\gamma < \sqrt{\gamma^2 - 1}$ влечет неравенство $\gamma^2 > \frac{4}{3}$, что приводит к противоречию.

11. Так как a нечетно, $y_1 + y_2$ должна быть четной. Чтобы избежать решения, где и y_1, и y_2 четные, положим $y_1 = x_1 + x_2$, $y_2 = x_1 - x_2$ и решим уравнение $x_1^2 + x_2^2 = m/\sqrt{3} - \epsilon$ при $x_1 \perp x_2$ и четном x_1. Соответствующий множитель a будет решением уравнения $(x_2 - x_1)a \equiv x_2 + x_1$ (по модулю 2^e). Не составляет труда доказать, что $a \equiv 1$ (по модулю 2^{k+1}) тогда и только тогда, когда $x_1 \equiv 0$ (по модулю 2^k), поэтому получим наилучший потенциал, когда $x_1 \bmod 4 = 2$. Проблема сводится к нахождению относительно простого решения $x_1^2 + x_2^2 = N$, где N — большое число вида $4k + 1$. Разлагая N на комплексные целые числа (гауссовские целые числа), мы получим, что решение существует тогда и только тогда, когда каждый простой множитель N (среди обычных целых чисел) имеет вид $4k + 1$.

В соответствии с известной теоремой Ферма каждое простое p вида $4k + 1$ можно с точностью до знаков u и v записать единственным способом: $p = u^2 + v^2 = (u + iv)(u - iv)$, где v четное. Числа u и v эффективно могут быть вычислены, если решить уравнение $x^2 \equiv -1$ (по модулю p) и затем вычислить $u + iv = \gcd(x + i, p)$ с помощью алгоритма Евклида по комплексным целым числам (гауссовским целым числам). [Можно взять $x = n^{(p-1)/4} \bmod p$ для почти половины всех целых n. Это применение алгоритма Евклида, по существу, такое же, как и нахождение наименьшей не равной нулю величины $u^2 + v^2$, такой, что $u \pm xv \equiv 0$ (по модулю p). Значения u и v также появляются, когда алгоритм Евклида для целых чисел применяется обычным образом к p и x; см. работу Ж. А. Серре и К. Эрмита (J. A. Serret and C. Hermite, *J. de Math. Pures et Appl.* **5** (1848), 12–15).] Если разложение на простые множители N имеет вид $p_1^{e_1} \dots p_r^{e_r} = (u_1 + iv_1)^{e_1}(u_1 - iv_1)^{e_1} \dots (u_r + iv_r)^{e_r}(u_r - iv_r)^{e_r}$, то получим 2^{r-1} различных решений $x_1^2 + x_2^2 = N$, $x_1 \perp x_2$, x_1 четное, полагая $|x_2| + i|x_1| = (u_1 + iv_1)^{e_1}(u_2 \pm iv_2)^{e_2} \dots (u_r \pm iv_r)^{e_r}$. Подобным методом можно получить все такие решения.

Замечание. Аналогичная процедура может использоваться, когда $m = 10^e$, но для этого потребуется в пять раз больше работы, так как необходимо хранить информацию до нахождения решения с $x_1 \equiv 0$ (по модулю 10). Например, когда $m = 10^{10}$, получим $\lfloor m/\sqrt{3} \rfloor = 5773502691$ и $5773502689 = 53 \cdot 108934013 = (7 + 2i)(7 - 2i)(2203 + 10202i) \times (2203 - 10202i)$. Из решений $|x_2| + i|x_1| = (7 + 2i)(2203 + 10202i)$ и $(7 + 2i)(2203 - 10202i)$ первое дает $|x_1| = 67008$ (не хорошее) и второе — $|x_1| = 75820$, $|x_2| = 4983$ (это подходит). Строка 9 из табл. 1 была получена, когда мы положили $x_1 = 75820$, $x_2 = -4983$.

Строка 14 таблицы была получена следующим образом. $\lfloor 2^{32}/\sqrt{3} \rfloor = 2479700524$; спустимся вниз к $N = 2479700521$, равному $37 \cdot 797 \cdot 84089$, и получим четыре решения: $N = 4364^2 + 49605^2 = 26364^2 + 42245^2 = 38640^2 + 31411^2 = 11960^2 + 48339^2$. Соответствующими множителями будут 2974037721, 2254986297, 4246248609 и 956772177. Проверим также $N - 4$, но это число не подходит, поскольку оно не делится на 3. С другой стороны, простое число $N - 8 = 45088^2 + 21137^2$ приводит к получению множителя 3825140801. Аналогично получим дополнительные множители для $N - 20$, $N - 44$, $N - 48$ и т. д. Множитель строки 14 — наилучший из первых шестнадцати множителей, найденных с помощью данной процедуры; это один из четырех множителей для $N - 68$.

12. $U_j' \cdot U_j' = U_j \cdot U_j + 2\sum_{i \ne j} q_i(U_i \cdot U_j) + \sum_{i \ne j}\sum_{k \ne j} q_iq_k(U_i \cdot U_k)$. Взята вторая частная производная по q_k от левой части (26). Если достигается минимум, то частная производная должна обращаться в нуль.

13. $u_{11} = 1$, $u_{21} =$ иррациональны, $u_{12} = u_{22} = 0$.

14. После трехкратного применения алгоритма Евклида получим $\nu_2^2 = 5^2 + 5^2$. Затем после выполнения шага S4 получим

$$U = \begin{pmatrix} -5 & 5 & 0 \\ -18 & -2 & 0 \\ 1 & -2 & 1 \end{pmatrix}, \qquad V = \begin{pmatrix} -2 & 18 & 38 \\ -5 & -5 & -5 \\ 0 & 0 & 100 \end{pmatrix}.$$

Результатом преобразования $(j, q_1, q_2, q_3) = (1, *, 0, 2), (2, -4, *, 1), (3, 0, 0, *), (1, *, 0, 0)$ будет

$$U = \begin{pmatrix} -3 & 1 & 2 \\ -5 & -8 & -7 \\ 1 & -2 & 1 \end{pmatrix}, \qquad V = \begin{pmatrix} -22 & -2 & 18 \\ -5 & -5 & -5 \\ 9 & -31 & 29 \end{pmatrix}, \qquad Z = (0 \quad 0 \quad 1).$$

Таким образом, $\nu_3 = \sqrt{6}$, как уже известно из упр. 3.

15. Наибольшее достижимое q в (11) минус наименьшее достижимое плюс 1 равно $|u_1| + \cdots + |u_t| - \delta$, где $\delta = 1$, если $u_i u_j < 0$ для некоторых i и j. В остальных случаях $\delta = 0$. Например, если $t = 5$, $u_1 > 0$, $u_2 > 0$, $u_3 > 0$, $u_4 = 0$ и $u_5 < 0$, наибольшее достижимое значение q равно $q = u_1 + u_2 + u_3 - 1$, а наименьшее равно $q = u_5 + 1 = -|u_5| + 1$.

[Заметим, что число гиперплоскостей остается прежним при изменении c. Следовательно, ответ будет таким же, если вместо L_0 покрыть L. Однако приведенная формула *не* всегда точна для покрытия L_0, поскольку не во всех гиперплоскостях, пересекающих единичный гиперкуб, содержатся точки из L_0. В приведенном выше примере можно никогда не достичь значения $q = u_1 + u_2 + u_3 - 1$ в L_0, если $u_1 + u_2 + u_3 > m$. Это значение достигается тогда и только тогда, когда существует решение $m - u_1 - u_2 - u_3 = x_1 u_1 + x_2 u_2 + x_3 u_3 + x_4 |u_5|$, где (x_1, x_2, x_3, x_4) — неотрицательные целые числа. Возможно, верно, что данный предел всегда достижим, когда сумма $|u_1| + \cdots + |u_t|$ минимальна, но это не кажется очевидным.]

16. Достаточно найти все решения (15), имеющие минимум $|u_1| + \cdots + |u_t|$, и вычесть 1, если какое-нибудь из этих решений имеет компоненты с противоположным знаком.

Вместо положительно определенной квадратичной формы будем использовать функцию, в некотором смысле аналогичную ей, $(f(x_1, \ldots, x_t) = |x_1 U_1 + \cdots + x_t U_t|)$, определяя $|Y| = |y_1| + \cdots + |y_t|$. Неравенство (21) можно заменить неравенством $|x_k| \le f(y_1, \ldots, y_t) \times (\max_{1 \le j \le t} |v_{kj}|)$.

Таким образом может быть получен следующий работающий алгоритм. Заменить шаг S1 шагом S3: "Присвоить $U \leftarrow (m)$, $V \leftarrow (1)$, $r \leftarrow 1$, $s \leftarrow m$, $t \leftarrow 1$". (Здесь U и V — матрицы размера 1×1. Следовательно, двумерный случай сводится к общему методу. Для $t = 2$ можно, конечно, использовать частную процедуру; см. работу, приведенную в ответе к упр. 5.) На шагах S4 и S7 присвоить $s \leftarrow \min(s, |U_k|)$. На шаге S7 присвоить $z_k \leftarrow \lfloor \max_{1 \le j \le t} |v_{kj}| \, s/m \rfloor$. На шаге S9 присвоить $s \leftarrow \min(s, |Y| - \delta)$ и на шаге S10 выход, $s = N_t$. Иначе — оставьте алгоритм в том виде, в каком он сформулирован, так как он уже производит достаточно короткие векторы. [*Math. Comp.* **29** (1975), 827–833.]

17. Когда $k > t$ на шаге S9 и $Y \cdot Y \le s$, взять в качестве выхода Y и $-Y$. Кроме того, если $Y \cdot Y < s$, взять в качестве выхода предыдущие векторы для этого t. [При подготовке табл. 1 автор заметил, что существовал точно один вектор (и он же со знаком "минус") на выходе для ν_t, кроме случаев, когда $y_1 = 0$ или $y_t = 0$.]

18. (a) Пусть $x = m$, $y = (1 - m)/3$, $v_{ij} = y + x\delta_{ij}$, $u_{ij} = -y + \delta_{ij}$. Тогда $V_j \cdot V_k = \frac{1}{3}(m^2 - 1)$ для $j \ne k$, $V_k \cdot V_k = \frac{2}{3}(m^2 + \frac{1}{2})$, $U_j \cdot U_j = \frac{1}{3}(m^2 + 2)$, $z_k \approx \sqrt{\frac{2}{9}}\, m$. (Этот пример удовлетворяет (28) с $a = 1$ и работает для всех $m \equiv 1$ (по модулю 3).)

(b) Необходимо поменять местами U и V на шаге S5. После замены присвоить также $s \leftarrow \min(s, U_i \cdot U_i)$ для всех U_i. Например, когда $m = 64$, это преобразование с $j = 1$, примененное к матрице из (a), сводит

$$V = \begin{pmatrix} 43 & -21 & -21 \\ -21 & 43 & -21 \\ -21 & -21 & 43 \end{pmatrix}, \quad U = \begin{pmatrix} 22 & 21 & 21 \\ 21 & 22 & 21 \\ 21 & 21 & 22 \end{pmatrix}$$

к

$$V = \begin{pmatrix} 1 & 1 & 1 \\ -21 & 43 & -21 \\ -21 & -21 & 43 \end{pmatrix}, \quad U = \begin{pmatrix} 22 & 21 & 21 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}.$$

[Поскольку преобразование может привести к увеличению длины V_j, алгоритм, объединяющий оба преобразования, должен быть таким, чтобы не допускалось зацикливание. См. также упр. 23.]

19. Нет, так как произведение не единичных матриц со всеми неотрицательными элементами вне диагонали и со всеми диагональными элементами, равными единице, не может быть единичным.

[Однако зацикливание было бы возможным, если бы последующее преобразование с $q = -1$ осуществлялось тогда, когда $-2V_i \cdot V_j = V_j \cdot V_j$. Правило округления должно быть несимметричным относительно знака, если допустимо несокращенное преобразование.]

20. Когда $a \bmod 8 = 5$, точки $2^{-e}(x, s(x), \ldots, s^{[t-1]}(x))$ для x в периоде — это те же самые точки, что и $2^{2-e}(y, \sigma(y), \ldots, \sigma^{t-1}(y))$ для $0 \le y < 2^{e-2}$ плюс $(X_0 \bmod 4)/2^e$, где $\sigma(y) = (ay + \lfloor a/4 \rfloor) \bmod 2^{e-2}$. Поэтому в рассматриваемом случае следовало бы использовать алгоритм S с $m = 2^{e-2}$.

Когда $a \bmod 8 = 3$, максимальное расстояние между параллельными гиперплоскостями, содержащими точки $2^{-e}(x, s(x), \ldots, s^{[t-1]}(x))$ по модулю 1, такое же, как и максимальное расстояние между параллельными гиперплоскостями, содержащими точки $2^{-e}(x, -s(x), \ldots, (-1)^{t-1} s^{[t-1]}(x))$, поскольку замена знака координат не изменяет расстояния. Последней будет точка $2^{2-e}(y, \sigma(y), \ldots, \sigma^{t-1}(y))$, где $\sigma(y) = (-ay - \lceil a/4 \rceil) \bmod 2^{e-2}$ плюс постоянное отклонение. Снова применим алгоритм S с $m = 2^{e-2}$; замена a на $m - a$ не сказывается на результате.

21. $X_{4n+4} \equiv X_{4n}$ (по модулю 4), поэтому сейчас удобно положить $V_1 = (4, 4a^2, 4a^3)/m$, $V_2 = (0, 1, 0)$, $V_3 = (0, 0, 1)$ и определить соответствующую решетку L_0.

24. Пусть $m = p^e$; можно произвести анализ, подобный приведенному в разделе. Например, когда $t = 4$, то $X_{n+3} = ((a^2 + b)X_{n+1} + abX_n) \bmod m$, и необходимо так минимизировать $u_1^2 + u_2^2 + u_3^2 + u_4^2 \neq 0$, чтобы $u_1 + bu_3 + abu_4 \equiv u_2 + au_3 + (a^2 + b)u_4 \equiv 0$ (по модулю m).

Заменим шаги S1–S3 операцией присвоения

$$U \leftarrow \begin{pmatrix} m & 0 \\ 0 & m \end{pmatrix}, \qquad V \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad R \leftarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \qquad s \leftarrow m^2, \qquad t \leftarrow 2$$

и на выходе получим $\nu_2 = m$. Заменим шаг S4 шагом S4′.

S4′. [Продвижение t.] Если $t = T$, алгоритм завершен. Иначе — присвоить $t \leftarrow t + 1$ и $R \leftarrow R\left(\begin{smallmatrix} 0 & b \\ 1 & a \end{smallmatrix}\right) \bmod m$. Присвоим U_t новой строке $(-r_{12}, -r_{22}, 0, \ldots, 0, 1)$ из t элементов и присвоим $u_{it} \leftarrow 0$ для $1 \le i < t$. Присвоим V_t новой строке $(0, \ldots, 0, m)$. Для $1 \le i < t$ присвоим $q \leftarrow$ округление$((v_{i1}r_{12} + v_{i2}r_{22})/m)$, $v_{it} \leftarrow v_{i1}r_{12} + v_{i2}r_{22} - qm$ и $U_t \leftarrow U_t + qU_i$. Наконец, присвоить $s \leftarrow \min(s, U_t \cdot U_t)$, $k \leftarrow t$, $j \leftarrow 1$.

[Подобное обобщение применяется ко всем последовательностям длиной $p^k - 1$, удовлетворяющим линейным рекуррентным соотношениям 3.2.2–(8). Дополнительные числовые

примеры приведены в работах A. Grube, *Zeitschrift für angewandte Math. und Mechanik* **53** (1973), T223–T225; , L'Ecuyer, Blouin, and Couture *ACM Trans. Modeling and Comp. Simul.* **3** (1993), 87–98.]

25. Рассматриваемая сумма меньше либо равна удвоенной сумме $\sum_{0\le k\le m/2d} r(dk) = 1 + \frac{1}{d}f(m/d)$, где

$$f(m) = \frac{1}{m}\sum_{1\le k\le m/2} \csc(\pi k/m)$$
$$= \frac{1}{m}\int_1^{m/2} \csc(\pi x/m)\,dx + O\Big(\frac{1}{m}\Big) = \frac{1}{\pi}\ln\tan\Big(\frac{\pi}{2m}x\Big)\Big|_1^{m/2} + O\Big(\frac{1}{m}\Big).$$

[Когда $d = 1$, справедливо равенство $\sum_{0\le k<m} r(k) = (2/\pi)\ln m + 1 + (2/\pi)\ln(2e/\pi) + O(1/m)$.]

26. Когда $m = 1$, использовать (52) нельзя, так как k будет равняться нулю. Если $\gcd(q, m) = d$ (gcd — наибольший общий делитель), то доказательство остается прежним, только m заменяется на m/d. Предположим, что справедливо равенство $m = p_1^{e_1}\dots p_r^{e_r}$, $\gcd(a-1, m) = p_1^{f_1}\dots p_r^{f_r}$ и $d = p_1^{d_1}\dots p_r^{d_r}$. Если m заменить на m/d, то s заменяется на $p_1^{\max(0,e_1-f_1-d_1)}\dots p_r^{\max(0,e_r-f_r-d_r)}$.

27. Удобно использовать следующие функции: $\rho(x) = 1$, если $x = 0$; $\rho(x) = x$, если $0 < x \le m/2$; $\rho(x) = m - x$, если $m/2 < x < m$; усечение$(x) = \lfloor x/2\rfloor$, если $0 \le x \le m/2$; усечение$(x) = m - \lfloor (m-x)/2\rfloor$, если $m/2 < x < m$; $L(x) = 0$, если $x = 0$; $L(x) = \lfloor \lg x\rfloor + 1$, если $0 < x \le m/2$; $L(x) = -(\lfloor \lg(m-x)\rfloor + 1)$, если $m/2 < x < m$, и $l(x) = \max(1, 2^{|x|-1})$. Заметим, что $l(L(x)) \le \rho(x) < 2l(L(x))$ и $2\rho(x) \le 1/r(x) = m\sin(\pi x/m) < \pi\rho(x)$ для $0 < x < m$.

Скажем, что вектор $(u_1, \dots, u_t)$ *плохой*, если он не равен нулю и удовлетворяет (15). Пусть $\rho_{\min}$ — минимальное значение $\rho(u_1)\dots\rho(u_t)$ по всем плохим $(u_1, \dots, u_t)$. Вектор $(u_1, \dots, u_t)$ относят к классу $(L(u_1), \dots, L(u_t))$. Поэтому существует не более $(2\lg m + 1)^t$ классов и в классе $(L_1, \dots, L_t)$ содержится самое большее $l(L_1)\dots l(L_t)$ векторов. Чтобы получить требуемое утверждение, достаточно доказать, что плохие векторы в каждом фиксированном классе привносят самое большее $2/\rho_{\min}$ в сумму $\sum r(u_1, \dots, u_t)$; это позволяет получить желаемую грань, так как $1/\rho_{\min} < \pi^t r_{\max}$.

Пусть $\mu = \lfloor \lg \rho_{\min}\rfloor$. *$\mu$-кратный оператор усечения*, применяемый к вектору, определяется следующей операцией, выполненной μ раз: "Пусть j — такой минимальный индекс, что $\rho(u_j) > 1$. Заменим u_j на trunc(u_j), но ничего не будем делать, если $\rho(u_j) = 1$ для всех j". (Эта операция, по существу, отбрасывает один двоичный разряд информации от $(u_1, \dots, u_t)$.) Если $(u_1', \dots, u_t')$ и $(u_1'', \dots, u_t'')$ — два вектора из одного и того же класса, имеющие одно и то же μ-кратное усечение, то мы говорим, что они *подобны*; в этом случае справедливо неравенство $\rho(u_1' - u_1'')\dots\rho(u_t' - u_t'') < 2^\mu \le \rho_{\min}$. Например, любые два вектора вида $((1x_2x_1)_2, 0, m - (1x_3)_2, (101x_5x_4)_2, (1101)_2)$ подобны, когда m большое и $\mu = 5$; μ-кратный оператор усечения последовательно удаляет x_1, x_2, x_3, x_4, x_5. Так как разность двух плохих векторов удовлетворяет (15), невозможно, чтобы два неравных плохих вектора были подобными. Поэтому класс $(L_1, \dots, L_t)$ может содержать самое большее $\max(1, l(L_1)\dots l(L_t)/2^\mu)$ плохих векторов. Если в классе $(L_1, \dots, L_t)$ содержится точно один плохой вектор $(u_1, \dots, u_t)$, то справедливы неравенства $r(u_1, \dots, u_t) \le r_{\max} \le 1/\rho_{\min}$; если в нем содержится $\le l(L_1)\dots l(L_t)/2^\mu$ плохих векторов, то для каждого из них выполняется $r(u_1, \dots, u_t) \le 1/\rho(u_1)\dots\rho(u_t) \le 1/l(L_1)\dots l(L_t)$ и справедливо неравенство $1/2^\mu < 2/\rho_{\min}$.

28. Пусть $\zeta = e^{2\pi i/(m-1)}$ и пусть $S_{kl} = \sum_{0\le j<m-1} \omega^{x_{j+l}}\zeta^{jk}$. Аналогом (51) будет равенство $|S_{k0}| = \sqrt{m}$. Значит, аналогом (53) является

$$\left| N^{-1} \sum_{0\le n<N} \omega^{x_n} \right| = O((\sqrt{m}\log m)/N).$$

Теорема, аналогичная теореме N, сейчас утверждает, что

$$D_N^{(t)} = O\left(\frac{\sqrt{m}\,(\log m)^{t+1}}{N}\right) + O\left((\log m)^t r_{\max}\right), \qquad D_{m-1}^{(t)} = O((\log m)^t r_{\max}).$$

На самом деле $D_{m-1}^{(t)} \le \frac{m-2}{m-1}\sum r(u_1,\dots,u_t)$ [суммируем по всем не равным нулю решениям (15)] $+ \frac{1}{m-1}\sum r(u_1,\dots,u_t)$ [суммируем по всем не равным нулю $(u_1,\dots,u_t)$]. Из упр. 25, если положить $d = 1$, следует, что последняя сумма будет иметь порядок $O(\log m)^t$. С полученной суммой поступим, как в упр. 27.

Рассмотрим величину $R(a) = \sum r(u_1,\dots,u_t)$, где суммирование выполняется по ненулевым решениям (15). Так как m — простое число, каждый $(u_1,\dots,u_t)$ может быть решением (15) для самое большее $t-1$ значений a. Следовательно, $\sum_{0<a<m} R(a) \le (t-1)\sum r(u_1,\dots,u_t) = O(t(\log m)^t)$. Отсюда получаем, что среднее значение $R(a)$, взятое по всем $\varphi(m-1)$ первообразным корням, равно $O(t(\log m)^t/\varphi(m-1))$.

Замечание. Справедливо соотношение $1/\varphi(n) = O(\log\log n/n)$, поэтому необходимо доказать, что *для всех простых чисел m и для всех T существует первообразный корень a по модулю m, такой, что линейная конгруэнтная последовательность $(1, a, 0, m)$ имеет разброс $D_{m-1}^{(t)} = O(m^{-1}T(\log m)^T\log\log m)$ для $1 \le t \le T$.* Этот метод доказательства *нельзя* расширить, чтобы получить подобные результаты для линейного конгруэнтного генератора с периодом 2^e по модулю 2^e, так как, например, вектор $(1,-3,3,-1)$ будет решением (15) для приблизительно $2^{2e/3}$ значений a.

29. Чтобы получить верхнюю границу, позволим ненулевым компонентам $u = (u_1,\dots,u_t)$ быть любыми действительными величинами $1 \le |u_j| \le \frac{1}{2}m$. Если k компонент не равны нулю, то $r(u) \le 1/(2^k\rho(u))$ в обозначениях ответа к упр. 27. И, если $u_1^2 + \cdots + u_t^2$ равно данному значению ν^2, минимизируем $\rho(u)$, взяв $u_1 = \cdots = u_{k-1} = 1$ и $u_k^2 = \nu^2 - k + 1$. Таким образом, $r(u) \le 1/2^k\sqrt{\nu^2-k+1}$. Однако $2^k\sqrt{\nu^2-k+1} \ge \sqrt{8}\nu$, поскольку $\nu \ge k \ge 2$.

30. Сначала минимизируем $q|aq-mp|$ для $1 \le q < m$ и $0 \le p < a$. В обозначениях упр. 4.5.3–42 справедливо равенство $aq_n - mp_n = (-1)^n K_{s-n-1}(a_{n+2},\dots,a_s)$ для $0 \le n \le s$. В области $q_{n-1} \le q < q_n$ справедливо неравенство $|aq-mp| \ge |aq_{n-1}-mp_{n-1}|$; значит, $q|aq-mp| \ge q_{n-1}|aq_{n-1}-mp_{n-1}|$ и минимум равен $\min_{0\le n<s} q_n|aq_n-mp_n| = \min_{0\le n<s} K_n(a_1,\dots,a_n)\,K_{s-n-1}(a_{n+2},\dots,a_s)$. Из упр. 4.5.3–32 следует равенство $m = K_n(a_1,\dots,a_n)\,a_{n+1}\,K_{s-n-1}(a_{n+2},\dots,a_s) + K_n(a_1,\dots,a_n)\,K_{s-n-2}(a_{n+3},\dots,a_s) + K_{n-1}(a_1,\dots,a_{n-1})\,K_{s-n-1}(a_{n+2},\dots,a_s)$; и задача, по существу, состоит в нахождении максимума величины $m/K_n(a_1,\dots,a_n)\,K_{s-n-1}(a_{n+2},\dots,a_s)$, лежащей между a_{n+1} и $a_{n+1}+2$.

Пусть сейчас $A = \max(a_1,\dots,a_s)$. Так как $r(m-u) = r(u)$, можно предположить, что $r_{\max} = r(u)r(au \bmod m)$ для некоторого u с $1 \le u \le \frac{1}{2}m$. Полагая $u' = \min(au \bmod m, (-au) \bmod m)$, получим $r_{\max} = r(u)r(u')$. Из предыдущего раздела известно, что $uu' \ge qq'$, где $A/m \le 1/qq' \le (A+2)/m$. Кроме того, $2u \le r(u)^{-1} \le \pi u$ для $0 < u \le \frac{1}{2}m$, тогда $r_{\max} \le 1/(4uu')$. Следовательно, $r_{\max} \le (A+2)/(4m)$. (Существует подобная нижняя грань, а именно — $r_{\max} > A/(\pi^2 m)$.)

31. Эквивалентное предположение состоит в том, что все большие m могут быть записаны в виде $m = K_n(a_1,\dots,a_n)$ для некоторого n и некоторых $a_i \in \{1,2,3\}$. Для фиксированного n 3^n чисел $K_n(a_1,\dots,a_n)$ имеют среднее значение порядка $(1+\sqrt{2}\,)^n$, и их стандартное

отклонение имеет порядок $(2.51527)^n$; так что предположение почти всюду верно. С. К. Заремба (S. K. Zaremba) в 1972 году предположил, что все m могут иметь такое представление с $a_i \le 5$; Т. В. Кузик (T. W. Cusick) достиг определенного прогресса в решении этой задачи в работе *Mathematika* **24** (1977), 166–172. Оказалось, что только в случаях, когда $m = 54$ и $m = 150$, требуются $a_i = 5$, и самые большие m_j, для которых необходимо $a_i = 4$, — это 2052, 2370, 5052 и 6234; по крайней мере, автор нашел представление с $a_i \le 3$ для всех других целых чисел, меньших 2 000 000. Чтобы выполнялось неравенство $a_i \le 2$, среднее $K_n(a_1, \ldots, a_n)$ должно быть равно $\frac{4}{5}2^n + \frac{1}{5}(-2)^{-n}$, в то время как стандартное отклонение растет как $(2.04033)^n$. Оказалось, что плотность таких чисел в эксперименте автора (автор рассмотрел 2^6 блоков из 2^{14} чисел каждый для $m \le 2^{20}$) изменялась между .50 и .65.

[См. работу I. Borosh and H. Niederreiter, *BIT* **25** (1980), 193–208, в которой рассматривается вычислительный метод нахождения множителей с малым частичным отношением. Авторы нашли представление с a_i для $m = 2^e$, где $25 \le e \le 35$.]

32. (a) $U_n - Z_n/m_1 \equiv (m_2 - m_1)Y_n/m_1m_2$ (по модулю 1) и $(m_1 - m_2)/m_1m_2 \approx 2^{-54}$. (Поэтому можно анализировать самый старший двоичный разряд Z_n, анализируя U_n. Младшие разряды, вероятно, также случайны, но эти соображения к ним не применяются.) (b) Справедливо равенство $U_n = W_n/m$ для всех n. Из китайской теоремы об остатках следует, что достаточно только проверить соотношения $W_n \equiv X_nm_2$ (по модулю m_1) и $W_n \equiv -Y_nm_1$ (по модулю m_2), так как $m_1 \perp m_2$. [Pierre L'Ecuyer and Shu Tezuka, *Math. Comp.* **57** (1991), 735–746.]

РАЗДЕЛ 3.4.1

1. $\alpha + (\beta - \alpha)U$.

2. Пусть $U = X/m$, тогда $\lfloor kU \rfloor = r \iff r \le kX/m < r + 1 \iff mr/k \le X < m(r + 1)/k \iff \lceil mr/k \rceil \le X < \lceil m(r + 1)/k \rceil$. Точная вероятность задается формулой $(1/m)(\lceil m(r + 1)/k \rceil - \lceil mr/k \rceil) = 1/k + \epsilon$, где $|\epsilon| < 1/m$.

3. Если заданы случайные числа, длина которых равна машинному слову, то результат будет отличаться от правильного распределения самое большее на $1/m$, как показано в упр. 2, но все эксцессы приводят к наименьшим отличиям. Так, если $k \approx m/3$, результат будет меньше $k/2$ приблизительно в $\frac{2}{3}$ раз. Намного лучше получить совершенно равномерное распределение, отбросив U, если $U > k\lfloor m/k \rfloor$ [см. D. E. Knuth, *The Stanford GraphBase* (New York: ACM Press, 1994), 221].

С другой стороны, если использовалась линейная конгруэнтная последовательность, то k и модуль m должны быть взаимно простыми числами, иначе числа имеют очень короткий период, как следует из раздела 3.2.1.1. Например, если $k = 2$ и m четное, наилучшими числами будут попеременно повторяемые числа 0 и 1. Метод слабее (1) почти в каждом случае, так что мы его не рекомендуем.

К сожалению, однако, операция "himult" в (1) отсутствует во многих языках высокого уровня (см. упр. 3.2.1.1–3). Деление m/k, возможно, наилучшее, когда "himult" отсутствует.

4. $\max(X_1, X_2) \le x$ тогда и только тогда, когда $X_1 \le x$ и $X_2 \le x$; $\min(X_1, X_2) \ge x$ тогда и только тогда, когда $X_1 \ge x$ и $X_2 \ge x$. Вероятность того, что два независимых события происходят одновременно, равна произведению вероятностей этих событий.

5. Получим независимые равномерно распределенные случайные величины U_1 и U_2. Положим $X \leftarrow U_2$. Если $U_1 \ge p$, положим $X \leftarrow \max(X, U_3)$, где U_3 — третья равномерно распределенная величина. Если $U_1 \ge p + q$, положить также $X \leftarrow \max(X, U_4)$, где U_4 — четвертая равномерно распределенная случайная величина. Этот метод можно очевидным образом обобщить для любой случайной величины с полиномиальной функцией распре-

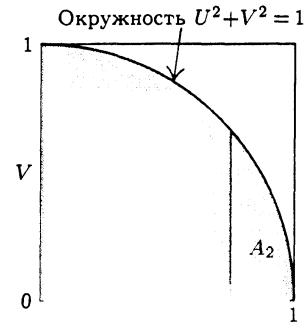


Рис. А–4. "Область принятия" для алгоритма из упр. 6.

деления и даже с функцией распределения, представимой в виде степенного ряда (как показано, например, в алгоритме S, вместо максимизации использующем минимизацию).

Можем поступить и следующим образом (предложение М. Д. Мак-Ларена (M. D. MacLaren)): если $U_1 < p$, присвоить $X \leftarrow U_1/p$; иначе, если $U_1 < p+q$, присвоить $X \leftarrow \max((U_1-p)/q, U_2)$; иначе присвоить $X \leftarrow \max((U_1-p-q)/r, U_2, U_3)$. Этот метод по сравнению с другими требует меньше времени на получение равномерно распределенных случайных чисел, несмотря на то что он требует больше арифметических операций и несколько менее устойчив численно.

6. $F(x) = A_1/(A_1+A_2)$, где A_1 и A_2 — площади, показанные на рис. А–4, так что

$$F(x) = \frac{\int_0^x \sqrt{1-y^2}\,dy}{\int_0^1 \sqrt{1-y^2}\,dy} = \frac{2}{\pi}\arcsin x + \frac{2}{\pi}x\sqrt{1-x^2}.$$

Вероятность окончания процедуры на шаге 2 равна $p = \pi/4$. Процедура продолжается до завершения на шаге 2, поэтому число выполнений шага 2 имеет геометрическое распределение. Характеристиками этого числа являются (min 1, ave $4/\pi$, max ∞, dev $(4/\pi)\sqrt{1-\pi/4}$) согласно упр. 17.

7. Если $k = 1$, тогда $n_1 = n$, и задача тривиальна. В противном случае всегда можно найти $i \neq j$, такое, что $n_i \le n \le n_j$. Заполните B_i n_i кубиками цвета C_i и $n - n_i$ цвета C_j. Затем замените n_j на $n - n_i$ и исключите цвет C_i. Теперь перед нами стоит та же задача, но k меньше на 1. Следовательно, по индукции задача имеет решение.

Следующий алгоритм можно использовать, чтобы подсчитать P и Y для таблиц из (3). Образуем набор пар $(p_1,1)\ldots(p_k,k)$ и рассортируем их по первой компоненте, получив набор $(q_1,a_1)\ldots(q_k,a_k)$, где $q_1 \le \cdots \le q_k$. Присвоим $n \leftarrow k$ и будем повторять следующие операции до тех пор, пока не получим $n = 0$: присвоим $P[a_1-1] \leftarrow kq_1$ и $Y[a_1-1] \leftarrow x_{a_n}$. Удалим (q_1,a_1) и (q_n,a_n), поместим новый элемент $(q_n-(1/k-q_1),a_n)$ на его собственное место в наборе и уменьшим n на 1.

(Если $p_j < 1/k$, то алгоритм никогда не поместит x_j в таблицу для Y. Этот факт безоговорочно используется в алгоритме М. Алгоритм пытается максимизировать вероятность того, что $V < P_K$ в (3), отбирая у самого "богатого" отброшенного элемента и отдавая самому "бедному". Однако очень трудно определить абсолютный минимум этой вероятности, поскольку подобная задача, по крайней мере, так же трудна, как "проблема упаковки корзин"; см. раздел 7.9.)

8. Нужно поменять местами P_j и $(j+P_j)/k$ для $0 \le j < k$.

9. Рассмотрите знак второй производной $f''(x) = \sqrt{2/\pi}\,(x^2-1)e^{-x^2/2}$.

10. Пусть $S_j = (j-1)/5$ для $1 \le j \le 16$ и $p_{j+15} = F(S_{j+1}) - F(S_j) - p_j$ для $1 \le j \le 15$. Пусть также $p_{31} = 1 - F(3)$ и $p_{32} = 0$. (Формула (15) определяет $p_1, \ldots, p_{15}$.) Алгоритм

из упр. 7 можно использовать здесь с $k = 32$ для вычисления P_j и Y_j, после чего получим $1 \le Y_j \le 15$ для $1 \le j \le 32$. Присвоим $P_0 \leftarrow P_{32}$ (которое равно 0) и $Y_0 \leftarrow Y_{32}$. Затем присвоим $Z_j \leftarrow 1/(5 - 5P_j)$ и $Y_j \leftarrow \frac{1}{5}Y_j - Z_j$ для $0 \le j < 32$ и $Q_j \leftarrow 1/(5P_j)$ для $1 \le j \le 15$.

Пусть $h = \frac{1}{5}$ и $f_{j+15}(x) = \sqrt{2/\pi}(e^{-x^2/2} - e^{-j^2/50})/p_{j+15}$ для $S_j \le x \le S_j + h$. Также пусть $a_j = f_{j+15}(S_j)$ для $1 \le j \le 5$, $b_j = f_{j+15}(S_j)$ для $6 \le j \le 15$; $b_j = -hf'_{j+15}(S_j + h)$ для $1 \le j \le 5$ и $a_j = f_{j+15}(x_j) + (x_j - S_j)b_j/h$ для $6 \le j \le 15$, где x_j — корень уравнения $f'_{j+15}(x_j) = -b_j/h$. Наконец присвоим $D_{j+15} \leftarrow a_j/b_j$ для $1 \le j \le 15$, $E_{j+15} \leftarrow 25/j$ для $1 \le j \le 5$ и $E_{j+15} \leftarrow 1/(e^{(2j-1)/50} - 1)$ для $6 \le j \le 15$.

Табл. 1 была подсчитана с использованием следующих промежуточных величин: $(p_1, \ldots, p_{31}) = (.156, .147, .133, .116, .097, .078, .060, .044, .032, .022, .014, .009, .005, .003, .002, .002, .005, .007, .009, .010, .009, .009, .008, .006, .005, .004, .002, .002, .001, .001, .003)$; $(x_6, \ldots, x_{15}) = (1.115, 1.304, 1.502, 1.700, 1.899, 2.099, 2.298, 2.497, 2.697, 2.896)$; $(a_1, \ldots, a_{15}) = (7.5, 9.1, 9.5, 9.8, 9.9, 10.0, 10.0, 10.1, 10.1, 10.1, 10.1, 10.2, 10.2, 10.2, 10.2)$; $(b_1, \ldots, b_{15}) = (14.9, 11.7, 10.9, 10.4, 10.1, 10.1, 10.2, 10.3, 10.4, 10.5, 10.6, 10.7, 10.7, 10.8, 10.9)$.

11. Пусть $g(t) = e^{9/2}te^{-t^2/2}$ для $t \ge 3$. Так как $G(x) = \int_3^x g(t)\,dt = 1 - e^{-(x^2-9)/2}$, случайную величину X с плотностью g можно вычислить, если положить $X \leftarrow G^{[-1]}(1 - V) = \sqrt{9 - 2\ln V}$. Сейчас $e^{-t^2/2} \le (t/3)e^{-t^2/2}$ для $t \ge 3$. Таким образом будет получен обоснованный метод отбраковки, если принять X с вероятностью $f(X)/cg(X) = 3/X$.

12. Справедливо равенство $f'(x) = xf(x) - 1 < 0$ для $x \ge 0$, так как $f(x) = x^{-1} - e^{x^2/2}\int_x^\infty e^{-t^2/2}\,dt/t^2$ для $x > 0$. Пусть $x = a_{j-1}$ и $y^2 = x^2 + 2\ln 2$, тогда

$$\sqrt{2/\pi}\int_y^\infty e^{-t^2/2}\,dt = \tfrac{1}{2}\sqrt{2/\pi}\,e^{-x^2/2}f(y) < \tfrac{1}{2}\sqrt{2/\pi}\,e^{-x^2/2}f(x) = 2^{-j}.$$

Следовательно, $y > a_j$.

13. Возьмем $b_j = \mu_j$; рассмотрим сейчас задачу с $\mu_j = 0$ для каждого j. В матричных обозначениях, если $Y = AX$, где $A = (a_{ij})$, необходимо, чтобы выполнялось $AA^T = C = (c_{ij})$. (В других обозначениях, если $Y_j = \sum a_{jk}X_k$, среднее значение Y_iY_j равно $\sum a_{ik}a_{jk}$.) Если это матричное уравнение может быть решено для A, то оно может быть решено и тогда, когда A — треугольная матрица, так как $A = BU$ для некоторой ортогональной матрицы U и некоторой треугольной матрицы B, а $BB^T = C$. Требуемое решение в виде треугольной матрицы может быть получено в результате решения уравнений $a_{11}^2 = c_{11}$, $a_{11}a_{21} = c_{12}$, $a_{21}^2 + a_{22}^2 = c_{22}$, $a_{11}a_{31} = c_{13}$, $a_{21}a_{31} + a_{22}a_{32} = c_{23}$, ... последовательно для a_{11}, a_{21}, a_{22}, a_{31}, a_{32} и т. д. [*Замечание.* Ковариационная матрица должна быть неотрицательно определена, так как среднее значение величины $\left(\sum y_jY_j\right)^2$ равно сумме $\sum c_{ij}y_iy_j$, которая должна быть неотрицательна. И решение всегда существует, когда C неотрицательно определена, так как $C = U^{-1}\mathrm{diag}(\lambda_1, \ldots, \lambda_n)U$, где собственные числа λ_j неотрицательны, и $U^{-1}\mathrm{diag}(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_n})U$ и будет решением.]

14. $F(x/c)$, если $c > 0$; ступенчатая функция $[x \ge 0]$, если $c = 0$ или $1 - F(x/c)$, если $c < 0$.

15. Функция распределения — $\int_{-\infty}^\infty F_1(x - t)\,dF_2(t)$. Плотность — $\int_{-\infty}^\infty f_1(x - t)f_2(t)\,dt$. Это называется *сверткой* заданных распределений.

16. Ясно, что, как и требуется $f(t) \le cg(t)$ для всех t. Так как $\int_0^\infty g(t)\,dt = 1$, то $g(t) = Ct^{a-1}$ для $0 \le t < 1$, Ce^{-t} для $t \ge 1$, где $C = ae/(a+e)$. Случайную величину с плотностью g легко получить, как смесь распределений $G_1(x) = x^a$ для $0 \le x < 1$ и $G_2(x) = 1 - e^{1-x}$ для $x \ge 1$.

G1. [Инициализация.] Присвоить $p \leftarrow e/(a + e)$. (Это вероятность того, что используется распределение G_1.)

G2. [Генерирование случайной величины с распределением G.] Генерировать независимые равномерно распределенные случайные величины U и V, где $V \ne 0$. Если

$U < p$, присвоить $X \leftarrow V^{1/a}$ и $q \leftarrow e^{-X}$; иначе — присвоить $X \leftarrow 1 - \ln V$ и $q \leftarrow X^{a-1}$. (Сейчас X имеет плотность g, а $q = f(X)/cg(X)$.)

G3. [Отбросить?] Генерировать новую равномерно распределенную случайную величину U. Если $U \geq q$, возвратиться к шагу G2. ▮

Среднее число итераций равно $c = (a + e)/(e\Gamma(a + 1)) < 1.4$.

Существует несколько способов улучшения этой процедуры. Можно заменить V случайной величиной Y, имеющей показательное распределение со средним 1, которая генерируется, допустим, с помощью алгоритма S, а затем присвоить $X \leftarrow e^{-Y/a}$ или $X \leftarrow 1 + Y$ в обоих случаях. Более того, если присвоить $q \leftarrow pe^{-X}$ в первом случае и $q \leftarrow p + (1 - p)X^{a-1}$ во втором, то можно использовать первоначальную случайную величину U вместо того, чтобы снова генерировать ее на шаге G3. Наконец, если $U < p/e$, можно принять $V^{1/a}$ немедленно, не расходуя на вычисление q около 30% времени.

17. (a) $F(x) = 1 - (1 - p)^{\lfloor x \rfloor}$ для $x \geq 0$. (b) $G(z) = pz/(1 - (1 - p)z)$. (c) Среднее равно $1/p$, среднеквадратичное отклонение равно $\sqrt{1-p}/p$. Для дальнейших вычислений заметим, что если $H(z) = q + (1 - q)z$, то $H'(1) = 1 - q$ и $H''(1) + H'(1) - (H'(1))^2 = q(1 - q)$, поэтому среднее и дисперсия $1/H(z)$ равны $q-1$ и $q(q-1)$ соответственно (см. раздел 1.2.10). В этом случае $q = 1/p$, дополнительный множитель z в знаменателе $G(z)$ добавляет к среднему 1.

18. Присвоить $N \leftarrow N_1 + N_2 - 1$, где N_1 и N_2 — независимые случайные величины, имеющие геометрическое распределение с параметром p. (Рассмотрите производящую функцию.)

19. Присвоить $N \leftarrow N_1 + \cdots + N_t - t$, где N_j (независимые. — *Прим. ред.*) — случайные величины, имеющие геометрическое распределение с параметром p. (Это число неудач перед t-м успехом, когда осуществляется последовательность независимых испытаний, каждое из которых приводит к успеху с вероятностью p.)

Для $t = p = \frac{1}{2}$ и вообще, когда среднее значение (т. е. $t(1 - p)/p$) распределения мало́, можно упростить вычисление вероятностей $p_n = \binom{t-1+n}{n} p^t (1 - p)^n$ последовательно для $n = 0, 1, 2, \ldots$, как показано в следующем алгоритме.

B1. [Инициализация.] Присвоить $N \leftarrow 0$, $q \leftarrow p^t$, $r \leftarrow q$ и генерировать равномерно распределенную случайную величину U. (Будет выполняться $q = p_N$ и $r = p_0 + \cdots + p_N$ на протяжении всего алгоритма, выполнение которого остановится, как только получим $U < r$.)

B2. [Итерация.] Если $U \geq r$, присвоить $N \leftarrow N + 1$, $q \leftarrow q(1-p)(t-1+N)/N$, $r \leftarrow r + q$ и повторить этот шаг. Иначе — возврат N и конец. ▮

[Интересный метод для отрицательного биномиального распределения с произвольно большим действительным значением t предложил Р. Леже (R. Léger). Сначала нужно генерировать случайную величину X, имеющую гамма-распределение порядка t, а затем положить N равной случайной величине с пуассоновским распределением со средним, равным $X(1 - p)/p$.]

20. $\mathrm{R1} = 1 + (1 - A/R) \cdot \mathrm{R1}$. Когда выполнен шаг R2, алгоритм завершается с вероятностью I/R; когда выполнен шаг R3, переход к шагу R1 происходит с вероятностью E/R. Справедливо следующее.

R1	R/A	R/A	R/A	R/A
R2	0	R/A	0	R/A
R3	0	0	R/A	$R/A - I/A$
R4	R/A	$R/A - I/A$	$R/A - E/A$	$R/A - I/A - E/A$

21. $R = \sqrt{8/e} \approx 1.71553$; $A = \sqrt{\pi/2} \approx 1.25331$. Так как

$$\int u\sqrt{a-bu}\,du = (a-bu)^{3/2}\left(\tfrac{2}{5}(a-bu)-\tfrac{2}{3}\right)/b^2,$$

получим $I = \int_0^{a/b} u\sqrt{a-bu}\,du = \frac{4}{15}a^{5/2}/b^2$, где $a = 4(1+\ln c)$ и $b = 4c$. Когда $c = e^{1/4}$, I принимает максимальное значение $\frac{5}{6}\sqrt{5/e} \approx 1.13020$. Наконец, чтобы вычислить E, понадобятся следующие формулы для интегрирования:

$$\int\sqrt{bu-au^2}\,du = \tfrac{1}{8}b^2a^{-3/2}\arcsin(2ua/b-1) + \tfrac{1}{4}ba^{-1}\sqrt{bu-au^2}\,(2ua/b-1),$$

$$\int\sqrt{bu+au^2}\,du = -\tfrac{1}{8}b^2a^{-3/2}\ln(\sqrt{bu+au^2}+u\sqrt{a}+b/2\sqrt{a}) + \tfrac{1}{4}ba^{-1}\sqrt{bu+au^2}\,(2ua/b+1),$$

где $a, b > 0$. Пусть на шаге R3 выполняется проверка "$X^2 \geq 4e^{x-1}/U - 4x$", тогда внешняя область попадет в верхнюю часть прямоугольника, когда $u = r(x) = (e^x - \sqrt{e^{2x}-2ex})/2ex$. ($r(x)$ случайно принимает максимальное значение в точке $x = 1/2$, в которой $r(x)$ *не* дифференцируема!) Справедливо $E = \int_0^{r(x)}(\sqrt{2/e} - \sqrt{bu-au^2})\,du$, где $b = 4e^{x-1}$ и $a = 4x$. E принимает максимальное значение возле точки $x = -.35$, где оно приближенно равно $E \approx .29410$.

22. (Решение Дж. Марсальи (G. Marsaglia).) Рассмотрим "непрерывное пуассоновское распределение", определенное следующим образом: $G(x) = \int_\mu^\infty e^{-t}t^{x-1}\,dt/\Gamma(x)$ для $x > 0$. Если X имеет это распределение, то и $\lfloor X \rfloor$ имеет пуассоновское распределение, так как $G(x+1) - G(x) = e^{-\mu}\mu^x/x!$. Если μ большое, G приближенно нормально. Следовательно, $G^{[-1]}(F_\mu(x))$ приближенно линейно, когда $F_\mu(x)$ — функция распределения нормальной случайной величины со средним и дисперсией, равными μ, т. е. $F_\mu(x) = F((x-\mu)/\sqrt{\mu})$, где $F(x)$ — функция нормального распределения (10). Пусть $g(x)$ — реально вычисляемая функция, такая, что $|G^{[-1]}(F_\mu(x)) - g(x)| < \epsilon$ для $-\infty < x < \infty$. Сейчас можно эффективно генерировать пуассоновскую случайную величину следующим образом: генерировать нормальную случайную величину X и присвоить $Y \leftarrow g(\mu + \sqrt{\mu}\,X)$, $N \leftarrow \lfloor Y \rfloor$, $M \leftarrow \lfloor Y + \frac{1}{2} \rfloor$. Затем, если $|Y - M| > \epsilon$, получаем на выходе N, иначе на выходе будет $M - [G^{[-1]}(F(X)) < M]$.

Этот подход применим также к биномиальному распределению с

$$G(x) = \int_p^1 u^{x-1}(1-u)^{n-x}\,du\;\frac{\Gamma(t+1)}{\Gamma(x)\,\Gamma(t+1-x)},$$

поскольку $\lfloor G^{[-1]}(U) \rfloor$ — величина, имеющая биномиальное распределение с параметрами (t, p) и G приближенно нормально.

[См. также альтернативный метод, предложенный Аренсом и Дитером (Ahrens and Dieter, *Computing* **25** (1980), 193–208).]

23. Да. Второй метод дает распределение $|\cos 2\theta|$, где θ — равномерно распределенная случайная величина между 0 и $\pi/2$. (Предполагается, что $U = r\cos\theta$, $V = r\sin\theta$.)

25. $\frac{21}{32} = (.10101)_2$. Обычно двоичное представление формируется с использованием 1 для $\vee$ и 0 — для $\wedge$ слева направо, затем прибавляется 1. Этот метод [см. (Точер К. Д.) K. D. Tocher, *J. Roy. Stat. Soc.* **B16** (1954), 49] может привести к эффективному генерированию независимых двоичных разрядов с заданной вероятностью p, а также использоваться при генерировании случайных величин с геометрическим и биномиальным распределением.

26. (a) Верно: $\sum_k \Pr(N_1 = k)\Pr(N_2 = n-k) = e^{-\mu_1-\mu_2}(\mu_1+\mu_2)^n/n!$. (b) Неверно, поскольку, если $\mu_2 \neq 0$, $N_1 - N_2$ может быть отрицательным.

27. Пусть двоичное представление p имеет вид $(.b_1b_2b_3\ldots)_2$. Поступим далее в соответствии со следующими правилами.

B1. [Инициализация.] Присвоить $m \leftarrow t$, $N \leftarrow 0$, $j \leftarrow 1$. (В этом алгоритме m обозначает количество моделируемых равномерно распределенных величин, для которых соотношение с p еще неизвестно, так как их старшие $j-1$-двоичные разряды совпадают с этими же разрядами числа p, N — число моделируемых случайных величин, о которых известно, что они меньше p.)

B2. [Взгляд на следующий столбец двоичных разрядов.] Генерировать случайное целое число M с биномиальным распределением $(m, \frac{1}{2})$. (Сейчас M означает число неизвестных случайных величин, таких, что их j-й разряд не совпадает с b_j.) Присвоить $m \leftarrow m - M$, если $b_j = 1$, то присвоить $N \leftarrow N + M$.

B3. [Сделано?] Если $m = 0$ или если остающиеся двоичные разряды $(.b_{j+1}b_{j+2}\ldots)_2$ p все равны нулю, алгоритм завершен. Иначе — присвоить $j \leftarrow j+1$ и возвратиться к шагу B2. ▮

[Когда $b_j = 1$ для бесконечного числа j, среднее число итераций A_t удовлетворяет равенствам

$$A_0 = 0; \qquad A_n = 1 + \frac{1}{2^n}\sum_k \binom{n}{k} A_k \quad \text{для } n \geq 1.$$

Положив $A(z) = \sum A_n z^n/n!$, получим $A(z) = e^z - 1 + A(\frac{1}{2}z)e^{z/2}$ Поэтому $A(z)e^{-z} = 1 - e^{-z} + A(\frac{1}{2}z)e^{-z/2} = \sum_{k\geq 0}(1 - e^{-z/2^k}) = 1 - e^{-z} - \sum_{n\geq 1}(-z)^n/(n!(2^n - 1))$ и

$$A_m = 1 + \sum_{k\geq 1}\binom{n}{k}\frac{(-1)^{k+1}}{2^k - 1} = 1 + \frac{V_{n+1}}{n+1} = \lg n + \frac{\gamma}{\ln 2} + \frac{1}{2} + f_0(n) + O(n^{-1})$$

в обозначениях упр. 5.2.2–48.]

28. Генерируем случайную точку $(y_1, \ldots, y_n)$ на единичной сфере и предположим, что $\rho = \sqrt{\sum a_k y_k^2}$. Генерируем независимую равномерно распределенную случайную величину U. Если $\rho^{n+1}U < K\sqrt{\sum a_k^2 y_k^2}$, то на выходе получится точка $(y_1/\rho, \ldots, y_n/\rho)$; в остальных случаях начинаем сначала. Здесь $K^2 = \min\{(\sum a_k y_k^2)^{n+1}/(\sum a_k^2 y_k^2) \mid \sum y_k^2 = 1\} = a_n^{n-1}$, если $na_n \geq a_1$, $((n+1)/(a_1 + a_n))^{n+1}(a_1 a_n/n)^n$ — в остальных случаях.

29. Предположим, что $X_{n+1} = 1$, затем присвоим $X_k \leftarrow X_{k+1}U_k^{1/k}$ или $X_k \leftarrow X_{k+1}e^{-Y_k/k}$ для $k = n, n-1, \ldots, 1$, где U_k — равномерно распределенная случайная величина или Y_k — случайная величина с показательным распределением. [*ACM Trans. Math. Software* **6** (1980), 359–364. Этот метод введен в употребление в 60-х годах Давидом Сенешолом (David Seneschol); см. работу *Amer. Statistician* **26**, 4 (October, 1972), 56–57. Альтернатива состоит в генерировании n равномерно распределенных случайных чисел и их сортировке наиболее быстрым методом. Предложенный здесь метод является особенно полезным, если только требуется несколько наибольших или наименьших X_j. Заметим, что $(F^{[-1]}(X_1), \ldots, F^{[-1]}(X_n))$ соответствуют упорядоченным случайным величинам с функцией распределения F.]

30. Генерировать случайные числа $Z_1 = -\mu^{-1}\ln U_1$, $Z_2 = Z_1 - \mu^{-1}\ln U_2$, ... до тех пор, пока не выполнится $Z_{m+1} \geq 1$. На выходе получим $(X_j, Y_j) = f(Z_j)$ для $1 \leq j \leq m$, где $f((.b_1b_2\ldots b_{2r})_2) = ((.b_1b_2\ldots b_r)_2, (.b_{r+1}b_{r+2}\ldots b_{2r})_2)$. Если менее старшие двоичные разряды значительно менее случайны, чем более старший двоичный разряд, то надежнее (но медленнее) положить, что $f((.b_1b_2\ldots b_{2r})_2) = ((.b_1b_3\ldots b_{2r-1})_2, (.b_2b_4\ldots b_{2r})_2)$.

31. (a) Достаточно рассмотреть случай, когда $k = 2$, так как $a_1X_1 + \cdots + a_kX_k = X\cos\theta + Y\sin\theta$ при $X = X_1$, $\cos\theta = a_1$ и $Y = (a_2X_2 + \cdots + a_kX_k)/\sin\theta$. И справедливо равенство

$$\Pr(X\cos\theta + Y\sin\theta \le x) = \frac{1}{2\pi}\int_{s,t} e^{-s^2/2-t^2/2}ds\,dt\,[s\cos\theta + t\sin\theta \le x]$$
$$= \frac{1}{2\pi}\int_{u,v} e^{-u^2/2-v^2/2}du\,dv\,[u \le x] = (10)$$

после подстановки $u = s\cos\theta + t\sin\theta$, $v = -s\sin\theta + t\cos\theta$.

(b) Существуют числа $\alpha > 1$ и $\beta > 1$, такие, что $(\alpha^{-24} + \alpha^{-55})/\sqrt{2} = 1$ и $\frac{3}{5}\beta^{-24} + \frac{4}{5}\beta^{-55} = 1$, поскольку числа X_n растут экспоненциально по n согласно свойствам линейных рекуррентных соотношений.

Если отказаться от формы линейного рекуррентного соотношения, скажем, используя рекуррентное соотношение $X_n = X_{n-24}\cos\theta_n + X_{n-55}\sin\theta_n$, где θ_n выбрано равномерно в $[0\,..\,2\pi)$, можно получить подходящий результат, но потребуется выполнить намного больше вычислений.

(c) Начните, пожалуй, с 2 048 нормальных случайных величин $X_0, \ldots, X_{1023}, Y_0, \ldots, Y_{1023}$. После использования около 1/3 из них генерируйте еще 2 048 случайных величин следующим образом. Выберите целые числа a, b, c и d независимо в $[0\,..\,1024)$, причем a и c должны быть нечетными. Затем присвойте

$$X'_j \leftarrow X_{(aj+b) \bmod 1024}\cos\theta + Y_{(cj+d) \bmod 1024}\sin\theta,$$
$$Y'_j \leftarrow -X_{(aj+b) \bmod 1024}\sin\theta + Y_{(cj+d) \bmod 1024}\cos\theta$$

для $0 \le j < 1024$, где $\cos\theta$ и $\sin\theta$ — случайные отношения $(U^2 - V^2)/(U^2 + V^2)$ и $2UV/(U^2 + V^2)$, выбранные, как в упр. 23. Можно отбросить U и V, кроме случаев, когда $|\cos\theta| \ge \frac{1}{2}$ и $|\sin\theta| \ge \frac{1}{2}$. 2 048 новых случайных величин заменят старые. Заметим, что для получения новой случайной величины необходимо выполнить лишь несколько операций.

Этот метод не расходится подобно последовательностям, содержащимся в (b), так как сумма квадратов $\sum(X_j^2 + Y_j^2) = \sum((X'_j)^2 + (Y'_j)^2)$ остается постоянной со значением $S \approx 2048$, исключая незначительную ошибку округления. С другой стороны, постоянство S на самом деле является недостатком метода, поскольку сумма квадратов действительно должна иметь χ^2-распределение с 2 048 степенями свободы. Чтобы преодолеть возникшую проблему, следовало бы использовать не нормальные случайные величины X_j, а αX_j, где $\alpha^2 = \frac{1}{2}(Y_{1023} + \sqrt{4095})^2/S$ является заранее вычисленным нормирующим множителем. (Величина $\frac{1}{2}(Y_{1023} + \sqrt{4095})^2$ будет приемлемо приближать требуемую χ^2 случайную величину.)

Литература. C. S. Wallace, *ACM Trans. on Math. Software* **22** (1996), 119–127; R. P. Brent, *Lecture Notes in Comp. Sci.* **1470** (1998), 1–20.

32. (a) Преобразование $(X', Y') = f(X, Y)$ является взаимно однозначным соответствием преобразованию множества $\{x, y \ge 0\}$ самого в себя, такого, что $x' + y' = x + y$ и $dx'\,dy' = dx\,dy$. Получим

$$\frac{X'}{X'+Y'} = \left(\frac{X}{X+Y} - \lambda\right) \bmod 1, \qquad \frac{Y'}{X'+Y'} = \left(\frac{Y}{X+Y} + \lambda\right) \bmod 1.$$

(b) Это преобразование является таким соответствием (two-to-one) от двух к одному, что $x' + y' = x + y$ и $dx'\,dy' = 2\,dx\,dy$.

(c) Достаточно рассмотреть "j-транспонированное" преобразование

$$X' = (\ldots x_{j+2}x_{j+1}x_jy_{j-1}y_{j-2}y_{j-3}\ldots)_2,$$
$$Y' = (\ldots y_{j+2}y_{j+1}y_jx_{j-1}x_{j-2}x_{j-3}\ldots)_2$$

для фиксированных целых j и затем составить j-транспонирования для $j = 0, 1, -1, 2, -2, \ldots$, принимая во внимание, что совместные вероятностные распределения X' и Y' сходятся при $|j| \to \infty$. Каждое j-транспонирование является взаимно однозначным с операциями $x' + y' = x + y$ и $dx'\,dy' = dx\,dy$.

33. Использовать U_1 как начальное значение для *других* генераторов случайных чисел (возможен линейный конгруэнтный генератор с другим множителем); в качестве множителя взять одну из $U_2, U_3, \ldots$.

РАЗДЕЛ 3.4.2

1. Существует $\binom{N-t}{n-m}$ способов выбора $n-m$ записей из последних $N-t$ и $\binom{N-t-1}{n-m-1}$ способов выбора $n-m-1$ записей из $N-t-1$ после выбора $(t+1)$-й записи.

2. Переход от шага S3 к шагу S5 невозможен, если количество записей, которые осталось проверить, равно $n-m$.

3. Не будем путать условную и безусловную вероятности. Значение m зависит от случайного выбора первых t элементов. Если взять среднее по всем возможным выборам, которые могут появиться среди этих элементов, то можно найти, что $(n-m)/(N-t)$ в среднем точно равно n/N. Например, рассмотрим второй элемент. Если первый элемент отобран в выборку (это происходит с вероятностью n/N), то второй элемент выбирается с вероятностью $(n-1)/(N-1)$; если первый элемент не выбирается, то второй выбирается с вероятностью $n/(N-1)$. Полная вероятность выбора второго элемента равна $(n/N)((n-1)/(N-1)) + (1-n/N)(n/(N-1)) = n/N$.

4. Из алгоритма следует, что

$$p(m,t+1) = \Big(1 - \frac{n-m}{N-t}\Big)\,p(m,t) + \frac{n-(m-1)}{N-t}\,p(m-1,t).$$

Требуемая формула может быть доказана индукцией по t, в частности $p(n,N) = 1$.

5. В обозначениях упр. 4 вероятность того, что $t = k$, по окончании работы алгоритма равна $q_k = p(n,k) - p(n,k-1) = \binom{k-1}{n-1}/\binom{N}{n}$. Среднее равно $\sum_{k=0}^{N} kq_k = (N+1)n/(n+1)$.

6. Так же, как в упр. 5, получим $\sum_{k=0}^{N} k(k+1)q_k = (N+2)(N+1)n/(n+2)$; дисперсия, следовательно, равна $(N+1)(N-n)n/(n+2)(n+1)^2$.

7. Предположим, есть выбор $1 \le x_1 < x_2 < \cdots < x_n \le N$. Пусть $x_0 = 0$, $x_{n+1} = N+1$. Выбор получен с вероятностью $p = \prod_{1\le t\le N} p_t$, где

$$p_t = \begin{cases} (N-(t-1)-n+m)/(N-(t-1)) & \text{для } x_m < t < x_{m+1};\\ (n-m)/(N-(t-1)) & \text{для } t = x_{m+1}. \end{cases}$$

Знаменатель произведения p_t равен $N!$, числитель содержит члены $N-n, N-n-1, \ldots, 1$ для тех t_j, которые не равны x_j, а члены $n, n-1, \ldots, 1$ для тех t_k, которые *равны* x_k. Следовательно, $p = (N-n)!\,n!/N!$.

Пример. $n = 3$, $N = 8$, $(x_1, x_2, x_3) = (2, 3, 7)$; $p = \frac{5}{8}\frac{3}{7}\frac{2}{6}\frac{4}{5}\frac{3}{4}\frac{2}{3}\frac{1}{2}\frac{1}{1}$.

8. (a) $p(0,k) = \binom{N-k}{n}/\binom{N}{n} = \binom{N-n}{k}/\binom{N}{k}$ из $\binom{N}{n}$ выборок, если пропустить первые k записей.

(b) Присвоить $X \leftarrow k-1$, где k является минимальным с $U \ge \Pr(X \ge k)$. Затем выполнять присвоения $X \leftarrow 0$, $p \leftarrow N-n$, $q \leftarrow N$, $R \leftarrow p/q$ и т. д. до тех пор, пока $U < R$. Присвоить $X \leftarrow X+1$, $p \leftarrow p-1$, $q \leftarrow q-1$, $R \leftarrow Rp/q$. (Этот метод хорош, когда n/N, скажем, $\ge 1/5$. Можно предположить, что $n/N \le 1/2$, иначе лучше выбрать $N-n$ *невыбранных* записей.)

(c) $\Pr(\min(Y_N, \ldots, Y_{N-n+1}) \ge k) = \prod_{j=0}^{n-1} \Pr(Y_{N-j} \ge k) = \prod_{j=0}^{n-1} (N-j-k)/(N-j)$. (Этот метод хорош, если, скажем, $n \le 5$.)

(d) (См. упр. 3.4.1–29.) Значение $X \leftarrow \lfloor N(1 - U^{1/n}) \rfloor$ требуется отбросить с вероятностью, равной только $O(n/N)$. Точные детали тщательно проработаны в *CACM* **27** (1984), 703–718, а практическое осуществление приведено в *ACM Trans. Math. Software* **13** (1987), 58–67. (Этот метод хорош, когда, скажем, $5 < n < \frac{1}{5}N$.)

После пропуска X записей и выбора следующих присвоим $n \leftarrow n - 1$, $N \leftarrow N - X - 1$ и будем повторять процесс до тех пор, пока $n = 0$. Подобное приближение ускоряет метод резервуара [см. *ACM Trans. Math. Software* **11** (1985), 37–57].

9. В резервуар будет помещено семь записей: 1, 2, 3, 5, 9, 13, 16. В окончательной выборке содержатся записи 2, 5, 16.

10. Удалить шаг R6 и переменную m. Заменить таблицу I таблицей записей, инициализированных первых n записей на шаге R1, и, заменив M-е значение таблицы, перейти в алгоритме к шагу R4.

11. Поступая, как в разделе 1.2.10, в котором рассматривался частный случай, когда $n = 1$, получаем, что производящая функция равна

$$G(z) = z^n\Big(\frac{1}{n+1} + \frac{n}{n+1}z\Big)\Big(\frac{2}{n+2} + \frac{n}{n+2}z\Big)\ldots\Big(\frac{N-n}{N} + \frac{n}{N}z\Big).$$

Среднее значение равно $n + \sum_{n<t\le N}(n/t) = n(1 + H_N - H_n)$, а дисперсия оказывается равной $n(H_N - H_n) - n^2(H_N^{(2)} - H_n^{(2)})$.

12. Заметим, что $\pi^{-1} = (b_t t)\ldots(b_3 3)(b_2 2)$, поэтому находим алгоритм, который переводит представление π в π^{-1}. Присвоить $b_j \leftarrow j$ для $1 \le j \le t$. Затем для $j = 2, 3, \ldots, t$ (в таком порядке) произвести взаимный обмен $b_j \leftrightarrow b_{a_j}$. Наконец для $j = t, \ldots, 3, 2$ (в таком порядке) присвоить $b_{a_j} \leftarrow b_j$. (Алгоритм основывается на том факте, что $(a_t t)\pi_1 = \pi_1(b_t t)$.)

13. Перенумеровав колоду $0, 1, \ldots, 2n-2$, находим, что номер $(2x) \bmod (2n-1)$ s раз присвоен карте с номером x, в то время как номер $(x+1) \bmod (2n-1)$ c раз присвоен карте с номером x. Получим (c следует из s) $= cs = sc^2$. Значит, произведение любого количества c и s можно привести к виду $s^i c^k$. Также $2^{\varphi(2n-1)} \equiv 1$ по модулю $(2n-1)$. Поскольку $s^{\varphi(2n-1)}$ и c^{2n-1} — тождественные перестановки, то возможно не более $(2n-1)\varphi(2n-1)$ компоновок. (*Точное* число различных компоновок равно $(2n-1)k$, где k равно порядку 2 по модулю $(2n-1)$. Для случая, когда $s^k = c^j$, c^j устанавливает карту 0, $s^k = c^j =$ тождество.) Дополнительные детали приводятся в *SIAM Review* **3** (1961), 293–297.

14. (a) $\begin{smallmatrix}9\\2\end{smallmatrix}$. Это можно установить, не обращая внимания на то, где именно была сдвинута карта, кроме случая, когда карта была взята из первых трех или последних двух позиций. (b) $\begin{smallmatrix}5\\\diamond\end{smallmatrix}$. Три разрезания и перетасовки приведут к перемешиванию самое большее восьми циклически возрастающих подпоследовательностей $a_{x_j} a_{(x_j+1) \bmod n} \cdots a_{(x_{j+1}-1) \bmod n}$. Значит, подпоследовательности $\begin{smallmatrix}6\\\diamond\end{smallmatrix}$ $\begin{smallmatrix}5\\\diamond\end{smallmatrix}$ $\begin{smallmatrix}4\\\diamond\end{smallmatrix}$ заблокированы. [Несколько магических трюков основано на том факте, что три разрезания и перетасовки являются совсем неслучайными; см. Martin Gardner, *Mathematical Magic Show* (Knopf, 1977), гл. 7.]

15. Присвоить $Y_j \leftarrow j$ для $t - n < j \le t$. Затем для $j = t, t-1, \ldots, t-n+1$ проделать следующие операции. Присвоить $k \leftarrow \lfloor jU \rfloor + 1$. Если $k > t - n$, присвоить $X_j \leftarrow Y_k$ и $Y_k \leftarrow Y_j$, иначе, если $k = X_i$ для некоторых $i > j$ (можно использовать таблицу идентификаторов алгоритма), присвоить $X_j \leftarrow Y_i$ и $Y_i \leftarrow Y_j$; иначе — присвоить $X_j \leftarrow k$. (Идея основана на предположении,что $Y_{t-n+1}, \ldots, Y_j$ представляют $X_{t-n+1}, \ldots, X_j$, и, если $i > j$ и $X_i \le t - n$, на предположении, что Y_i представляют X_{X_i} в исполнении алгоритма P. Интересно доказать правильность алгоритма Дахла. Алгоритм основан на наблюдении, что на шаге P2 из $X_k \ne k$ следует, что $X_k > j$ для $1 \le k \le j$.)

16. Можно предположить, что $n \le \frac{1}{2}N$, иначе достаточно найти $N - n$ элементов, *не* входящих в выборку. Идея заключается в том, чтобы, используя таблицу случайных данных размерности $2n$, генерировать случайные числа между 1 и N, хранить их в таблице и выбрасывать дубликаты до тех пор, пока не будут сгенерированы n различных чисел. Среднее число генерируемых случайных чисел равно $N/N + N/(N-1) + \cdots + N/(N-n+1) < 2n$ согласно упр. 3.3.2–10, а среднее время обработки каждого числа равно $O(1)$. Требуется получить выходные результаты в порядке возрастания, что можно сделать следующим образом. Если использовать таблицу упорядоченных случайных данных (упр. 6.4–66) с линейным зондированием, таблица случайных данных сформируется, как только значения будут включаться в порядке возрастания, и общее среднее число проб будет меньше $\frac{5}{2}n$. Так, если использовать монотонные случайные адреса, например $\lfloor 2n(k-1)/N \rfloor$, для ключа k, получится вывод ключей в упорядоченном виде в результате самое большее двух просмотров таблицы. [См. *CACM* **29** (1986), 366–367.]

17. Покажите по индукции перед шагом j, что множество S является случайной выборкой $j - N - 1 + n$ целых чисел из $\{1, \ldots, j-1\}$. [*CACM* **30** (1987), 754–757. Метод Флойда может быть использован для ускорения выполнения упр. 16. Это, по существу, двойной алгоритм Дахла из упр. 15, который оперирует *убывающими* значениями j; см. упр. 12.]

РАЗДЕЛ 3.5

1. b-ичная последовательность — да (см. упр. 2); последовательность $[0\,..\,1)$ — нет (так как предполагается только конечное множество значений элементов).

2. Последовательность 1- и 2-распределенная, только не 3-распределенная (двоичное число 111 никогда не появляется).

3. Повторите последовательность из упр. 3.2.2–17 с периодом длиной 27.

4. Если $\nu_1(n)$, $\nu_2(n)$, $\nu_3(n)$, $\nu_4(n)$ считать соответствующими четырем вероятностям, то получим $\nu_1(n) + \nu_2(n) = \nu_3(n) + \nu_4(n)$ для всех n. Так что требуемый результат вытекает из сложения пределов.

5. Последовательность начинается так: $\frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}$ и т. д. Когда $n = 1, 3, 7, 15, \ldots$, получим $\nu(n) = 1, 1, 5, 5, \ldots$, так что $\nu(2^{2k-1} - 1) = \nu(2^{2k} - 1) = (2^{2k}-1)/3$. Значит, $\nu(n)/n$ колеблется между $\frac{1}{3}$ и приблизительно $\frac{2}{3}$ и предел не существует. Вероятность не определена. [Методы из раздела 4.2.4 показывают, однако, что численное значение *может* быть определено так: $\Pr(U_n < \frac{1}{2}) = \Pr($старший разряд представления $n+1$ в системе счисления с основанием 4 равен 1), т. е. $\log_4 2 = \frac{1}{2}$.]

6. По индукции и согласно упр. 5

$$\Pr(S_j(n) \text{ для некоторого } j,\ 1 \le j \le k) = \sum_{j=1}^{k} \Pr(S_j(n)).$$

Когда $k \to \infty$, последняя является монотонной последовательностью, ограниченной 1, так что она сходится и

$$\underline{\Pr}(S_j(n) \text{ для некоторых } j \ge 1) \ge \sum_{j=1}^{k} \Pr(S_j(n))$$

для всех k. В качестве контрпримера, показывающего, что равенство будет не всегда, не трудно устроить так, что $S_j(n)$ будет всегда верно для *некоторых* j, однако $\Pr(S_j(n)) = 0$ для *всех* j.

7. Пусть $p_i = \sum_{j\ge1} \Pr(S_{ij}(n))$. Результат предыдущего упражнения можно обобщить так: $\underline{\Pr}(S_j(n)$ для некоторого $j \ge 1) \ge \sum_{j\ge1} \underline{\Pr}(S_j(n))$ для *любых* непересекающихся утверждений $S_j(n)$. Так что получим $1 = \Pr(S_{ij}(n)$ для некоторых $i, j \ge 1) \ge \sum_{i\ge1} \underline{\Pr}(S_{ij}(n)$для некоторого $j \ge 1) \ge \sum_{i\ge1} p_i = 1$ и, следовательно, $\underline{\Pr}(S_{ij}(n)$ для некоторого $j \ge 1) = p_i$. Зададим $\epsilon > 0$; пусть I достаточно велико, так, что $\sum_{i=1}^{I} p_i \ge 1-\epsilon$. Пусть

$$\phi_i(N) = (\text{число } n < N \text{ с } S_{ij}(n) \text{ справедливых для некоторого } j \ge 1)/N.$$

Очевидно, что $\sum_{i=1}^{I} \phi_i(N) \le 1$, и для всех достаточно больших N получим $\sum_{i=2}^{I} \phi_i(N) \ge \sum_{i=2}^{I} p_i - \epsilon$; следовательно, $\phi_1(N) \le 1 - \phi_2(N) - \cdots - \phi_I(N) \le 1 - p_2 - \cdots - p_I + \epsilon \le 1 - (1 - \epsilon - p_1) + \epsilon = p_1 + 2\epsilon$. Это доказывает, что $\overline{\Pr}(S_{1j}(n)$ для некоторого $j \ge 1) \le p_1 + 2\epsilon$. Значит, $\Pr(S_{1j}(n)$ для некоторого $j \ge 1) = p_1$ и требуемый результат получается для $i = 1$. Из симметрии гипотез следует, что он справедлив для любого значения i.

8. Сложите вероятности для j, $j+d$, $j+2d$, $\ldots$, $m+j-d$ в определении E.

9. $\limsup_{n\to\infty}(a_n + b_n) \le \limsup_{n\to\infty} a_n + \limsup_{n\to\infty} b_n$; отсюда найдем, что

$$\limsup_{n\to\infty}((y_{1n}-\alpha)^2 + \cdots + (y_{mn}-\alpha)^2) \le m\alpha^2 - 2m\alpha^2 + m\alpha^2 = 0,$$

и это может происходить только тогда, когда каждая $(y_{jn} - \alpha)$ стремится к нулю.

10. В оценке суммы в равенстве (22).

11. $\langle U_{2n}\rangle$ k-распределена, если $\langle U_n\rangle$ $(2, 2k-1)$-распределена.

12. Примените теорему B с $f(x_1, \ldots, x_k) = [u \le \max(x_1, \ldots, x_k) < v]$.

13. Пусть

$$\begin{aligned} p_k &= \Pr(\text{с } U_n \text{ начинается серия длиной } k-1) \\ &= \Pr(U_{n-1} \in [\alpha\,..\,\beta),\ U_n \notin [\alpha\,..\,\beta),\ \ldots,\ U_{n+k-2} \notin [\alpha\,..\,\beta),\ U_{n+k-1} \in [\alpha\,..\,\beta)) \\ &= p^2(1-p)^{k-1}. \end{aligned}$$

Остается преобразовать это выражение в вероятность того, что $f(n) - f(n-1) = k$. Пусть $\nu_k(n) = (\text{число } j \le n \text{ с } f(j) - f(j-1) = k)$; пусть $\mu_k(n) = (\text{число } j \le n \text{ с } U_j$ — началом серии длиной $k-1)$ и пусть $\mu(n)$ также равно числу $1 \le j \le n$ с $U_j \in [\alpha\,..\,\beta)$. Получим $\mu_k(f(n)) = \nu_k(n)$, $\mu(f(n)) = n$. Когда $n \to \infty$, мы должны получить $f(n) \to \infty$. Следовательно,

$$\nu_k(n)/n = (\mu_k(f(n))/f(n)) \cdot (f(n)/\mu(f(n))) \to p_k/p = p(1-p)^{k-1}.$$

[Здесь используется только тот факт, что последовательность $(k+1)$-распределена.]

14. Пусть $p_k = \Pr(U_n$ начало серии длиной $k)$

$$\begin{aligned} &= \Pr(U_{n-1} > U_n < \cdots < U_{n+k-1} > U_{n+k}) \\ &= \frac{1}{(k+2)!}\left(\binom{k+2}{1}\binom{k+1}{1} - \binom{k+2}{1} - \binom{k+2}{1} + 1\right) \\ &= \frac{k}{(k+1)!} - \frac{k+1}{(k+2)!} \end{aligned}$$

(см. упр. 3.3.2–13). Сейчас поступим, как в предыдущем упражнении, чтобы преобразовать это выражение в $\Pr(f(n) - f(n-1) = k)$. [Только нужно предположить, что последовательность $(k+2)$-распределенная.]

15. Пусть для $s, t \geq 0$

$$p_{st} = \Pr(X_{n-2t-3} = X_{n-2t-2} \neq X_{n-2t-1} \neq \cdots \neq X_{n-1} \text{ и } X_n = \cdots = X_{n+s} \neq X_{n+s+1}) = 2^{-s-2t-3};$$

для $t \geq 0$ пусть $q_t = \Pr(X_{n-2t-2} = X_{n-2t-1} \neq \cdots \neq X_{n-1}) = 2^{-2t-1}$. Согласно упр. 7

$$\Pr(X_n \text{ не начало множества купонов}) = \textstyle\sum_{t\geq 0} q_t = \frac{2}{3};$$
$$\Pr(X_n \text{ начало множества купонов длиной } s+2) = \textstyle\sum_{t\geq 0} p_{st} = \frac{1}{3}\cdot 2^{-s-1}.$$

Поступим, как в упр. 13.

16. (Решение Р. П. Стенли (R. P. Stanley).) Всякий раз, когда появляется подпоследовательность $S = (b-1), (b-2), \ldots, 1, 0, 0, 1, \ldots, (b-2), (b-1)$, множество купонов должно закончиться в правой части S, так как некоторое множество купонов находится полностью в первой половине S. Вычислим вероятность того, что множество купонов начинается с позиции n, используя вероятность, что последнее предшествующее появление S произошло на позиции $n-1$, $n-2$ и т. д., как в упр. 15.

18. Поступите, как в доказательстве теоремы A, чтобы вычислить $\underline{\Pr}$ и $\overline{\Pr}$.

19. (Решение Т. Герцога (T. Herzog).) Да. Например, примените упр. 33 к последовательности $\langle U_{\lfloor n/2\rfloor}\rangle$, когда $\langle U_n\rangle$ удовлетворяет определению R4 (или даже его слабой версии).

20. (a) 2 и $\frac{1}{2}$. (Когда n возрастает, разделяем $l_n^{(1)}$ пополам).

(b) Каждая новая точка разделяет один интервал на две части. Допустим, ρ равно $\max_{k=0}^{n-1}((n+k)l_{n+k}^{(1)})$. Тогда $1 = \sum_{k=1}^{n} l_n^{(k)} \leq \sum_{k=0}^{n-1} l_{n+k}^{(1)} \leq \sum_{k=0}^{n-1} \rho/(n+k) = \rho \ln 2 + O(1/n)$. Так что для бесконечного множества m выполняется $ml_m^{(1)} \geq 1/\ln 2 + O(1/m)$.

(c) Чтобы проверить указание, предположим, что $l_{2n}^{(k)}$ выбирается из интервала с конечными точками U_m и $U_{m'}$, и положим $a_k = \max(m-n, m'-n, 1)$. Тогда, если $\rho = \min_{m=n+1}^{2n} ml_m^{(m)}$, $1 = \sum_{k=1}^{2n} l_{2n}^{(k)} \geq \sum_{k=1}^{2n} \rho/(n+a_k) \geq 2\rho \sum_{k=1}^{n} 1/(n+k)$; следовательно, $2\rho \leq 1/(H_{2n} - H_n) = 1/\ln 2 + O(1/n)$.

(d) Мы получим $(l_n^{(1)}, \ldots, l_n^{(n)}) = (\lg \frac{n+1}{n}, \lg \frac{n+2}{n+1}, \ldots, \lg \frac{2n}{2n-1})$, так как $(n+1)$-я точка всегда делит наибольший интервал на интервалы длиной $\lg \frac{2n+1}{2n}$ и $\lg \frac{2n+2}{2n+1}$. [*Indagationes Math.* **11** (1949), 14–17.]

21. (a) Нет! Мы получим $\overline{\Pr}(W_n < \frac{1}{2}) \geq \limsup_{n\to\infty} \nu(\lceil 2^{n-1/2}\rceil)/\lceil 2^{n-1/2}\rceil = 2-\sqrt{2}$ и $\underline{\Pr}(W_n < \frac{1}{2}) \leq \liminf_{n\to\infty} \nu(2^n)/2^n = \sqrt{2}-1$, поскольку $\nu(\lceil 2^{n-1/2}\rceil) = \nu(2^n) = \frac{1}{2}\sum_{k=0}^{n}(2^{k+1/2} - 2^k) + O(n)$.

(b,c) См. *Indagationes Math.* **40** (1978), 527–541.

22. Если последовательность k-распределена, то предел равен нулю согласно теореме B и значению интеграла. Обратно, заметим, что если $f(x_1, \ldots, x_k)$ разлагается в абсолютно сходящийся ряд Фурье

$$f(x_1, \ldots, x_k) = \sum_{-\infty < c_1, \ldots, c_k < \infty} a(c_1, \ldots, c_k) \exp(2\pi i(c_1x_1 + \cdots + c_kx_k)),$$

то мы получим $\lim_{N\to\infty} \frac{1}{N} \sum_{0\leq n<N} f(U_n, \ldots, U_{n+k-1}) = a(0, \ldots, 0) + \epsilon_r$, где

$$|\epsilon_r| \leq \sum_{\max\{|c_1|,\ldots,|c_k|\}>r} |a(c_1, \ldots, c_k)|,$$

так что ϵ_r можно сделать произвольно малым. Следовательно, предел равен

$$a(0, \ldots, 0) = \int_0^1 \cdots \int_0^1 f(x_1, \ldots, x_k)\, dx_1 \ldots dx_k$$

и (8) выполняется для всех достаточно гладких функций f. Осталось доказать, что функцию в (9) можно аппроксимировать гладкими функциями с любой требуемой точностью.

23. (a) Немедленно следует из упр. 22. (b) Аналогичным путем используйте дискретное преобразование Фурье; см. D. E. Knuth, *AMM* **75** (1968), 260–264.

24. (a) Пусть c — любое не равное нулю целое число. Покажем согласно упр. 22, что

$$\frac{1}{N}\sum_{n=0}^{N-1} e^{2\pi i c U_n} \to 0 \qquad \text{при } N \to \infty.$$

Это выполняется потому, что если K — любое положительное целое число, то получим $\sum_{k=0}^{K-1}\sum_{n=0}^{N-1} e^{2\pi i c U_{n+k}} = K\sum_{n=0}^{N-1} e^{2\pi i c U_n} + O(K^2)$. Следовательно, по неравенству Коши

$$\begin{aligned}\frac{1}{N^2}\left|\sum_{n=0}^{N-1} e^{2\pi i c U_n}\right|^2 &= \frac{1}{K^2N^2}\left|\sum_{n=0}^{N-1}\sum_{k=0}^{K-1} e^{2\pi i c U_{n+k}}\right|^2 + O\Big(\frac{K}{N}\Big)\\ &\le \frac{1}{K^2N}\sum_{n=0}^{N-1}\left|\sum_{k=0}^{K-1} e^{2\pi i c U_{n+k}}\right|^2 + O\Big(\frac{K}{N}\Big)\\ &= \frac{1}{K} + \frac{2}{K^2N}\Re\Bigg(\sum_{0\le j<k<K}\sum_{n=0}^{N-1} e^{2\pi i c(U_{n+k}-U_{n+j})}\Bigg) + O\Big(\frac{K}{N}\Big) \to \frac{1}{K}.\end{aligned}$$

(b) Когда $d = 1$, то из упр. 22 следует, что $\langle(\alpha_1 n + \alpha_0) \bmod 1\rangle$ равнораспределена тогда и только тогда, когда α_1 — иррациональное число. При $d > 1$ можно воспользоваться (a) и индукцией по d. [*Acta Math.* **56** (1931), 373–456. Результат в (b) ранее был получен более сложным методом Г. Вэйлом (H. Weyl, *Nachr. Gesellschaft der Wiss. Göttingen*, Math.-Phys. Kl. (1914), 234–244). С помощью подобных аргументов доказывается, что полиномиальная последовательность равнораспределена, если по крайней мере один из коэффициентов $\alpha_d, \ldots, \alpha_1$ — иррациональное число.]

25. Если последовательность равнораспределена, то знаменатель в следствии S приближается к $\frac{1}{12}$, а числитель — к значению, полученному в этом упражнении.

26. См. *Math. Comp.* **17** (1963), 50–54. [Рассмотрим также следующий пример А. Дж. Вотермена (A. G. Waterman): пусть $\langle U_n\rangle$ — равнораспределенная $[0\,..\,1)$-последовательность и $\langle X_n\rangle$ — ∞-распределенная двоичная последовательность. Пусть $V_n = U_{\lceil\sqrt{n}\rceil}$ или $1 - U_{\lceil\sqrt{n}\rceil}$ соответственно, когда X_n равно 0 или 1. Тогда $\langle V_n\rangle$ равнораспределена и белая, однако $\Pr(V_n = V_{n+1}) = \frac{1}{2}$. Пусть $W_n = (V_n - \epsilon_n) \bmod 1$, где $\langle\epsilon_n\rangle$ — любая убывающая монотонно к 0 последовательность, тогда $\langle W_n\rangle$ равнораспределена и белая, однако $\Pr(W_n < W_{n+1}) = \frac{3}{4}$.]

28. Пусть $\langle U_n\rangle$ ∞-распределена. Рассмотрите последовательность $\langle\frac{1}{2}(X_n + U_n)\rangle$. Она 3-распределена, если использовать тот факт, что $\langle U_n\rangle$ $(16, 3)$-распределена.

29. Если $x = x_1x_2\ldots x_t$ — любое двоичное число, то можно рассмотреть число $\nu_x^E(n)$ случаев, когда $X_p\ldots X_{p+t-1} = x$, где $1 \le p \le n$ и p четное. Аналогично пусть $\nu_x^O(n)$ — число случаев, когда p нечетное. Пусть $\nu_x^E(n) + \nu_x^O(n) = \nu_x(n)$. Тогда

$$\nu_0^E(n) = \sum \nu^E_{0**\ldots*}(n) \approx \sum \nu^O_{*0*\ldots*}(n) \approx \sum \nu^E_{**0\ldots*}(n) \approx \cdots \approx \sum \nu^O_{***\ldots0}(n),$$

где ν_i в этих суммах имеют $2k$ нижних индексов, $2k-1$ из которых — звездочки (обозначающие, что по ним суммируют — каждая сумма берется по 2^{2k-1} комбинаций нулей и единиц), и где "$\approx$" означает приближенное равенство (за исключением ошибки самое большее $2k$ вследствие условий на концах). Поэтому находим, что

$$\tfrac{1}{n}2k\nu_0^E(n) = \tfrac{1}{n}\left(\sum \nu_{*0*\ldots*}(n) + \cdots + \sum \nu_{***\ldots0}(n)\right) \tfrac{1}{n}\sum_x (r(x)-s(x))\nu_x^E(n) + O\left(\tfrac{1}{n}\right),$$

где $x = x_1 \ldots x_{2k}$ содержит $r(x)$ нулей на нечетных позициях и $s(x)$ нулей на четных позициях. Согласно $(2k)$-распределенности величина в скобках стремится к $k(2^{2k-1})/2^{2k} = k/2$. Оставшаяся сумма, очевидно, максимальна, если $\nu_x^E(n) = \nu_x(n)$, когда $r(x) > s(x)$, а $\nu_x^E(n) = 0$, когда $r(x) < s(x)$. Так что максимум правой части равен

$$\frac{k}{2} + \sum_{0\le s<r\le k} (r-s)\binom{k}{r}\binom{k}{s}\Big/2^{2k} = \frac{k}{2} + k\binom{2k-1}{k}\Big/2^{2k}.$$

Сейчас $\overline{\Pr}(X_{2n}=0) \le \limsup_{n\to\infty} \nu_0^E(2n)/n$. Таким образом, доказательство завершено. Заметим, что получено

$$\sum_{r,s}\binom{n}{r}\binom{n}{s}\max(r,s) = 2n2^{2n-2} + n\binom{2n-1}{n};$$
$$\sum_{r,s}\binom{n}{r}\binom{n}{s}\min(r,s) = 2n2^{2n-2} - n\binom{2n-1}{n}.$$

30. Постройте дираф с 2^{2k} вершинами, обозначенными $(Ex_1\ldots x_{2k-1})$ и $(Ox_1\ldots x_{2k-1})$, где каждое x_j равно либо 0, либо 1. Пусть $1+f(x_1,x_2,\ldots,x_{2k})$ — ориентированные ребра из $(Ex_1\ldots x_{2k-1})$ к $(Ox_2\ldots x_{2k})$, а $1-f(x_1,x_2,\ldots,x_{2k})$ — ориентированные ребра, ведущие из $(Ox_1\ldots x_{2k-1})$ к $(Ex_2\ldots x_{2k})$, где $f(x_1,x_2,\ldots,x_{2k}) = \operatorname{sign}(x_1-x_2+x_3-x_4+\cdots-x_{2k})$. Мы обнаружим, что каждая вершина имеет столько же ребер, ведущих к ней, сколько ребер, ведущих от нее. Например, $(Ex_1\ldots x_{2k-1})$ имеет $1-f(0,x_1,\ldots,x_{2k-1})+1-f(1,x_1,\ldots,x_{2k-1})$ ведущих к ней ребер, $1+f(x_1,\ldots,x_{2k-1},0)+1+f(x_1,\ldots,x_{2k-1},1)$ ребер, ведущих от нее, и $f(x,x_1,\ldots,x_{2k-1}) = -f(x_1,\ldots,x_{2k-1},x)$. Опустим все вершины, не имеющие путей, ведущих к ним либо исходящих из них, т. е. $(Ex_1\ldots x_{2k-1})$, если $f(0,x_1,\ldots,x_{2k-1}) = +1$, или $(Ox_1\ldots x_{2k-1})$, если $f(1,x_1,\ldots,x_{2k-1}) = -1$. Полученный ориентированный граф является связным, так как мы можем добраться из любой вершины к $(E1010\ldots1)$ и из этой точки — к любой требуемой вершине. Согласно теореме 2.3.4.2G существует циклический путь, проходящий через каждое ребро длиной 2^{2k+1}, и можно предположить, что он начинается в вершине $(E00\ldots0)$. Построим циклическую последовательность с $X_1 = \cdots = X_{2k-1} = 0$ и $X_{n+2k-1} = x_{2k}$, если n-е ребро — это путь из $(Ex_1\ldots x_{2k-1})$ к $(Ox_2\ldots x_{2k})$ или из $(Ox_1\ldots x_{2k-1})$ к $(Ex_2\ldots x_{2k})$. Например, граф для $k=2$ показан на рис. А–5; ребра циклического пути пронумерованы от 1 до 32 и циклическая последовательность имеет вид

$$(00001000110010101001101110111110)(00001\ldots).$$

Заметим, что в этой последовательности $\Pr(X_{2n}=0) = \frac{11}{16}$. Очевидно, что последовательность $(2k)$-распределена, так как каждая строка размерности $(2k)$ $x_1x_2\ldots x_{2k}$ появляется

$$1+f(x_1,\ldots,x_{2k})+1-f(x_1,\ldots,x_{2k}) = 2$$

раз за цикл. Тот факт, что $\Pr(X_{2n}=0)$ имеет требуемое значение, вытекает из факта, что при этом построении достигается максимальное значение правой части равенства в доказательстве предыдущего упражнения.

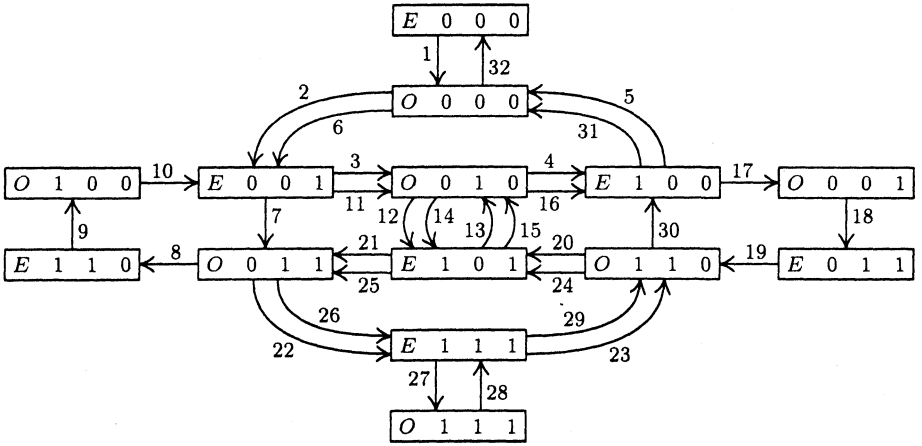


Рис. А–5. Ориентированный граф из упр. 30.

31. Используйте алгоритм W с правилом $\mathcal{R}_1$ выбора полной последовательности. [Для обобщения этого типа неслучайного поведения в R5-последовательностях обратитесь к работе Jean Ville, *Étude Critique de la Notion de Collectif* (Paris, 1939), 55–62. Возможно, определение R6 также слишком слабое с этой точки зрения, однако такой контрпример неизвестен.]

32. Если $\mathcal{R}, \mathcal{R}'$ — исчисляемые правила подпоследовательностей, то существует $\mathcal{R}'' = \mathcal{R}\mathcal{R}'$, определенное такими функциями: $f''_n(x_0, \ldots, x_{n-1}) = 1$ тогда и только тогда, когда $\mathcal{R}$ определяет подпоследовательность $x_{r_1}, \ldots, x_{r_k}$ из последовательности $x_0, \ldots, x_{n-1}$, где $k \geq 0$, $0 \leq r_1 < \cdots < r_k < n$ и $f'_k(x_{r_1}, \ldots, x_{r_k}) = 1$.

Тогда $\langle X_n \rangle \mathcal{R}\mathcal{R}'$ равна $(\langle X_n \rangle \mathcal{R})\mathcal{R}'$. Результат следует незамедлительно.

33. Зададим $\epsilon > 0$ и найдем такое N_0, при котором неравенство $N > N_0$ влечет оба неравенства $|\nu_r(N)/N - p| < \epsilon$ и $|\nu_s(N)/N - p| < \epsilon$. Затем найдем такое N_1, при котором из $N > N_1$ следует, что t_N равно r_M или s_M для какого-либо $M > N_0$. Из $N > N_1$ следует, что

$$\left|\frac{\nu_t(N)}{N} - p\right| = \left|\frac{\nu_r(N_r) + \nu_s(N_s)}{N} - p\right| = \left|\frac{\nu_r(N_r) - pN_r + \nu_s(N_s) - pN_s}{N_r + N_s}\right| < \epsilon.$$

34. Например, если двоичным представлением t является $(1\,0^{b-2}\,1\,0^{a_1}\,1\,1\,0^{a_2}\,1\,\ldots\,1\,0^{a_k})_2$, где "$0^a$" — обозначение последовательности из a нулей. Пусть правило $\mathcal{R}_t$ принимает U_n тогда и только тогда, когда $\lfloor bU_{n-k} \rfloor = a_1, \ldots, \lfloor bU_{n-1} \rfloor = a_k$.

35. Пусть $a_0 = s_0$ и $a_{m+1} = \max\{s_k \mid 0 \leq k < 2^{a_m}\}$. Построим правило подпоследовательностей, выбирающее элемент X_n тогда и только тогда, когда $n = s_k$ для некоторого $k < 2^{a_m}$, когда n принадлежит интервалу $a_m \leq n < a_{m+1}$. Тогда $\lim_{m\to\infty} \nu(a_m)/a_m = \frac{1}{2}$.

36. Пусть b и k — произвольные, однако фиксированные целые числа, большие, чем 1. Пусть $Y_n = \lfloor bU_n \rfloor$. Для произвольной бесконечной подпоследовательности $\langle Z_n \rangle = \langle Y_{s_n} \rangle \mathcal{R}$, определенной алгоритмами $\mathcal{S}$ и $\mathcal{R}$ (как в доказательстве теоремы M), существует простое, но трудно записываемое соответствие алгоритмам $\mathcal{S}'$ и $\mathcal{R}'$, которые просматривают X_t,

X_{t+1}, ..., X_{t+s} и/или выбирают X_t, X_{t+1}, ..., $X_{t+\min(k-1,s)}$ из $\langle X_n\rangle$ тогда и только тогда, когда $\mathcal{S}$ и $\mathcal{R}$ просматривают и/или выбирают Y_s, где $U_s = (0.X_tX_{t+1}\ldots X_{t+s})_2$. Алгоритмы $\mathcal{S}'$ и $\mathcal{R}'$ определяют бесконечную 1-распределенную подпоследовательность $\langle X_n\rangle$, и фактически (как в упр. 32) эта подпоследовательность является ∞-распределенной, поскольку она $(k,1)$-распределена. Таким образом, мы определили, что $\underline{\Pr}(Z_n = a)$ и $\overline{\Pr}(Z_n = a)$ отличаются от $1/b$ менее чем на $1/2^k$.

[Результат этого упражнения справедлив, если "R6" заменить последовательно на "R4" или "R5", но он не верен, если использовать "R1," так как $X_{\binom{n}{2}}$ может быть тождественно равен нулю.]

37. Для $n \geq 2$ замените U_{n^2} на $\frac{1}{2}(U_{n^2} + \delta_n)$, где $\delta_n = 0$ или 1 в зависимости от того, четное или нечетное число элементов, меньших $\frac{1}{2}$, содержит множество $\{U_{(n-1)^2+1}, \ldots, U_{n^2-1}\}$. [*Advances in Math.* **14** (1974), 333–334; см. также Ph. D. thesis Томаса Н. Герцога (Thomas N. Herzog, Univ. of Maryland, 1975).]

39. См. *Acta Arithmetica* **21** (1972), 45–50. Наилучшее возможное значение c неизвестно.

40. Так как F_k зависимы только на $B_1\ldots B_k$, получим $P(A_k^P, \$_N) = \frac{1}{2}$. Пусть $q(B_1\ldots B_k) = \Pr(B_{k+1} = 1 \mid B_1\ldots B_k)$, где вероятность берется по всем элементам S, имеющим $B_1\ldots B_k$ в качестве первых k двоичных разрядов. Аналогично пусть $q_b(B_1\ldots B_k) = \Pr(F_k = 1$ и $B'_{k+1} = b \mid B_1\ldots B_k)$. Тогда получим $\Pr(A_k^P{=}1 \mid B_1\ldots B_k){=}\Pr((F_k{+}B_{k+1}{+}B'_{k+1}) \bmod 2{=}1 \mid B_1\ldots B_k) = q\cdot(\frac{1}{2} - q_0 + q_1) + (1-q)\cdot(q_0 + \frac{1}{2} - q_1) = \frac{1}{2} - (q_0 + q_1) + 2(qq_1 + (1-q)q_0) = \frac{1}{2} - \Pr(F_k = 1 \mid B_1\ldots B_k) + 2\Pr(F_k = 1$ и $B'_{k+1} = B_{k+1} \mid B_1\ldots B_k)$. Следовательно, $\Pr(A_k^P = 1) = \sum_{B_1\ldots B_k} \Pr(B_1\ldots B_k)\Pr(A_k^P = 1 \mid B_1\ldots B_k) = \frac{1}{2} - \Pr(F_k = 1) + \Pr(F_{k+1} = 1)$. [См. теорему 4 в работе Goldreich, Goldwasser, and Micali, *JACM* **33** (1986), 792–807.]

41. Выберите k равномерно из $\{0,\ldots,N-1\}$ и воспользуйтесь построением из доказательства леммы P1. Тогда из доказательства P1 будет следовать, что A' равно 1 с вероятностью $\sum_{k=0}^{N-1}(\frac{1}{2} - p_k + p_{k+1})/N$.

42. (a) Пусть $X = X_1 + \cdots + X_n$. Очевидно, что $\mathrm{E}(X) = n\mu$, и мы имеем $\mathrm{E}((X-n\mu)^2) = \mathrm{E}\,X^2 - n^2\mu^2 = n\,\mathrm{E}\,X_j^2 + 2\sum_{1\leq i<j\leq n}(\mathrm{E}\,X_i)(\mathrm{E}\,X_j) - n^2\mu^2 = n\,\mathrm{E}\,X_j^2 - n\mu^2 = n\sigma^2$. Также $\mathrm{E}((X-n\mu)^2) = \sum_{x\geq 0} x\Pr((X-n\mu)^2 = x) \geq \sum_{x\geq tn\sigma^2} x\Pr((X-n\mu)^2 = x) \geq \sum_{x\geq tn\sigma^2} tn\sigma^2 \times \Pr((X-n\mu)^2 = x) = tn\sigma^2\Pr((X-n\mu)^2 \geq tn\sigma^2)$.

(b) Существует индекс i, такой, что $c_i \neq c'_i$, скажем, $c_i = 0$ и $c'_i = 1$. Тогда существует такой индекс j, что $c_j = 1$. Для любого фиксированного набора из $k-2$ строк в матрице B, номер которых не равен i или j, получим $(cB, c'B) = (d, d')$ тогда и только тогда, когда строки i и j имеют частные значения (это происходит с вероятностью $1/2^{2R}$).

(c) В обозначениях алгоритма L возьмем $n = 2^k - 1$ и $X_c = (-1)^{G(cB+e_i)}$, тогда $\mu = s$ и $\sigma^2 = 1 - s^2$. Вероятность, что $X = \sum_{c\neq 0} X_c$ отрицательна, не больше вероятности того, что $(X-n\mu)^2 \geq n^2\mu^2$. Согласно (a) это не больше, чем $\sigma^2/(n\mu^2)$.

43. Заключение для фиксированного M не представляет интереса, так как, очевидно, существует алгоритм для нахождения множителей любого фиксированного M (т. е. алгоритм для нахождения множителей). Теория применима ко *всем* алгоритмам, имеющим короткое время счета, а не только к алгоритмам, которые эффективно находят множители.

44. Если каждое изменение одного числа в случайной таблице приводит к случайной таблице, то все таблицы случайны (или ни одна не существует). Если мы не допускаем степеней случайности, то ответ должен, следовательно, быть "Не всегда".

```
1. RANDI STJ  9F          Запоминание выходной ячейки.
       STA  8F            Запоминание значения k.
       LDA  XRAND         rA ← X.
       MUL  7F            rAX ← aX.
       INCX 1009          rX ← (aX + c) mod m.
       JOV  *+1           Гарантия, что переполнение выключено.
       SLAX 5             rA ← (aX + c) mod m.
       STA  XRAND         Запоминание X.
       MUL  8F            rA ← ⌊kX/m⌋.
       INCA 1             Добавить 1, чтобы 1 ≤ Y ≤ k.
   9H  JMP  *             Возврат.
XRAND  CON  1             Значение X; X_0 = 1.
   8H  CON  0             Быстрое запоминание k.
   7H  CON  3141592621    Множитель a.  ▌
```

2. Помещение генератора случайных чисел в программу дает, по существу, непредсказуемый для программиста результат. Если бы поведение машины в каждой задаче было известно заранее, немногие программы были бы когда-нибудь написаны. Как говорил Тьюринг (Turing), действия компьютера очень часто *преподносят* сюрприз программисту, в особенности при отладке.

Так что мир должен быть более бдительным.

7. Фактически вам необходимо только 2 двуразрядных значения $\lfloor X_n/2^{16} \rfloor \bmod 4$; см. D. E. Knuth, *IEEE Trans.* **IT-31** (1985), 49–52. J. Reeds, *Cryptologia* **1** (1977), 20–26, **3** (1979), 83–95. Д. Рид первым начал изучать родственные задачи. (См. также L. Blum, M. Blum, and M. Shub, *SICOMP* **15** (1986), 364–383; J. Boyar, *J. Cryptology* **1** (1989), 177–184.) В своей работе Фриз, Хастед, Кеннан, Лагарис и Шамир (Frieze, Håstad, Kannan, Lagarias, and Shamir, *SICOMP* **17** (1988), 262–280) обсуждают общие технические приемы, используемые в аналогичных задачах.

8. Можно, допустим, генерировать $X_{1000000}$, произведя миллион последовательных вызовов программы, и сравнивать с точным значением $(a^{1000000}X_0 + (a^{1000000} - 1)c/(a-1)) \bmod m$, которое также можно выразить в виде $((a^{1000000}(X_0(a-1)+c) - c) \bmod (a-1)m)/(a-1)$. Последний можно быстро оценить независимым методом (см. алгоритм 4.6.3A), например $48271^{1000000} \bmod 2147483647 = 1263606197$. Большая часть ошибок обнаруживается, так как рекуррентное соотношение (1) не является самокорректирующимся.

9. Значения X_0, X_1, ..., X_{99} не все четные. $z^{100} + z^{37} + 1$ — первообразный полином (см. раздел 3.2.2); следовательно, существует число $h(s)$ такое, что $P_0(z) \equiv z^{h(s)}$ (по модулю 2 и $z^{100}+z^{37}+1$). Тогда $zP_{n+1}(z) = P_n(z) - X_n z^{37} - X_{n+63} + X_{n+63}z^{100} + X_{n+100}z^{37} \equiv P_n(z) + X_{n+63}(z^{100} + z^{37} + 1)$ (по модулю 2), так что результат получается по индукции.

(b) Операции "возведение в квадрат" и "умножение на z" в *ran_start* меняют $p(z) = x_{99}z^{99} + \cdots + x_1 z + x_0$ на $p(z)^2$ и $zp(z)$ соответственно по модулю 2 и $z^{100} + z^{37} + 1$, так как $p(z)^2 \equiv p(z^2)$. (Мы рассмотрели здесь только младшие двоичные разряды. Другие двоичные разряды преобразуются специальным методом, который стремится сохранить и/или увеличить тот беспорядок, в котором они находятся.) Следовательно, если $s = (1s_j \ldots s_1 s_0)_2$, то получим $h(s) = (1s_0 s_1 \ldots s_j 1)_2 \cdot 2^{69}$.

(c) $z^{h(s)-n} \equiv z^{h(s')-n'}$ (по модулю 2 и $z^{100} + z^{37} + 1$) подразумевает, что $h(s) - n \equiv h(s') - n'$ (по модулю $2^{100} - 1$). Так как $2^{69} \le h(s) < 2^{100} - 2^{69}$, имеем $|n - n'| \ge |h(s) - h(s')| \ge 2^{70}$.

[Этот метод инициализации инспирирован комментариями Р. П. Брента (R. P. Brent, *Proc. Australian Supercomputer Conf.* **5** (1992), 95–104), хотя алгоритм Брента полностью отличается от этого. Вообще, если запаздывание $k > l$, если $0 \le s < 2^e$ и если отдельный параметр t удовлетворяет неравенству $t + e \le k$, этот метод доказательства показывает, что $|n - n'| \ge 2^t - 1$ с $2^t - 1$ выполняется только тогда, когда $\{s, s'\} = \{0, 2^e - 1\}$.]

10. Следующие операции относятся к упрощенному языку Subset FORTRAN, как установлено American National Standards Institute, за исключением использования утверждения PARAMETER для удобства.

```
      SUBROUTINE RNARRY(AA,N)
      IMPLICIT INTEGER (A-Z)
      DIMENSION AA(*)
      PARAMETER (KK=100)
      PARAMETER (LL=37)
      PARAMETER (MM=2**30)
      COMMON /RSTATE/ RANX(KK)
      SAVE /RSTATE/
      DO 1 J=1,KK
1        AA(J)=RANX(J)
      DO 2 J=KK+1,N
         AA(J)=AA(J-KK)-AA(J-LL)
         IF (AA(J) .LT. 0) AA(J)=AA(J)+MM
2     CONTINUE
      DO 3 J=1,LL
         RANX(J)=AA(N+J-KK)-AA(N+J-LL)
         IF (RANX(J) .LT. 0) RANX(J)=RANX(J)+MM
3     CONTINUE
      DO 4 J=LL+1,KK
         RANX(J)=AA(N+J-KK)-RANX(J-LL)
         IF (RANX(J) .LT. 0) RANX(J)=RANX(J)+MM
4     CONTINUE
      END

      SUBROUTINE RNSTRT(SEED)
      IMPLICIT INTEGER (A-Z)
      PARAMETER (KK=100)
      PARAMETER (LL=37)
      PARAMETER (MM=2**30)
      PARAMETER (TT=70)
      PARAMETER (KKK=KK+KK-1)
      DIMENSION X(KKK)
      COMMON /RSTATE/ RANX(KK)
      SAVE /RSTATE/
      IF (SEED .LT. 0) THEN
         SSEED=MM-1-MOD(-1-SEED,MM)
      ELSE
         SSEED=MOD(SEED,MM)
      END IF
      SS=SSEED-MOD(SSEED,2)+2
```

```
      DO 1 J=1,KK
         X(J)=SS
         SS=SS+SS
         IF (SS .GE. MM) SS=SS-MM+2
1     CONTINUE
      DO 2 J=KK+1,KKK
2        X(J)=0
      X(2)=X(2)+1
      SS=SSEED
      T=TT-1
10    DO 12 J=KK,2,-1
12       X(J+J-1)=X(J)
      DO 13 J=KKK,KK-LL+1,-2
13       X(KKK-J+2)=X(J)-MOD(X(J),2)
      DO 14 J=KKK,KK+1,-1
         IF (MOD(X(J),2) .EQ. 1) THEN
            X(J-(KK-LL))=X(J-(KK-LL))-X(J)
            IF (X(J-(KK-LL)) .LT. 0) X(J-(KK-LL))=X(J-(KK-LL))+MM
            X(J-KK)=X(J-KK)-X(J)
            IF (X(J-KK) .LT. 0) X(J-KK)=X(J-KK)+MM
         END IF
14    CONTINUE
      IF (MOD(SS,2) .EQ. 1) THEN
         DO 16 J=KK,1,-1
16       X(J+1)=X(J)
         X(1)=X(KK+1)
         IF (MOD(X(KK+1),2) .EQ. 1) THEN
            X(LL+1)=X(LL+1)-X(KK+1)
            IF (X(LL+1) .LT. 0) X(LL+1)=X(LL+1)+MM
         END IF
      END IF
      IF (SS .NE. 0) THEN
         SS=SS/2
      ELSE
         T=T-1
      END IF
      IF (T .GT. 0) GO TO 10
      DO 20 J=1,LL
20       RANX(J+KK-LL)=X(J)
      DO 21 J=LL+1,KK
21       RANX(J-LL)=X(J)
      END
```

11. Арифметика с плавающей точкой на операндах с 64-мя двоичными разрядами соответствует ANSI/IEEE Standard 754 и позволяет вычислять $U_n = (U_{n-100} - U_{n-37}) \bmod 1$ с прекрасной точностью для дробей U_n, которые являются целыми кратными 2^{-53}. Тем не менее следующая программа использует *аддитивное* рекуррентное соотношение $U_n = (U_{n-100} + U_{n-37}) \bmod 1$ для целых кратных 2^{-52}, так как конвейерные компьютеры могут вычитать целую часть более быстро, чем выполнять условный переход по знаку промежуточного результата. Теория из упр. 9 применима также к этой последовательности. Основной новой идеей программы *ranf_start* является хранение копии *ul*

младших значащих двоичных разрядов дробей в u. Перевод на язык программирования FORTRAN, подобно операциям из упр. 10, генерирует такие же числа, как и программа на языке C.

```
#define KK 100                              /* длинное запаздывание */
#define LL  37                              /* короткое запаздывание */
#define mod_sum(x,y) (((x)+(y))-(int)((x)+(y)))   /* (x+y) mod 1.0 */

double ran_u[KK];          /* состояние генератора */

void ranf_array(double aa[],int n) { /* aa присвоить n случайных
                                        дробей */
register  int  i,j;
for  (j=0;j<KK;j++)   aa[j]=ran_u[j];
for  (;j<n;j++)   aa[j]=mod_sum(aa[j-KK],aa[j-LL]);
for  (i=0;i<LL;i++,j++)
  ran_u[i]=mod_sum(aa[j-KK],aa[j-LL]);
for     (;i<KK;i++,j++)
  ran_u[i]=mod_sum(aa[j-KK],ran_u[i-LL]);
}

#define TT  70              /* гарантирует разделение потоков */
#define is_odd(s) ((s)&1)

void ranf_start(long seed) {  /* сделать до использования ranf_array */
  register int t,s,j;
  double u[KK+KK-1],ul[KK+KK-1];
  double ulp=(1.0/(1L<<30))/(1L<<22);  /* от 2 к -52 */
  double ss=2.0*ulp*(seed+2);

  for (j=0;j<KK;j++) {
    u[j]=ss; ul[j]=0.0;                  /* инициализация буфера */
    ss+=ss; if (ss>=1.0) ss-=1.0-2*ulp;  /* циклический сдвиг на
                                            51 двоичный разряд */
  }
  for (;j<KK+KK-1;j++) u[j]=ul[j]=0.0;
  u[1]+=ulp;ul[1]=ulp;   /* получаем u[1] (и только u[1]) "нечетное" */
  s=seed;
  t=TT-1; while (t) {
    for (j=KK-1;j>0;j--) ul[j+j]=ul[j],u[j+j]=u[j];
                                        /*  "возведение в квадрат" */
for (j=KK+KK-2;j>KK-LL;j-=2)
        ul[KK+KK-1-j]=0.0,u[KK+KK-1-j]=u[j]-ul[j];
    for (j=KK+KK-2;j>=KK;j--) if(ul[j]) {
      ul[j-(KK-LL)]=ulp-ul[j-(KK-LL)],
        u[j-(KK-LL)]=mod_sum(u[j-(KK-LL)],u[j]);
      ul[j-KK]=ulp-ul[j-KK],u[j-KK]=mod_sum(u[j-KK],u[j]);
    }
    if (is_odd(s)) {                    /* "умножение на z" */
      for (j=KK;j>0;j--)  ul[j]=ul[j-1],u[j]=u[j-1];
      ul[0]=ul[KK],u[0]=u[KK];          /* циклический сдвиг буфера */
      if (ul[KK]) ul[LL]=ulp-ul[LL],u[LL]=mod_sum(u[LL],u[KK]);
    }
    if (s) s>>=1; else t--;
```

```
  }
  for (j=0;j<LL;j++) ran_u[j+KK-LL]=u[j];
  for (;j<KK;j++) ran_u[j-LL]=u[j];
}

main() { register int m; double a[2009]; /* элементарный критерий */
  ranf_start(310952);
  for (m=0;m<2009;m++) ranf_array(a,1009);
  printf("%.20f\n", ran_u[0]);            /* 0.27452626307394156768 */
  ranf_start(310952);
  for (m=0;m<1009;m++) ranf_array(a,2009);
  printf("%.20f\n", ran_u[0]); }          /* 0.27452626307394156768 */
```

12. Простой линейный конгруэнтный генератор, подобный (1), не подходит, так как m чересчур мало́. Хороший результат возможен при комбинации трех (не двух) таких генераторов с множителями и модулями (157, 32363), (146, 31727) и (142, 31657), как советует П. Лекуер (P. L'Ecuyer, *CACM* **31** (1988), 747–748). Тем не менее лучшим методом является использование программ на языке C *ran_array* и *ran_start* со следующей заменой, чтобы сохранить все числа в области: '`long`' становятся '`int`'; '`MM`' определяется как '`(1U<<15)`' и тип переменной *ss* должен быть `unsigned int`. Тогда генерируются целые числа, содержащие 15 двоичных разрядов, из которых можно использовать все двоичные разряды. Начальное значение сейчас ограничено областью $[0\,..\,32765]$. "Программа элементарной проверки" напечатает $X_{1009\times 2009} = 9387$ при заданном начальном значении 12509.

13. Программа вычитания с заимствованием очень похожа на программу *ran_array*, но работает медленнее, поскольку сохраняет содержимое. Как в упр. 11, арифметику с плавающей точкой можно использовать с совершенной точностью. Это позволяет гарантировать "непересечение" последовательностей, полученных от различных начальных значений s путем инициализации генератора с $(-n)$-м элементом последовательности, где $n = 2^{70s}$. Этого требует вычисление $b^n \bmod (b^k - b^l \pm 1)$. Возведение в квадрат числа с основанием b по mod $b^k - b^l \pm 1$, тем не менее, значительно сложнее, чем аналогичная операция в программе *ran_start*, и для k оно практически выполняется за приблизительно $k^{1.6}$ операций вместо $O(k)$.

Оба метода, вероятно, практически генерируют последовательности одинакового качества, когда у них приблизительно те же самые значения k. Единственным различием между ними является лучшее теоретическое обоснование и, возможно, огромный период метода вычитания с заимствованием, анализ генератора Фибоначчи с запаздыванием менее полный. Опыт показывает, что можно было бы не уменьшать значение k в методе вычитания с заимствованием только из-за его теоретических преимуществ. Когда все это сказано и сделано, генератор Фибоначчи с запаздыванием кажется предпочтительнее с практической точки зрения. Метод вычитания с заимствованием является ценным, главным образом, потому, что его понимание дается нам благодаря превосходному поведению более простого приближения.

14. Имеем $X_{n+200} \equiv (X_n + X_{n+126})$ (по модулю 2); см. упр. 3.2.2–32. Следовательно, $Y_{n+100} \equiv Y_n + Y_{n+26}$, когда $n \bmod 100 > 73$. Аналогично $X_{n+200} \equiv X_n + X_{n+26} + X_{n+89}$. Значит, $Y_{n+100} \equiv Y_n + Y_{n+26} + Y_{n+89}$, когда $n \bmod 100 < 11$. Таким образом, Y_{n+100} является суммой только двух или трех элементов $\{Y_n, \ldots, Y_{n+99}\}$ в $26\% + 11\%$ всех случаев и преобладание нулей приводит к тенденции делать так, чтобы выполнялось равенство $Y_{n+100} = 0$.

Точнее, рассмотрим последовательность $\langle u_1, u_2, \ldots \rangle = \langle 126, 89, 152, 115, 78, \ldots, 100, 63, 126, \ldots \rangle$, где $u_{n+1} = u_n - 37 + 100[u_n < 100]$. Тогда

$$X_{n+200} = (X_n + X_{n+v_1} + \cdots + X_{n+v_{k-2}} + X_{u_{k-1}}) \bmod 2,$$

где $v_j = u_j + (-1)^{[u_j \geq 100]}100$, например $X_{n+200} \equiv X_n + X_{n+26} + X_{n+189} + X_{n+152} \equiv X_n + X_{n+26} + X_{n+189} + X_{n+52} + X_{n+115}$. Если все индексы $< n+t$ и $\geq n+100+t$, то получим k членов выражения Y_{n+100}, когда $n \bmod 100 = 100-t$ для $1 \leq t \leq 100$. Случай, когда $t = 63$, является исключением, так как $X_n + X_{n+1} + \cdots + X_{n+62} + X_{n+163} + X_{n+164} + \cdots + X_{n+199} \equiv 0$. Здесь Y_{n+100} не зависит от $\{Y_n, \ldots, Y_{n+99}\}$. Случай, когда $t = 64$, интересен потому, что он дает 99 членов соотношения $Y_{n+100} \equiv Y_{n+1} + Y_{n+2} + \cdots + Y_{n+99}$, которые имеют тенденцию становиться нулями несмотря на большое количество членов. Это происходит потому, что большая часть строк размерности 100, которые имеют 40 или меньше единиц, имеют одинаковую четность.

Когда существует соотношение из k членов, вероятность, что $Y_{n+100} = 1$, равна

$$p_k = \sum_{l=0}^{40} \sum_{j=1}^{k} \binom{100-k}{l-j} \binom{k}{j} [j \text{ odd}] \Big/ \sum_{l=0}^{40} \binom{100}{l}.$$

Величина t принимает значения 100, 99, ..., 1, 100, 99, ..., 1, ... во время печати двоичных разрядов. Итак, мы определили, что ожидаемое число напечатанных единиц равно $10^6(26p_2+11p_3+26p_4+11p_6+11p_9+4p_{12}+4p_{20}+3p_{28}+p_{47}+p_{74}+p_{99}+1/2)/100 \approx 14043$. Ожидаемое число напечатанных знаков равно $10^6 \sum_{l=0}^{40} \binom{100}{l}/2^{100} \approx 28444$, ожидаемое число нулей равно ≈ 14401.

Замеченное смещение пропадает, если отбрасывается больше элементов. Например, если использовать только 100 элементов программы *ran_array*(a,300), вероятность может быть $(26p_5 + 22p_6 + 19p_{10} + \cdots)/100$. С программой *ran_array*(a,400) дела обстоят хуже: здесь вероятность $(15p_3 + 37p_6 + 15p_9 + \cdots)/100$, так как $X_{n+400} \equiv X_n + X_{n+252}$. С программой *ran_array*(a,1009), рекомендованной в разделе, получаем вероятность $(17p_7 + 10p_{11} + 2p_{12} + \cdots)/100$, что может быть обнаружено в ходе таких экспериментов, если порог для печати поднимается от 60 до, скажем, 75, но тогда предполагаемое число выходов равно всего лишь около 0.28 из миллиона испытаний.

[Это упражнение основано на идеях Й. Курита (Y. Kurita), Х. Лииба (H. Leeb) и М. Мацумото (M. Matsumoto), сообщенных автору в 1997 году.]

15. Следующая программа позволяет получить новые случайные целые числа, которые выражаются *ran_arr_next*(), начав с программы *ran_start*.

```
#define QUALITY 1009 /* рекомендуется уровень качества для
                        высокорезультативного использования */
long ran_arr_buf[QUALITY];
long ran_arr_sentinel=-1;
long *ran_arr_ptr=&ran_arr_sentinel; /* следующее случайное
                                        число или -1 */
#define ran_arr_next() (*ran_arr_ptr>=0? *ran_arr_ptr++: ran_arr_cycle())
long ran_arr_cycle()
{
  ran_array(ran_arr_buf,QUALITY);
  ran_arr_buf[100]=-1;
  ran_arr_ptr=ran_arr_buf+1;
  return ran_arr_buf[0];
}
```

РАЗДЕЛ 4.1

1. $(1010)_{-2}$, $(1011)_{-2}$, $(1000)_{-2}$, ..., $(11000)_{-2}$, $(11001)_{-2}$, $(11110)_{-2}$.

2. (a) $-(110001)_2$, $-(11.001001001001\ldots)_2$, $-(11.00100100001111110110101\ldots)_2$.
(b) $(11010011)_{-2}$, $(1101.001011001011\ldots)_{-2}$, $(111.0110010001000000101\ldots)_{-2}$.
(c) $(\bar{1}11\bar{1}\bar{1})_3$, $(\bar{1}0.0\bar{1}\bar{1}0110\bar{1}\bar{1}011\ldots)_3$, $(10.011\bar{1}111\bar{1}000\bar{1}011\bar{1}101\bar{1}1111110\ldots)_3$.
(d) $-(9.4)_{1/10}$, $-(\ldots 7582417582413)_{1/10}$, $(\ldots 346264832397985356295141 3)_{1/10}$.

3. $(1010113.2)_{2i}$.

4. (a) Между rA и rX. (b) У остатка в регистре rX разделяющая точка находится между байтами 3 и 4; у частного в регистре rA разделяющая точка расположена на один байт правее младшего разряда регистра.

5. Представление в обратном коде получается путем вычитания из $999\ldots9 = 10^p - 1$, вместо вычитания из $1000\ldots0 = 10^p$.

6. (a, c) $2^{p-1} - 1$, $-(2^{p-1} - 1)$; (b) $2^{p-1} - 1$, -2^{p-1}.

7. Представление в дополнительном коде для отрицательного числа x может быть получено, если взять число $10^n + x$ (n достаточно велико, чтобы число было положительным) и продолжить его влево при помощи бесконечного количества девяток. Представление в обратном коде можно получить обычным способом. (Эти два представления совпадают для бесконечных десятичных дробей, в противном случае представление в обратном коде имеет вид $\ldots(a)99999\ldots$, а представление в дополнительном коде — $\ldots(a+1)0000\ldots$.) Данные представления имеют смысл, если считать, что значение бесконечной суммы $N = 9 + 90 + 900 + 9000 + \cdots$ равно -1, поскольку $N - 10N = 9$.

См. также упр. 31, в котором рассматриваются системы p-адических чисел. Для чисел, представление которых по основанию p конечно, p-адическое представление совпадает с рассмотренным здесь представлением в дополнительном коде, но между полем p-адических чисел и полем вещественных чисел не существует никакой прямой связи.

8. $\sum_j a_j b^j = \sum_j (a_{kj+k-1}b^{k-1} + \cdots + a_{kj})b^{kj}$.

9. A BAD ADOBE FACADE FADED. [*Примечание.* Вот другие "числовые фразы": DO A DEED A DECADE; A CAD FED A BABE BEEF, COCOA, COFFEE; BOB FACED A DEAD DODO.]

10. $\begin{bmatrix}\ldots, a_3, a_2, a_1, a_0; a_{-1}, a_{-2}, \ldots\\ \ldots, b_3, b_2, b_1, b_0; b_{-1}, b_{-2}, \ldots\end{bmatrix} = \begin{bmatrix}\ldots, A_3, A_2, A_1, A_0; A_{-1}, A_{-2}, \ldots\\ \ldots, B_3, B_2, B_1, B_0; B_{-1}, B_{-2}, \ldots\end{bmatrix}$, если

$$A_j = \begin{bmatrix} a_{k_{j+1}-1}, a_{k_{j+1}-2}, \ldots, a_{k_j} \\ b_{k_{j+1}-2}, \ldots, b_{k_j}\end{bmatrix}, \qquad B_j = b_{k_{j+1}-1}\ldots b_{k_j},$$

где $\langle k_n\rangle$ — произвольная бесконечная последовательность целых чисел, удовлетворяющих неравенствам $k_{j+1} > k_j$.

11. (Описываемый ниже алгоритм выполняет как сложение, так и вычитание в зависимости от выбора знака "плюс" или "минус".)

Сначала устанавливается $k \leftarrow a_{n+1} \leftarrow a_{n+2} \leftarrow b_{n+1} \leftarrow b_{n+2} \leftarrow 0$; затем для $m = 0, 1, \ldots, n+2$ выполняется следующее: устанавливается $c_m \leftarrow a_m \pm b_m + k$; далее, если $c_m \geq 2$, устанавливается $k \leftarrow -1$ и $c_m \leftarrow c_m - 2$; если $c_m < 0$, устанавливается $k \leftarrow 1$ и $c_m \leftarrow c_m + 2$; а если $0 \leq c_m \leq 1$, то устанавливается $k \leftarrow 0$.

12. (a) Вычесть $\pm(\ldots a_3 0 a_1 0)_{-2}$ из $\pm(\ldots a_4 0 a_2 0 a_0)_{-2}$ в негадвоичной системе.
(В упр. 7.1-18 приводится оригинальное решение задачи, полученное с использованием битовых операций над полным словом.)

(b) Вычесть $(\ldots b_3 0 b_1 0)_2$ из $(\ldots b_4 0 b_2 0 b_0)_2$ в двоичной системе.

13. $(1.909090\ldots)_{-10} = (0.090909\ldots)_{-10} = \frac{1}{11}$.

14.

$$\begin{array}{rrrrrrrrrrl}
 & & & & & 1 & 1 & 3 & 2 & 1 & [5-4i] \\
 & & & & & 1 & 1 & 3 & 2 & 1 & [5-4i] \\
\hline
 & & & & & 1 & 1 & 3 & 2 & 1 & \\
 & & & & 1 & 1 & 2 & 0 & 2 & & \\
 & & & 1 & 2 & 1 & 2 & 3 & & & \\
 & & 1 & 1 & 3 & 2 & 1 & & & & \\
 & 1 & 1 & 3 & 2 & 1 & & & & & \\
\hline
0 & 1 & 0 & 3 & 1 & 1 & 2 & 0 & 1 & & [9-40i]
\end{array}$$

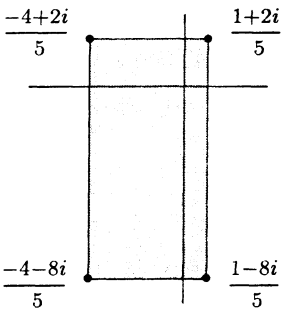


Рис. А–6. Фундаментальная область для мнимочетверичных чисел.

15. $[-\frac{10}{11} \mathinner{..} \frac{1}{11}]$ и прямоугольник справа (рис А-6).

16. Соблазнительно попробовать сделать это самым простым способом, предписав реализацию переносов по правилу $2 = (1100)_{i-1}$, но это приведет к непрекращающейся процедуре (зацикливанию) при попытке добавления единицы к числу $(11101)_{i-1} = -1$.

В приведенном ниже решении предлагаются четыре алгоритма (для сложения и вычитания 1 и i). Если α — строка из нулей и единиц, то полагается, что α^P — такая строка из нулей и единиц, что $(\alpha^P)_{i-1} = (\alpha)_{i-1} + 1$. Аналогичным образом определяются α^{-P}, α^Q, α^{-Q} путем замещения $+1$ соответственно на -1, $+i$ и $-i$. Тогда

$$\begin{array}{llll}
(\alpha 0)^P = \alpha 1; & (\alpha x 1)^P = \alpha^Q x 0. & (\alpha 0)^Q = \alpha^P 1; & (\alpha 1)^Q = \alpha^{-Q} 0. \\
(\alpha x 0)^{-P} = \alpha^{-Q} x 1; & (\alpha 1)^{-P} = \alpha 0. & (\alpha 0)^{-Q} = \alpha^Q 1; & (\alpha 1)^{-Q} = \alpha^{-P} 0.
\end{array}$$

Здесь x означает 0 или 1 и цепочки при необходимости дополняются слева нулями. Все эти процедуры очевидным образом заканчиваются. Таким образом, любое число вида $a + bi$ с целыми a и b представимо в системе по основанию $i - 1$.

17. Нет (вопреки упр. 28); число -1 не представимо в этой системе. Это можно доказать, построив соответствующее множество S, как на рис. 1. Имеем представления $-i = (0.1111\ldots)_{1+i}$, $i = (100.1111\ldots)_{1+i}$.

18. Пусть S_0 — множество точек $(a_7a_6a_5a_4a_3a_2a_1a_0)_{i-1}$, в котором каждое a_k есть 0 или 1. (Поэтому множество S_0 состоит из 256 внутренних точек, отмеченных на рис. 1, после 16-кратного увеличения.) Покажем сначала, что множество S замкнутое. Пусть $y_1, y_2, \ldots$ — произвольная последовательность точек множества S. Имеется

$$y_n = \textstyle\sum_{k\ge 1} a_{nk} 16^{-k},$$

где каждое a_{nk} принадлежит S_0. Построим дерево, вершинами которого будут наборы $(a_{n1}, \ldots, a_{nr})$ для $1 \le r \le n$, причем одна вершина является родителем, если она является по отношению к этой вершине начальным поднабором. Согласно теореме о бесконечных деревьях (теорема 2.3.4.3К) это дерево имеет бесконечный путь $(a_1, a_2, a_3, \ldots)$; соответственно $\sum_{k\ge 1} a_k 16^{-k}$ есть предельная точка множества $\{y_1, y_2, \ldots\}$, принадлежащая S.

В соответствии с ответом к упр. 16 все числа вида $(a+bi)/16^k$ представимы, если a и b целые. Поэтому при действительных x и y и $k \ge 1$ число $z_k = (\lfloor 16^k x \rfloor + \lfloor 16^k y \rfloor i)/16^k$ для некоторых целых m и n принадлежит $S + m + ni$. Можно показать, что всякая представимая точка вне S должна принадлежать множеству $S + m + ni$ при $(m, n) \ne (0, 0)$. Соответственно, если $|x|$ и $|y|$ фиксированы и k достаточно велико, получаем, что $z_k \in S$ и $\lim_{k\to\infty} z_k = x + yi$ принадлежит множеству S.

[B. Мандельброт (B. Mandelbrot) назвал множество S двуглавым драконом, подметив, что оно, по существу, формируется путем объединения двух "кривых дракона" (см. книгу

Fractals: Form, Chance, and Dimension (San Francisco: Freeman, 1977), 313–314, в которой Мандельброт установил, что размерность границы равна $2\lg x \approx 1.523627$, где $x = 1 + 2x^{-2} \approx 1.69562$. Другие свойства кривой дракона описаны в статье C. Davis and D. E. Knuth *J. Recr. Math.* **3** (1970), 66–81, 133–149. Множества S систем счисления по основаниям $\{0,1\}$ и другим комплексным основаниям построены и проанализированы Д. Гоффином (D. Goffinet) в *AMM* **98** (1991), 249–255.]

В статье I. Kátai and J. Szabó, *Acta Scient. Math.* **37** (1975), 255–260, показано, что основание $-d+i$ приводит к системам счисления с цифрами $\{0, 1, \ldots, d^2\}$. Другие свойства таких систем исследовались У. Дж. Гильбертом (W. J. Gilbert) в *Canadian J. Math.* **34** (1982), 1335–1348; *Math. Magazine* **57** (1984), 77–81. В. Нортон (V. Norton) предложил еще одну интересную систему счисления по основанию $2 + i$ с цифрами $\{0, 1, i, -1, -i\}$ [*Math. Magazine* **57** (1984), 250–251]. С системами счисления, основанными на более общих целых числах, можно ознакомиться в работах I. Kátai and B. Kovács, *Acta Math. Acad. Sci. Hung.* **37** (1981), 159–164, 405–407; *Acta ath. Hung.* **58** (1991), 113–120; A. Pethő, *Studia Scient. Math. Hung.* **27** (1992), 169–172.

19. Если $m > u$ или $m < l$, то найдем такое $a \in D$, что $m \equiv a$ (по модулю b); искомое представление будет представлением $m' = (m - a)/b$, за которым следует a. Заметим, что $m > u$ принадлежит интервалу $l < m' < m$; $m < l$ принадлежит $m < m' < u$, поэтому алгоритм конечен.

[При $b = 2$ решения нет. Представление будет единственным тогда и только тогда, когда $0 \in D$; неоднозначное представление появится, например, когда $D = \{-3, -1, 7\}$, $b = 3$, так как $(\alpha)_3 = (\overline{377}\overline{3}\alpha)_3$. Нетрудно показать, что при $b \geq 3$ имеется точно 2^{b-3} решающих множеств D, в которых $|a| < b$ для всех $a \in D$. Далее, множество $D = \{0,\ 1,\ 2 - \epsilon_2 b^n,\ 3 - \epsilon_3 b^n,\ \ldots,\ b - 2 - \epsilon_{b-2} b^n,\ b - 1 - b^n\}$ порождает единственное представление для всех $b \geq 3$ и $n \geq 1$, где любое ϵ_j есть 0 или 1. См. *Proc. IEEE Symp. Comp. Arith.* **4** (1978), 1–9; *JACM* **29** (1982), 1131–1143.]

20. (a) $0.\bar{1}\bar{1}\bar{1}\ldots = \bar{1}.888\ldots = \bar{1}8.\frac{111}{777}\ldots = \bar{1}8\frac{1}{7}.\frac{222}{666}\ldots = \cdots = \bar{1}8\frac{123456}{765432}.\frac{777}{111}\ldots$ имеет девять представлений. (b) D — дробная часть $.a_1a_2\ldots$, которая всегда принимает значения между $-1/9$ и $+71/9$. Пусть x имеет десять или более D — десятичных представлений. Тогда для достаточно большого k число $10^k x$ имеет десять представлений, отличающихся цифрами, которые расположены левее десятичной точки: $10^k x = n_1 + f_1 = \cdots = n_{10} + f_{10}$, где любое f_j есть D — дробная часть. Ввиду единственности представления целых чисел числа n_j различны, скажем, $n_1 < \cdots < n_{10}$; следовательно, $n_{10} - n_1 \geq 9$, но это число принадлежит интервалу $f_1 - f_{10} \geq 9 > 71/9 - (-1/9)$. Таким образом, мы пришли к противоречию, что и доказывает справедливость утверждения. (c) Любое число вида $0.a_1a_2\ldots$, где любое a_j есть -1 или 8, равно $\bar{1}.a_1'a_2'\ldots$ при $a_j' = a_j + 9$ (более того, оно имеет *еще* 6 представлений $\bar{1}8.a_1''a_2''\ldots$ и т. д.).

21. Такое представление можно получить, используя метод, аналогичный предложенному в тексте раздела для перевода в уравновешенную троичную систему счисления.

В отличие от систем, рассмотренных в упр. 20, нуль может быть представлен бесчисленным количеством способов, которые получаются в результате умножения на степень десять суммы $\frac{1}{2} + \sum_{k\geq 1}(-4\frac{1}{2}) \cdot 10^{-k}$ (или из такого же представления, но с противоположными знаками цифр). Представлениями единицы служат $1\frac{1}{2} - \frac{1}{2}^*$, $\frac{1}{2} + \frac{1}{2}^*$, $5 - 3\frac{1}{2} - \frac{1}{2}^*$, $5 - 4\frac{1}{2} + \frac{1}{2}^*$, $50 - 45 - 3\frac{1}{2} - \frac{1}{2}^*$, $50 - 45 - 4\frac{1}{2} + \frac{1}{2}^*$ и др., где $\pm\frac{1}{2}^* = (\pm 4\frac{1}{2})(10^{-1} + 10^{-2} + \cdots)$. [*AMM* **57** (1950), 90–93.]

22. Полагая, что имеется приближение $b_n \ldots b_1 b_0$ с погрешностью $\sum_{k=0}^{n} b_k 10^k - x > 10^{-t}$, где $t > 0$, покажем, как уменьшить ошибку примерно в 10^{-t} раз. (Процесс может быть начат с любого приближения, для которого $\sum_{k=0}^{n} b_k 10^k > x$; далее через конечное количество итераций ошибка станет меньше ϵ.) Просто выбираем $m > n$ настолько

большим, чтобы десятичное представление числа $-10^m\alpha$ имело единицу в позиции 10^{-t} и не имело единиц в позициях $10^{-t+1}, 10^{-t+2}, \ldots, 10^n$. Тогда $10^m\alpha +$ (некоторая подходящая сумма степеней 10 между 10^m и 10^n) $+ \sum_{k=0}^n b_k 10^k \approx \sum_{k=0}^n b_k 10^k - 10^{-t}$.

23. Пусть множество $S = \{\sum_{k\geq 1} a_k b^{-k} \mid a_k \in D\}$ замкнуто (как в упр. 18), следовательно, оно измеримо. Так как $bS = \bigcup_{a\in D}(a+S)$, имеем $b\mu(S) = \mu(bS) \leq \sum_{a\in D}\mu(a+S) = \sum_{a\in D}\mu(S) = b\mu(S)$, и поэтому должно быть справедливо $\mu((a+S)\cap(a'+S)) = 0$, если $a \neq a' \in D$. Тогда множество T — множество меры нуль, если $0 \in D$, так как множество T является объединением множеств вида $b^k(n + ((a+S)\cap(a'+S)))$, $a \neq a'$, каждое из которых — меры нуль. С другой стороны, как отмечал К. А. Брэкк (K. A. Brakke), каждое вещественное число (в системе счисления, рассмотренной в упр. 21) имеет бесконечное количество представлений.

[Множество T не может быть пустым, поэтому вещественные числа не могут быть записаны как счетное объединение замкнутых, разомкнутых и граничных множеств (см. *AMM* **84** (1977), 827–828; более детальный анализ приводится в работе Petkovšek, *AMM* **97** (1990), 408–411). Если множество D состоит из элементов, меньших b, то множество представлений чисел по основанию b и цифры из множества D имеют меру нуль. Если множество D состоит из элементов, бóльших, чем b, и из вещественных чисел, то оно имеет бесконечную меру.]

24. $\{2a\cdot 10^k + a' \mid 0 \leq a < 5, 0 \leq a' < 2\}$ или $\{5a'\cdot 10^k + a \mid 0 \leq a < 5, 0 \leq a' < 2\}$ для $k \geq 0$. [Р. Л. Грэхэм (R. L. Graham) показал, что не существует другого множества целых цифр, удовлетворяющих этим свойствам. Эндрю Одлыжко (Andrew Odlyzko) доказал, что ограничение в рассмотрении множеств целых чисел излишне в том смысле, что если два наименьших элемента множества D являются 0 и 1, то все цифры должны быть целыми.

Доказательство. Пусть $S = \{\sum_{k<0} a_k b^k \mid a_k \in D\}$ — множество "дробных частей" и пусть $X = \{(a_n \ldots a_0)_b \mid a_k \in D\}$ — множество "полных чисел". Тогда $[0\,..\,\infty) = \bigcup_{x\in X}(x+S)$ и $(x+S)\cap(x'+S)$ при $x \neq x' \in X$ имеет меру нуль. Получим $(0\,..\,1) \subseteq S$ и докажем индукцией по m, что $(m\,..\,m+1) \subseteq x_m + S$ для некоторого $x_m \in X$. Пусть $x_m \in X$ таково, что для любого $\epsilon > 0$ мера $(m\,..\,m+\epsilon)\cap(x_m+S)$ положительна. Тогда $x_m \leq m$ и x_m должно быть целым независимо от величины перекрытия множеством $x_{\lfloor x_m\rfloor}+S$ множества x_m+S. Если $x_m > 0$, то из того, что $(m-x_m\,..\,m-x_m+1)\cap S$ имеет положительную меру, по индукции следует, что эта мера равна 1 и $(m\,..\,m+1) \subseteq x_m + S$, так как множество S замкнутое. При $x_m = 0$ и $(m\,..\,m+1) \not\subseteq S$ мы должны получить $m < x'_m < m+1$ для любого $x'_m \in X$, где $(m\,..\,x'_m) \subseteq S$; но тогда $1+S$ перекрывает $x'_m + S$. (См. *Proc. London Math. Soc.* (3) **18** (1978), 581–595.)]

Примечание. Если снять ограничение $0 \in D$, *возникнет* много других достаточно интересных ситуаций, в частности $\{1,2,3,4,5,6,7,8,9,10\}$, $\{1,2,3,4,5,51,52,53,54,55\}$ и $\{2,3,4,5,6,52,53,54,55,56\}$. Если же допустить наличие отрицательных цифр, то при помощи метода, описанного в упр. 19, можно найти много других решений задачи, а также множества, содержащие необычные числа наподобие $\{-1,0,1,2,3,4,5,6,7,18\}$, которые не удовлетворяют оговоренным условиям. Появляются предпосылки для поиска изящных решений для множеств с отрицательными цифрами.

25. Положительное число, представление которого по основанию b содержит m последовательных цифр $(b-1)$, расположенных справа от разделяющей точки, должно иметь вид $c/b^n + (b^m-\theta)/b^{n+m}$, где c и n — неотрицательные целые числа и $0 < \theta \leq 1$. Поэтому, если u/v имеет такой вид, значит, равенство $b^{m+n}u = b^m cv + b^m v - \theta v$ выполнено. Следовательно, θv есть целое число, кратное b^m. Однако $0 < \theta v \leq v < b^m$. (При $0 \leq a < b-1$ могут встречаться произвольно длинные ряды цифр *aaaaa*, например, в представлении чисел $a/(b-1)$.)

26. Доказательство достаточности получается непосредственным обобщением на случай основания b обычного доказательства. Доказательство необходимости разбивается на две части. Если для некоторого n число β_{n+1} больше $\sum_{k\le n} c_k\beta_k$, то для малых ϵ число $\beta_{n+1}-\epsilon$ не допускает такого представления. Если $\beta_{n+1} \le sum_{k\le n} c_k\beta_k$ для всех n, но равенство выполняется не всегда, то можно показать, что для некоторого x существуют два представления (см. *Transactions of the Royal Society of Canada*, series III, **46** (1952), 45–55).

27. Доказательство выполняется индукцией по n. Если n четно, то должно быть $e_0 > 0$ и искомый результат получается по индукции, так как $n/2$ имеет единственное представление такого типа. Если n нечетно, то должно быть $e_0 = 0$ и задача сводится к представлению числа $-(n-1)/2$. Если это последнее равно либо 0, либо 1, то, очевидно, существует только один способ решения задачи. В противном случае по индукции доказывается, что число имеет единственное представление.

[Отсюда следует, что любое положительное целое число имеет ровно *два* таких представления с убывающим порядком $e_0 > e_1 > \cdots > e_t$: одно — с четным t, другое — с нечетным t.]

28. Доказательство может быть выполнено, как и в упр. 27. Обратите внимание, что $a+bi$ представляет собой произведение $1+i$ и некоторого комплексного целого числа тогда и только тогда, когда $a+b$ четное. Такое представление неявно связано с "кривой дракона", описанной в ответе к упр. 18.

29. Достаточно доказать, что любую совокупность $\{T_0,T_1,T_2,\ldots\}$, удовлетворяющую свойству B, можно получить с помощью "стягивания" некоторой совокупности $\{S_0, S_1, S_2, \ldots\}$, где $S_0 = \{0, 1, \ldots, b-1\}$, и что все элементы множеств S_1, S_2, ... кратны b.

При доказательстве последнего утверждения можно считать, что $1 \in T_0$ и существует наименьший элемент $b > 1$, такой, что $b \notin T_0$. Индукцией по n докажем, что если $nb \notin T_0$, то $nb+1$, $nb+2$, ..., $nb+b-1$ не принадлежат никакому из множеств T_j; если же $nb \in T_0$, то же самое верно и для чисел $nb+1$, ..., $nb+b-1$. Тогда искомой совокупностью будет $S_1 = \{nb \mid nb \in T_0\}$, $S_2 = T_1$, $S_3 = T_2$ и т. д., откуда следует результат.

Если $nb \notin T_0$, то $nb = t_0 + t_1 + \cdots$, где t_1, t_2, ... кратны b. Следовательно, $t_0 < nb$ кратно b. По индукции $(t_0+k)+t_1+t_2+\cdots$ есть представление числа $nb+k$ при $0 < k < b$, поэтому $nb+k \notin T_j$ для любого j.

Если $nb \in T_0$ и $0 < k < b$, то $t_0+t_1+\cdots$. Равенство $t_j = nb+k$ не может выполняться для $j \ge 1$, иначе $nb+b$ имело бы два представления $(b-k)+\cdots+(nb+k)+\cdots = (nb)+\cdots+b+\cdots$. По индукции $t_0 \bmod b = k$. Из представления $nb = (t_0-k)+t_1+\cdots$ следует $t_0 = nb+k$.

[См. *Nieuw Archief voor Wiskunde* (3) **4** (1956), 15–17. Конечный результат получен P. A. Мак-Магоном (P. A. MacMahon), *Combinatory Analysis* **1** (1915), 217–223.]

30. (a) Пусть A_j — множество чисел n, в представлении которых не содержится b_j; тогда согласно свойству единственности $n \in A_j$ тогда и только тогда, когда $n + b_j \notin A_j$. Следовательно, $n \in A_j$ тогда и только тогда, когда $n + 2b_jb_k \in A_j \cap A_k$. Пусть m — количество целых чисел $n \in A_j \cap A_k$, таких, что $0 \le n < 2b_jb_k$. Значит, в том же интервале найдется ровно m целых чисел, принадлежащих A_j, но не принадлежащих A_k, и ровно m, не принадлежащих ни A_j, ни A_k; поэтому $4m = 2b_jb_k$. Следовательно, b_j и b_k не могут быть нечетными одновременно. Однако одно из них, разумеется, нечетно, так как нечетные числа допускают представление в бинарном базисе.

(b) Согласно п. (a) можно так перенумеровать числа b, чтобы b_0 было нечетным, а b_1, b_2, ... — четными. Тогда ряд $\frac{1}{2}b_1$, $\frac{1}{2}b_2$, ... должен также образовывать базис и эту процедуру можно повторить.

(c) Если имеется бинарный базис, то для представления числа $\pm 2^n$ (для больших n) при достаточно больших k необходимо получить и положительные, и отрицательные d_k. Доказательство обратного утверждения приводится в следующем алгоритме.

S1. [Начальная установка.] Установить $k \leftarrow 0$.

S2. [Выполнить?] Если $n = 0$, завершить выполнение алгоритма.

S3. [Выбрать.] Если число n четное, установить $n \leftarrow n/2$; иначе — включить в представление $2^k d_k$ и установить $n \leftarrow (n - d_k)/2$.

S4. [Увеличить k.] Увеличить k на 1 и перейти к шагу S2. ▮

На шаге S3 приведенного алгоритма $|n|$ уменьшается до тех пор, пока не выполнится равенство $n = -d_k$; следовательно, алгоритм должен завершиться.

(d) Две итерации шагов S2–S4 алгоритма приводят к преобразованию $4m \to m$, $4m + 1 \to m + 5$, $4m + 2 \to m + 7$, $4m + 3 \to m - 1$. Рассуждая, как в упр. 19, остается только показать, что алгоритм завершит работу при $-2 \le n \le 8$; все остальные значения числа n сдвигаются в этот интервал. Для данного интервала имеем следующую структуру дерева: $3 \to -1 \to -2 \to 6 \to 8 \to 2 \to 7 \to 0$ и $4 \to 1 \to 5 \to 6$. Таким образом, $1 = 7 \cdot 2^0 - 13 \cdot 2^1 + 7 \cdot 2^2 - 13 \cdot 2^3 - 13 \cdot 2^5 - 13 \cdot 2^9 + 7 \cdot 2^{10}$.

Примечание. Выбор $d_0, d_1, d_2, \ldots = 5, -3, 3, 5, -3, 3, \ldots$ также дает бинарный базис. Более подробно с этим вопросом можно ознакомиться в работах *Math. Comp.* **18** (1964), 537–546; A. D. Sands, *Acta Math. Acad. Sci. Hung.* **8** (1957), 65–86.

31. (См. относящиеся к этому вопросу упр. 3.2.2–11, 4.3.2–13, 4.6.2–22.)

(a) Умножая числитель и знаменатель на подходящую степень 2, можно полагать, что $u = (\ldots u_2 u_1 u_0)_2$ и $v = (\ldots v_2 v_1 v_0)_2$, где $v_0 = 1$, являются 2-адическими целыми числами. Для определения w можно прибегнуть к следующему вычислительному методу, используя для целого числа $(u_{n-1} \ldots u_0)_2 = u \bmod 2^n$ обозначение $u^{(n)}$ $(n > 0)$.

Пусть $w_0 = u_0$ и $w^{(1)} = w_0$. Для $n = 1, 2, \ldots$ полагаем, что найдено целое число $w^{(n)} = (w_{n-1} \ldots w_0)_2$, такое, что $u^{(n)} \equiv v^{(n)} w^{(n)}$ (по модулю 2^n). Тогда $u^{(n+1)} \equiv v^{(n+1)} w^{(n)}$ (по модулю 2^n). Поэтому можно положить $w_n = 0$ или 1 в соответствии с тем, чему равно число $(u^{(n+1)} - v^{(n+1)} w^{(n)}) \bmod 2^{n+1}$: нулю или 2^n.

(b) Найдем наименьшее целое число k, такое, что $2^k \equiv 1$ (по модулю $2n + 1$). Тогда $1/(2n + 1) = m/(2^k - 1)$ для некоторого целого числа m, $1 \le m < 2^{k-1}$. Пусть α есть k-разрядное двоичное представление для m; тогда в двоичной системе число $(0.\alpha\alpha\alpha\ldots)_2$, умноженное на $2n + 1$, равно $(0.111\ldots)_2 = 1$, а в 2-адической системе $(\ldots\alpha\alpha\alpha)_2$, умноженное на $2n + 1$, равно $(\ldots 111)_2 = -1$.

(c) Если u рационально, скажем, $u = m/(2^e n)$, где n — нечетное и положительное число, то 2-адическое представление числа u периодично, так как множество чисел с периодическим разложением содержит $-1/n$ и замкнуто относительно операций изменения знака, деления на 2 и сложения. Наоборот, если для всех $N \ge \mu$ соблюдается $u_{N+\lambda} = u_N$, то 2-адическое представление числа $(2^\lambda - 1)2^{-\mu} u$ есть целое число.

(d) Квадрат любого числа вида $(\ldots u_2 u_1 1)_2$ имеет вид $(\ldots 001)_2$, поэтому сформулированное условие является необходимым. Достаточность доказывается наличием следующей процедуры вычисления $v = \sqrt{n}$ для случая, когда $n \bmod 8 = 1$.

H1. [Начальная установка.] Установить $m \leftarrow (n - 1)/8$, $k \leftarrow 2$, $v_0 \leftarrow 1$, $v_1 \leftarrow 0$, $v \leftarrow 1$. (При выполнении этого алгоритма получим $v = (v_{k-1} \ldots v_1 v_0)_2$ и $v^2 = n - 2^{k+1} m$.)

H2. [Преобразования.] Если m четно, установить $v_k \leftarrow 0$, $m \leftarrow m/2$. В противном случае установить $v_k \leftarrow 1$, $m \leftarrow (m - v - 2^{k-1})/2$, $v \leftarrow v + 2^k$.

H3. [Приращение k.] Увеличить k на 1 и вернуться к шагу H2. ▮

32. Более общий результат опубликован в журнале *Math. Comp.* **29** (1975), 84–86.

33. Пусть K_n — множество всех таких n-разрядных чисел, что $k_n = |K_n|$. Если множества S и T являются произвольными конечными множествами целых чисел, можно утверждать, что для некоторого целого числа x $S \sim T$ при $S = T + x$, и можно записать $k_n(S) = |\mathcal{K}_n(S)|$, где $\mathcal{K}_n(S)$ — семейство всех подмножеств K_n, принадлежащих $\sim S$. При $n = 0$ выполняется

$k_n(S) = 0$, однако $|S| \le 1$, так как нуль является единственным 0-разрядным числом. При $n \ge 1$ и $S = \{s_1, \ldots, s_r\}$ имеем

$$\mathcal{K}_n(S) = \bigcup_{0 \le j < b} \bigcup_{(a_1,\ldots,a_r)} \{\{t_1 b + a_1, \ldots, t_r b + a_r\} \mid \{t_1, \ldots, t_r\} \in K_{n-1}(\{(s_i + j - a_i)/b \mid 1 \le i \le r\})\},$$

где внутреннее объединение представляет собой полные последовательности цифр $(a_1, \ldots, a_r)$, удовлетворяющих условию $a_i \equiv s_i + j$ (по модулю b) при $1 \le i \le r$. Потребуем для однозначности определения индексов в этой формуле, чтобы $t_i - t_{i'} = (s_i - a_i)/b - (s_{i'} - a_{i'})/b$ при $1 \le i < i' \le r$. Согласно принципу включения и исключения получаем $k_n(S) = \sum_{0 \le j < b} \sum_{m \ge 1} (-1)^{m-1} f(S, m, j)$, где $f(S, m, j)$ — количество множеств целых чисел, которые могут быть выражены в виде $\{t_1 b + a_1, \ldots, t_r b + a_r\}$ описанным выше способом применительно к m различным последовательностям $(a_1, \ldots, a_r)$, причем выражение суммируется по всем выборкам из m различных последовательностей $(a_1, \ldots, a_r)$. Для полученных m различных последовательностей $(a_1^{(l)}, \ldots, a_r^{(l)})$ при $1 \le l \le m$ количество таких множеств равно $k_{n-1}(\{(s_i + j - a_i^{(l)})/b \mid 1 \le i \le r, 1 \le l \le m\})$. Итак, имеем набор множеств $\mathcal{T}(S)$, таких, что

$$k_n(S) = \sum_{T \in \mathcal{T}(S)} c_T \, k_{n-1}(T),$$

где каждое c_T есть целое число. Более того, если множество $T \in \mathcal{T}(S)$, то его элементы близки к элементам множества S и имеем

$$\min T \ge (\min S - \max D)/b \qquad \text{и} \qquad \max T \le (\max S + b - 1 - \min D)/b.$$

Таким образом, получены рекуррентные соотношения для последовательностей множеств $\langle k_n(S) \rangle$, в которых множество S порождает подмножества целых чисел $[l \,..\, u + 1]$ (в обозначениях упр. 19). Последовательность $\langle k_n \rangle$ появляется в процессе формирования этих рекуррентных соотношений, так как $k_n = k_n(S)$ для любого элемента множества S. Коэффициенты c_T могут быть вычислены через несколько начальных значений $k_n(S)$, так что можно получить систему уравнений для определения производящих функций $k_S(z) = \sum k_n(S) z^n = [|S| \le 1] + z \sum_{T \in \mathcal{T}(S)} c_T k_T(z)$ (см. *J. Algorithms* **2** (1981), 31–43).

Например, при $D = \{-1, 0, 3\}$ и $b = 3$ имеем $l = -\frac{3}{2}$ и $u = \frac{1}{2}$, так что искомыми множествами S являются $\{0\}$, $\{0, 1\}$, $\{-1, 1\}$ и $\{-1, 0, 1\}$. Соответствующие соотношения для $n \le 3$ имеют вид $\langle 1, 3, 8, 21 \rangle$, $\langle 0, 1, 3, 8 \rangle$, $\langle 0, 0, 1, 4 \rangle$ и $\langle 0, 0, 0, 0 \rangle$. Итак, получено

$$k_0(z) = 1 + z(3k_0(z) - k_{01}(z)), \qquad k_{02}(z) = z(k_{01}(z) + k_{02}(z)),$$
$$k_{01}(z) = zk_0(z), \qquad k_{012}(z) = 0$$

и $k(z) = 1/(1 - 3z + z^2)$. Тогда $k_n = F_{2n+2}$ и $k_n(\{0, 2\}) = F_{2n-1} - 1$.

34. Существует единственная цепочка α_n символов $\{\overline{1}, 0, 1\}$, такая, что $n = (\alpha_n)_2$ и в α_n нет ни ведущих нулей, ни последовательных ненулевых элементов: α_0 пусто; в противном случае $\alpha_{2n} = \alpha_n 0$, $\alpha_{4n+1} = \alpha_n 01$, $\alpha_{4n-1} = \alpha_n 0\overline{1}$. Любую цепочку, содержащую n, можно преобразовать в α_n при помощи сжатия $1\overline{1} \to 01$, $\overline{1}1 \to 0\overline{1}$, $01\ldots11 \to 10\ldots0\overline{1}$, $0\overline{1}\ldots\overline{1}1 \to \overline{1}0\ldots01$ и добавления или исключения ведущих нулей. Так как операции сжатия не увеличивают количество цифр, отличных от нуля, то в α_n содержится наименьшее количество. [*Advances in Computers* **1** (1960), 244–260.] Количество $\overline{\nu}(n)$ ненулевых цифр в α_n — это количество единиц в обычном представлении, перед которыми сразу же помещается либо нуль, либо подстрока $00(10)^k1$ для некоторого $k \ge 0$.

Обобщение для систем представления по основанию $b > 2$ предложено Й. фон цур Гатеном (J. von zur Gathen), *Computational Complexity* **1** (1991), 360–394.

РАЗДЕЛ 4.2.1

1. $N = (62, +.60\ 22\ 14\ 00)$; $h = (37, +.66\ 26\ 10\ 00)$. Обратите внимание на то, что значение $10h$ имело бы вид $(38, +.06\ 62\ 61\ 00)$.

2. $b^{E-q}(1-b^{-p})$, b^{-q-p}; $b^{E-q}(1-b^{-p})$, b^{-q-1}.

3. Если e не принимает своего наименьшего значения, то наиболее значимый разряд, в котором во всех таких нормализованных числах стоит единица, можно не включать в машинное слово (т. е. не хранить, а подразумевать при выполнении операций).

4. $(51, +.10209877)$; $(50, +.12346000)$; $(53, +.99999999)$. Если бы первый операнд равнялся $(45, -.50000000)$, то третий ответ был бы $(54, +.10000000)$, поскольку $b/2$ нечетно.

5. Если $x \sim y$ и m есть целое число, то $mb+x \sim mb+y$. Более того, если рассмотреть все возможные случаи, то окажется, что из $x \sim y$ следует $x/b \sim y/b$. Другое важное свойство состоит в том, что x и y будут округлены до одного и того же целого при любых $x \sim y$.

Теперь, если $b^{-p-2}F_v \neq f_v$, необходимо получить $(b^{p+2}f_v) \bmod b \neq 0$. Следовательно, преобразование оставляет f_v неизменным до тех пор, пока $e_u - e_v \geq 2$. Поскольку u не нормализовано, оно отлично от нуля и $|f_u + f_v| > b^{-1} - b^{-2} \geq b^{-2}$. Ведущий, отличный от нуля разряд $f_u + f_v$ должен находиться не более чем на два разряда правее разделяющей точки, и операция округления преобразует $b^{p+j}(f_u + f_v)$ в целое, где $j \leq 1$. Доказательство будет завершено, если показать, что $b^{p+j+1}(f_u+f_v) \sim b^{p+j+1}(f_u+b^{-p-2}F_v)$. Из предыдущего абзаца имеем $b^{p+2}(f_u + f_v) \sim b^{p+2}f_u + F_v = b^{p+2}(f_u + b^{-p-2}F_v)$, из чего следует искомый результат для всех $j \leq 1$. Аналогичное замечание справедливо и в отношении шага M2 алгоритма M.

Обратите внимание, что, когда $b > 2$ четное, такое целое число F_v существует всегда; но если $b = 2$, потребуется $p + 3$ бит (полагаем $2F_v$ целым). Если b нечетно, то целое F_v существует всегда, кроме случая деления в алгоритме M, когда возможен остаток $\frac{1}{2}b$.

6. (Рассмотрите случай, когда $e_u = e_v$, $f_u = -f_v$, в программе A.) Регистр A сохранит предыдущее значение знака, как и при выполнении команды `ADD`.

7. Будем говорить, что число нормализовано тогда и только тогда, когда оно равно нулю или его дробная часть находится в диапазоне $\frac{1}{6} < |f| < \frac{1}{2}$. Аккумулятор на $(p + 1)$ разрядов вполне подойдет для операций сложения и вычитания; округление (кроме случая деления) эквивалентно отбрасыванию лишних разрядов. Кажется, получилась очень привлекательная система! Можно использовать представление чисел порядком с избытком нуль, вставленным между первым и последующими разрядами дробной части и дополненным, если дробная часть отрицательна. Так что сохраняется порядок, принятый для чисел с фиксированной точкой.

8. (a) $(06, +.12345679) \oplus (06, -.12345678)$, $(01, +.10345678) \oplus (00, -.94000000)$;
(b) $(99, +.87654321) \oplus$ оно же, $(99, +.99999999) \oplus (91, +.50000000)$.

9. $a = c = (-50, +.10000000)$, $b = (-41, +.20000000)$, $d = (-41, +.80000000)$, $y = (11, +.10000000)$.

10. $(50, +.99999000) \oplus (55, +.99999000)$.

11. $(50, +.10000001) \otimes (50, +.99999990)$.

12. Если $0 < |f_u| < |f_v|$, то $|f_u| \leq |f_v| - b^{-p}$; следовательно, $1/b < |f_u/f_v| \leq 1 - b^{-p}/|f_v| < 1 - b^{-p}$. Если $0 < |f_v| \leq |f_u|$, имеем $1/b < |f_u/f_v|/b \leq ((1 - b^{-p})/(1/b))/b = 1 - b^{-p}$.

13. См. J. Michael Yohe, *IEEE Transactions* **C-22** (1973), 577–586, а также упр. 4.2.2–24.

14.
```
FIX STJ  9F          Подпрограмма ПТ-в-ФТ.
    STA  TEMP
    LD1  TEMP(EXP)   rI1 ← e.
    SLA  1           rA ← ± f f f f 0.
```

```
        JAZ   9F          Нуль на входе?
        DEC1  1
        CMPA  =0=(1:1)    Если ведущий байт равен нулю,
        JE    *-4           снова сместить влево.
        ENN1  -Q-4,1
        J1N   FIXOVFLO    Абсолютное значение слишком велико?
        ENTX  0
        SRAX  0,1
        CMPX  =1//2=
        JL    9F
        JG    *+2
        JAO   9F          В некоторых случаях становится нечетным,
                            поскольку b/2 четно.
        STA   *+1(0:0)    При необходимости округлить.
        INCA  1           Add ±1 (переполнение невозможно).
    9H  JMP   *           Выход из подпрограммы. ▌
```

15.
```
    FP  STJ   EXITF       Подпрограмма для выделения дробной части.
        JOV   OFLO        Снятие блокировки переполнения.
        STA   TEMP        TEMP ← u.
        ENTX  0
        SLA   1           rA ← f_u.
        LD2   TEMP(EXP)   rI2 ← e_u.
        DEC2  Q
        J2NP  *+3
        SLA   0,2         Удалить целую часть u.
        ENT2  0
        JANN  1F
        ENN2  0,2         Дробная часть отрицательна: найти ее
        SRAX  0,2           дополнение.
        ENT2  0
        JXNZ  *+3
        JAZ   *+2
        INCA  1
        ADD   WM1         Добавить размер слова минус единица.
    1H  INC2  Q           Подготовка к нормализации результата.
        JMP   NORM        Нормализовать, округлить и выйти из подпрограммы.
    8H  EQU   1(1:1)
    WM1 CON   8B-1,8B-1(1:4)  Размер слова минус единица ▌
```

16. Если $|c| \geq |d|$, то установить

$$r \leftarrow d \oslash c, \quad s \leftarrow c \oplus (r \otimes d); \qquad x \leftarrow (a \oplus (b \otimes r)) \oslash s, \quad y \leftarrow (b \ominus (a \otimes r)) \oslash s.$$

В противном случае установить

$$r \leftarrow c \oslash d, \quad s \leftarrow d \oplus (r \otimes c); \qquad x \leftarrow ((a \otimes r) \oplus b) \oslash s, \quad y \leftarrow ((b \otimes r) \ominus a) \oslash s.$$

Тогда $x + iy$ есть искомая аппроксимация $(a + bi)/(c + di)$. [*CACM* **5** (1963), 435. Другие алгоритмы для выполнения арифметических операций с комплексными числами и вычисления соответствующих функций описаны в работе P. Wynn, *BIT* **2** (1962), 232–255; см. также Paul Friedland, *CACM* **10** (1967), 665.]

17. (См. Robert Morris, *IEEE Transactions* **C-20** (1971), 1578–1579.) Для такой системы анализ ошибки более сложен, поскольку здесь предпочтительнее использовать арифметику интервалов.

18. Для положительных чисел решение таково. Сдвигать дробную часть влево до тех пор, пока не окажется, что $f_1 = 1$; затем округлить, затем, если дробная часть равна нулю (переполнение при округлении), снова сдвинуть вправо. Для отрицательных чисел решение таково. Сдвигать дробную часть влево до тех пор, пока не окажется, что $f_1 = 0$; затем округлить, затем, если дробная часть равна нулю (исчезновение значимости при округлении), снова сдвинуть вправо.

19. $(43 + [e_v < e_u] - [\text{переполнение дробной части}] - 10\,[\text{результат} - \text{нуль}] + 4\,[\text{абсолютная величина округлена до}] + [\text{первый округленный разряд равен } \frac{b}{2}] + 5\,[\text{округленные разряды равны } \frac{b}{2}0\ldots0] + 7\,[\text{переполнение при округлении}] + 7N + (11\,[N>0\,] - 1\,)\,[\text{rX принимает ненулевые разряды}])u$, где N есть число сдвигов влево при нормализации. Максимальное время $73u$ получается, если, например,

$$u = +50\ 01\ 00\ 00\ 00, \quad v = -46\ 49\ 99\ 99\ 99, \quad b = 100.$$

[Учитывая данные из раздела 4.2.4, время будет иметь порядок $45.5u$.]

РАЗДЕЛ 4.2.2

1. $u \ominus v = u \oplus -v = -v \oplus u = -(v \oplus -u) = -(v \ominus u)$.

2. $u \oplus x \geq u \oplus 0 = u$, как следует из (8), (2), (6), поэтому снова в силу (8) $(u \oplus x) \oplus v \geq u \oplus v$. Аналогично из (8) и (6) вместе с (2) следует, что $(u \oplus x) \oplus (v \oplus y) \geq (u \oplus x) \oplus v$.

3. $u = 8.0000001$, $v = 1.2500008$, $w = 8.0000008$; $(u \otimes v) \otimes w = 80.000064$, $u \otimes (v \otimes w) = 80.000057$.

4. Да; пусть $1/u \approx v = w$, где v велико.

5. Не всегда, например для чисел $u = v = 9$ в десятичной системе.

6. (a) Да. (b) Только для $b + p \leq 4$ (попробуйте проверить с $u = 1 - b^{-p}$). Но см. упр. 27.

7. Если u и v — последовательные числа с плавающей точкой, $u \oplus v = 2u$ или $2v$. Когда это $2v$, мы часто получаем $u^{②} \oplus v^{②} < 2v^{②}$. Например, $u = (.10\ldots001)_2$, $v = (.10\ldots010)_2$, $u \oplus v = 2v$ и $u^{②} + v^{②} = (.10\ldots011)_2$.

8. (a) $\sim, \approx$; (b) $\sim, \approx$; (c) $\sim, \approx$; (d) $\sim$; (e) $\sim$.

9. $|u-w| \leq |u-v|+|v-w| \leq \epsilon_1 \min(b^{e_u-q}, b^{e_v-q}) + \epsilon_2 \min(b^{e_v-q}, b^{e_w-q}) \leq \epsilon_1 b^{e_u-q} + \epsilon_2 b^{e_w-q} \leq (\epsilon_1 + \epsilon_2)\max(b^{e_u-q}, b^{e_w-q})$. В общем случае усилить этот результат нельзя. Например, можно было бы взять e_u очень малым по сравнению с обоими порядками e_v и e_w, но это означало бы, что $u - w$ слишком велико при сделанных предположениях.

10. Если $a_p \geq 1$ и $a_1 \geq \frac{b}{2}$, получим $(.a_1\ldots a_{p-1}a_p)_b \otimes (.9\ldots99)_b = (.a_1\ldots a_{p-1}(a_p-1))_b$; здесь "9" есть $b-1$. Далее, $(.a_1\ldots a_{p-1}a_p)_b \otimes (1.0\ldots0)_b = (.a_1\ldots a_{p-1}0)_b$, так что, если $b > 2$ и $a_p \geq 1 + [a_1 \geq \frac{b}{2}]$, умножение оказывается не монотонной операцией. Но в случае, когда $b = 2$, используя то же самое рассуждение, можно доказать, что умножение является монотонным. Очевидно, "некий компьютер" имеет $b > 2$.

11. Можно положить без ущерба для общности рассуждений, что x есть целое, $0 \leq x < b^p$. Если $e \leq 0$, то $t = 0$. Если $0 < e \leq p$, то $x - t$ имеет не более $p+1$ разрядов, причем наименее значимый равен нулю. Если $e > p$, то $x - t = 0$. [Результат справедлив и при более слабой исходной посылке $|t| < b^e$; в этом случае можно было бы получить $x - t = b^e$, если $e > p$.]

12. Предположим, что $e_u = p$, $e_v \leq 0$, $u > 0$. Случай 1: $u > b^{p-1}$. Случай (1, a): $w = u + 1$, $v \geq \frac{1}{2}$, $e_v = 0$. Тогда $u' = u$ или $u + 1$, $v' = 1$, $u'' = u$, $v'' = 1$ или 0. Случай (1, b):

$w = u$, $|v| \le \frac{1}{2}$. Тогда $u' = u$, $v' = 0$, $u'' = u$, $v'' = 0$. Если $|v| = \frac{1}{2}$ и допустимо более общее правило округления, можно было бы получить $u' = u \pm 1$, $v'' = \mp 1$. Случай (1, c): $w = u - 1$, $v \le -\frac{1}{2}$, $e_v = 0$. Тогда $u' = u$ или $u - 1$, $v' = -1$, $u'' = u$, $v'' = -1$ или 0. Случай 2: $u = b^{p-1}$. Случай (2, a): $w = u + 1$, $v \ge \frac{1}{2}$, $e_v = 0$, как в (1, a). Случай (2, b): $w = u$, $|v| \le \frac{1}{2}$, $u' \ge u$, как в (1, b). Случай (2, c): $w = u$, $|v| \le \frac{1}{2}$, $u' < u$. Тогда $u' = u - j/b$, где $v = j/b + v_1$ и $|v_1| \le \frac{1}{2}b^{-1}$ для некоторого положительного целого $j \le \frac{1}{2}b$; имеем $v' = 0$, $u'' = u$, $v'' = j/b$. Случай (2, d): $w < u$. Тогда $w = u - j/b$, где $v = -j/b + v_1$ и $|v_1| \le \frac{1}{2}b^{-1}$ для некоторого положительного целого $j \le b$; имеем $(v', u'') = (-j/b, u)$ и $(u', v'') = (u, -j/b)$ или $(u - 1/b, (1 - j)/b)$; последний вариант возможен, только если $v_1 = \frac{1}{2}b^{-1}$. В любом случае $u \ominus u' = u - u'$, $v \ominus v' = v - v'$, $u \ominus u'' = u - u''$, $v \ominus v'' = v - v''$, $\mathrm{round}(w - u - v) = w - u - v$.

13. Поскольку $\mathrm{round}(x) = 0$ тогда и только тогда, когда $x = 0$, нужно найти множество пар целых чисел (m, n), обладающих таким свойством: результат $m \oslash n$ есть целое число тогда и только тогда, когда целым числом является m/n. Предположим, что $|m|, |n| < b^p$. Если m/n есть целое, то $m \oslash n = m/n$ — также целое. И наоборот, если m/n не есть целое, а $m \oslash n$ — целое, имеем $1/|n| \le |m \oslash n - m/n| < \frac{1}{2}|m/n|b^{1-p}$. Следовательно, $|m| > 2b^{p-1}$. Значит, ответ таков: нужно потребовать, чтобы выполнялись условия $|m| \le 2b^{p-1}$ и $0 < |n| < b^p$. (Возможно и чуть менее жесткое ограничение.)

14. $|(u \otimes v) \otimes w - uvw| \le |(u \otimes v) \otimes w - (u \otimes v)w| + |w|\,|u \otimes v - uv| \le \delta_{(u\otimes v)\otimes w} + b^{e_w - q - l_w}\delta_{u\otimes v} \le (1 + b)\delta_{(u\otimes v)\otimes w}$. Теперь $|e_{(u\otimes v)\otimes w} - e_{u\otimes(v\otimes w)}| \le 2$; таким образом, в качестве искомого результата можно взять $\epsilon = \frac{1}{2}(1 + b)b^{2-p}$.

15. Из $u \le v$ следует, что $(u \oplus u) \oslash 2 \le (u \oplus v) \oslash 2 \le (v \oplus v) \oslash 2$. Таким образом, сформулированное в задаче условие остается справедливым для любых u и v тогда и только тогда, когда оно справедливо для любых $u = v$. Для основания $b = 2$ исходное условие справедливо всегда (за исключение случая, когда возникает переполнение); но при $b > 2$ существуют числа $v \ne w$, такие, что $v \oplus v = w \oplus w$. Значит, сформулированное в задаче условие не выполняется. [С другой стороны, формула $u \oplus ((v \ominus u) \oslash 2)$ действительно дает среднюю точку, лежащую внутри интервала. *Доказательство.* Достаточно показать, что $u + (v \ominus u) \oslash 2 \le v$, т. е. $(v \ominus u) \oslash 2 \le v - u$; легко проверить, что $\mathrm{round}(\frac{1}{2}\mathrm{round}(x)) \le x$ для любых $x \ge 0$.]

16. (a) Изменение порядка появится в суммах $\sum_{10} = 11.111111$, $\sum_{91} = 101.11111$, $\sum_{901} = 1001.1102$, $\sum_{9001} = 10001.020$, $\sum_{90009} = 100000.91$, $\sum_{900819} = 1000000.0$; таким образом, $\sum_{1000000} = 1109099.1$.

(b) После определения суммы $\sum_{k=1}^{n} 1.2345679 = 1224782.1$ вычисления в соответствии с (14) приведут к попытке определить квадратный корень из $-.0053187053$. Но в этом случае вычисления в соответствии с (15) и (16) будут точными. [Если $x_k = 1 + \lfloor (k-1)/2 \rfloor 10^{-7}$, то вычисления в соответствии с (15) и (16) дадут ошибку порядка n. Более детальный анализ точности вычисления стандартного отклонения приведен в работе Chan and Lewis, *CACM* **22** (1979), 526–531.]

(c) Нужно показать, что $u \oplus ((v \ominus u) \oslash k)$ находится между u и v; см. упр. 15.

17.
```
FCMP STJ  9F          Подпрограмма для сравнения чисел в формате
                        с плавающей точкой.
     JOV  OFLO        Убедиться, что индикатор переполнения выключен.
     STA  TEMP
     LDAN TEMP        v ← −v.
     (Сюда скопируйте строки 07–20 из программы 4.2.1A.)
     LDX  FV(0:0)     Присвоить rX значение "нуль" со знаком, совпадающим
                        со знаком f_v.
```

```
        DEC1 5
        J1N  *+2
        ENT1 0              Заменить большое значение разности порядков
        SRAX 5,1                меньшим.
        ADD  FU             rA ← разность операндов.
        JOV  7F             Переполнение при вычислении дробной части: не ∼.
        CMPA EPSILON(1:5)
        JG   8F             Переход, если нет ∼.
        JL   6F             Переход, если ∼.
        JXZ  9F             Переход, если ∼.
        JXP  1F             Если |rA| = ε, проверить знак rA × rX.
        JAP  9F             Переход, если ∼ (rA ≠ 0).
        JMP  8F
7H      ENTX 1
        SRC  1              Установить в rA ненулевое значение с тем же значением.
        JMP  8F
1H      JAP  8F             Переход, если не ∼ (rA ≠ 0).
6H      ENTA 0
8H      CMPA =0=            Установить индикатор сравнения.
9H      JMP  *              Выход из подпрограммы. ▮
```

19. Пусть $\gamma_k = \delta_k = \eta_k = \sigma_k = 0$ при $k > n$. Достаточно найти коэффициент при x_1, поскольку коэффициент при x_k будет точно таким же, если все индексы увеличить на $k-1$. Пусть (f_k, g_k) означают соответственно коэффициенты при x_1 в $(s_k - c_k, c_k)$. Тогда $f_1 = (1+\eta_1)(1-\gamma_1-\gamma_1\delta_1-\gamma_1\sigma_1-\delta_1\sigma_1-\gamma_1\delta_1\sigma_1)$, $g_1 = (1+\delta_1)(1+\eta_1)(\gamma_1+\sigma_1+\gamma_1\sigma_1)$ и $f_k = (1-\gamma_k\sigma_k-\delta_k\sigma_k-\gamma_k\delta_k\sigma_k)f_{k-1} + (\gamma_k-\eta_k+\gamma_k\delta_k+\gamma_k\eta_k+\gamma_k\delta_k\eta_k+\gamma_k\eta_k\sigma_k+\delta_k\eta_k\sigma_k+\gamma_k\delta_k\eta_k\sigma_k)g_{k-1}$, $g_k = \sigma_k(1+\gamma_k)(1+\delta_k)f_{k-1} - (1+\delta_k)(\gamma_k+\gamma_k\eta_k+\eta_k\sigma_k+\gamma_k\eta_k\sigma_k)g_{k-1}$, где $1 < k \le n$. Тогда $f_n = 1+\eta_1-\gamma_1+(4n$ членов второго порядка$)+($члены более высоких порядков$) = 1+\eta_1-\gamma_1+O(n\epsilon^2)$ будет достаточно мало́. [Формула суммирования Кахана впервые была опубликована в *CACM* **8** (1965), 40; см. также *Proc. IFIP Congress* (1971), **2**, 1232, а продолжение — в работе K. Ozawa, *J. Information Proc.* **6** (1983), 226–230. Другой подход к организации точного суммирования описан в работе R. J. Hanson, *CACM* **18** (1975), 57–58. Когда одни x_k отрицательны, а другие положительны, можно с пользой для дела скомбинировать их, как рекомендуется в работе T. O. Espelid, *SIAM Review* **37** (1995), 603–607. См. также работу G. Bohlender, *IEEE Trans.* **C-26** (1977), 621–632, где приводятся алгоритмы, которые вычисляют $\mathrm{round}(x_1+\cdots+x_n)$ и $\mathrm{round}(x_1\ldots x_n)$ *точно*, по заданным $\{x_1,\ldots,x_n\}$.]

20. Как следует из доказательства теоремы C, соотношение (47) не выполняется при $e_w = p$, только если $|v| + \frac{1}{2} \ge |w-u| \ge b^{p-1}+b^{-1}$, поэтому $|f_u| \ge |f_v| \ge 1-(\frac{1}{2}b-1)b^{-p}$. Теперь приходим к выводу, что необходимое и достаточное условие "невыполнения" (47) есть округление $|f_w|$, по сути, почти до 2 в процессе нормализации (фактически до $2/b$ после масштабирования посредством сдвига вправо для получения переполнения дробной части). Случай, конечно же, чрезвычайно редкий!

21. (Решение Г. В. Вельткампа (G. W. Veltkamp).) Пусть $c = 2^{\lceil p/2\rceil}+1$; можно предположить, что $p \ge 2$, так что c репрезентативно. Сначала вычислим $u' = u \otimes c$, $u_1 = (u \ominus u') \oplus u'$, $u_2 = u \ominus u_1$; аналогично $v' = v \otimes c$, $v_1 = (v \ominus v') \oplus v'$, $v_2 = v \ominus v_1$. Затем присвоим $w \leftarrow u \otimes v$, $w' \leftarrow (((u_1 \otimes v_1 \ominus w) \oplus (u_1 \otimes v_2)) \oplus (u_2 \otimes v_1)) \oplus (u_2 \otimes v_2)$.

Достаточно доказать это для случая, когда $u, v > 0$ и $e_u = e_u = p$, так что u и v есть целые $\in [2^{p-1}\,..\,2^p)$. Тогда $u = u_1 + u_2$, где $2^{p-1} \le u_1 \le 2^p$, $u_1 \bmod 2^{\lceil p/2\rceil} = 0$ и $|u_2| \le 2^{\lceil p/2\rceil-1}$; аналогично $v = v_1 + v_2$. Операции, выполняемые при вычислении w', являются

точными, поскольку $w-u_1v_1$ кратно 2^{p-1}, причем $|w-u_1v_1| \le |w-uv|+|u_2v_1+u_1v_2+u_2v_2| \le 2^{p-1}+2^{p+\lceil p/2\rceil}+2^{p-1}$; аналогично $|w-u_1v_1-u_1v_2| \le |w-uv|+|u_2v| < 2^{p-1}+2^{\lceil p/2\rceil-1+p}$, где $w-u_1v_1-u_1v_2$ кратно $2^{\lceil p/2\rceil}$.

22. Можно предположить, что $b^{p-1} \le u, v < b^p$. Если $uv \le b^{2p-1}$, то $x_1 = uv - r$, где $|r| \le \frac{1}{2}b^{p-1}$. Значит, $x_2 = \text{round}(u - r/v) = x_0$ (since $|r/v| \le \frac{1}{2}b^{p-1}/b^{p-1} \le \frac{1}{2}$ и при равенстве следует $v = b^{p-1}$, откуда $r = 0$). Если $uv > b^{2p-1}$, то $x_1 = uv - r$, где $|r| \le \frac{1}{2}b^p$. Поэтому $x_1/v = u - r/v < b^p + \frac{1}{2}b$ и $x_2 \le b^p$. Если $x_2 = b^p$, то $x_3 = x_1$ (поскольку из условия $(b^p - \frac{1}{2})v \le x_1$ следует, что x_1 кратно b^p и имеем $x_1 < b^p(v+\frac{1}{2})$). Если $x_2 < b^p$ и $x_1 > b^{2p-1}$, то пусть $x_2 = x_1/v + q$, где $|q| \le \frac{1}{2}$; получим $x_3 = \text{round}(x_1 + qv) = x_1$. Окончательно, если $x_2 < b^p$, $x_1 = b^{2p-1}$ и $x_3 < b^{2p-1}$, то $x_4 = x_2$ как следствие первого из рассмотренных выше случаев. Такая ситуация может возникнуть, например, когда $b = 10$, $p = 2$, $u = 19$, $v = 55$, $x_1 = 1000$, $x_2 = 18$, $x_3 = 990$.

23. Если $u \ge 0$ или $u \le -1$, имеем $u \bigcirc\!\!\!\!\text{mod}\; 1 = u \bmod 1$, так что равенство справедливо. Если $-1 < u < 0$, то $u \bigcirc\!\!\!\!\text{mod}\; 1 = u \oplus 1 = u + 1 + r$, где $|r| \le \frac{1}{2}b^{-p}$; равенство справедливо тогда и только тогда, когда $\text{round}(1 + r) = 1$, так что оно будет иметь место всегда, когда выполняется округление до четного. Если использовать правило округления, приведенное в тексте раздела, то равенство не будет выполняться тогда и только тогда, когда b кратно 4, $-1 < u < 0$ и $u \bmod 2b^{-p} = \frac{3}{2}b^{-p}$ (например, $p = 3$, $b = 8$, $u = -(.0124)_8$).

24. Пусть $u = [u_l \,..\, u_r]$, $v = [v_l \,..\, v_r]$. Когда $u \oplus v = [u_l \bigtriangledown v_l \,..\, u_r \bigtriangleup v_r]$, где $x \bigtriangleup y = y \bigtriangleup x$, $x \bigtriangleup +0 = x$ для всех x, $x \bigtriangleup -0 = x$ для всех $x \ne +0$, $x \bigtriangleup +\infty = +\infty$ для всех $x \ne -\infty$, а $x \bigtriangleup -\infty$ не нуждается в определении; $x \bigtriangledown y = -((-x) \bigtriangleup (-y))$. Если $x \oplus y$ приведет к переполнению при выполнении операции согласно обычным алгоритмам работы в формате с плавающей точкой, поскольку $x + y$ слишком велико, то $x \bigtriangleup y$ есть $+\infty$ и $x \bigtriangledown y$ является наибольшим по величине представимым числом.

Для вычитания положим $u \ominus v = u \oplus (-v)$, где $-v = [-v_r \,..\, -v_l]$.

С умножением дела обстоят сложнее. Правильный результат даст такая процедура: пусть $u \otimes v = [\min(u_l \bigtriangledown v_l, u_l \bigtriangledown v_r, u_r \bigtriangledown v_l, u_r \bigtriangledown v_r) \,..\, \max(u_l \bigtriangleup v_l, u_l \bigtriangleup v_r, u_r \bigtriangleup v_l, u_r \bigtriangleup v_r)]$, где $x \bigtriangleup y = y \bigtriangleup x$, $x \bigtriangleup (-y) = -(x \bigtriangledown y) = (-x) \bigtriangleup y$; $x \bigtriangleup +0 = (+0$ при $x > 0$, -0 при $x < 0)$; $x \bigtriangleup -0 = -(x \bigtriangleup +0)$; $x \bigtriangleup +\infty = (+\infty$ при $x > +0$, $-\infty$ при $x < -0)$. (Можно определить минимум и максимум, просто просмотрев знаки u_l, u_r, v_l и v_r и вычислив таким образом только два из восьми произведений, кроме ситуации, когда $u_l < 0 < u_r$ и $v_l < 0 < v_r$; в последнем случае нужно вычислить четыре произведения и результатом будет $[\min(u_l \bigtriangledown v_r, u_r \bigtriangledown v_l) \,..\, \max(u_l \bigtriangleup v_l, u_r \bigtriangleup v_r)]$.)

Окончательно получаем: $u \oslash v$ неопределенно, если $v_l < 0 < v_r$; иначе — будем использовать формулы для умножения, заменив v_l и v_r соответственно значениями v_r^{-1} и v_l^{-1}, где $x \bigtriangleup y^{-1} = x \bigtriangleup y$, $x \bigtriangledown y^{-1} = x \bigtriangledown y$, $(\pm 0)^{-1} = \pm\infty$, $(\pm\infty)^{-1} = \pm 0$.

[См. E. R. Hansen, *Math. Comp.* **22** (1968), 374–384. Альтернативная схема, в которой деление на 0 не приводит к сообщению об ошибке и интервалы могут быть окрестностями ∞, предложена У. М. Каханом (W. M. Kahan). По схеме Кахана, например, обращение $[-1 \,..\, +1]$ дает результат $[+1 \,..\, -1]$, а попытка умножения интервала, включающего 0, на интервал, включающий ∞, дает результат $[-\infty \,..\, +\infty]$, т. е. множество всех чисел. См. *Numerical Analysis*, Univ. Michigan Engineering Summer Conf. Notes No. 6818 (1968).]

25. Это явление связано с ошибками в *предшествующих* операциях вычисления u и v. Например, если ϵ мало́, $f(x + \epsilon) \ominus f(x)$ часто вычисляется с низкой точностью, поскольку округление при вычислении $f(x + \epsilon)$ приводит к утрате информации об ϵ. Рекомендуется переписать эту формулу в виде $\epsilon \otimes g(x, \epsilon)$, где $g(x, \epsilon) = (f(x + \epsilon) - f(x))/\epsilon$ определяется сначала в символьном виде. Таким образом, если $f(x) = x^2$, то $g(x, \epsilon) = 2x + \epsilon$; если $f(x) = \sqrt{x}$, то $g(x, \epsilon) = 1/(\sqrt{x + \epsilon} + \sqrt{x}\,)$.

26. Пусть $e = \max(e_u, e_{u'})$, $e' = \max(e_v, e_{v'})$, $e'' = \max(e_{u\oplus v}, e_{u'\oplus v'})$, и предположим, что $q = 0$. Тогда $(u \oplus v) - (u' \oplus v') \le u + v + \frac{1}{2}b^{e''-p} - u' - v' + \frac{1}{2}b^{e''-p} \le \epsilon b^e + \epsilon b^{e'} + b^{e''-p}$ и $e'' \ge \max(e, e')$. Следовательно, $u \oplus v \sim u' \oplus v'$ $(2\epsilon + b^{-p})$.

Если $b = 2$, эта оценка может быть уточнена до $1.5\epsilon + b^{-p}$. При условии, что $\epsilon + b^{-p}$ есть верхняя граница, если $u - u'$ и $v - v'$, получим противоположные знаки, а в другом случае не может оказаться $e = e' = e''$.

27. Сформулированное тождество есть следствие того, что $1 \oslash (1 \oslash u) = u$ при $b^{-1} \le f_u \le b^{-1/2}$. Если последнее ложно, значит, могут существовать целые числа x и y, такие, что $b^{p-1} < x < b^{p-1/2}$, и либо $y - \frac{1}{2} \le b^{2p-1}/x < b^{2p-1}/(x - \frac{1}{2}) \le y$, либо $y \le b^{2p-1}/(x + \frac{1}{2}) < b^{2p-1}/x \le y + \frac{1}{2}$. Но последнее явно невозможно, если только не выполняется соотношение $x(x + \frac{1}{2}) > b^{2p-1}$, а это последнее условие влечет за собой $y = \lfloor b^{p-1/2} \rfloor = x$.

28. См. *Math. Comp.* **32** (1978), 227–232.

29. Когда $b = 2$, $p = 1$ и $x > 0$, имеем $\text{round}(x) = 2^{e(x)}$, где $e(x) = \lfloor \lg \frac{4}{3}x \rfloor$. Пусть $f(x) = x^\alpha$ и пусть $t(n) = \lfloor \lfloor \alpha n + \lg \frac{4}{3} \rfloor / \alpha + \lg \frac{4}{3} \rfloor$. Тогда $\hat{h}(2^e) = 2^{t(e)}$. Когда $\alpha = .99$, получим $\hat{h}(2^e) = 2^{e-1}$ для $41 < e \le 58$.

31. Как следует из теории, которая будет изложена в разделе 4.5.3, "сходимость" бесконечной дроби $\sqrt{3} = 1 + //1, 2, 1, 2, \ldots //$ есть $p_n/q_n = K_{n+1}(1, 1, 2, 1, 2, \ldots)/K_n(1, 2, 1, 2, \ldots)$. Это прекрасная аппроксимация для $\sqrt{3}$, поскольку $3q_n^2 \approx p_n^2$, а фактически $3q_n^2 - p_n^2 = 2 - 3(n \bmod 2)$. Заданный в условии пример таков:

$$2p_{31}^2 + (3q_{31}^2 - p_{31}^2)(3q_{31}^2 + p_{31}^2) = 2p_{31}^2 - (p_{31}^2 - 1 + p_{31}^2) = 1.$$

Вычитание в формате с плавающей точкой p_{31}^2 из $3q_{31}^2$ дает нуль, если только нельзя представить $3q_{31}^2$ почти точно. Вычитание p_{31}^4 из $9q_{31}^4$ дает в общем случае ошибку округления, значительно бóльшую, чем $2p_{31}^2$. Аналогичный пример можно сформулировать, основываясь на аппроксимации бесконечной дробью любого алгебраического числа.

РАЗДЕЛ 4.2.3

1. $(w_m, w_l) = (.573, .248)$; тогда $w_m v_l / v_m = .290$. Таким образом, результат равен $(.572, .958)$. Он верен с точностью до шести десятичных знаков.

2. Это не окажет никакого воздействия на результат, поскольку подпрограмма нормализации обрезает до восьми разрядов и никогда не принимает в расчет байт в этой позиции. (Сдвиг влево происходит в ходе нормализации не более одного раза, так как предполагается, что входные данные уже нормализованы.)

3. При выполнении команды в строке 9 переполнение, очевидно, произойти не может, поскольку складываются двухбайтовые величины. Не может оно произойти и при выполнении команды в строке 22, так как складываются четырехбайтовые величины. Команда в строке 30 вычисляет сумму трех четырехбайтовых величин, что не может привести к переполнению. Наконец, при выполнении команды в строке 32 переполнение невозможно, так как произведение $f_u f_v$ должно быть меньше единицы.

4. Вставьте команду "`JOV OFLO; ENT1 0`" между строками 03 и 04. Замените строки 21–22 на "`ADD TEMP(ABS); JNOV *+2; INC1 1`", а строки 28–31 на "`SLAX 5; ADD TEMP; JNOV *+2; INC1 1; ENTX 0,1; SRC 5`". Таким образом, ко времени выполнения будет добавлено пять строк кода и только 1, 2 или 3 машинных цикла.

5. Вставьте "`JOV OFLO`" после строки 06. Замените строки 22, 31, 39 на "`SRAX 0,1`", "`SLAX 5`", "`ADD ACC`" соответственно. Между строками 40 и 41 вставьте команды "`DEC2 1; JNOV DNORM; INC2 1; INCX 1; SRC 1`". (Так и подмывает удалить "`DEC2 1`" и заменить ее командой "`STZ EXPO`", но тогда команда "`INC2 1`" может вызвать переполнение регистра rI2!)

Таким образом, программа увеличится на шесть строк; время выполнения *уменьшится* на $3u$, если только не возникнет переполнения при работе с дробной частью. В этом последнем случае время выполнения увеличится на $7u$.

```
6. DOUBLE STJ  EXITDF      Преобразование в формат с удвоенной точностью.
          ENTX 0           Очистка rX.
          STA  TEMP
          LD2  TEMP(EXP)   rI2 ← e.
          INC2 QQ-Q        Поправка на разность избытков.
          STZ  EXPO        EXPO ← 0.
          SLAX 1           Удалить порядок.
          JMP  DNORM       Нормализовать и выйти из подпрограммы.

   SINGLE STJ  EXITF       Преобразование в формат с однократной точностью.
          JOV  OFLO        Убедиться, что индикатор переполнения выключен.
          STA  TEMP
          LD2  TEMP(EXPD)  rI2 ← e.
          DEC2 QQ-Q        Поправка на разность избытков.
          SLAX 2           Удалить порядок.
          JMP  NORM        Нормализовать, округлить и выйти из подпрограммы. ▮
```

7. Все три программы дают нуль в качестве результата тогда и только тогда, когда точный результат должен равняться нулю, так что не стоит беспокоиться о нулевых знаменателях в выражениях для относительной ошибки. Наихудший случай для программы сложения действительно плох. Перейдем для наглядности к десятичным обозначениям. Если входные значения слагаемых — 1.0000000 и .99999999, то ответ будет равен b^{-7} вместо b^{-8}. Таким образом, максимальная относительная ошибка δ_1 равна $b-1$, где b — размер байта.

Что касается умножения и деления, то можно предполагать, что оба операнда положительны и порядки обоих операндов равны QQ. Максимальную ошибку умножения легко можно найти, проанализировав рис. 2. Если $uv \geq 1/b$, то $0 \leq uv - u \otimes v < 3b^{-9} + (b-1)b^{-9}$, так что относительная ошибка ограничена значением $(b+2)b^{-8}$. Если $1/b^2 \leq uv < 1/b$, то $0 \leq uv - u \otimes v < 3b^{-9}$, так что в этом случае относительная ошибка ограничена величиной $3b^{-9}/uv \leq 3b^{-7}$. Возьмем в качестве δ_2 бóльшую из этих двух оценок, а именно — $3b^{-7}$.

В случае деления требуется более тщательный анализ программы D. Величина, действительно вычисленная подпрограммой, будет равна $\alpha - \delta - b\epsilon((\alpha-\delta'')(\beta-\delta')-\delta''')-\delta_n$, где $\alpha = (u_m + \epsilon u_l)/bv_m$, $\beta = v_l/bv_m$; неотрицательные ошибки отбрасывания, равные $(\delta, \delta', \delta'', \delta''')$, не превышают $(b^{-10}, b^{-5}, b^{-5}, b^{-6})$ соответственно; наконец, δ_n (ошибка, возникающая при отбрасывании в процессе нормализации) неотрицательна и меньше либо b^{-9}, либо b^{-8} в зависимости от того, происходит ли сдвиг. Действительное значение частного есть $\alpha/(1+b\epsilon\beta) = \alpha - b\epsilon\alpha\beta + b^2\alpha\beta^2\delta'''' $, где δ'''' — неотрицательная ошибка, возникающая при отбрасывании бесконечного ряда в (2). Здесь $\delta'''' < \epsilon^2 = b^{-10}$, так как ряд знакопеременный. Следовательно, относительная ошибка равна абсолютному значению выражения $(b\epsilon\delta' + b\epsilon\delta''\beta/\alpha + b\epsilon\delta'''/\alpha) - (\delta/\alpha + b\epsilon\delta'\delta''/\alpha + b^2\beta^2\delta'''' + \delta_n/\alpha)$, умноженному на $(1+b\epsilon\beta)$. Положительные члены этого выражения ограничены величиной $b^{-9}+b^{-8}+b^{-8}$, а отрицательные ограничены (по модулю) величиной $b^{-8} + b^{-12} + b^{-8}$ плюс вклад от фазы нормализации, величина которого может достигать b^{-7}. Таким образом, ясно, что потенциально наибольшая составляющая относительной ошибки возникает в процессе выполнения нормализации, и можно считать, что значение $\delta_3 = (b+2)b^{-8}$ является надежной верхней границей для относительной ошибки.

8. Сложение. Если $e_u \le e_v + 1$, то вся относительная ошибка возникает в процессе нормализации и она ограничена значением b^{-7}. Если $e_u \ge e_v + 2$ и если знаки операндов совпадают, то, опять же, вся ошибка может быть отнесена на счет нормализации; если знаки противоположны, то ошибка, связанная со сдвигом разрядов из регистра, противоположна ошибке, возникающей после этого в ходе нормализации. Обе составляющие ограничены значением b^{-7}; следовательно, $\delta_1 = b^{-7}$. (Это существенно лучше результата упр. 7.)

Умножение. Анализ, аналогичный выполненному в упр. 7, дает $\delta_2 = (b+2)b^{-8}$.

РАЗДЕЛ 4.2.4

1. Так как переполнение дробной части может происходить только при операндах одного знака, искомая вероятность равна вероятности переполнения дробной части, деленной на вероятность совпадения знаков операндов, т. е. $7\%/(\frac{1}{2}(91\%)) \approx 15\%$.

3. $\log_{10} 2.4 - \log_{10} 2.3 \approx 1.84834\%$.

4. Страницы были бы потрепаны равномерно.

5. Вероятность того, что $10f_U \le r$, равна $(r-1)/10 + (r-1)/100 + \cdots = (r-1)/9$. Поэтому в рассматриваемом случае ведущие цифры распределены *равномерно*, например ведущая цифра 1 встречается с вероятностью $\frac{1}{9}$.

6. Вероятность того, что имеется три нулевых ведущих бита, равна $\log_{16} 2 = \frac{1}{4}$; вероятность того, что два ведущих бита нулевые, равна $\log_{16} 4 - \log_{16} 2 = \frac{1}{4}$; аналогично и для двух других случаев. "Среднее" число нулевых ведущих битов равно $1\frac{1}{2}$, так что "среднее" число "значащих битов" есть $p + \frac{1}{2}$. Однако наихудший случай, $p - 1$ значащих битов, встречается с довольно высокой вероятностью. На практике, как правило, оценки ошибок необходимо проводить на основе наихудшего случая, поскольку цепь вычислений настолько "прочна", насколько прочно слабейшее звено. Как следует из анализа ошибок в разделе 4.2.2, верхняя граница относительной ошибки округления равна 2^{1-p} для шестнадцатеричной системы счисления. При работе в двоичной системе счисления можно получить $p + 1$ значащих битов во всех нормализованных числах с относительной ошибкой округления, ограниченной значением 2^{-1-p} (см. упр. 4.2.1–3). Обширные вычислительные эксперименты подтверждают утверждение, что вычисления в двоичном формате с плавающей точкой дают более точные результаты, чем аналогичные вычисления в шестнадцатеричном формате с плавающей точкой, даже когда точность представления двоичных чисел равна p бит, а не $p + 1$.

Из табл. 1 и 2 видно, что шестнадцатеричная арифметика может быть сделана чуть более быстрой, так как для нее необходимо меньше циклов при масштабировании посредством сдвига вправо или нормализации посредством сдвига влево. Но этот фактор оказывается несущественным в сравнении с теми неоспоримыми преимуществами, которые имеет выбор $b = 2$ над другими основаниями (см. также теорему 4.2.2C и упр. 4.2.2–13, 15, 21), в особенности, если принять во внимание, что можно достичь той же скорости вычислений в двоичном формате с плавающей точкой, что и в шестнадцатеричном, ценой очень незначительного повышения суммарной стоимости процессора.

7. Предположим, что

$$\sum_m (F(10^{km} \cdot 5^k) - F(10^{km})) = \log 5^k / \log 10^k$$

и что

$$\sum_m (F(10^{km} \cdot 4^k) - F(10^{km})) = \log 4^k / \log 10^k;$$

тогда

$$\sum_m (F(10^{km} \cdot 5^k) - F(10^{km} \cdot 4^k)) = \log_{10} \tfrac{5}{4}$$

для всех k. По данному малому положительному числу ϵ выберем $\delta > 0$ так, чтобы $F(x) < \epsilon$ при $0 < x < \delta$, и выберем $M > 0$ так, чтобы $F(x) > 1 - \epsilon$ при $x > M$. Можно взять k

столь большим, что будут выполняться неравенства $10^{-k}\cdot 5^k < \delta$ и $4^k > M$; следовательно, в связи с монотонностью функции F получаем

$$\sum_m (F(10^{km}\cdot 5^k) - F(10^{km}\cdot 4^k))$$
$$\le \sum_{m\le 0} (F(10^{km}\cdot 5^k) - F(10^{k(m-1)}\cdot 5^k)) + \sum_{m\ge 0} (F(10^{k(m+1)}\cdot 4^k) - F(10^{km}\cdot 4^k))$$
$$= F(10^{-k}5^k) + 1 - F(10^k 4^k) < 2\epsilon.$$

8. Если $s > r$, то $P_0(10^n s)$ равно 1 при малых n и равно 0, если $\lfloor 10^n s\rfloor > \lfloor 10^n r\rfloor$. Наименьшее n, для которого так бывает, может быть произвольно большим, так что для $N_0(\epsilon)$ нельзя дать никакой равномерной оценки, не зависящей от s. (Вообще говоря, в учебниках по анализу доказывается, что из такой равномерной оценки следовало бы, что предельная функция $S_0(s)$ непрерывна, а это не так.)

9. Пусть $q_1, q_2, \ldots$ таковы, что $P_0(n) = q_1\binom{n-1}{0} + q_2\binom{n-1}{1} + \cdots$ для всех n. Тогда $P_m(n) = 1^{-m}q_1\binom{n-1}{0} + 2^{-m}q_2\binom{n-1}{1} + \cdots$ для всех m и n.

10. Если $1 < r < 10$, производящая функция $C(z)$ имеет простые полюсы в точках $1 + w_n$, где $w_n = 2\pi ni/\ln 10$. Следовательно,

$$C(z) = \frac{\log_{10} r - 1}{1-z} + \sum_{n\ne 0} \frac{1+w_n}{w_n} \frac{e^{-w_n \ln r} - 1}{(\ln 10)(z - 1 - w_n)} + E(z),$$

где $E(z)$ есть аналитическая функция на всей плоскости.

Таким образом, если $\theta = \arctan(2\pi/\ln 10)$, то

$$c_m = \log_{10} r - 1 - \frac{2}{\ln 10} \sum_{n>0} \Re\left(\frac{e^{-w_n \ln r} - 1}{w_n(1+w_n)^m}\right) + e_m$$
$$= \log_{10} r - 1 + \frac{\sin(m\theta + 2\pi \log_{10} r) - \sin(m\theta)}{\pi(1 + 4\pi^2/(\ln 10)^2)^{m/2}} + O\left(\frac{1}{(1 + 16\pi^2/(\ln 10)^2)^{m/2}}\right).$$

11. Если величина $(\log_b U) \bmod 1$ равномерно распределена в интервале $[0\,..\,1)$, то так же распределена и величина $(\log_b 1/U) \bmod 1 = (1 - \log_b U) \bmod 1$.

12. Имеем

$$h(z) = \int_{1/b}^{z} f(x)\,dx\, g(z/bx)/bx + \int_z^1 f(x)\,dx\, g(z/x)/x;$$

следовательно,

$$\frac{h(z) - l(z)}{l(z)} = \int_{1/b}^{z} f(x)\,dx\, \frac{g(z/bx) - l(z/bx)}{l(z/bx)} + \int_z^1 f(x)\,dx\, \frac{g(z/x) - l(z/x)}{l(z/x)}.$$

Так как $f(x) \ge 0$, $|(h(z) - l(z))/l(z)| \le \int_{1/b}^{z} f(x)\,dx\, A(g) + \int_z^1 f(x)\,dx\, A(g)$ для всех z, то $A(h) \le A(g)$. В силу симметрии $A(h) \le A(f)$. [*Bell System Tech. J.* **49** (1970), 1609–1625.]

13. Пусть $X = (\log_b U) \bmod 1$ и $Y = (\log_b V) \bmod 1$, так что X и Y независимо и равномерно распределены в интервале $[0\,..\,1)$. Ни одного сдвига влево не потребуется тогда и только тогда, когда $X + Y \ge 1$, и это происходит с вероятностью $\frac{1}{2}$.

(Аналогично результат выполнения деления по алгоритму 4.2.1M не нуждается в нормализующих сдвигах с вероятностью $\frac{1}{2}$; здесь достаточно более слабого предположения о *совпадении* распределений независимых операндов.)

14. Для удобства вычисления проводятся здесь для $b = 10$. Если $k = 0$, то вероятность переноса равна

$$\Big(\frac{1}{\ln 10}\Big)^2 \int_{\substack{1\le x,y\le 10\\ x+y\ge 10}} \frac{dx}{x}\,\frac{dy}{y}$$

(рис. А–7). Интеграл равен

$$\int_0^{10} \frac{dy}{y} \int_{10-y}^{10} \frac{dx}{x} - 2\int_0^1 \frac{dy}{y}\int_{10-y}^{10}\frac{dx}{x}$$

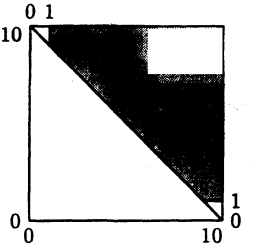


Рис. А–7.

и

$$\int_0^t \frac{dy}{y}\,\ln\Big(\frac{1}{1-y/10}\Big) = \int_0^t \Big(\frac{1}{10}+\frac{y}{200}+\frac{y^2}{3000}+\cdots\Big)\,dy = \frac{t}{10}+\frac{t^2}{400}+\frac{t^3}{9000}+\cdots.$$

(Последний интеграл, по существу, является "билогарифмическим".) Значит, вероятность переноса при $k = 0$ равна $(1/\ln 10)^2(\pi^2/6 - 2\sum_{n\ge1} 1/n^2 10^n) \approx .27154$. [*Замечание.* Если $b = 2$ и $k = 0$, переполнение дробной части происходит *всегда*, так что этот вывод подтверждает справедливость соотношения $\sum_{n\ge1} 1/n^2 2^n = \pi^2/12 - (\ln 2)^2/2$.]

Если $k > 0$, вероятность равна

$$\Big(\frac{1}{\ln 10}\Big)^2 \int_{10^{-k}}^{10^{1-k}} \frac{dy}{y}\int_{10-y}^{10}\frac{dx}{x} = \Big(\frac{1}{\ln 10}\Big)^2\Big(\sum_{n\ge1}\frac{1}{n^2 10^{nk}} - \sum_{n\ge1}\frac{1}{n^2 10^{n(k+1)}}\Big).$$

Таким образом, если $b = 10$, переполнение дробной части должно возникать с вероятностью приблизительно $.272p_0 + .017p_1 + .002p_2 + \cdots$. Когда $b = 2$, соответствующие величины равны $p_0 + .655p_1 + .288p_2 + .137p_3 + .067p_4 + .033p_5 + .016p_6 + .008p_7 + .004p_8 + .002p_9 + .001p_{10} + \cdots$.

Если теперь воспользоваться значениями вероятностей из табл. 1, разделив их на .91 для устранения нулевых операндов, и принять, что вероятности не зависят от знака операндов, можно при $b = 10$ предсказать значение вероятности около 14% против 15% из упр. 1. При $b = 2$ мы предсказываем значение, приблизительно равное 48%, в то время как таблица даст 44%. Эти расхождения, конечно, лежат в допускаемых пределах, если учтена ошибка эксперимента.

15. Если $k = 0$, то старшая цифра равна 1 тогда и только тогда, когда возникает перенос. (Возможно, что при $b \ge 4$ в результате переполнения дробной части и последующего округления появится ведущий разряд, равный 2, но в этом упражнении округление игнорируется.) Как показано в предыдущем упражнении, вероятность переполнения дробной части равна приблизительно .272, а $.272 < \log_{10} 2$.

Если $k > 0$, то ведущий разряд равен 1 с вероятностью

$$\Big(\frac{1}{\ln 10}\Big)^2\Big(\int_{10^{-k}}^{10^{1-k}}\frac{dy}{y}\int_{\substack{1\le x<2-y\\ \text{или } 10-y\le x<10}}\frac{dx}{x}\Big) < \Big(\frac{1}{\ln 10}\Big)^2\Big(\int_{10^{-k}}^{10^{1-k}}\frac{dy}{y}\int_{1\le x\le 2}\frac{dx}{x}\Big) = \log_{10} 2.$$

16. Для доказательства положения, сформулированного в указании [его авторство принадлежит Ландау (Landau, *Prace Matematyczno-Fizyczne* **21** (1910), 103–113)], сначала предположим, что $\limsup a_n = \lambda > 0$. Пусть $\epsilon = \lambda/(\lambda + 4M)$, и выберем N так, что $|a_1 + \cdots + a_n| < \frac{1}{10}\epsilon\lambda n$ для всех $n > N$. Пусть $n > N/(1-\epsilon)$, $n > 5/\epsilon$ будет таким, что $a_n > \frac{1}{2}\lambda$. Тогда по индукции $a_{n-k} \ge a_n - kM/(n-\epsilon n) > \frac{1}{4}\lambda$ для $0 \le k < \epsilon n$ и $\sum_{n-\epsilon n<k\le n} a_k \ge \frac{1}{4}\lambda(\epsilon n - 1) > \frac{1}{5}\lambda\epsilon n$. Но

$$\Big|\sum_{n-\epsilon n<k\le n} a_k\Big| = \Big|\sum_{1\le k\le n} a_k - \sum_{1\le k\le n-\epsilon n} a_k\Big| \le \tfrac{1}{5}\epsilon\lambda n,$$

так как $n - \epsilon n > N$. Аналогичное противоречие возникает и если $\liminf a_n < 0$.

Предполагая, что $P_{m+1}(n) \to \lambda$ при $n \to \infty$, положим $a_k = P_m(k) - \lambda$. Если $m > 0$, то a_k удовлетворяют гипотезе, сформулированной в указании, поскольку $0 \le P_m(k) \le 1$ (см. 4.2.2–(15)); отсюда $P_m(n) \to \lambda$.

17. См. *J. Math. Soc. Japan* **4** (1952), 313–322. (Тот факт, что гармоническая вероятность расширяет понятие обычной вероятности, следует из теоремы Чезаро (Cesàro, *Atti della Reale Accademia dei Lincei, Rendiconti* (4) **4** (1888), 452–457). Перси Дьяконис (Persi Diaconis, Ph. D. thesis, Harvard University, 1974) среди прочего показал, что определение вероятности через повторяющееся усреднение является более слабым, чем определение гармонической вероятности, в смысле следующей строгой формулировки. Если $\lim_{m\to\infty} \liminf_{n\to\infty} P_m(n) = \lim_{m\to\infty} \limsup_{n\to\infty} P_m(n) = \lambda$, то гармоническая вероятность равна λ. С другой стороны, утверждение "$10^{k^2} \le n < 10^{k^2+k}$ для целых чисел $k > 0$" имеет гармоническую вероятность $\frac{1}{2}$, в то время как повторяющееся усреднение никогда не присвоит этому утверждению некоторой конкретной вероятности.)

18. Пусть $p(a) = P(L_a)$ и $p(a,b) = \sum_{a\le k<b} p(k)$ для $1 \le a < b$. Поскольку $L_a = L_{10a} \cup L_{10a+1} \cup \cdots \cup L_{10a+9}$ для всех a, имеем $p(a) = p(10a, 10(a+1))$ вследствие (i). Далее, поскольку $P(S) = P(2S) + P(2S+1)$ вследствие (i)–(iii), имеем $p(a) = p(2a, 2(a+1))$. Отсюда следует, что $p(a,b) = p(2^m 10^n a, 2^m 10^n b)$ для всех $m, n \ge 0$.

Если $1 < b/a < b'/a'$, то $p(a,b) \le p(a',b')$. Причина в том, что существуют целые числа m, n, m', n', такие, что $2^{m'}10^{n'}a' \le 2^m 10^n a < 2^m 10^n b \le 2^{m'}10^{n'}b'$ как следствие из того факта, что $\log 2/\log 10$ является иррациональным числом; следовательно можно применить аксиому (v). (См. упр. 3.5–22, в котором нужно положить $k = 1$ и $U_n = n \log 2/\log 10$.) В частности, $p(a) \ge p(a+1)$ и отсюда следует, что $p(a,b)/p(a,b+1) \ge (b-a)/(b+1-a)$. (См. формулу 4.2.2–(15).)

Теперь можно доказать, что $p(a,b) = p(a',b')$, если только $b/a = b'/a'$; для любого большого значения n выполняется $p(a,b) = p(10^n a, 10^n b) \le c_n p(10^n a, 10^n b - 1) \le c_n p(a',b')$, где $c_n = 10^n(b-a)/(10^n(b-a)-1) = 1 + O(10^{-n})$.

Для любого положительного целого числа n имеем

$$p(a^n, b^n) = p(a^n, ba^{n-1}) + p(ba^{n-1}, b^2a^{n-2}) + \cdots + p(b^{n-1}a, b^n) = np(a,b).$$

Если $10^m \le a^n \le 10^{m+1}$ и $10^{m'} \le b^n \le 10^{m'+1}$, то $p(10^{m+1}, 10^{m'}) \le p(a^n, b^n) \le p(10^m, 10^{m'+1})$ согласно (v). Но $p(1,10) = 1$ вследствие (iv); отсюда $p(10^m, 10^{m'}) = m' - m$ для всех $m' \ge m$. Приходим к заключению, что $\lfloor \log_{10} b^n \rfloor - \lfloor \log_{10} a^n \rfloor - 1 \le np(a,b) \le \lfloor \log_{10} b^n \rfloor + \lfloor \log_{10} a^n \rfloor + 1$ для всех n и $p(a,b) = \log_{10}(b/a)$.

[На это упражнение автора вдохновил Д. И. А. Кохен (D. I. A. Cohen), который доказал несколько более слабый результат в *J. Combinatorial Theory* **A20** (1976), 367–370.]

19. Эквивалентно утверждению, что $\langle (\log 10 F_n) \bmod 1 \rangle$ имеет равномерное распределение в смысле определения 3.5B. Поскольку $\log_{10} F_n = n \log 10\phi - \log_{10}\sqrt{5} + O(\phi^{-2n})$ согласно 1.2.8–(14), это эквивалентно равномерному распределению $\langle n \log_{10} \phi \rangle$, что следует из упр. 3.5–22. [*Fibonacci Quarterly* **5** (1967), 137–140.] То же доказательство показывает, что последовательности $\langle b^n \rangle$ подчиняются логарифмическому закону для всех целых чисел $b > 1$, которые не являются степенями 10 [Яглом А. М. и Яглом И. М., *Неэлементарные задачи в элементарном изложении* (М: Гостехиздат, 1954; English translation, 1964), Задача 91b].

Замечание. Этим же свойством обладают многие другие последовательности целых чисел. Например, в работе Persi Diaconis, *Annals of Probability* **5** (1977), 72–81, показано, что одной из них является $\langle n! \rangle$ и что последовательность биномиальных коэффициентов

также подчиняется логарифмическому закону в том смысле, что

$$\lim_{n\to\infty}\frac{1}{n+1}\sum_{k=0}^{n}[10f_{\binom{n}{k}}<r]=\log_{10}r\,.$$

В работе P. Schatte, *Math. Nachrichten* **148** (1990), 137–144, доказано, что знаменатели в непрерывных разложениях дробей имеют логарифмические дробные части, если только частичные отношения имеют повторяющуюся форму с полиномиальной вариацией, как в упр. 4.5.3–16. Очень интересен вопрос, который до сих пор остается открытым: будет ли последовательность $\langle 2!, (2!)!, ((2!)!)!, \ldots\rangle$ иметь логарифмические дробные части; см. J. H. Conway и M. J. T. Guy, *Eureka* **25** (1962), 18–19.

РАЗДЕЛ 4.3.1

2. Если i-м слагаемым является $u_i = (u_{i(n-1)}\ldots u_{i1}u_{i0})_b$, то используем алгоритм A, в котором шаг A2 модифицирован следующим образом.

A2′. [Сложить разряды.] Присвоить

$$w_j \leftarrow (u_{1j}+\cdots+u_{mj}+k) \bmod b \quad\text{и}\quad k\leftarrow\lfloor(u_{1j}+\cdots+u_{mj}+k)/b\rfloor.$$

(Максимальное значение k равно $m-1$, поэтому при $m>b$ потребуется изменить шаг A3.)

3.
```
    ENN1 N          1
    JOV  OFLO       1     Сбросить индикатор переполнения.
    ENTX 0          1     k ← 0.
2H  SLAX 5          N     (rX ≡ следующее значение k.)
    ENT3 M*N,1      N     (LOC(u_ij) ≡ U + n(i − 1) + j.)
3H  ADD  U,3        MN    rA ← rA + u_ij.
    JNOV *+2        MN
    INCX 1          K     Перенос единицы.
    DEC3 N          MN    Повторить для m ≥ i ≥ 1.
    J3NN 3B         MN    (rI3 ≡ n(i − 1) + j.)
    STA  W+N,1      N     w_j ← rA.
    INC1 1          N
    J1N  2B         N     Повторить для 0 ≤ j < n.
    STX  W+N        1     Запомнить последний перенос в w_n.   ▮
```

Время выполнения в предположении, что $K=\frac{1}{2}MN$, равно $5.5MN+7N+4$ циклов.

4. Перед шагом A1 можно сделать следующее утверждение: "$n\ge 1$ и $0\le u_i,v_i<b$ при $0\le i<n$". Перед шагом A2 справедливо утверждение "$0\le j<n$; $0\le u_i,v_i<b$ при $0\le i<n$; $0\le w_i<b$ при $0\le i<j$; $0\le k\le 1$; $(u_{j-1}\ldots u_0)_b+(v_{j-1}\ldots v_0)_b=(kw_{j-1}\ldots w_0)_b$". Более точная форма последнего утверждения такова:

$$\sum_{0\le l<j}u_lb^l+\sum_{0\le l<j}v_lb^l=kb^j+\sum_{0\le l<j}w_lb^l.$$

Перед шагом A3 справедливо утверждение "$0\le j<n$; $0\le u_i,v_i<b$ при $0\le i<n$; $0\le w_i<b$ при $0\le i\le j$; $0\le k\le 1$ и $(u_j\ldots u_0)_b+(v_j\ldots v_0)_b=(kw_j\ldots w_0)_b$". После шага A3 справедливо утверждение, что $0\le w_i<b$ при $0\le i<n$; $0\le w_n\le 1$ и $(u_{n-1}\ldots u_0)_b+(v_{n-1}\ldots v_0)_b=(w_n\ldots w_0)_b$.

После этого можно довольно просто закончить доказательство, проверив справедливость нужных импликаций для утверждений и показав, что выполнение алгоритма всегда завершается.

5. B1. Присвоить $j \leftarrow n-1$, $w_n \leftarrow 0$.

B2. Присвоить $t \leftarrow u_j + v_j$, $w_j \leftarrow t \bmod b$, $i \leftarrow j$.

B3. Если $t \geq b$, присвоить $i \leftarrow i+1$, $t \leftarrow w_i + 1$, $w_i \leftarrow t \bmod b$ и повторять этот шаг до тех пор, пока не будет выполнено неравенство $t < b$.

B4. Уменьшить j на единицу и, если $j \geq 0$, вернуться к шагу B2. ▮

6. C1. Присвоить $j \leftarrow n-1$, $i \leftarrow n$, $r \leftarrow 0$.

C2. Присвоить $t \leftarrow u_j + v_j$. Если $t \geq b$, присвоить $w_i \leftarrow r+1$ и $w_k \leftarrow 0$ при $i > k > j$; затем присвоить $i \leftarrow j$ и $r \leftarrow t \bmod b$. В противном случае, если $t < b-1$, присвоить $w_i \leftarrow r$ и $w_k \leftarrow b-1$ при $i > k > j$. После этого присвоить $i \leftarrow j$ и $r \leftarrow t$.

C3. Уменьшить j на единицу. Если $j \geq 0$, вернуться к шагу C2, в противном случае присвоить $w_i \leftarrow r$ и $w_k \leftarrow b-1$ при $i > k \geq 0$. ▮

7. Если, к примеру, $j = n-3$, то $k = 0$ с вероятностью $(b+1)/2b$; $k = 1$ с вероятностью $((b-1)/2b)(1-1/b)$, а это вероятность того, что произошел перенос и предшествующий разряд не был равен $b-1$; $k = 2$ с вероятностью $((b-1)/2b)(1/b)(1-1/b)$ и $k = 3$ с вероятностью $((b-1)/2b)(1/b)(1/b)(1)$. При фиксированных k можно просуммировать все вероятности по параметру j, который изменяется от $n-1$ до 0; это даст среднее число случаев, когда перенос распространяется на k разрядов:

$$m_k = \frac{b-1}{2b^k}\left((n+1-k)\left(1-\frac{1}{b}\right)+\frac{1}{b}\right).$$

Для проверки найдем среднее число переносов

$$m_1 + 2m_2 + \cdots + nm_n = \frac{1}{2}\left(n - \frac{1}{b-1}\left(1-\left(\frac{1}{b}\right)^n\right)\right),$$

что согласуется с формулой (6).

8.

```
    ENT1 N-1   1          3H LDA  W,2  K
    JOV  OFLO  1             INCA 1    K
    STZ  W+N   1             STA  W,2  K
  2H LDA  U,1  N             INC2 1    K
    ADD  V,1   N             JOV  3B   K
    STA  W,1   N           4H DEC1 1    N
    JNOV 4F    N             J1NN 2B   N   ▮
    ENT2 1,1   L
```

Время выполнения программы зависит от L — числа позиций, в которых $u_j + v_j \geq b$, и от K — общего числа переносов. Нетрудно заметить, что K — это та же самая величина, которая появляется в программе A. Анализ в тексте раздела показывает, что среднее значение L равно $N((b-1)/2b)$, а среднее значение K равно $\frac{1}{2}(N - b^{-1} - b^{-2} - \cdots - b^{-n})$. Таким образом, если пренебречь членами порядка $1/b$, время выполнения программы будет равно $9N + L + 7K + 3 \approx 13N + 3$ циклам.

9. На шаге A2 везде заменить "b" на "b_j".

10. Если поменять местами строки 06 и 07, то почти всегда можно получить переполнение. При этом регистр A в ходе выполнения команды в строке 08 может иметь отрицательное значение, так что программа работать не будет. Если поменять местами строки 05 и 06, то последовательность переполнений, происходящих в ходе работы программы, будет в некоторых случаях иной, но программа даст правильный результат.

11. Эта задача равносильна задаче лексикографического сравнения цепочек: (i) присвоить $j \leftarrow n-1$; (ii) если $u_j < v_j$, закончить с результатом $[u < v]$; если $u_j = v_j$ и $j = 0$, закончить

с результатом $[u = v]$; если $u_j = v_j$ и $j > 0$, присвоить $j \leftarrow j - 1$ и повторить (ii); если $u_j > v_j$, то закончить $[u > v]$. Этот алгоритм оказывается довольно быстрым, так как обычно очень мала вероятность того, что j станет очень большим раньше, чем возникнет случай, когда $u_j \neq v_j$.

12. Используем алгоритм B при $u_j = 0$ и $v_j = w_j$. В конце этого алгоритма появится другое "заимствование", но на этот раз им можно пренебречь.

13.
```
        ENN1 N      1       MUL  V      N       STA  W+N,1  N
        JOV  OFLO   1       SLC  5      N       INC1 1      N
        ENTX 0      1       ADD  CARRY  N       J1N  2B     N
     2H STX  CARRY  N       JNOV *+2    N       STX  W+N    1  ∎
        LDA  U+N,1  N       INCX 1      K
```
Время выполнения программы равно $23N + K + 5$ циклам, а грубая оценка K есть $\frac{1}{2}N$.

14. Ключевым является индуктивное утверждение, которое должно быть справедливым в начале шага M4. Все остальные утверждения легко выводятся из этого утверждения, которое выглядит так: $0 \le i < m$; $0 \le j < n$; $0 \le u_l < b$ при $0 \le l < m$; $0 \le v_l < b$ при $0 \le l < n$; $0 \le w_l < b$ при $0 \le l < j + m$; $0 \le k < b$; и в обозначениях, введенных в ответе к упр. 4,

$$(w_{j+m-1} \dots w_0)_b + kb^{i+j} = u \times (v_{j-1} \dots v_0)_b + (u_{i-1} \dots u_0)_b \times v_j b^j.$$

15. Ошибка неотрицательна и меньше $(n-2)b^{-n-1}$. (Аналогично, если проигнорировать произведения при $i+j > n+3$, ошибка будет ограничена величиной $(n-3)b^{-n-2}$, и т. д. Но, вообще говоря, для получения правильно округленного результата необходимо вычислять все произведения.) Дальнейший анализ показывает, что корректно округленный результат перемножения дробных частей чисел в формате с плавающей точкой почти всегда может быть получен при вычислительных затратах, почти вдвое меньших, чем при вычислении полного произведения с удвоенной точностью. Более того, выполнив проверку, можно убедиться, что случаи, в которых необходима полная точность, крайне редки. [См. W. Krandick, J. R. Johnson, *Proc. IEEE Symp. Computer Arithmetic* **11** (1993), 228–233.]

16. **S1.** Присвоить $r \leftarrow 0$, $j \leftarrow n - 1$.

S2. Присвоить $w_j \leftarrow \lfloor (rb + u_j)/v \rfloor$, $r \leftarrow (rb + u_j) \bmod v$.

S3. Уменьшить j на 1 и, если $j \ge 0$, вернуться к шагу S2. ∎

17. $u/v > u_n b^n/(v_{n-1}+1)b^{n-1} = b(1 - 1/(v_{n-1}+1)) > b(1 - 1/(b/2)) = b - 2$.

18. $(u_n b + u_{n-1})/(v_{n-1}+1) \le u/(v_{n-1}+1)b^{n-1} < u/v$.

19. $u - \hat{q}v \le u - \hat{q}v_{n-1}b^{n-1} - \hat{q}v_{n-2}b^{n-2} = u_{n-2}b^{n-2} + \cdots + u_0 + \hat{r}b^{n-1} - \hat{q}v_{n-2}b^{n-2} < b^{n-2}(u_{n-2} + 1 + \hat{r}b - \hat{q}v_{n-2}) \le 0$. Поскольку $u - \hat{q}v < 0$, то $q < \hat{q}$.

20. Если $q \le \hat{q} - 2$, то $u < (\hat{q}-1)v < \hat{q}(v_{n-1}b^{n-1} + (v_{n-2}+1)b^{n-2}) - v < \hat{q}v_{n-1}b^{n-1} + \hat{q}v_{n-2}b^{n-2} + b^{n-1} - v \le \hat{q}v_{n-1}b^{n-1} + (b\hat{r} + u_{n-2})b^{n-2} + b^{n-1} - v = u_n b^n + u_{n-1}b^{n-1} + u_{n-2}b^{n-2} + b^{n-1} - v \le u_n b^n + u_{n-1}b^{n-1} + u_{n-2}b^{n-2} \le u$. Другими словами, выходит, что $u < u$, а это невозможно.

21. (Получено Г. К. Гоялом (G. K. Goyal).) Из неравенства $\hat{q}v_{n-2} \le b\hat{r} + u_{n-2}$ следует $\hat{q} \le (u_n b^2 + u_{n-1}b + u_{n-2})/(v_{n-1}b + v_{n-2}) \le u/((v_{n-1}b + v_{n-2})b^{n-2})$. Отсюда $u \bmod v = u - qv = v(1-\alpha)$, где $0 < \alpha = 1 + q - u/v \le \hat{q} - u/v \le u(1/((v_{n-1}b + v_{n-2})b^{n-2}) - 1/v) = u(v_{n-3}b^{n-3} + \cdots)/((v_{n-1}b + v_{n-2})b^{n-2}v) < u/(v_{n-1}bv) \le \hat{q}/(v_{n-1}b) \le (b-1)/(v_{n-1}b)$, которое ограничено величиной $2/b$, так как $v_{n-1} \ge \frac{1}{2}(b-1)$.

22. Пусть $u = 4100$, $v = 588$. Возьмем сначала $\hat{q} = \lfloor \frac{41}{5} \rfloor = 8$. Однако видим, что $8 \cdot 8 > 10(41 - 40) + 0$. Тогда полагаем $\hat{q} = 7$ и получаем $7 \cdot 8 < 10(41 - 35) + 0$. Но число 588, умноженное на 7, равно 4116, так что правильное частное будет $q = 6$. (Между прочим,

данный пример показывает, что для $b = 10$ теорема B при данных предположениях не может быть улучшена.)

23. Очевидно, что $v\lfloor b/(v+1)\rfloor < (v+1)\lfloor b/(v+1)\rfloor \le b$, поэтому при $v \ge b/2$ выполняется левая часть неравенства. В противном случае $v\lfloor b/(v+1)\rfloor \ge v(b-v)/(v+1) \ge (b-1)/2 > \lfloor b/2\rfloor - 1$.

24. Приближенная вероятность равна всего лишь $\log_b 2$, а не $\frac{1}{2}$. (Например, если $b = 2^{32}$, вероятность того, что $v_{n-1} \ge 2^{31}$, приблизительно равна $\frac{1}{32}$. Тем не менее этого еще достаточно много для того, чтобы оправдать выполнение специальной проверки условия $d = 1$ на шагах D1 и D8 алгоритма D.)

25.

```
002      ENTA 1              1
003      ADD  V+N-1          1
004      STA  TEMP           1
005      ENTA 1              1
006      JOV  1F             1        Переход, если v_{n-1} = b - 1.
007      ENTX 0              1
008      DIV  V+N-1          1        Иначе — вычислить ⌊b/(v+1)⌋.
009      JOV  DIVBYZERO      1        Переход, если v_{n-1} = 0.
010  1H  STA  D              1
011      DECA 1              1
012      JANZ *+3            1        Переход, если d ≠ 1.
013      STZ  U+M+N        1 - A      Присвоить u_{m+n} ← 0.
014      JMP  D2           1 - A
015      ENN1 N              A        Умножить v на d.
016      ENTX 0              A
017  2H  STX  CARRY         AN
018      LDA  V+N,1         AN
019      MUL  D             AN
...                                   (Как в упр. 13.)
026      J1N  2B            AN
027      ENN1 M+N            A        (Теперь rX = 0.)
028  2H  STX  CARRY       A(M+N)     Умножить u на d.
029      LDA  U+M+N,1     A(M+N)
...                                   (Как в упр. 13.)
037      J1N  2B          A(M+N)
038      STX  U+M+N          A
```

∎

26. (См. алгоритм в упр. 16.)

```
101  D8  LDA  D          1     (Остаток сохранится в ячейках
102      DECA 1          1        от U до U+N-1.)
103      JAZ  DONE       1     Если d = 1, то закончить.
104      ENT1 N-1        A     rI1 ≡ j; j ← n - 1.
105      ENTA 0          A     r ← 0.
106  1H  LDX  U,1        AN    rAX ← rb + u_j.
107      DIV  D          AN
108      STA  U,1        AN
109      SLAX 5          AN    (u_j, r) ← (⌊rAX/d⌋, rAX mod d).
110      DEC2 1          AN    j ← j + 1.
111      J2NN 1B         AN    Повторить для n > j ≥ 0.
```

∎

На этом программа деления завершается, причем, как будет показано в следующем упражнении, rAX = 0.

27. Это число равно $du \bmod dv = d(u \bmod v)$.

28. Предположим (для удобства), что v имеет десятичную точку слева, т. е. что $v = (v_n.v_{n-1}v_{n-2}\ldots)_b$. После завершения шага N1 получим $\frac{1}{2} \le v < 1 + 1/b$. Тогда

$$v\left\lfloor\frac{b+1}{v_{n-1}+1}\right\rfloor \le \frac{v(b+1)}{v_{n-1}+1} = \frac{v(1+1/b)}{(1/b)(v_{n-1}+1)} < 1 + \frac{1}{b}$$

и

$$v\left\lfloor\frac{b+1}{v_{n-1}+1}\right\rfloor \ge \frac{v(b+1-v_{n-1})}{v_{n-1}+1} \ge \frac{1}{b}\frac{v_{n-1}(b+1-v_{n-1})}{v_{n-1}+1}.$$

Величина в последнем соотношении принимает наименьшее значение при $v_{n-1} = 1$, так как это выпуклая функция и другое ее экстремальное значение больше.

Формулу на шаге N2 можно переписать в виде $v \leftarrow \left\lfloor\frac{b(b+1)}{v_{n-1}+1}\right\rfloor\frac{v}{b}$, поэтому, как и выше, находим, что v никогда не будет $\ge 1 + 1/b$.

После одной итерации шага N2 минимальное значение v не меньше, чем

$$\left(\frac{b(b+1)-v_{n-1}}{v_{n-1}+1}\right)\frac{v}{b} \ge \left(\frac{b(b+1)-v_{n-1}}{v_{n-1}+1}\right)\frac{v_{n-1}}{b^2} = \left(\frac{b(b+1)+1-t}{t}\right)\left(\frac{t-1}{b^2}\right)$$
$$= 1 + \frac{1}{b} + \frac{2}{b^2} - \frac{1}{b^2}\left(t + \frac{b(b+1)+1}{t}\right),$$

при $t = v_{n-1} + 1$. Минимум этой величины достигается при $t = b/2 + 1$, нижняя граница равна $1 - 3/2b$. Следовательно, после одной итерации шага N2 имеем $v_{n-1} \ge b - 2$. В итоге получаем $(1-3/2b)(1+1/b)^2 > 1$, где $b \ge 5$, так что потребуется еще не более двух итераций. В случае, когда $b < 5$, утверждение легко проверяется непосредственно.

29. Это утверждение верно, так как $(u_{j+n}\ldots u_j)_b < v$.

30. При выполнении алгоритмов A и S такое перекрытие возможно, если алгоритмы слегка видоизменить. К примеру, в алгоритме A можно так переписать шаг A2: "Присвоить $t \leftarrow u_j + v_j + k$, $w_j \leftarrow t \bmod b$, $k \leftarrow \lfloor t/b \rfloor$".

При выполнении алгоритма M значение v_j может храниться в том же месте, что и w_{j+n}. При реализации алгоритма D удобнее всего (как в программе D в упр. 26) принять, что значения $r_{n-1}\ldots r_0$ хранятся там же, где и $u_{n-1}\ldots u_0$. Можно также считать $q_m\ldots q_0$ такими же, как и $u_{m+n}\ldots u_n$, при условии, что на шаге D6 значения переменных u_{j+n} не изменились. (Строка 098 программы D может быть без вреда заменена на "`J1N2B`", так как величина u_{j+n} в вычислениях, выполняемых на этом шаге, не используется.)

31. Рассмотрите ситуацию, приведенную на рис. 6, положив $u = (u_{j+n}\ldots u_{j+1}u_j)_3$, как в алгоритме D. Если ведущие ненулевые разряды чисел u и v имеют один знак, то присваиваем $r \leftarrow u - v$, $q \leftarrow 1$; в противном случае присваиваем $r \leftarrow u + v$, $q \leftarrow -1$. Если теперь $|r| > |u|$ или $|r| = |u|$ и знак первого ненулевого разряда чисел $u_{j-1}\ldots u_0$ совпадает с первым ненулевым разрядом числа r, то присваиваем $q \leftarrow 0$; в противном случае присваиваем $u_{j+n}\ldots u_j$ значения разрядов числа r.

32. См. M. Nadler, *CACM* **4** (1961), 192–193; Z. Pawlak and A. Wakulicz, *Bull. de l'Acad. Polonaise des Sciences*, Classe III, **5** (1957), 233–236 (см. также с. 803–804), и упр. 4.1–15.

34. См., например, R. E. Maeder, *The Mathematica Journal* **6**, 2 (Spring, 1996), 32–40; **6**, 3 (Summer, 1996), 37–43.

36. Имея число ϕ, заданное с точностью $\pm 2^{-2n}$, ϕ^{-1}, ϕ^{-2}, ..., значение $\ln\phi$ можно вычислить, выполняя вычитания до тех пор, пока $\phi^{-k} < 2^{-n}$. Накапливаемая при этом ошибка не превысит 2^{1-n}. Затем можно использовать ряд $\ln\phi = \ln((1+\phi^{-3})/(1-\phi^{-3})) = 2(\phi^{-3} + \frac{1}{3}\phi^{-9} + \frac{1}{5}\phi^{-15} + \cdots)$. [См. статью William Schooling, *Napier Tercentenary Memorial*,

edited by C. G. Knott (London: Longmans, 1915), 337–344.] Но еще лучше (предложено в 1965 году Дж. У. Ренчем (мл.) (J. W. Wrench, Jr.)) вычислить

$$\ln\phi = \tfrac{1}{2}\ln((1+5^{-1/2})/(1-5^{-1/2})) = (2\phi-1)(5^{-1}+\tfrac{1}{3}5^{-2}+\tfrac{1}{5}5^{-3}+\cdots).$$

37. Пусть $d = 2^e$, так что $b > dv_{n-1} \geq b/2$. Вместо нормализации u и v на шаге D1 определяем два ведущих разряда $v'v''$ числа $2^e(v_{n-1}v_{n-2}v_{n-3})_b$ посредством его сдвига влево на e бит. На шаге D3 вместо (v_{n-1}, v_{n-2}) используем (v', v''), а вместо $(u_{j+n}, u_{j+n-1}, u_{j+n-2})$ используем (u', u'', u'''). Значения разрядов $u'u''u'''$ вычисляются путем сдвига влево на e бит числа $(u_{j+n}\ldots u_{j+n-3})_b$. Деление на d на шаге D8 опускаем. (В сущности, числа u и v сдвигаются "виртуально". Такой метод снижает объем вычислений, если m мало́ по сравнению с n.)

38. Присвоим $k \leftarrow n$, $r \leftarrow 0$, $s \leftarrow 1$, $t \leftarrow 0$, $w \leftarrow u$. Тогда сохраняется инвариантное отношение $uv = 2^{2k}(r+s^2-s) + 2^{2k-n}t + 2^{2k-2n}vw$ при $0 \leq t, w < 2^n$ и при $0 < r \leq 2s$. Если условие $(r,s) = (0,1)$ больше не выполняется, то до тех пор, пока $k > 0$, положить $4w = 2^n w' + w''$ и $4t + w'v = 2^n t' + t''$, где $0 \leq w'', t'' < 2^n$ и $0 \leq t' \leq 6$. Затем присвоить $t \leftarrow t''$, $w \leftarrow w''$, $s \leftarrow 2s$, $r \leftarrow 4r + t' - s$, $k \leftarrow k-1$. Если $r \leq 0$, присвоить $s \leftarrow s-1$ и $r \leftarrow r+2s$, в противном случае, если $r > 2s$, присвоить $r \leftarrow r-2s$ и $s \leftarrow s+1$ (эта поправка может понадобиться дважды). Повторять до тех пор, пока не получим $k = 0$. Тогда $uv = r + s^2 - s$, так как w всегда умножается на 2^{2n-2k}. Соответственно, $r = 0$ тогда и только тогда, когда $uv = 0$. В противном случае результат будет равен s, так как $uv - s \leq s^2 < uv + s$.

39. Положим $S_j = \sum_{k\geq 0} 16^{-k}/(8k+j)$. Необходимо выяснить, выполняется ли неравенство $2^{n-1}\pi \bmod 1 < \frac{1}{2}$. Так как $\pi = 4S_1 - 2S_4 - S_5 - S_6$, этого достаточно для наличия хороших оценок $2^{n-1}S_j \bmod 1$. Теперь $2^{n-1}S_j$ конгруэнтно (по модулю 1)

$$\sum_{0\leq k<n/4} a_{njk}/(8k+j) + \sum_{k\geq n/4} 2^{n-1-k}/(8k+j),$$

где $a_{njk} = 2^{n-1-4k} \bmod (8k+j)$. Каждый член в первой сумме может быть аппроксимирован посредством вычисления a_{njk} при помощи $O(\log n)$ операций (раздел 4.6.3) в пределах 2^{-m}. После этого получится промасштабированное частное $\lfloor 2^m a_{njk}/(8k+j)\rfloor$. Вторая сумма может быть аппроксимирована в пределах 2^{-m} в результате вычисления 2^m раз ее первых $m/4$ членов. Если $m \approx 2\lg n$, интервал неопределенности будет равен $\approx 1/n$, что почти всегда обеспечивает достаточную точность. [*Math. Comp.* **66** (1997), 903–913.]

Примечание. Пусть $\zeta = e^{\pi i/4} = (1+i)/\sqrt{2}$ равно корню степени 8 из единицы. Рассмотрим значения $l_j = \ln(1-\zeta^j/\sqrt{2})$. Тогда $l_0 = \ln(1-1/\sqrt{2})$, $l_1 = \bar{l}_7 = \frac{1}{2}\ln\frac{1}{2} - i\arctan 1$, $l_2 = \bar{l}_6 = \frac{1}{2}\ln\frac{3}{2} - i\arctan(1/\sqrt{2})$, $l_3 = \bar{l}_5 = \frac{1}{2}\ln\frac{5}{2} - i\arctan(1/3)$, $l_4 = \ln(1+1/\sqrt{2})$. Кроме того, $-S_j/2^{j/2} = \frac{1}{8}(l_0 + \zeta^{-j}l_1 + \cdots + \zeta^{-7j}l_7)$ при $1 \leq j \leq 8$ вследствие 1.2.9–(13). Отсюда $4S_1 - 2S_4 - S_5 - S_6 = 2l_0 - (2-2i)2l_1 + 2l_4 + (2+2i)l_7 = \pi$. Другие интересные тождества приведены ниже:

$$\begin{aligned}
\ln 2 &= S_2 + \tfrac{1}{2}S_4 + \tfrac{1}{4}S_6 + \tfrac{1}{8}S_8;\\
\ln 3 &= 2S_2 + \tfrac{1}{2}S_6;\\
\ln 5 &= 2S_2 + 2S_4 + \tfrac{1}{2}S_6;\\
\sqrt{2}\ln(\sqrt{2}+1) &= S_1 + \tfrac{1}{2}S_3 + \tfrac{1}{4}S_5 + \tfrac{1}{8}S_7;\\
\sqrt{2}\arctan(1/\sqrt{2}) &= S_1 - \tfrac{1}{2}S_3 + \tfrac{1}{4}S_5 - \tfrac{1}{8}S_7;\\
\arctan(1/3) &= S_1 - S_2 - \tfrac{1}{2}S_4 - \tfrac{1}{4}S_5;\\
0 &= 8S_1 - 8S_2 - 4S_3 - 8S_4 - 2S_5 - 2S_6 + S_7.
\end{aligned}$$

В общем случае получаем

$$\sum_{k\ge0}\frac{z^{8k+1}}{8k+1}=A+B+C+D,\qquad \sum_{k\ge0}\frac{z^{8k+5}}{8k+5}=A-B+C-D,$$

$$\sum_{k\ge0}\frac{z^{8k+3}}{8k+3}=A-B-C+D,\qquad \sum_{k\ge0}\frac{z^{8k+7}}{8k+7}=A+B-C-D,$$

где

$$A=\frac18\ln\frac{1+z}{1-z},\qquad B=\frac{1}{2^{7/2}}\ln\frac{1+\sqrt2z+z^2}{1-\sqrt2z+z^2},$$

$$C=\frac14\arctan z,\qquad D=\frac{1}{2^{5/2}}\arctan\frac{\sqrt2z}{1-z^2};$$

и

$$\sum_{k\ge0}\frac{z^{mk+a}}{mk+a}=-\frac1m\Big(\ln(1-z)+(-1)^a[m\text{ even}]\ln(1+z)+f_{am}(z)\Big),$$

$$f_{am}(z)=\sum_{k=1}^{\lfloor(m-1)/2\rfloor}\Big(\cos\frac{2\pi ka}{m}\ln\Big(1-2z\cos\frac{2\pi k}{m}+z^2\Big)$$

$$-2\sin\frac{2\pi ka}{m}\arctan\frac{z\sin(2\pi k/m)}{1-z\cos(2\pi k/m)}\Big).$$

40. Чтобы получить $n/2$ старших разрядов числа, необходимо выполнить около $\sum_{k=1}^{n/2}\approx\frac18n^2$ основных операций (см. упр. 15). Используя b-адический метод, где b — степень 2, можно получить $n/2$ младших разрядов числа (см. упр. 4.1–31). Эта задача легко сводится к случаю, когда v нечетное. Пусть имеется три числа: $u=(\dots u_2u_1u_0)_b$, $v=(\dots v_2v_1v_0)_b$ и $w=(\dots w_2w_1w_0)_b$. Необходимо найти $u=vw$ (по модулю $b^{n/2}$). Вычислим v' так, чтобы $v'v \bmod b=1$ (см. упр. 4.5.2–17). В этом случае $w_0=v'u_0 \bmod b$ и можно вычислить $u'=u-w_0v$, $w_1=v'u_0' \bmod b$ и т. д. Понадобится выполнить примерно $\frac18n^2$ основных операций, чтобы вычислить $n/2$ разрядов справа. Таким образом, общее количество операций равно $\frac14n^2+O(n)$, в то время как для выполнения алгоритма D требуется $n^2+O(n)$ операций. Для реализации прямого метода вычисления всех n разрядов справа налево необходимо $\frac12n^2+O(n)$ операций. [См. T. Jebelean, *J. Symbolic Comp.* **15** (1993), 169–180; A. Schönhage and E. Vetter, *Lecture Notes in Comp. Sci.* **855** (1994), 448–459.]

41. (a) Если $m=0$, то положим $v=u$. В противном случае вычтем xw из $(u_{m+n-1}\dots u_1u_0)_b$, где $x=u_0w' \bmod b$. Эти нули расположены вне разрядов представления числа, так что мы эффективно уменьшили m на 1. (Эта операция тесно связана с операцией вычисления в b-адической арифметике частного u/w, так как для некоторого целого числа q оно равно $u/w=q+b^mv/w$ (см. упр. 4.1–31). Преимущество такого деления заключается в том, что нет необходимости в коррекции пробного делителя.)

(b) Применим алгоритм (a) для получения uv. Необходимый объем памяти останется неизменным, если объединить операции умножения и вычисления остатка по модулю следующим образом. Присвоим $k\leftarrow0$, $t\leftarrow0$. Пока $k<n$, присваиваем $t\leftarrow t+u_kv$, $t\leftarrow(t-xw)/b$, $k\leftarrow k+1$, где $x=t_0w' \bmod b$ выбирается так, чтобы $t-xw$ было кратным b. Тем самым обеспечим инвариантность отношения $b^kt\equiv(u_{k-1}\dots u_0)v$ (по модулю w). При этом принимаем, что числа t, u и v представлены в прямом коде со знаком. Можно оперировать и неотрицательными числами $<2w$ или числами, представленными в виде дополнений, как описано Шэндом (Shand) и Вуйлеменом (Vuillemin), а также Корнерупом

(Kornerup) [*IEEE Symp. Computer Arithmetic* **11** (1993), 252–259, 277–283]. При большом n для повышения скорости умножения может быть применена методика, описанная в разделе 4.3.3.

(c) Представим все числа, конгруэнтные u (по модулю w), внутренним значением $r(u)$ ($r(u) \equiv b^n u$). Далее операции сложения и вычитания выполняются обычным образом, а операция умножения — в виде $r(uv) = \text{bmult}(r(u), r(v))$, где bmult — операция из алгоритма (b). В начале вычислений каждый операнд u заменяется на $r(u) = \text{bmult}(u, a)$, где $a = b^{2n} \bmod w$ — константа, полученная до начала вычислений. И наконец заменим каждое $r(u)$ на $u = \text{bmult}(r(u), 1)$. [В приложении к RSA-кодированию программа переделана так, чтобы необходимость в предварительных и заключительных вычислениях отпала (см. раздел 4.5.4).]

42. Дж. М. Холт (J. M. Holte) в работе, опубликованной в журнале *AMM* **104** (1997), 138–149, получил точную формулу

$$P_{nk} = \frac{1}{m!} \sum_j \left[{m \atop m-j} \right] b^{-jn} \sum_{r=0}^{k} \binom{m+1}{r} (k+1-r)^{m-j}.$$

Внутренняя сумма равна $\sum_{r=0}^{k} (-1)^r \binom{m+1}{r} (k+1-r)^m = \left\langle {m \atop k} \right\rangle$ при $j = 0$. (В упр. 5.1.3–25 поясняется причина появления чисел Эйлера.)

43. Согласно упр. 1.2.4–35 получаем $w = \lfloor W/2^{16} \rfloor$, где $W = (2^8+1)t = (2^8+1)(uv+2^7)$. Поэтому, если $xy/255 > c + \frac{1}{2}$, имеем $c < 2^8$. Следовательно, $w \geq \lfloor (2^{16}(c+1) + 2^8 - c)/2^{16} \rfloor \geq c+1$. Если $xy/255 < c + \frac{1}{2}$, то получаем $w \leq \lfloor (2^{16}(c+1) - c - 1)/2^{16} \rfloor = c$. [См. J. F. Blinn, *IEEE Computer Graphics and Applic.* **14**, 6 (November, 1994), 78–82.]

РАЗДЕЛ 4.3.2

1. Решение единственно, так как $7 \cdot 11 \cdot 13 = 1001$. Из конструктивного доказательства теоремы C следует, что ответом будет $((11 \cdot 13)^6 + 6 \cdot (7 \cdot 13)^{10} + 5 \cdot (7 \cdot 11)^{12}) \bmod 1001$. Но этот ответ не совсем точный! В соответствии с (24) имеем $v_1 = 1$, $v_2 = (6-1) \cdot 8 \bmod 11 = 7$, $v_3 = ((5-1) \cdot 2 - 7) \cdot 6 \bmod 13 = 6$, так что $u = 6 \cdot 7 \cdot 11 + 7 \cdot 7 + 1 = 512$.

2. Нет. Найдется хотя бы одно число u, для которого условия теоремы не удовлетворяются. Необходимым и достаточным является дополнительное условие $u_1 \equiv \cdots \equiv u_r$ (по модулю 1). Из этого следует, что такое обобщение не представляет большого интереса.

3. Из того, что $u \equiv u_i$ (по модулю m_i), следует $u \equiv u_i$ (по модулю $\gcd(m_i, m_j)$). Так что, если решение существует, то условие $u_i \equiv u_j$ (по модулю $\gcd(m_i, m_j)$) должно непременно выполняться. Далее, если $u \equiv v$ (по модулю m_j) при всех j, то $u - v$ кратно $\text{lcm}(m_1, \ldots, m_r) = m$; следовательно, имеется не более одного решения.

Доказательство можно завершить неконструктивным способом, подсчитав число различных r-наборов $(u_1, \ldots, u_r)$, которые удовлетворяют условиям $0 \leq u_j < m_j$ и $u_i \equiv u_j$ (по модулю $\gcd(m_i, m_j)$). Если это число равно m, то решение должно существовать, так как при изменении числа u от a до $a+m-1$ набор $(u \bmod m_1, \ldots, u \bmod m_r)$ принимает m различных значений. Предположим, что выбранные $u_1, \ldots, u_{r-1}$ удовлетворяют перечисленным условиям. Необходимо подобрать $u_r \equiv u_j$ (по модулю $\gcd(m_j, m_r)$) для $1 \leq j < r$. В соответствии с обобщенной китайской теоремой об остатках для $r-1$ элементов это можно сделать

$$\begin{aligned} m_r / \text{lcm}(\gcd(m_1, m_r), \ldots, \gcd(m_{r-1}, m_r)) &= m_r / \gcd(\text{lcm}(m_1, \ldots, m_{r-1}), m_r) \\ &= \text{lcm}(m_1, \ldots, m_r) / \text{lcm}(m_1, \ldots, m_{r-1}) \end{aligned}$$

способами. (Данное доказательство основано на тождествах (10), (11), (12), (14) из раздела 4.5.2.)

Конструктивное доказательство [A. S. Fraenkel, *Proc. Amer. Math. Soc.* **14** (1963), 790–791]; обобщающее (25), можно провести следующим образом. Пусть $M_j = \operatorname{lcm}(m_1, \ldots, m_j)$; необходимо найти $u = v_r M_{r-1} + \cdots + v_2 M_1 + v_1$, где $0 \le v_j < M_j/M_{j-1}$. Предположим, что $v_1, \ldots, v_{j-1}$ уже определены. Тогда нужно решить уравнение

$$v_j M_{j-1} + v_{j-1} M_{j-2} + \cdots + v_1 \equiv u_j \pmod{m_j}.$$

Здесь $v_{j-1} M_{j-2} + \cdots + v_1 \equiv u_i \equiv u_j \pmod{\gcd(m_i, m_j)}$ при $i < j$ по предположению. Так что $c = u_j - (v_{j-1} M_{j-2} + \cdots + v_1)$ кратно

$$\operatorname{lcm}(\gcd(m_1, m_j), \ldots, \gcd(m_{j-1}, m_j)) = \gcd(M_{j-1}, m_j) = d_j.$$

Поэтому нужно решить $v_j M_{j-1} \equiv c \pmod{m_j}$. В соответствии с алгоритмом Евклида найдется число c_j, такое, что $c_j M_{j-1} \equiv d_j \pmod{m_j}$; следовательно, можно взять

$$v_j = (c_j\, c)/d_j \bmod (m_j/d_j).$$

Обратите внимание на то, что, как и при неконструктивном способе доказательства, получено $m_j/d_j = M_j/M_{j-1}$.

4. (После вычисления $m_4 = 91 = 7 \cdot 13$ мы исчерпали все произведения двух или более нечетных простых чисел, меньших 100, поэтому все числа $m_5, \ldots$ должны быть простыми.)

$$\begin{array}{lllll} m_7 = 79, & m_8 = 73, & m_9 = 71, & m_{10} = 67, & m_{11} = 61, \\ m_{12} = 59, & m_{13} = 53, & m_{14} = 47, & m_{15} = 43, & m_{16} = 41, \\ m_{17} = 37, & m_{18} = 31, & m_{19} = 29, & m_{20} = 23, & m_{21} = 17. \end{array}$$

После этого процесс прерывается ($m_{22} = 1$ не подходит).

5. Нет. Очевидная верхняя граница равна

$$3^4 5^2 7^2 11^1 \cdots = \prod_{\substack{p \text{ нечетное} \\ p \text{ простое}}} p^{\lfloor \log_p 100 \rfloor}$$

и достигается при $m_1 = 3^4$, $m_2 = 5^2$ и т. д. (Труднее, однако, определить максимальное значение $m_1 \ldots m_r$ в случае, когда r фиксировано, либо максимизировать $e_1 + \cdots + e_r$ со взаимно простыми e_j, если используются модули $2^{e_j} - 1$.)

6. (a) Если $e = f + kg$, то $2^e = 2^f (2^g)^k \equiv 2^f \cdot 1^k \pmod{2^g - 1}$. Поэтому при $2^e \equiv 2^f \pmod{2^g - 1}$ имеем $2^{e \bmod g} \equiv 2^{f \bmod g} \pmod{2^g - 1}$; а так как эти последние величины расположены между нулем и $2^g - 1$, должно быть $e \bmod g = f \bmod g$. (b) Согласно п. (a) имеем $(1 + 2^d + \cdots + 2^{(c-1)d}) \cdot (2^e - 1) \equiv (1 + 2^d + \cdots + 2^{(c-1)d}) \cdot (2^d - 1) = 2^{cd} - 1 \equiv 2^{ce} - 1 \equiv 2^1 - 1 = 1 \pmod{2^f - 1}$.

7. Имеем $v_j m_{j-1} \ldots m_1 \equiv u_j - (v_{j-1} m_{j-2} \ldots m_1 + \cdots + v_1)$ и $C_j m_{j-1} \ldots m_1 \equiv 1 \pmod{m_j}$ из (23), (25) и (26); см. P. A. Pritchard, *CACM* **27** (1984), 57.

Этот метод преобразования формул включает столько же арифметических операций, но меньше констант. Количество констант будет меньше только в том случае, если упорядочить модули таким образом, чтобы удовлетворялось неравенство $m_1 < m_2 < \cdots < m_r$; в противном случае потребуется таблица значений $m_i \bmod m_j$. Может показаться, что это упорядочение модулей связано с бóльшими вычислительными затратами, чем если бы мы придавали наибольшие значения сначала модулю m_1, затем — m_2 и т. д., так как операций по модулю m_r нужно выполнить значительно больше, чем по модулю m_1. Но поскольку v_j может быть столь же большим, как и $m_j - 1$, упорядочение $m_1 < m_2 < \cdots < m_r$ лучше ввести и в (24). Таким образом, желательно применить эту идею и к другим формулам в тексте, хотя, как показано в разделе 4.3.3B, они полезны и тогда, когда модули представлены в виде (14).

8. Модуль m_j: $m_{j-1}\dots m_1 v_j \equiv m_{j-1}\dots m_1(\dots((u_j - v_1)c_{1j} - v_2)c_{2j} - \cdots - v_{j-1})c_{(j-1)j} \equiv m_{j-2}\dots m_1(\dots(u_j - v_1)c_{1j} - \cdots - v_{j-2})c_{(j-2)j} - v_{j-1}m_{j-2}\dots m_1 \equiv \cdots \equiv u_j - v_1 - v_2 m_1 - \cdots - v_{j-1}m_{j-2}\dots m_1$.

9. $u_r \leftarrow ((\dots(v_r m_{r-1} + v_{r-1})\,m_{r-2} + \cdots)\,m_1 + v_1) \bmod m_r,\ \dots,$
$u_2 \leftarrow (v_2 m_1 + v_1) \bmod m_2,\ u_1 \leftarrow v_1 \bmod m_1$.

(Вычисления следует выполнять именно в таком порядке, если u_j и v_j должны располагаться в одних и тех же ячейках памяти, что и допускается в (24).)

10. Если переопределить оператор "mod" так, чтобы он выдавал значения в симметричной области, то основные формулы для арифметических операций (2)–(4) и уравнений перевода (24) и (25) останутся теми же, а число u в (25) будет принадлежать области (10). (Здесь (25) — *представление в уравновешенной позиционной системе счисления со смешанным основанием*, являющееся обобщением уравновешенной троичной системы.) Сравнение двух чисел можно по-прежнему выполнять слева направо простым способом, описанным в тексте. Далее, если в компьютере реализован прямой код, можно хранить значение числа u в одном машинном слове, даже в том случае, когда m_j почти в два раза больше машинного слова. Однако арифметические операции, аналогичные (11) и (12), осуществляются сложнее. Создается впечатление, что применение этой идеи на большинстве компьютеров приведет к некоторому замедлению выполнения операций.

11. Умножим на $\frac{1}{2}(m+1) = (\frac{1}{2}(m_1+1), \dots, \frac{1}{2}(m_r+1))$.

12. Заменим в (11) число m_j числом m. [Можно также выполнить проверку переполнения, если m нечетно, сохраняя внешние биты $u_0 = u \bmod 2$ и $v_0 = v \bmod 2$. В таком случае переполнение наступит тогда и только тогда, когда $u_0 + v_0 \not\equiv w_1 + \cdots + w_r \frac{1}{2}{}^* IeC$помодулю2, где $(w_1, \dots, w_r)$ — значения разрядов, соответственно с $u+v$ представленные в системе со смешанным основанием.

13. (a) Пусть равенство $x^2 - x = (x-1)x \equiv 0\frac{1}{2}{}^* IeC$помодулю$10^n$ эквивалентно равенству $(x-1)x \equiv 0\frac{1}{2}{}^* IeC$помодулю$p^n$ для $p = 2$ и 5. Из двух чисел x и $x-1$ одно должно быть кратным числу p, и тогда другое из них будет взаимно простым с p^n; поэтому либо x, либо $x-1$ должно быть кратным числу p^n. Если $x \bmod 2^n = x \bmod 5^n = 0$ или 1, то должно выполняться $x \bmod 10^n = 0$ или 1. Следовательно, свойством автоморфа обладает число, для которого $x \bmod 2^n \neq x \bmod 5^n$. (b) Если $x = qp^n + r$, где $r = 0$ или 1, то $r \equiv r^2 \equiv r^3$, так что $3x^2 - 2x^3 \equiv (6qp^n r + 3r) - (6qp^n r + 2r) \equiv r\frac{1}{2}{}^* IeC$помодулю$p^{2n}$. (c) Положим c' равным $(3(cx)^2 - 2(cx)^3)/x^2 = 3c^2 - 2c^3 x$.

Примечание. Так как последние k цифр n-разрядного автоморфного числа образуют k-разрядное автоморфное число, можно говорить о двух ∞-разрядных автоморфных числах x и $1-x$, которые являются 10-адическими числами (см. упр. 4.1–31). Ряд 10-адических чисел эквивалентен ряду упорядоченных пар (u_1, u_2) чисел, представленных в модулярном виде, где u_1 есть 2-адическое число, а u_2 есть 5-адическое число.

14. Найдем циклическую свертку $(z_0, z_1, \dots, z_{n-1})$ приближений к $(a_0u_0, a_1u_1, \dots, a_{n-1}u_{n-1})$ и $(a_0v_0, a_1v_1, \dots, a_{n-1}v_{n-1})$ в формате с плавающей точкой, где константы $a_k = 2^{-(kq \bmod n)/n}$ уже вычислены. Теперь из тождеств $u = \sum_{k=0}^{n-1} u_k a_k 2^{kq/n}$ и $v = \sum_{k=0}^{n-1} v_k a_k 2^{kq/n}$ следует $w = \sum_{k=0}^{n-1} t_k a_k 2^{kq/n}$, где $t_k \approx z_k/a_k$. При сохранении требуемой точности каждое из чисел будет очень близко к целому. Представление числа w может быть легко получено через эти целые числа. [См. R. Crandall and B. Fagin, *Math. Comp.* **62** (1994), 305–324.]

РАЗДЕЛ 4.3.3

1.

12×23:	34×41:	22×18:	1234×2341:
02	12	02	0276
02	12	02	0276
−01	+03	+00	−0396
06	04	16	1394
06	04	16	1394
0276	1394	0396	2888794

2. $\sqrt{Q+\lfloor\sqrt{Q}\rfloor}\le\sqrt{Q+\sqrt{Q}}<\sqrt{Q+2\sqrt{Q}+1}=\sqrt{Q}+1$, так что $\lfloor\sqrt{Q+R}\rfloor\le\lfloor\sqrt{Q}\rfloor+1$.

3. При $k\le 2$ неравенство выполняется, поэтому предположим, что $k>2$. Пусть $q_k=2^{Q_k}$, $r_k=2^{R_k}$, так что $R_k=\lfloor\sqrt{Q_k}\rfloor$ и $Q_k=Q_{k-1}+R_{k-1}$. Необходимо показать, что $1+(R_k+1)2^{R_k}\le 2^{Q_{k-1}}$. Отметим, что это неравенство грубое. Возможный способ доказательства — проанализировать, выполняется ли $1+(R_k+1)2^{R_k}\le 1+2^{2R_k}$ и $2R_k<Q_{k-1}$ при $k>2$. (Тот факт, что $2R_k<Q_{k-1}$, легко доказывается по индукции, поскольку $R_{k+1}-R_k\le 1$ и $Q_k-Q_{k-1}\ge 2$.)

4. Для $j=1,\ldots,r$ вычисляем $U_e(j^2)$, $jU_o(j^2)$, $V_e(j^2)$, $jV_o(j^2)$; рекурсивно обращаясь к алгоритму умножения, вычисляем

$$W(j)=(U_e(j^2)+jU_o(j^2))(V_e(j^2)+jV_o(j^2)),$$
$$W(-j)=(U_e(j^2)-jU_o(j^2))(V_e(j^2)-jV_o(j^2)).$$

Тогда получаем $W_e(j^2)=\frac{1}{2}(W(j)+W(-j))$, $W_o(j^2)=\frac{1}{2}(W(j)-W(-j))$. Вычисляем также $W_e(0)=U(0)V(0)$. Затем строим таблицы разностей для полиномов W_e и W_o, степени которых равны r и $r-1$ соответственно.

Этот метод позволяет уменьшить размер обрабатываемых чисел и количество операций сложения и умножения. Единственный минус — более длинная программа (так как усложняется управление процессом и некоторые вычисления должны выполняться над числами со знаком).

Другая возможность заключается в определении значений полиномов W_e и W_o в точках 1^2, 2^2, 4^2, …, $(2^r)^2$. Несмотря на то что величины обрабатываемых чисел здесь больше, вычисления выполняются быстрее, поскольку все операции умножения заменяются операциями сдвига, а все операции деления выполняются над двоичными числами вида $2^j(2^k-1)$. (Деление на такие числа осуществляется с помощью простых средств.)

5. Начнем построение последовательностей q и r с достаточно больших начальных значений q_0 и q_1 так, чтобы выполнялось неравенство из упр. 3. После этого из формул, аналогичных формулам, которые предшествовали теореме B, можно найти, что $\eta_1\to 0$ и $\eta_2=(1+1/(2r_k))2^{1+\sqrt{2Q_k}-\sqrt{2Q_{k+1}}}(Q_k/Q_{k+1})$. При $k\to\infty$ множитель $Q_k/Q_{k+1}\to 1$, и поэтому его можно не учитывать, ибо необходимо доказать, что $\eta_2<1-\epsilon$ при всех больших k. Итак, получаем $\sqrt{2Q_{k+1}}=\sqrt{2Q_k+2\lceil\sqrt{2Q_k}\rceil+2}\ge\sqrt{(2Q_k+2\sqrt{2Q_k}+1)+1}\ge\sqrt{2Q_k}+1+1/(3R_k)$. Отсюда при достаточно больших k $\eta_2\le(1+1/(2r_k))2^{-1/(3R_k)}$ и $\lg\eta_2<0$.

Замечание. Алгоритм T тоже можно модифицировать так, чтобы он находил последовательность сходного типа q_0, q_1, …, построенную на базе n в том смысле, что после шага T1 $n\approx q_k+q_{k+1}$. Такая модификация приводит к оценке (20).

6. Любой общий делитель чисел $6q+d_1$ и $6q+d_2$ должен также быть делителем их разности d_2-d_1. Такими $\binom{6}{2}$ разностями будут 2, 3, 4, 6, 8, 1, 2, 4, 6, 1, 3, 5, 2, 4, 2, поэтому необходимо только показать, что на каждое из простых чисел 2, 3, 5 делится не более чем

одно из данных чисел. Ясно, что только число $6q+2$ четно и только число $6q+3$ кратно 3. Имеется также одно самое большое число, кратное 5, поскольку $q_k \not\equiv 3$ (по модулю 5).

7. Пусть $p_{k-1} < n \le p_k$. Тогда для некоторого постоянного числа c $t_k \le 6t_{k-1} + ck3^k$. Поэтому $t_k/6^k \le t_{k-1}/6^{k-1} + ck/2^k \le t_0 + c\sum_{j\ge 1} j/2^j = M$. Таким образом, $t_k \le M \cdot 6^k = O(p_k^{\log_3 6})$.

8. Ложно. Для этого достаточно посмотреть на результат при $k = 2$.

9. $\tilde{u}_s = \hat{u}_{(qs) \bmod K}$. В частности, при $q = -1$ получаем $\hat{u}_{(-r) \bmod K}$, что при выполнении обратного преобразования позволяет избежать обращения данных (побитового инвертирования кода).

10. $A^{[j]}(s_{k-1}, \ldots, s_{k-j}, t_{k-j-1}, \ldots, t_0)$ можно записать в виде

$$\sum_{0\le t_{k-1},\ldots,t_{k-j}\le 1} \omega^{2^{k-j}(s_{k-j}\ldots s_{k-1})2\cdot(t_{k-1}\ldots t_{k-j})2} \left(\sum_{0\le p<K} \omega^{tp} u_p\right)\left(\sum_{0\le q<K} \omega^{tq} v_q\right),$$

а это равно $\sum_{p,q} u_p v_q S(p,q)$, где $|S(p,q)| = 0$ или 2^j. Для точных значений $2^{2k}/2^j$ чисел p и q получаем $|S(p,q)| = 2^j$.

11. Автомат не может иметь $z_2 = 1$ до того, как у него не будет $c \ge 2$, а это в момент $3j-1$ случится сначала для автомата M_j. Отсюда следует, что автомат M_j не может иметь $z_2 z_1 z_0 \ne 000$ до момента $3(j-1)$. Далее, если в момент t в автомате M_j компонента $z_0 \ne 0$, то нельзя сделать $z_0 = 0$, не повлияв при этом на выходные данные. Но на выходные данные нельзя повлиять при указанном значении z_0, по крайней мере до момента $t+j-1$, поэтому должно выполняться неравенство $t+j-1 \le 2n$. Поскольку мы доказали, что $3(j-1) \le t$, должно быть $4(j-1) \le 2n$ или, что то же самое, $j-1 \le n/2$, т. е. $j \le \lfloor n/2 \rfloor + 1$. Поскольку для обработки входных данных $u = v = 2^n - 1$ необходимо использовать автоматы M_j для всех $j \le \lfloor n/2 \rfloor + 1$, полученная оценка является наилучшей из возможных. (Например, из табл. 2 видно, что автомат M_2 необходим для умножения двухбитовых чисел в момент 3.)

12. Можно "просмотреть" K списков команд для компьютера наподобие MIX, выполняя первую команду из каждого списка за $O(K + (N \log N)^2)$ шагов следующим образом. (i) Алгоритм поразрядной сортировки списка (раздел 5.2.5) сгруппирует все идентичные команды за время $O(K+N)$. (ii) Каждый из наборов j идентичных команд может быть выполнен за $O(\log N)^2 + O(j)$ шагов, а имеется $O(N^2)$ таких наборов. Все списки будут просмотрены за конечное число просмотров. Остаются очевидные детали. Например, арифметические операции можно промоделировать, переведя числа p и q в двоичную систему счисления. [*SICOMP* **9** (1980), 490–508.]

13. Если на перемножение двух n-битовых чисел затрачивается $T(n)$, то умножение m-битового числа на n-битовое можно выполнить за $\lceil n/m \rceil T(m) + O(n+m)$ операций, разбив для этого n-битовое число на $\lceil n/m \rceil$ групп m-битовых чисел. Из результатов, полученных в этом разделе, следует, что для машины Тьюринга оценка времени равна $O(n \log m \log\log m)$, для машин со случайной выборкой слов ограниченного размера — $O(n \log m)$, а для машин с указателями — $O(n)$.

15. Известная наилучшая верхняя оценка, равная $O(n(\log n)^2 \log\log n)$, предложена М. Дж. Фишером (M. J. Fischer) и Л. Дж. Штокмейером (L. J. Stockmeyer) [*J. Comp. and Syst. Sci.* **9** (1974), 317–331]. Рассматриваемая ими конструкция ориентирована на машину Тьюринга с множеством лент; для машин с указателями эта оценка равна $O(n \log n)$. М. С. Патерсон (M. S. Paterson), М. Дж. Фишер (M. J. Fischer) и А. Р. Мейер (A. R. Meyer) получили наилучшую нижнюю оценку порядка $n \log n / \log\log n$, опубликованную в *SIAM/AMS Proceedings* **7** (1974), 97–111. Она применима только к машинам Тьюринга с множеством лент.

16. Пусть 2^k — наименьшая степень 2, которая превышает $2K$. Присвоим $a_t \leftarrow \omega^{-t^2/2} u_t$ и $b_t \leftarrow \omega^{(2K-2-t)^2/2}$, где $u_t = 0$ для $t \geq K$. Нужно вычислить свертки $c_r = \sum_{j=0}^{r} a_j b_{r-j}$ для $r = 2K-2-s$ при $0 \leq s < K$. Свертки можно найти при помощи трех быстрых преобразований Фурье порядка 2^k, как описано в разделе. [Заметим, что такой способ, который иногда называют "chirp-преобразование", пригоден для оперирования любыми комплексными числами ω, а не только корнями из единицы. [См. L. I. Bluestein, *Northeast Electronics Res. and Eng. Meeting Record* **10** (1968), 218–219; D. H. Bailey and P. N. Swarztrauber, *SIAM Review* **33** (1991), 389–404.]

17. Значение $D_n = K_{n+1} - K_n$ удовлетворяет $D_1 = 2$, $D_{2n} = 2D_n$ и $D_{2n+1} = D_n$. Следовательно, $D_n = 2^{e_1 - t + 2}$, если n имеет указанный вид. Отсюда по индукции следует, что $K_n = 3^{e_1} + \sum_{l=2}^{t} 3^{e_l} 2^{e_1 - e_l - l + 3}$.

Между прочим, K_n нечетно и можно умножать n-разрядное целое число на $(n+1)$-разрядное целое число, умножив $(K_n + K_{n+1})/2$ 1-разрядных чисел. Производящая функция $K(z) = \sum_{n \geq 1} K_n z^n$ удовлетворяет уравнению $zK(z) + z^2 = K(z^2)(z+1)(z+2)$; отсюда $K(-1) = 1$ и $K(1) = \frac{1}{5}$.

18. В следующей схеме используется $3N + S_N$ позиций в рабочем пространстве. Здесь в обозначениях предыдущего упражнения $S_1 = 0$, $S_{2n} = S_n$ и $S_{2n-1} = S_n + 1$, т. е. $S_n = e_1 - e_t - t + 2 - [t = 1]$. Пусть $N = 2n - \epsilon$, где ϵ равно 0 или 1, и предположим, что $N > 1$. Для заданных N-разрядных чисел $u = 2^n U_1 + U_0$ и $v = 2^n V_1 + V_0$ сначала формируются $|U_0 - U_1|$ и $|V_0 - V_1|$ в двух n-разрядных областях, начиная с позиций 0- и n-й $(3N + S_N)$-разрядной рабочей области. После этого произведение помещается в рабочую область, начиная с позиции $3n + S_n$. Следующий шаг заключается в формировании $2(n-\epsilon)$-разрядного произведения $U_1 V_1$, начиная с позиции 0. С помощью этого произведения изменяем $3n - 2\epsilon$ разрядов, начиная с позиции $3n + S_n$, и записываем в них значение $U_1 V_1 - (U_0 - U_1)(V_0 - V_1) + 2^n U_1 V_1$. (Заметим, что $3n - 2\epsilon + 3n + S_n = 3N + S_N$.) В итоге формируется $2n$-разрядное произведение $U_0 V_0$, начиная с позиции 0. Оно добавляется к частичному результату, начиная с позиций $2n + S_n$ и $3n + S_n$. Кроме того, необходимо переместить $2N$-разрядный результат на его окончательное место, сдвинув его вниз на $2n + S_n$ позиций.

Окончательное перемещение может быть запрещено при помощи оригинального способа, который циклически сдвигает выходное значение на заданную величину внутри сформированной рабочей области. Если $2N$-разрядное произведение не должно присоединяться к вспомогательной рабочей области, необходимо иметь еще около N разрядов памяти (т. е. общее количество разрядов для ввода, вывода и временного хранения должно в этом случае равняться примерно $6N$ разрядам вместо $5N$). См. R. Maeder, *Lecture Notes in Comp. Sci.* **722** (1993), 59–65.

19. Пусть $m = s^2 + r$ при $-s < r \leq s$. Можно использовать (2) при $U_1 = \lfloor u/s \rfloor$, $U_0 = u \bmod s$, $V_1 = \lfloor v/s \rfloor$, $V_0 = v \bmod s$, а s предоставить роль 2^n. Если известны знаки чисел $U_1 - U_0$ и $V_1 - V_0$, то известно также, как вычислить произведение $|U_1 - U_0|\,|V_1 - V_0|$, которое $< m$ и нужно ли добавлять его или вычитать. Остается только умножить результат на s и $s^2 \equiv -r$. Каждая из этих операций может быть выполнена путем четырех умножений и/или делений при помощи способа, описанного в упр. 3.2.1.1–9, но при этом потребуется только семь операций, так как одно из умножений, необходимых для вычисления $sx \bmod m$, — это умножение на r или $r + s$. Таким образом, достаточно 14 операций умножения и/или деления (либо 12 в случае, если $u = v$ или если u — константа). Не имея возможности сравнивать операнды, мы смогли выполнить работу без единого дополнительного умножения за счет раздельного вычисления $U_0 V_1$ и $U_1 V_0$.

РАЗДЕЛ 4.4

1. Выполнив сложение и умножение в системе счисления с основанием B_j, получаем $(\ldots(a_m b_{m-1} + a_{m-1})b_{m-2} + \cdots + a_1)b_0 + a_0$*.

	T.	= 20(cwt.	= 8(st.	= 14(lb.	= 16 oz.)))
Начать с нуля	0	0	0	0	0
Прибавить 3	0	0	0	0	3
Умножить на 24	0	0	0	4	8
Прибавить 9	0	0	0	5	1
Умножить на 60	0	2	5	9	12
Прибавить 12	0	2	5	10	8
Умножить на 60	8	3	1	0	0
Прибавить 37	8	3	1	2	5

(Операции сложения и умножения на константу в системе счисления со смешанным основанием легко выполняются при помощи простого обобщения обычного правила переноса; см. упр. 4.3.1–9.)

2. Вычислим $\lfloor u/B_0 \rfloor$, $\lfloor \lfloor u/B_0 \rfloor / B_1 \rfloor$ и т. д.; остатки суть A_0, A_1 и т. д. Деление выполнено в системе счисления с основанием b_j.

	d.	= 24(h.	= 60(m.	= 60 s.))	
Начать с u	3	9	12	37	
Разделить на 16	0	5	4	32	Остаток = 5
Разделить на 14	0	0	21	45	Остаток = 2
Разделить на 8	0	0	2	43	Остаток = 1
Разделить на 20	0	0	0	8	Остаток = 3
Разделить на ∞	0	0	0	0	Остаток = 8

Результат. 8 длинных тонн, 3 хандредвейта, 1 стоун, 2 фунта, 5 унций.

3. Предложенная Г. Л. Стилом (мл.) (G. L. Steele, Jr.) и Йоном Л. Уайтом (Jon L. White) и впервые опубликованная в журнале *CACM* **2**, 7 (July, 1959), 27, процедура обобщает алгоритм Д. Таранто для $B = 2$.

A1. [Начальная установка.] Присвоить $M \leftarrow 0$, $U_0 \leftarrow 0$.

A2. [Выполнено?] Если $u < \epsilon$ или $u > 1 - \epsilon$, то перейти к шагу A4. (В противном случае M-разрядная дробь будет удовлетворять заданным условиям.)

A3. [Преобразовать.] Присвоить $M \leftarrow M + 1$, $U_{-M} \leftarrow \lfloor Bu \rfloor$, $u \leftarrow Bu \bmod 1$, $\epsilon \leftarrow B\epsilon$ и возвратиться к шагу A2. (Это преобразование возвращает нас, по сути, в предыдущее состояние. Остается решить проблему преобразования дроби u в дробь U по основанию B с минимальным числом разрядов, но таким, чтобы удовлетворялось неравенство $|U - u| < \epsilon$. Заметим, однако, что теперь ϵ может быть ≥ 1; в таком случае можно сразу перейти к шагу A4 и не сохранять новое значение ϵ.)

A4. [Округлить.] Если $u \geq \frac{1}{2}$, то увеличить U_{-M} на 1. (Если равенство $u = \frac{1}{2}$ выполняется точно, то предпочтительнее пользоваться другим правилом округления: "увеличить U_{-M} на 1 только тогда, когда оно нечетно". См. раздел 4.2.2.) ▮

---

* В таблице приняты следующие обозначения: *T.* — длинные тонны, *cwt.* — хандредвейты, *st.* — стоуны, *lb.* — фунты, *oz.* — унции. — *Прим. перев.*

Число U_{-M} на шаге A4 никогда не будет увеличиваться с $B-1$ до B. В этом случае, если $(U_{-M} = B-1)$ должно быть $M > 0$, ни одна из $(M-1)$-разрядных дробей не обеспечивает достаточной точности. Стил и Уайт в работе *SIGPLAN Notices* **25**, 6 (June, 1990), 112–126, продолжили анализ преобразований в формате с плавающей точкой. (См. также работу Д. Э. Кнута в сборнике *Beauty is Our Business*, edited by W. H. J. Feijen et al. (New York: Springer, 1990), 233–242.)

4. (a) $1/2^k = 5^k/10^k$. (b) Все простые делители числа b делят B.

5. Это возможно только в том случае, когда $10^n - 1 \le c < w$; см. (3).

7. $\alpha u \le ux \le \alpha u + u/w \le \alpha u + 1$; следовательно, $\lfloor \alpha u \rfloor \le \lfloor ux \rfloor \le \lfloor \alpha u + 1 \rfloor$. Далее, для частного случая, оговоренного выше, имеем $ux < \alpha u + \alpha$ и $\lfloor \alpha u \rfloor = \lfloor \alpha u + \alpha - \epsilon \rfloor$ для $0 < \epsilon \le \alpha$.

8.
```
    ENT1 0
    LDA  U
  1H MUL  =1//10=
  3H STA  TEMP
    MUL  =-10=
    SLAX 5
    ADD  U
    JANN 2F
    LDA  TEMP       (Может случиться только
    DECA 1              на первой итерации
    JMP  3B             согласно упр. 7.)
  2H STA  ANSWER,1  (Может быть минус нуль.)
    LDA  TEMP
    INC1 1
    JAP  1B
```

9. Пусть $N = 2^{2^{k+1}}$. При выполнении вычислений происходит присвоение

$$v \leftarrow \left\lfloor \frac{(2^2-1)}{2^1}\frac{(2^4+1)}{2^4}\frac{(2^8+1)}{2^8}\cdots\frac{(2^{2^k}+1)}{2^{2^k}}(u+1) \right\rfloor = \left\lfloor \frac{8}{5}\frac{N-1}{N}(u+1) \right\rfloor .$$

Поэтому $q = \lfloor u/10 + \epsilon_u \rfloor$, где $\epsilon_u = \frac{1}{10}(1-(u+1)/N)$. Так как $N \bmod 10 = 6$ и $0 \le \epsilon_u < 1/10$ при $0 \le u < N$, то $q = \lfloor u/10 \rfloor$ при $0 \le u < N+4$.

Когда u принадлежит этому интервалу, получаем $r = u \bmod 10 + \lfloor 1 - (u+1+5\theta_u)/N \rfloor$, где $\theta_u = \frac{N-1}{5}(u+1) \bmod \frac{N}{8}$. Если θ_u велико (к примеру, $\theta_u = N/8 - t$, где $0 < t < N/40$), получаем $u + 1 \equiv 5t \pmod{N/8}$. Таким образом, $u + 1 + 5\theta_u \le N$ при $u < N/2$. В противном случае $\theta_u \le N/8 - N/40 = N/10$, и снова получаем $u + 1 + \theta_u \le N$ при $u < N/2$. Случаи, когда $u = N/2$, $N/2+1$, $N/2+2$ и $N/2+3$, как легко видеть, не создают проблем. Но когда $u = N/2 + 4$, находим $u \bmod 10 = 2$ и $r = 1$.

[Альтернативный вариант $r \leftarrow u - 8q$, $r \leftarrow r - 2q$ будет выполняться в бóльшем интервале, но на 8-битовом компьютере немного медленнее. Это упражнение основано на идеях Р. А. Воувелса (R. A. Vowels), *Australian Comp. J.* **24** (1992), 81–85.]

10. (i) Сдвинуть вправо на один разряд; (ii) извлечь левый бит каждой группы; (iii) сдвинуть на два разряда результат, полученный в (ii); (iv) сдвинуть этот результат вправо на один разряд и сложить его с результатом (iii); (v) Вычесть результат (iv) из результата (i).

11.

$$
\begin{array}{rl}
 & 5.7\,7\,2\,1 \\
- & 1\,0 \\
\hline
 & 4\,7.7\,2\,1 \\
- & 9\,4 \\
\hline
 & 3\,8\,3.2\,1 \\
- & 7\,6\,6 \\
\hline
 & 3\,0\,6\,6.1 \\
- & 6\,1\,3\,2 \\
\hline
 & 2\,4\,5\,2\,9
\end{array}
$$

Результат: $(24529)_{10}$.

12. Сначала преобразуем число, представленное в троичной системе, в девятеричную систему, а затем поступаем так же, как при преобразовании числа из восьмеричной системы счисления в десятичную, но без удвоений. Преобразование из десятичной системы в девятеричную выполняется аналогично. В данном примере имеем следующее.

$$
\begin{array}{rl}
 & 1.7\,6\,4\,7\,2\,3 \\
- & 1 \\
\hline
 & 1\,6.6\,4\,7\,2\,3 \\
- & 1\,6 \\
\hline
 & 1\,5\,0.4\,7\,2\,3 \\
- & 1\,5\,0 \\
\hline
 & 1\,3\,5\,4.7\,2\,3 \\
- & 1\,3\,5\,4 \\
\hline
 & 1\,2\,1\,9\,3.2\,3 \\
- & 1\,2\,1\,9\,3 \\
\hline
 & 1\,0\,9\,7\,3\,9.3 \\
- & 1\,0\,9\,7\,3\,9 \\
\hline
 & 9\,8\,7\,6\,5\,4
\end{array}
$$

Результат: $(987654)_{10}$.

$$
\begin{array}{rl}
 & 9.8\,7\,6\,5\,4 \\
+ & 9 \\
\hline
 & 1\,1\,8.7\,6\,5\,4 \\
+ & 1\,1\,8 \\
\hline
 & 1\,3\,1\,6.6\,5\,4 \\
+ & 1\,3\,1\,6 \\
\hline
 & 1\,4\,4\,8\,3.5\,4 \\
+ & 1\,4\,4\,8\,3 \\
\hline
 & 1\,6\,0\,4\,2\,8.4 \\
+ & 1\,6\,0\,4\,2\,8 \\
\hline
 & 1\,7\,6\,4\,7\,2\,3
\end{array}
$$

Результат: $(1764723)_9$.

13.

```
BUF    ALF  .␣␣␣␣      (Разделяющая точка в первой строке.)
       ORIG *+39
START  JOV  OFLO       Сбросить признак переполнения.
       ENT2 -40        Установить указатель буфера.
8H     ENT3 10         Установить счетчик циклов.
1H     ENT1 m          Начать программу умножения.
       ENTX 0
2H     STX  CARRY
       ...             (См. упр. 4.3.1–13 с
       J1P  2B             v = 10^9 и W = U.)
       SLAX 5          rA ← следующие девять разрядов.
       CHAR
       STA  BUF+40,2(2:5)  Запомнить следующие девять разрядов.
       STX  BUF+41,2
       INC2 2          Увеличить указатель буфера.
       DEC3 1
       J3P  1B         Повторить десять раз.
       OUT  BUF+20,2(PRINTER)
       J2N  8B         Повторять до тех пор,
                           пока печатаются обе строки.  ∎
```

14. Пусть $K(n)$ — число шагов, требуемых для преобразования n-разрядного десятичного числа в представление в двоичном формате и одновременного вычисления двоичного представления числа 10^n. Тогда $K(2n) \le 2K(n) + O(M(n))$. *Доказательство.* Для данного числа $U = (u_{2n-1} \ldots u_0)_{10}$ за $2K(n)$ шагов вычисляем $U_1 = (u_{2n-1} \ldots u_n)_{10}$ и $U_0 = (u_{n-1} \ldots u_0)_{10}$, а также 10^n. Затем за $O(M(n))$ шагов вычисляем $U = 10^n U_1 + U_0$ и $10^{2n} = 10^n \cdot 10^n$ Отсюда следует, что $K(2^n) = O(M(2^n) + 2M(2^{n-1}) + 4M(2^{n-2}) + \cdots) = O(nM(2^n))$.

[Подобным образом, как показал Шёнхаге, за $O(nM(2^n))$ шагов можно преобразовать $(2^n \lg 10)$-битовое число U из двоичного представления в десятичное. Сначала за $O(M(2^{n-1}) + M(2^{n-2}) + \cdots) = O(M(2^n))$ шагов формируем число $V = 10^{2^{n-1}}$, затем еще за $O(M(2^n))$ шагов вычисляем $U_0 = (U \bmod V)$ и $U_1 = \lfloor U/V \rfloor$. После этого преобразуем U_0 и U_1.]

17. См работу У. Д. Клингера (W. D. Clinger), *SIGPLAN Notices* **25**, 6 (June, 1990), 92–101, а также работу Стила (Steele) и Уайта (White), цитируемую в ответе к упр. 3.

18. Пусть $U = \text{round}_B(u, P)$ и $v = \text{round}_b(U, p)$. Можно положить, что $u > 0$, так что $U > 0$ и $v > 0$. *Случай 1, $v < u$:* определим e и E такими, чтобы выполнялось неравенство $b^{e-1} < u \le b^e$, $B^{E-1} \le U < B^E$. Тогда $u \le U + \frac{1}{2}B^{E-P}$ и $U \le u - \frac{1}{2}b^{e-p}$; следовательно, $B^{P-1} \le B^{P-E}U < B^{P-E}u \le b^{p-e}u \le b^p$. *Случай 2, $v > u$:* определим e и E так, чтобы получить $b^{e-1} \le u < b^e$, $B^{E-1} < U \le B^E$. Тогда $u \ge U - \frac{1}{2}B^{E-P}$ и $U \ge u + \frac{1}{2}b^{e-p}$; следовательно, $B^{P-1} \le B^{P-E}(U - B^{E-P}) < B^{P-E}u \le b^{p-e}u < b^p$. Таким образом, доказано, что $B^{P-1} < b^p$ когда $v \ne u$.

Обратно, если $B^{P-1} < b^p$, то при доказательстве, проведенном выше, предполагалось, что наиболее подходящим примером, для которого $u \ne v$, будет тот, в котором число u представляется по степеням b и в то же время близко к степени B. Получаем $B^{P-1}b^p < B^{P-1}b^p + \frac{1}{2}b^p - \frac{1}{2}B^{P-1} - \frac{1}{4} = (B^{P-1} + \frac{1}{2})(b^p - \frac{1}{2})$; следовательно, $1 < \alpha = 1/(1 - \frac{1}{2}b^{-p}) < 1 + \frac{1}{2}B^{1-P} = \beta$. Согласно результатам упр. 4.5.3–50 существуют целые числа e и E, такие, что $\log_B \alpha < e \log_B b - E < \log_B \beta$. Отсюда следует, что для некоторых e и E выполняется неравенство $\alpha < b^e/B^E < \beta$. Получаем $\text{round}_B(b^e, P) = B^E$ и $\text{round}_b(B^E, p) < b^e$. [*CACM* **11** (1968), 47–50; *Proc. Amer. Math. Soc.* **19** (1968), 716–723.]

Например, если $b^p = 2^{10}$ и $B^P = 10^4$, то число $u = 2^{6408} \approx .100049 \cdot 10^{1930}$ округляется до $U = .1 \cdot 10^{1930} \approx (.11111111101111111111)_2 \cdot 2^{6408}$, которое округляется до $2^{6408} - 2^{6398}$. (*Наименьший* пример округления $\text{round}((.1111111001)_2 \cdot 2^{784}) = .1011 \cdot 10^{236}$, $\text{round}(.1011 \cdot 10^{235}) = (.11111110010)_2 \cdot 2^{784}$ найден Фредом Дж. Тайдеманом (Fred J. Tydeman).)

19. $m_1 = (\text{F0F0F0F0})_{16}$, $c_1 = 1 - 10/16$ формирует $U = \big((u_7u_6)_{10} \ldots (u_1u_0)_{10}\big)_{256}$; тогда $m_2 = (\text{FF00FF00})_{16}$, $c_2 = 1 - 10^2/16^2$ формирует $U = \big((u_7u_6u_5u_4)_{10}(u_3u_2u_1u_0)_{10}\big)_{65536}$; а $m_3 = (\text{FFFF0000})_{16}$, $c_3 = 1 - 10^4/16^4$ завершает процедуру. [Ср. с алгоритмом Шёнхаге в упр. 14. Эту процедуру предложил Рой А. Кейр (Roy A. Keir) приблизительно в 1958 году.]

РАЗДЕЛ 4.5.1

1. Учитывая, что знаменатели положительны, проверяем выполнение неравенства

$$uv' < u'v.$$

2. Если на $c > 1$ делятся как u/d, так и v/d, то на cd делятся u и v.

3. Положим, что число p простое. Если p^e является делителем чисел uv и $u'v'$ при $e \ge 1$, то либо $p^e \backslash u$ и $p^e \backslash v'$, либо $p^e \backslash u'$ и $p^e \backslash v$; следовательно, $p^e \backslash \gcd(u, v')\gcd(u', v)$. Обратное утверждение доказывается путем обращения утверждения.

4. Пусть $d_1 = \gcd(u, v)$, $d_2 = \gcd(u', v')$; тогда ответом будет $w = (u/d_1)(v'/d_2)\text{sign}(v)$, $w' = |(u'/d_2)(v/d_1)|$ с сообщением об ошибке "Деление на нуль" при $v = 0$.

5. $d_1 = 10$, $t = 17 \cdot 7 - 27 \cdot 12 = -205$, $d_2 = 5$, $w = -41$, $w' = 168$.

6. Пусть $u'' = u'/d_1$, $v'' = v'/d_1$. Задача состоит в том, чтобы показать, что $\gcd(uv'' + u''v,\, d_1) = \gcd(uv'' + u''v,\, d_1u''v'')$. Если число p простое и является делителем числа u'', то p не будет делителем числа u или v'', а потому p не делит $uv'' + u''v$. Подобное же рассуждение действительно для простых делителей числа v''. Таким образом, ни один из простых делителей числа $u''v''$ не оказывает влияния на наибольшее общее кратное.

7. $(N-1)^2 + (N-2)^2 = 2N^2 - (6N-5)$. Если исходными числами являются n-битовые двоичные целые числа, то для представления числа t может понадобиться $2n+1$ бит.

8. При умножении и делении эти величины будут подчиняться правилам $x/0 = \mathrm{sign}(x)\infty$, $(\pm\infty) \times x = x \times (\pm\infty) = (\pm\infty)/x = \pm\mathrm{sign}(x)\infty$, $x/(\pm\infty) = 0$, где x — произвольное конечное число, не равное нулю. Описанные в разделе алгоритмы остаются без изменений. Затем эти алгоритмы можно легко модифицировать так, что $0/0 = 0\times(\pm\infty) = (\pm\infty)\times 0 =$ "(0/0)", где "0/0" служит представлением "неопределенности". Если один из операндов не определен, результат также должен быть не определен.

Поскольку подпрограммы умножения и деления могут удовлетворять этим естественным правилам расширенных арифметических операций, иногда имеет смысл модифицировать также операции сложения и вычитания, чтобы они удовлетворяли правилам $x \pm \infty = \pm\infty$, $x \pm (-\infty) = \mp\infty$ для конечных x; $(\pm\infty) + (\pm\infty) = \pm\infty - (\mp\infty) = \pm\infty$; более того, $(\pm\infty) + (\mp\infty) = (\pm\infty) - (\pm\infty) = (0/0)$; если одним или обоими операндами служит $(0/0)$, результат также должен быть $(0/0)$. Аналогично можно модифицировать операции проверки равенства и сравнения.

Сделанные выше замечания не относятся к признакам "переполнения". Если для признака переполнения используется ∞, то нельзя полагать $1/\infty$ равным нулю, чтобы не принять неверные результаты за правильные. Значительно лучше представить признак переполнения через $(0/0)$ и условиться, что результат любой операции будет неопределенным, если хотя бы один из вводимых операндов является неопределенным. Такой способ индикации переполнения имеет то преимущество, что данные на выходе расширенных операций точно указывают, какой из результатов определен, а какой — нет.

9. Если $u/u' \neq v/v'$, то $1 \le |uv' - u'v| = u'v'|u/u' - v/v'| < |2^{2n}u/u' - 2^{2n}v/v'|$. Две величины, различающиеся более чем на единицу, не могут иметь одинаковую "грань снизу". (Другими словами, $2n$ первых битов справа от двоичной точки достаточно для того, чтобы охарактеризовать значение двоичной дроби при наличии n-битовых знаменателей. Нельзя повысить ее точность до $2n-1$ бит, так как при $n = 4$ получим $\frac{1}{13} = (.00010011\ldots)_2$, $\frac{1}{14} = (.00010010\ldots)_2$.)

11. Чтобы разделить ненулевые числа v и v' на $(v + v'\sqrt{5}\,)/v''$, нужно умножить их на обратное значение $(v - v'\sqrt{5}\,)v''/(v^2 - 5v'^2)$ и выполнить возможные сокращения.

12. $((2^{q-1}-1)/1)$; $\mathrm{round}(x) = (0/1)$ тогда и только тогда, когда $|x| \le 2^{1-q}$. Аналогично $\mathrm{round}(x) = (1/0)$ тогда и только тогда, когда $x \ge 2^{q-1}$.

13. Один подход заключается в ограничении размеров числителя и знаменателя величиной 27 бит. Тогда нужно будет сохранять только 26 бит (так как ведущий бит знаменателя равен 1, за исключением случая, когда длина знаменателя равна 0). При таком подходе остается место для знака и 5 бит для запоминания размера знаменателя. Другой подход заключается в использовании 28 бит для числителя и знаменателя, которые занимают не более семи шестнадцатеричных разрядов вместе со знаком и 3-битовое поле для запоминания количества шестнадцатеричных разрядов в знаменателе.

[С учетом формул, приведенных в следующем упражнении, первый подход дает точно 2 140 040 119 конечных представимых чисел, а второй — 1 830 986 459. Первый подход предпочтительнее, так как он более простой и обеспечивает гладкий переход

между интервалами. При оперировании 64-битовыми словами общая длина числителя и знаменателя ограничивается величиной $64 - 6 = 58$ бит.]

14. Количество чисел, кратных n в интервале $(a\,..\,b]$, равно $\lfloor b/n \rfloor - \lfloor a/n \rfloor$. Отсюда, используя метод включений и исключений, получаем решение задачи в виде $S_0 - S_1 + S_2 - \cdots$, где S_k равно $\sum(\lfloor M_2/P \rfloor - \lfloor M_1/P \rfloor)(\lfloor N_2/P \rfloor - \lfloor N_1/P \rfloor)$, просуммированному по всем произведениям P различных простых чисел k. Можно также выразить ответ в виде

$$\sum_{n=1}^{\min(M_2,N_2)} \mu(n)\,(\lfloor M_2/n \rfloor - \lfloor M_1/n \rfloor)\,(\lfloor N_2/n \rfloor - \lfloor N_1/n \rfloor).$$

РАЗДЕЛ 4.5.2

1. Везде замените min, max, + на gcd, lcm, × соответственно (предварительно убедитесь, что все тождества справедливы и тогда, когда любая переменная равна нулю).

2. Для простого числа p пусть u_p, v_{1p}, ..., v_{np} — степени числа p в каноническом разложении чисел u, v_1, ..., v_n. По предположению $u_p \le v_{1p} + \cdots + v_{np}$. Необходимо показать, что $u_p \le \min(u_p, v_{1p}) + \cdots + \min(u_p, v_{np})$, но это неравенство действительно выполняется, если u_p не меньше каждого v_{jp} или u_p меньше некоторого v_{jp}.

3. *Решение 1.* Если $n = p_1^{e_1} \ldots p_r^{e_r}$, это число в каждом случае равно $(2e_1 + 1) \ldots (2e_r + 1)$. *Решение 2.* Если положить $u = \gcd(d, n)$ и $v = n^2/\operatorname{lcm}(d, n)$ для каждого делителя d числа n^2, то можно получить взаимно однозначное соответствие. [E. Cesàro, *Annali di Matematica Pura ed Applicata* (2) **13** (1885), 235–250, §12.]

4. См. упр. 3.2.1.2–15(a).

5. Будем выполнять сдвиг вправо чисел u и v до тех пор, пока не получим, что ни одно из них не кратно 3. Запоминаем соответствующее значение степени 3, которое появится в наибольшем общем делителе. Затем на каждой итерации присваиваем $t \leftarrow u + v$ или $t \leftarrow u - v$ (в зависимости от того, какая из этих величин кратна 3), сдвигаем t вправо до тех пор, пока оно не перестанет быть кратным 3, после чего заменяем $\max(u, v)$ полученным результатом.

u	v	t
13634	24140	10506, 3502
13634	3502	17136, 5712, 1904
1904	3502	5406, 1802
1904	1802	102, 34
34	1802	1836, 612, 204, 68
34	68	102, 34
34	34	0

Свидетельство того, что $\gcd(40902, 24140) = 34$, поразительно!

6. Вероятность того, что оба числа u и v четны, равна $\frac{1}{4}$; вероятность того, что оба числа кратны четырем, равна $\frac{1}{16}$, и т. д. Таким образом, величина A имеет распределение, задаваемое производящей функцией

$$\frac{3}{4} + \frac{3}{16}z + \frac{3}{64}z^2 + \cdots = \frac{3/4}{1 - z/4}.$$

Среднее значение равно $\frac{1}{3}$, а среднеквадратичное отклонение — $\sqrt{\frac{2}{9} + \frac{1}{3} - \frac{1}{9}} = \frac{2}{3}$. Если u и v независимо и равномерно распределены в интервале $1 \le u, v < 2^N$, необходимо внести

в расчет небольшие поправки, тогда среднее в действительности будет равно

$$(2^N-1)^{-2}\sum_{k=1}^{N}(2^{N-k}-1)^2=\frac{1}{3}-\frac{4}{3}(2^N-1)^{-1}+N(2^N-1)^{-2}.$$

7. Если числа u и v не являются оба четными, то равновероятен каждый из случаев (четное, нечетное), (нечетное, четное), (нечетное, нечетное) и при этом имеем соответственно $B=1,\ 0,\ 0$. Следовательно, в среднем $B=\frac{1}{3}$. Если уж быть совсем точным, то, когда $1\le u,v<2^N$, необходимо внести небольшую поправку: вероятность того, что $B=1$, в действительности будет равна

$$(2^N-1)^{-2}\sum_{k=1}^{N}(2^{N-k}-1)2^{N-k}=\frac{1}{3}-\frac{1}{3}(2^N-1)^{-1},$$

как следует из упр. 6.

8. Обозначим через F число шагов вычитания, в которых $u>v$. Тогда $E=F+B$. Если заменить исходные данные (u,v) на (v,u), то значение C не изменится, а число F станет равным $C-1-F$. Следовательно, $E_{\rm ave}=\frac{1}{2}(C_{\rm ave}-1)+B_{\rm ave}$.

9. В первый раз бинарный алгоритм попадает на шаг B6 при значениях $u=1963$, $v=1359$; тогда $t\leftarrow 604,\ 302,\ 151$ и т. д. Наибольший общий делитель равен 302. Применяя алгоритм X, находим, что $2\cdot 31408-23\cdot 2718=302$.

10. (a) Два целых числа взаимно просты тогда и только тогда, когда оба они не делятся одновременно ни на одно простое число. (b) Перегруппируем сумму в (a), принимая знаменатели равными $k=p_1\ldots p_r$. (Каждая из сумм в (a) и (b) на самом деле конечна.) (c) Так как $(n/k)^2-\lfloor n/k\rfloor^2=O(n/k)$, то $q_n-\sum_{k=1}^{n}\mu(k)(n/k)^2=\sum_{k=1}^{n}O(n/k)=O(nH_n)$. Далее, $\sum_{k>n}(n/k)^2=O(n)$. (d) $\sum_{d\backslash n}\mu(d)=\delta_{1n}$. [Фактически имеет место более общий, чем в (b), результат

$$\sum_{d\backslash n}\mu(d)\left(\frac{n}{d}\right)^s=n^s-\sum\left(\frac{n}{p}\right)^s+\sum\left(\frac{n}{pq}\right)^s-\cdots,$$

где суммирование в правой части берется по простым делителям n, и эта сумма равна $n^s(1-1/p_1^s)\ldots(1-1/p_r^s)$, если $n=p_1^{e_1}\ldots p_r^{e_r}$.]

Замечание. По аналогии найдем множество целых чисел k, которое с вероятностью $1/\zeta(k)=1/(\sum_{n\ge 1}1/n^k)$ является множеством, состоящим из простых чисел. Это доказательство теоремы D предложено Ф. Мертенсом (F. Mertens, *Crelle* **77** (1874), 289–291). Такая методика дает гораздо более строгий результат, а именно: для произвольных f и g $6\pi^{-2}mn+O(n\log m)$ пары целых чисел $u\in[f(m)\,..\,f(m)+m)$, $v\in[g(n)\,..\,g(n)+n)$ являются взаимно простыми при $m\le n$.

11. (a) Искомая вероятность равна $6/\pi^2$, умноженному на $1+\frac{1}{4}+\frac{1}{9}$, т. е. $49/(6\pi^2)\approx .82746$. (b) Среднее значение равно $6/\pi^2$, умноженному на $1/1+2/4+3/9+\cdots$, т. е. ∞. (Это верно, несмотря на результаты упр. 12 и 14.)

12. [*Annali di Mat.* (2) **13** (1885), 235–250, §3.] Пусть $\sigma(n)$ — количество положительных делителей числа n. Тогда ответ будет таким:

$$\sum_{k\ge 1}\sigma(k)\cdot\frac{6}{\pi^2k^2}=\frac{6}{\pi^2}\left(\sum_{k\ge 1}\frac{1}{k^2}\right)^2=\frac{\pi^2}{6}.$$

[Следовательно, среднее значение *меньше* 2, хотя в случае, когда u и v не являются взаимно простыми, они имеют по меньшей мере два общих делителя.]

13. $1+\frac{1}{9}+\frac{1}{25}+\cdots=1+\frac{1}{4}+\frac{1}{9}+\cdots-\frac{1}{4}(1+\frac{1}{4}+\frac{1}{9}+\cdots)$.

14. (a) $L = (6/\pi^2)\sum_{d\ge 1} d^{-2}\ln d = -\zeta'(2)/\zeta(2) = \sum_{p\text{ простое}}(\ln p)/(2^p-1) \approx 0.56996$.
(b) $(8/\pi^2)\sum_{d\ge 1}[d \text{ нечетное}]\, d^{-2}\ln d = L - \frac{1}{3}\ln 2 \approx 0.33891$.

15. $v_1 = \pm v/u_3$, $v_2 = \mp u/u_3$ (знак зависит от того, четно или нечетно число итераций). Это следует из того факта, что числа v_1 и v_2 взаимно просты (на протяжении всего процесса выполнения алгоритма) и $v_1 u = -v_2 v$. [Следовательно, в момент завершения выполнения алгоритма $v_1 u = \operatorname{lcm}(u,v)$, но такой метод — не лучший путь вычисления наименьшего общего кратного. Обобщение этого метода рассмотрено в упр. 4.6.1–18.]

Более подробно с данным вопросом можно ознакомиться, рассмотрев упр. 4.5.3–48.

16. В результате применения к числам v и m алгоритма X вычисляем такие значения x, при которых $xv \equiv 1$ (по модулю m). (Это можно сделать путем упрощения алгоритма X за счет отказа от вычисления u_2, v_2 и t_2, поскольку данные величины в ответе не присутствуют.) Затем присвоим $w \leftarrow ux \bmod m$. [Отсюда следует, как в упр. 4.5.3–45, что для реализации этого процесса при его применении к большим n-битовым числам необходимо затратить $O(n^2)$ единиц времени. В упр. 17 и 39 рассмотрены алгоритмы, альтернативные алгоритму X.]

17. По аналогии с методом Ньютона можно положить, что $u' = (2u - vu^2) \bmod 2^{2e}$ (см. окончание раздела 4.3.1). Точно так же при $uv \equiv 1 + 2^e w$ (по модулю 2^{2e}) полагаем $u' = 1 + 2^e((-uw) \bmod 2^e)$.

18. Пусть в дополнение к числам u и v числа u_1, u_2, u_3, v_1, v_2, v_3 — переменные с многократной точностью. Расширенный алгоритм будет выполнять над числами u_3 и v_3 те же операции, что и алгоритм L над числами u и v. Новыми операциями многократной точности будут: присвоение на шаге L4 $t \leftarrow Au_j$, $t \leftarrow t + Bv_j$, $w \leftarrow Cu_j$, $w \leftarrow w + Dv_j$, $u_j \leftarrow t$, $v_j \leftarrow w$ для всех j. Кроме того, если на этом шаге $B = 0$, выполняем присвоение $t \leftarrow u_j - qv_j$, $u_j \leftarrow v_j$, $v_j \leftarrow t$ для всех j и для $q = \lfloor u_3/v_3 \rfloor$. При малых значениях v_3 подобным образом модифицируется и шаг L1. Внутренний цикл (шаги L2 и L3) остается неизменным.

19. (a) Пусть $t_1 = x + 2y + 3z$. Тогда $3t_1 + y + 2z = 1$, $5t_1 - 3y - 20z = 3$. Исключим y, и тогда $14t_1 - 14z = 6$. Решений нет. (b) На этот раз $14t_1 - 14z = 0$. Выполняем деление на 14 и исключаем t_1; общее решение имеет вид $x = 8z - 2$, $y = 1 - 5z$ (z выбирается произвольно).

20. Предположим, что $m \ge n$. При $m > n = 0$ получим $(m-t, 0)$ с вероятностью 2^{-t}, а при $1 \le t < m$ получим $(0,0)$ с вероятностью 2^{1-m}. *Valida vi** при $n > 0$ можно получить следующие значения.

Случай 1, $m = n$. Из (n,n) при $2 \le t < n$ переходим в $(n-t, n)$ с вероятностью $t/2^t - 5/2^{t+1} + 3/2^{2t}$. (Этими значениями будут $\frac{1}{16}$, $\frac{7}{64}$, $\frac{27}{256}$, ...) Вероятность попадания в интервал $(0,n)$ равна $n/2^{n-1} - 1/2^{n-2} + 1/2^{2n-2}$. Вероятность попадания в интервал (n,k) такая же, как и вероятность попадания в интервал (k,n). Вероятность прекращения выполнения алгоритма равна $1/2^{n-1}$.

Случай 2, $m = n + 1$. Из интервала $(n+1, n)$ в интервал (n,n) можно попасть при $n > 1$ с вероятностью $\frac{1}{8}$ или при $n = 1$ с вероятностью 0. Вероятность попадания в интервал $(n-t,\ n)$ равна $11/2^{t+3} - 3/2^{2t+1}$ для $1 \le t < n-1$. (Этими значениями будут $\frac{5}{16}$, $\frac{1}{4}$, $\frac{19}{128}$, ...) Вероятность попадания в интервал $(1,n)$ при $n > 1$ равна $5/2^{n+1} - 3/2^{2n-1}$; вероятность попадания в интервал $(0,n)$ равна $3/2^n - 1/2^{2n-1}$.

---

* Здесь: после соответствующих усилий (*лат.*). — *Прим. перев.*

Случай 3, $m \geq n+2$. Вероятности, полученные для этого случая, приведены в следующей таблице.

$$\begin{aligned}
&(m-1,\,n): && 1/2 - 3/2^{m-n+2} - \delta_{n1}/2^{m+1};\\
&(m-t,\,n): && 1/2^t + 3/2^{m-n+t+1}, \qquad 1<t<n;\\
&(m-n,\,n): && 1/2^n + 1/2^m, \qquad n>1;\\
&(m-n-t,\,n): && 1/2^{n+t} + \delta_{t1}/2^{m-1}, \qquad 1\leq t<m-n;\\
&(0,\,n): && 1/2^{m-1}.
\end{aligned}$$

Единственная особенность этих результатов, которая обращает на себя внимание, заключается в том, что они чрезвычайно неупорядочены. Именно это обстоятельство делает их неинтересными.

21. Покажите, что при фиксированном v и при $2^m < u < 2^{m+1}$ для больших m каждый цикл "вычитание и сдвиг" рассматриваемого алгоритма уменьшает $\lfloor \lg u \rfloor$ в среднем на два.

22. После выполнения операции сдвига вправо числа u, лежащего в интервале $1 \leq u < 2^N$, до тех пор, пока оно не станет нечетным, ровно для $(N-m)2^{m-1+\delta_{m0}}$ целых чисел выполняется равенство $\lfloor \lg u \rfloor = m$. Следовательно,

$$\begin{aligned}
(2^N-1)^2 C = N^2 C_{00} &+ 2N \sum_{1\leq n\leq N} (N-n)2^{n-1} C_{n0}\\
&+ 2 \sum_{1\leq n<m\leq N} (N-m)(N-n)2^{m+n-2} C_{mn} + \sum_{1\leq n\leq N} (N-n)^2 2^{2n-2} C_{nn}.
\end{aligned}$$

(Та же формула справедлива для D в обозначениях D_{mn}.)

Средняя сумма равна $2^{2N-2} \sum_{0\leq m<n<N} mn2^{-m-n}((\alpha+\beta)N+\gamma-\alpha m-\beta n)$. Поскольку

$$\sum_{0\leq m<n} m2^{-m} = 2-(n+1)2^{1-n} \quad \text{и} \quad \sum_{0\leq m<n} m(m-1)2^{-m} = 4-(n^2+n+2)2^{1-n},$$

сумма по m равна

$$\begin{aligned}
2^{2N-2} \sum_{0\leq n<N} n2^{-n} &\Big((\gamma-\alpha-\beta n+(\alpha+\beta)N)(2-(n+1)2^{1-n}) - \alpha\big(4-(n^2+n+2)2^{1-n}\big)\Big)\\
&= 2^{2N-2}\Big((\alpha+\beta)N \sum_{n\geq 0} n2^{-n}(2-(n+1)2^{1-n}) + O(1)\Big).
\end{aligned}$$

Таким образом, в ответе коэффициент при $(\alpha+\beta)N$ равен $2^{-2}(4-(\frac{4}{3})^3) = \frac{11}{27}$.

Замечание. *Точное* значение сумм может быть получено после ряда скучных вычислений на основе общей формулы суммирования по частям:

$$\sum_{0\leq k<n} k^{\underline{m}}\, z^k = \frac{m!\, z^m}{(1-z)^{m+1}} - \sum_{k=0}^{m} \frac{m^{\underline{k}}\, n^{\underline{m-k}}\, z^{n+k}}{(1-z)^{k+1}}.$$

23. При $x \leq 1$ выполняется соотношение $\Pr(u \geq v \text{ и } v/u \leq x) = \frac{1}{2}(1-G_n(x))$. Если же $x \geq 1$, то $\frac{1}{2} + \Pr(u \leq v \text{ и } v/u \geq 1/x) = \frac{1}{2} + \frac{1}{2}G_n(1/x)$; в соответствии с (40) это также равно $\frac{1}{2}(1-G_n(x))$.

24. $\sum_{k\geq 1} 2^{-k} G(1/(2^k+1)) = S(1)$. Это значение, не имеющее отношения к классической константе, приближенно равно 0.5432582959.

25. Как заметил Ричард Брент (Richard Brent), функция $G(e^{-y})$ — нечетная аналитическая функция для всех вещественных значений y. Если положить $G(e^{-y}) = \lambda_1 y + \lambda_3 y^3 + \lambda_5 y^5 + \cdots = \rho(e^{-y} - 1)$, то получим $-\rho_1 = \lambda_1 = \lambda$, $\rho_2 = \frac{1}{2}\lambda$, $-\rho_3 = \frac{1}{3}\lambda + \lambda_3$, $\rho_4 = \frac{1}{4}\lambda + \frac{3}{2}\lambda_3$, $-\rho_5 = \frac{1}{5}\lambda + \frac{7}{4}\lambda_3 + \lambda_5$;

$$(-1)^n \rho_n = \sum_k \left[{n \atop k}\right] \frac{k!}{n!} \lambda_k; \qquad \lambda_n = -\sum_k \left\{{n \atop k}\right\} \frac{k!}{n!} \rho_k.$$

Приведем несколько первых значений:

$$\lambda_1 \approx .3979226812, \quad \lambda_3 \approx -.0210096400, \quad \lambda_5 \approx .0013749841, \quad \lambda_7 \approx -.0000960351.$$

Фантастическое предположение: $\lim_{k\to\infty}(-\lambda_{2k+1}/\lambda_{2k-1}) = 1/\pi^2$.

26. Согласно (39) левая часть равна $2S(1/x) - 5S(1/2x) + 2S(1/4x) - 2S(x) + 5S(2x) - 2S(4x)$; согласно (44) правая часть равна $S(2x) - 2S(4x) + 2S(1/x) - S(1/2x) - 2S(x) + 4S(2x) - 4S(1/2x) + 2S(1/4x)$. Самыми интересными являются, вероятно, случаи, когда $x = 1$, $x = 1/\sqrt{2}$ и $x = \phi$. Например, при $x = \phi$ получаем $2G(4\phi) - 5G(2\phi) + G(\phi^2/2) - G(\phi^3) = 2G(2\phi^2)$.

27. При $n > 1$ в соответствии с упр. 1.2.11.2–4 имеем

$$2\psi_n = [z^n]\, z \sum_{k\ge 0} 2^{-2k} \sum_{j=0}^{2^k-1} \sum_{l\ge 0} (jz/2^k)^l = \sum_{k\ge 1} 2^{-k(n+1)} \sum_{j=0}^{2^k-1} j^{n-1}$$

$$= \sum_{k\ge 1} 2^{-k(n+1)} \sum_{l=0}^{n-1} \binom{n}{l} B_l 2^{k(n-l)}/n$$

и, конечно, $\sum_{k\ge 1} 2^{-k(l+1)} = 1/(2^{l+1} - 1)$.

28. Следуя обозначениям упр. 6.3–34(b), положим, что $S_n(m) = \sum_{k=1}^{m-1} (1 - k/m)^n$ и $T_n(m) = 1/(e^{n/m} - 1)$. Получаем $S_n(m) = T_n(m) + O(e^{-n/m} n/m^2)$ и $2\psi_{n+1} = \sum_{j\ge 1} 2^{-2j} S_n(2^j) = \tau_n + O(n^{-3})$, где $\tau_n = \sum_{j\ge 1} 2^{-2j} T_n(2^j)$. Из того, что $\tau_{n+1} < \tau_n$, а $4\tau_{2n} - \tau_n = 1/(e^n - 1)$ положительное, но малое число, следует $\tau_n = \Theta(n^{-2})$. Более подробную информацию можно получить, если записать

$$\sum_{j\ge 1} \frac{1}{2^{2j}} \frac{1}{e^{n/2^j} - 1} = \frac{1}{2\pi i} \sum_{j\ge 1} \int_{3/2-i\infty}^{3/2+i\infty} \frac{\zeta(z)\Gamma(z) n^{-z}}{2^{j(2-z)}}\, dz = \frac{1}{2\pi i} \int_{3/2-i\infty}^{3/2+i\infty} \frac{\zeta(z)\Gamma(z) n^{-z}}{2^{2-z} - 1}\, dz.$$

Интеграл равен сумме вычетов $2 + 2\pi i k/\ln 2$, т. е. произведению n^{-2} и $\pi^2/(6\ln 2) + f(n)$, где

$$f(n) = 2 \sum_{k\ge 1} \Re(\zeta(2 + 2\pi i k/\ln 2)\Gamma(2 + 2\pi i k/\ln 2) \exp(-2\pi i k \lg n)/\ln 2)$$

есть периодическая функция $\lg n$ со "средним" значением, равным нулю.

29. (Решение П. Флажоле (P. Flajolet) и Б. Валле (B. Vallée).) Если при соответствующих условиях $f(x) = \sum_{k\ge 1} 2^{-k} g(2^k x)$ и $g^*(s) = \int_0^\infty g(x) x^{s-1} dx$, то $f^*(s) = \sum_{k\ge 1} 2^{-k(s+1)} g^*(s) = g^*(s)/(2^{s+1} - 1)$ и $f(x) = \frac{1}{2\pi i} \int_{c-i\infty}^{c+i\infty} f^*(s) x^{-s} ds$. В этом случае, полагая, что $g(x) = 1/(1+x)$, найдем преобразование в виде $g^*(s) = \pi/\sin \pi s$ при $0 < \Re s < 1$. Таким образом,

$$f(x) = \sum_{k=1}^{\infty} \frac{1}{2^k} \frac{1}{1 + 2^k x} = \frac{1}{2\pi i} \int_{1/2-i\infty}^{1/2+i\infty} \frac{\pi x^{-s}\, ds}{(2^{s+1} - 1)\sin \pi s}.$$

Отсюда следует, что $f(x)$ — сумма вычетов $\frac{\pi}{\sin \pi s} x^{-s}/(2^{s+1}-1)$ при $\Re s \le 0$, т. е. $1 + x \lg x + \frac{1}{2}x + xP(\lg x) - \frac{2}{1}x^2 + \frac{4}{3}x^3 - \frac{8}{7}x^4 + \cdots$, где функция

$$P(t) = \frac{2\pi}{\ln 2} \sum_{m=1}^{\infty} \frac{\sin 2\pi m t}{\sinh(2m\pi^2/\ln 2)}$$

есть периодическая функция, абсолютные значения которой никогда не превышают 8×10^{-12}. (Ввиду малости функции $P(t)$ Брент в своей первой работе не обратил на нее внимания.)

Преобразование Меллина функции $f(1/x)$ имеет вид $f^*(-s) = \pi/((1-2^{1-s})\sin \pi s)$ при $-1 < \Re s < 0$, поэтому $f(1/x) = \frac{1}{2\pi i} \int_{-1/2-i\infty}^{-1/2+i\infty} \frac{\pi}{\sin \pi s} x^{-s} ds/(1-2^{1-s})$. Найдем теперь при $\Re s \le -1$ вычеты подынтегральной функции $f(1/x) = \frac{1}{3}x - \frac{1}{7}x^2 + \cdots$. [Эта формула может быть получена и непосредственно.] Имеем $S_1(x) = 1 - f(x)$, и отсюда следует

$$G_1(x) = f(x) - f(1/x) = x \lg x + \frac{1}{2}x + xP(\lg x) - \frac{x^2}{1+x} + (1-x^2)\phi(x),$$

где $\phi(x) = \sum_{k=0}^{\infty} (-1)^k x^k/(2^{k+1}-1)$.

30. Имеем $G_2(x) = \Sigma_1(x) - \Sigma_1(1/x) + \Sigma_2(x) - \Sigma_2(1/x)$, где

$$\Sigma_1(x) = \sum_{k,l \ge 1} \frac{1}{2^{k+l}} \frac{1}{1 + 2^l(1+2^k x)}, \qquad \Sigma_2(x) = \sum_{k,l \ge 1} \frac{1}{2^k} \frac{1}{1 + 2^l + 2^k x}.$$

Преобразования Меллина будут иметь вид

$$\Sigma_1^*(s) = \frac{\pi}{\sin \pi s} a(s)/(2^{s+1}-1), \quad \Sigma_2^*(s) = \frac{\pi}{\sin \pi s} b(s)/(2^{s+1}-1),$$

где

$$a(s) = \sum_{l \ge 1} \frac{(1+2^{-l})^{s-1}}{2^{2l}} = \sum_{k \ge 0} \binom{s-1}{k} \frac{1}{2^{k+2}-1},$$

$$b(s) = \sum_{l \ge 1} (2^l + 1)^{s-1} = \sum_{k \ge 0} \binom{s-1}{k} \frac{1}{2^{k+1-s}-1}.$$

Таким образом, при $0 \le x \le 1$ получаем следующие выражения:

$$\Sigma_1(x) = a(0) + a(-1)x(\lg x + \tfrac{1}{2}) - a'(1)x/\ln 2 + xA(\lg x) - \sum_{k \ge 2} \frac{2^{k-1}}{2^{k-1}-1} a(-k)(-x)^k,$$

$$\Sigma_2(x) = b(0) + b(-1)x(\lg x + \tfrac{1}{2}) - b'(1)x/\ln 2 + xB(\lg x) - \sum_{k \ge 2} \frac{2^{k-1}}{2^{k-1}-1} b(-k)(-x)^k,$$

$$\Sigma_1(1/x) = \sum_{k \ge 1} \frac{-a(k)(-x)^k}{2^{k+1}-1},$$

$$\Sigma_2(1/x) = \sum_{k \ge 1} \frac{(-x)^k}{2^{k+1}-1} \left(\lg x - \hat{b}(k) - \frac{1}{2} - \frac{1}{2^{k+1}-1} + \frac{H_{k-1}}{\ln 2} + P_k(\lg x) \right),$$

$$\hat{b}(s) = \sum_{k=0}^{s-2} \binom{s-1}{k} \frac{1}{2^{k+1-s}-1},$$

$$A(t) = \frac{1}{\ln 2} \sum_{m\ge 1} \Re\left(\frac{2\pi i}{\sinh(2m\pi^2/\ln 2)}\, a(-1+2m\pi i/\ln 2)\, e^{-2m\pi i t} \right),$$

$$B(t) = \frac{1}{\ln 2} \sum_{m\ge 1} \Re\left(\frac{2\pi i}{\sinh(2m\pi^2/\ln 2)}\, b(-1+2m\pi i/\ln 2)\, e^{-2m\pi i t} \right),$$

$$P_k(t) = \frac{1}{\ln 2} \sum_{m\ge 1} \Re\left(\frac{2\pi i}{\sinh(2m\pi^2/\ln 2)} \binom{k-1-2m\pi i/\ln 2}{k-1} e^{-2m\pi i t} \right).$$

34. Бригитта Валле (Brigitte Vallée) с помощью приближения, существенно отличающегося от приближения, использованного Брентом, предложила изящный и строгий анализ алгоритма B, который опубликован в журнале *Algorithmica* **22** (1998), 660–685. Действительно, ее методы доказательства существенно отличаются от известных до сих пор методов предсказания поведения алгоритма B наподобие эвристических моделей Брента. Таким образом, впервые была строго решена задача анализа бинарного алгоритма нахождения наибольшего общего делителя, которая до сих пор доставляла математикам массу хлопот.

35. По индукции при $m \ge n$ длина равна $m + \lfloor n/2 \rfloor + 1 - [m = n = 1]$. Однако из результата упр. 37 следует, что алгоритм не может выполняться столь медленно.

36. Пусть $a_n = (2^n - (-1)^n)/3$; тогда $a_0, a_1, a_2, \ldots = 0, 1, 1, 3, 5, 11, 21, \ldots$. (В двоичном представлении этой последовательности чисел содержится интересный набор нулей и единиц. Обратите внимание на то, что $a_n = a_{n-1} + 2a_{n-2}$ и $a_n + a_{n+1} = 2^n$.) Положим, что $u = 2^{m+1} - a_{n+2}$, $v = a_{n+2}$ при $m > n$. При $m = n > 0$ положим $u = \max(a_{n+2}, 2a_{n+1})$ и $v = a_{n+3} - u$. Еще один пример для случая, когда $m = n > 0$, имеет вид $u = 2^{n+1} - 2$, $v = 2^{n+1} - 1$. При таком выборе требуется выполнить больше сдвигов, что дает $B = 1$, $C = n + 1$, $D = 2n$, $E = n$. Это наихудший с точки зрения времени выполнения случай для алгоритма B.

37. (Решение предложено Дж. О. Шэллитом (J. O. Shallit).) Это одна из тех задач, в которых для того, чтобы доказать то, что требуется, необходимо доказать *больше* того, что требуется. Пусть $S(u, v)$ — число шагов, на которых осуществляется операция вычитания, выполненных алгоритмом B над входными величинами u и v. Докажем, что $S(u, v) \le \lg(u + v)$. Отсюда следует, что $S(u, v) \le \lfloor \lg(u + v) \rfloor \le \lfloor \lg 2 \max(u, v) \rfloor = 1 + \lfloor \lg \max(u, v) \rfloor$ по построению.

Заметим, что $S(u, v) = S(v, u)$. Если u четно, то $S(u, v) = S(u/2, v)$. Следовательно, можно положить, что u и v нечетны. Можно также положить, что $u > v$, поскольку $S(u, u) = 1$. Тогда по индукции $S(u, v) = 1 + S((u - v)/2, v) \le 1 + \lg((u - v)/2 + v) = \lg(u + v)$.

А это означает, что наименьшей парой чисел, требующей n шагов вычитаний, будет $u = 2^{n-1} + 1$, $v = 2^{n-1} - 1$.

38. При формировании матрицы A (размера 2×2) целых чисел наподобие $A\binom{u}{v} = \binom{u'w}{v'w}$, таких, что w — размер машинного слова, следим за наиболее значимыми и наименее значимыми словами операндов (наиболее значимые используются для обозначения знака t, а наименее значимые — для определения общего числа сдвигов вправо), где u' и v' меньше u и v. (Вместо того чтобы разделить моделируемые нечетные операнды на 2, умножаем четные операнды на 2 до тех пор, пока не вычислим число, кратное w, в результате выполнения точно $\lg w$ сдвигов.) Проведенные эксперименты показали, что хотя бы на одном компьютере этот алгоритм выполняется в четыре раза быстрее алгоритма L. При использовании подобного алгоритма в упр. 40 отпадает необходимость в наиболее значимых словах.

Алгоритмы, возможно, более быстрые описаны в работах J. Sorenson, *J. Algorithms* **16** (1994), 110–144; Shallit and Sorenson, *Lecture Notes in Comp. Sci.* **877** (1994), 169–183.

39. (Решение предложено Майклом Пенком (Michael Penk).)

Y1. [Найти степень 2.] То же, что на шаге B1.

Y2. [Начальная установка.] Присвоить $(u_1, u_2, u_3) \leftarrow (1, 0, u)$ и $(v_1, v_2, v_3) \leftarrow (v, 1-u, v)$. Если число u нечетно, присвоить $(t_1, t_2, t_3) \leftarrow (0, -1, -v)$ и перейти к шагу Y4. В противном случае присвоить $(t_1, t_2, t_3) \leftarrow (1, 0, u)$.

Y3. [Выполнить половинное деление t_3.] Если t_1 и t_2 четны, присвоить $(t_1, t_2, t_3) \leftarrow (t_1, t_2, t_3)/2$; иначе — присвоить $(t_1, t_2, t_3) \leftarrow (t_1 + v, t_2 - u, t_3)/2$. (В последнем случае $t_1 + v$ и $t_2 - u$ будут оба четными.)

Y4. [t_3 четно?] Если t_3 четно, вернуться к шагу Y3.

Y5. [Переустановить $\max(u_3, v_3)$.] Если t_3 положительно, присвоить $(u_1, u_2, u_3) \leftarrow (t_1, t_2, t_3)$; иначе — присвоить $(v_1, v_2, v_3) \leftarrow (v - t_1, -u - t_2, -t_3)$.

Y6. [Вычесть.] Присвоить $(t_1, t_2, t_3) \leftarrow (u_1, u_2, u_3) - (v_1, v_2, v_3)$. Затем, если $t_1 \le 0$, присвоить $(t_1, t_2) \leftarrow (t_1 + v, t_2 - u)$. Если $t_3 \neq 0$, вернуться к шагу Y3. В противном случае закончить выполнение алгоритма с результатом, равным $(u_1, u_2, u_3 \cdot 2^k)$. ▮

Ясно, что взаимосвязи в (16) сохраняются; кроме того, после каждого из шагов Y2–Y6 $0 \le u_1, v_1, t_1 \le v$, $0 \ge u_2, v_2, t_2 \ge -u$, $0 < u_3 \le u$, $0 < v_3 \le v$. Если после шага Y2 число u четно, то можно упростить шаг Y3, так как оба числа t_1 и t_2 четны тогда и только тогда, когда четно t_2. Точно так, если v нечетно, оба числа t_1 и t_2 четны тогда и только тогда, когда четно t_1. Значит, как и в алгоритме X, вычисления, связанные с получением чисел u_2, v_2 и t_2, выполняемые для нечетных чисел v после шага Y2, можно пропустить. Это условие зачастую известно наперед (т. е. если v четно, а вычисляется u^{-1} по модулю v).

См. также A. W. Bojanczyk, R. P. Brent, *Computers and Math.* **14** (1987), 233.

40. Пусть $m = \lg \max(|u|, |v|)$. По индукции можно показать, что после выполнения s раз на шаге K3 операции $c \leftarrow c+1$ получим $|u| \le 2^{m-(s-c)/2}$, $|v| \le 2^{m-(s+c)/2}$. Поэтому $s \le 2m$. Если шаг K2 выполняется t раз, получим $t \le s + 2$, так как s увеличивается каждый раз, но не на первом и последнем шагах. [См. *VLSI '83* (North-Holland, 1983), 145–154.]

Примечание. Если $u = 1$, $v = 3 \cdot 2^k - 1$ и $k \ge 2$, получаем $m = k + 2$, $s = 2k$, $t = k + 4$. При $u = u_j$ и $v = 2u_{j-1}$ в последовательности $u_0 = 3$, $u_1 = 1$, $u_{j+1} = \min(|3u_j - 16u_{j-1}|, |5u_j - 16u_{j-1}|)$ получаем $s = 2j + 2$, $t = 2j + 3$ и (эмпирически) $m \approx \phi j$. Может ли t в пределе быть больше, чем $2m/\phi$?

41. Учитывая, что $(a^u - 1) \bmod (a^v - 1) = a^{u \bmod v} 1$ (см. соотношение 4.3.2–(20)), для всех положительных целых чисел a найдем в общем случае $\gcd(a^m - 1, a^n - 1) = a^{\gcd(m,n)} - 1$.

42. Для всех $k = 1, 2, 3, \ldots$ вычитаем k-й столбец из $2k$-, $3k$-, $4k$-го столбца и т. д. В результате получим треугольную матрицу с элементами x_k на главной диагонали, где $m = \sum_{d \backslash m} x_d$. Отсюда следует, что $x_m = \varphi(m)$, так что определитель равен $\varphi(1)\varphi(2)\ldots\varphi(n)$.

[В общем случае точно так можно доказать, что для произвольной функции f "определитель Смита", в котором (i, j)-й элемент есть $f(\gcd(i, j))$, равен $\prod_{m=1}^{n} \sum_{d \backslash m} \mu(m/d) f(d)$. См. L. E. Dickson, *History of the Theory of Numbers* **1** (Carnegie Inst. of Washington, 1919), 122–123.]

РАЗДЕЛ 4.5.3

1. Время выполнения примерно равно $19.02T + 6$, что чуть меньше, чем для программы 4.5.2A.

2. $\begin{pmatrix} K_n(x_1, x_2, \ldots, x_{n-1}, x_n) & K_{n-1}(x_1, x_2, \ldots, x_{n-1}) \\ K_{n-1}(x_2, \ldots, x_{n-1}, x_n) & K_{n-2}(x_2, \ldots, x_{n-1}) \end{pmatrix}$.

3. $K_n(x_1, \ldots, x_n)$.

4. По индукции или путем вычисления определителя в упр. 2.

5. Когда положительны все x, то q_i в (9) также положительны и $q_{n+1} > q_{n-1}$. Следовательно, выражение (9) представляет собой знакопеременный ряд с убывающими членами, сходящийся тогда и только тогда, когда $q_n q_{n+1} \to \infty$. По индукции, если все числа x больше ϵ, то $q_n \geq (1+\epsilon/2)^n c$, где c выбирается достаточно малым, чтобы неравенство выполнялось для $n = 1$ и 2. Но если $x_n = 1/2^n$, то $q_n \leq 2 - 1/2^n$.

6. Достаточно доказать, что $A_1 = B_1$. Из того факта, что $0 \leq /\!/x_1, \ldots, x_n/\!/ < 1$, когда $x_1, \ldots, x_n$ — положительные целые числа, следует, что $B_1 = \lfloor 1/X \rfloor = A_1$.

7. Только $12\ldots n$ и $n\ldots 21$. (Переменная x_k появляется точно в $F_k F_{n-k}$ членах, следовательно, x_1 и x_n могут быть переставлены только с x_1 и x_n. Если x_1 и x_n не затрагиваются перестановкой, то по индукции следует, что остаются нетронутыми и x_2, $\ldots$, x_{n-1}.)

8. Доказываемое равенство эквивалентно равенству

$$\frac{K_{n-2}(A_{n-1}, \ldots, A_2) - XK_{n-1}(A_{n-1}, \ldots, A_1)}{K_{n-1}(A_n, \ldots, A_2) - XK_n(A_n, \ldots, A_1)} = -\frac{1}{X_n}$$

и в силу (6) эквивалентно

$$X = \frac{K_{n-1}(A_2, \ldots, A_n) + X_n K_{n-2}(A_2, \ldots, A_{n-1})}{K_n(A_1, \ldots, A_n) + X_n K_{n-1}(A_1, \ldots, A_{n-1})}.$$

9. (a) По определению. (b, d) Доказываем для $n = 1$, затем применяем (a), чтобы получить результат в общем случае. (c) Доказываем при $n = k+1$, а затем снова применяем (a).

10. Если $A_0 > 0$, то $B_0 = 0$, $B_1 = A_0$, $B_2 = A_1$, $B_3 = A_2$, $B_4 = A_3$, $B_5 = A_4$, $m = 5$. Если $A_0 = 0$, то $B_0 = A_1$, $B_1 = A_2$, $B_2 = A_3$, $B_3 = A_4$, $m = 3$. Если $A_0 = -1$ и $A_1 = 1$, то $B_0 = -(A_2 + 2)$, $B_1 = 1$, $B_2 = A_3 - 1$, $B_3 = A_4$, $m = 3$. Если $A_0 = -1$ и $A_1 > 1$, то $B_0 = -2$, $B_1 = 1$, $B_2 = A_1 - 2$, $B_3 = A_2$, $B_4 = A_3$, $B_5 = A_4$, $m = 5$. Если $A_0 < -1$, то $B_0 = -1$, $B_1 = 1$, $B_2 = -A_0 - 2$, $B_3 = 1$, $B_4 = A_1 - 1$, $B_5 = A_2$, $B_6 = A_3$, $B_7 = A_4$, $m = 7$. [В действительности последние три случая включают в себя еще восемь подслучаев: если какое-либо из чисел B оказывается равным нулю, то следует выполнить "стягивание" в соответствии с правилом (c) упр. 9. Например, если $A_0 = -1$ и $A_1 = A_3 = 1$, то фактически имеем $B_0 = -(A_2 + 2)$, $B_1 = A_4 + 1$; $m = 1$. Когда $A_0 = -2$ и $A_1 = 1$, нужно выполнить двойное стягивание.]

11. Пусть $q_n = K_n(A_1, \ldots, A_n)$, $q'_n = K_n(B_1, \ldots, B_n)$, $p_n = K_{n+1}(A_0, \ldots, A_n)$, $p'_n = K_{n+1}(B_0, \ldots, B_n)$. Учитывая уравнения (5) и (11), получаем

$$X = (p_m + p_{m-1}X_m)/(q_m + q_{m-1}X_m), \quad Y = (p'_n + p'_{n-1}Y_n)/(q'_n + q'_{n-1}Y_n);$$

поэтому в силу тождества (8), если $X_m = Y_n$, утверждаемая зависимость имеет место. Обратно, если $X = (qY + r)/(sY + t)$ и $|qt - rs| = 1$, можно считать, что $s \geq 0$, и индукцией по s показать, что частичные частные для X и Y в конечном счете совпадают. Согласно упр. 9, (d) при $s = 0$ результат очевиден. Для $s > 0$ положим $q = as + s'$, где $0 \leq s' < s$. Тогда $X = a + 1/((sY + t)/(s'Y + r - at))$; поскольку $s(r - at) - ts' = sr - tq$ и $s' < s$, по предположению индукции и в силу упр. 10 частичные частные для X и Y совпадут. [*J. de Math. Pures et Appl.* **15** (1850), 153–155. При внимательном изучении данного доказательства из того факта, что в упр. 10 число m всегда нечетно, становится ясно, что $X_m = Y_n$ тогда и только тогда, когда $X = (qY + r)/(sY + t)$, где $qt - rs = (-1)^{m-n}$.]

12. (a) Так как $V_n V_{n+1} = D - U_n^2$, известно, что $D - U_{n+1}^2$ кратно V_{n+1}; следовательно, по индукции $X_n = (\sqrt{D} - U_n)/V_n$, где U_n и V_n — целые числа. [*Замечание.* Основанный на таком процессе алгоритм используется во многих приложениях для целочисленного

решения квадратных уравнений. (См., например, H. Davenport, *The Higher Arithmetic* (London: Hutchinson, 1952); W. J. LeVeque, *Topics in Number Theory* (Reading, Mass.: Addison-Wesley, 1956); см. также раздел 4.5.4.) Согласно упр. 1.2.4–35 имеем $A_{n+1} = \lfloor(\lfloor\sqrt{D}\rfloor + U_n)/V_{n+1}\rfloor$, где $V_{n+1} > 0$, и $A_{n+1} = \lfloor(\lfloor\sqrt{D}\rfloor + 1 + U_n)/V_{n+1}\rfloor$, где $V_{n+1} < 0$. Следовательно, такой алгоритм выполняется только для положительных целых чисел $\lfloor\sqrt{D}\rfloor$. Более того, тождество $V_{n+1} = A_n(U_{n-1} - U_n) + V_{n-1}$ позволяет при определении V_{n+1} исключить операцию деления.]

(b) Пусть $Y = (-\sqrt{D} - U)/V$, $Y_n = (-\sqrt{D} - U_n)/V_n$. Заменив в доказательстве (a) $\sqrt{D}$ на $-\sqrt{D}$, видим, что сформулированное тождество выполняется. Имеем

$$Y = (p_n/Y_n + p_{n-1})/(q_n/Y_n + q_{n-1}),$$

где элементы p_n и q_n определены в п. (c) настоящего упражнения. Следовательно,

$$Y_n = (-q_n/q_{n-1})(Y - p_n/q_n)/(Y - p_{n-1}/q_{n-1}).$$

Но согласно (12) p_{n-1}/q_{n-1} и p_n/q_n очень близки к X; учитывая, что $X \neq Y$, величины $Y - p_n/q_n$ и $Y - p_{n-1}/q_{n-1}$ для всех больших значений n будут иметь тот же знак, что и $Y - X$. Это доказывает, что $Y_n < 0$ для всех больших значений n. Следовательно, $0 < X_n < X_n - Y_n = 2\sqrt{D}/V_n$ и V_n должно быть положительным. Так как $X_n > 0$, то и $U_n < \sqrt{D}$. Значит, $V_n < 2\sqrt{D}$, поскольку $V_n \leq A_n V_n < \sqrt{D} + U_{n-1}$.

Наконец покажем, что $U_n > 0$. Поскольку $X_n < 1$, то $U_n > \sqrt{D} - V_n$, так что достаточно рассмотреть случай, когда $V_n > \sqrt{D}$. Тогда $U_n = A_n V_n - U_{n-1} \geq V_n - U_{n-1} > \sqrt{D} - U_{n-1}$, а это, как уже установлено, величина положительная.

Замечание. В повторяющемся цикле имеем $\sqrt{D} + U_n = A_n V_n + (\sqrt{D} - U_{n-1}) > V_n$; отсюда $\lfloor(\sqrt{+}U_{n+1})/V_{n+1}\rfloor = \lfloor A_{n+1} + V_n/(\sqrt{D} + U_n)\rfloor = A_{n+1} = \lfloor(\sqrt{D} + U_n)/V_{n+1}\rfloor$. Другими словами, A_{n+1} определено значениями U_{n+1} и V_{n+1}; величину (U_n, V_n) можно определить через ее преемника (U_{n+1}, V_{n+1}) в периоде. Фактически из приведенных выше рассуждений следует, что когда $0 < V_n < \sqrt{D} + U_n$ и $0 < U_n < \sqrt{D}$, то $0 < V_{n+1} < \sqrt{D} + U_{n+1}$ и $0 < U_{n+1} < \sqrt{D}$. Более того, если пара (U_{n+1}, V_{n+1}) следует за парой (U', V'), для которой $0 < V' < \sqrt{D} + U'$ и $0 < U' < \sqrt{D}$, то $U' = U_n$ и $V' = V_n$. Таким образом, (U_n, V_n) *будет частью цикла тогда и только тогда, когда* $0 < V_n < \sqrt{D} + U_n$ *и* $0 < U_n < \sqrt{D}$.

(c) $$\frac{-V_{n+1}}{V_n} = X_n Y_n = \frac{(q_n X - p_n)(q_n Y - p_n)}{(q_{n-1} X - p_{n-1})(q_{n-1} Y - p_{n-1})}.$$

Для доказанного тождества имеется сопряженное тождество

$$V p_n p_{n-1} + U(p_n q_{n-1} + p_{n-1} q_n) + ((U^2 - D)/V) q_n q_{n-1} = (-1)^n U_n.$$

(d) Если $X_n = X_m$ для некоторого $n \neq m$, то X — иррациональное число, удовлетворяющее квадратному уравнению $(q_n X - p_n)/(q_{n-1} X - p_{n-1}) = (q_m X - p_m)/(q_{m-1} X - p_{m-1})$.

История идеи, изложенной в этом упражнении, восходит к Джайадеву из Индии (не позднее 1073 г.). (См. K. S. Shukla, *Ganita* **5** (1954), 1–20; C.-O. Selenius, *Historia Math.* **2** (1975), 167–184.) Некоторые из аспектов этих идей обнаружены также в Японии и датируются не позднее 1750 года. (См. Y. Mikami, *The Development of Mathematics in China and Japan* (1913), 223–229.) Однако основные принципы теории цепных дробей применительно к квадратным уравнениям разработаны Эйлером [*Novi Comment. Acad. Sci. Petrop.* **11** (1765), 28–66] и Лагранжем [*Hist. Acad. Sci.* **24** (Berlin, 1768), 111–180].

14. Как и в упр. 9, достаточно проверить указанные тождества для случая, когда c есть последнее частичное отношение, а эта проверка выполняется просто. Применив правило Гурвица, получаем $2/e = //1, 2, 1, 2, 0, 1, 1, 1, 1, 1, 0, 2, 3, 2, 0, 1, 1, 3, 1, 1, 0, 2, 5, \ldots //$. Перейдя к обратной величине и отбросив нули, как в упр. 9, получаем

$$e/2 = 1 + //\, 2, \overline{2m+1,\ 3,\ 1,\ 2m+1,\ 1,\ 3}//, \quad m \geq 0$$

(см. упр. 16). Обратите внимание на некоторую закономерность в полученном выражении. (*Schriften der phys.-ökon. Gesellschaft zu Königsberg* **32** (1891), 59–62.) Гурвиц также изложил в *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich* **41** (1896), Jubelband II, 34-64, §2, способ умножения на произвольное положительное целое число.

15. (Эта процедура обрабатывает значения четырех целых чисел (A, B, C, D), которые имеют инвариантный смысл: "все, что осталось выполнить, — вывести цепную дробь вида $(Ay + B)/(Cy + D)$, где y — число, которое будет введено".) Сначала присвоим $j \leftarrow k \leftarrow 0$, $(A, B, C, D) \leftarrow (a, b, c, d)$; затем введем x_j и будем присваивать $(A, B, C, D) \leftarrow (Ax_j + B,\ A,\ Cx_j + D,\ C)$, $j \leftarrow j + 1$ один или более раз, пока знак числа $C + D$ не станет равным знаку числа C. (Когда $j \geq 1$ и ввод не закончен, выполняется неравенство $1 < y < \infty$, а когда знак числа $C + D$ равен знаку числа C, то известно, что $(Ay + B)/(Cy + D)$ лежит между $(A + B)/(C + D)$ и A/C.) Теперь приступаем к выполнению основного шага. Если точно между числами $(A + B)/(C + D)$ и A/C нет ни одного целого числа, то выводим $X_k \leftarrow \min(\lfloor A/C \rfloor, \lfloor (A + B)/(C + D) \rfloor)$ и присваиваем $(A, B, C, D) \leftarrow (C,\ D,\ A - X_kC,\ B - X_kD)$, $k \leftarrow k + 1$; в противном случае вводим x_j и присваиваем $(A, B, C, D) \leftarrow (Ax_j + B,\ A,\ Cx_j + D,\ C)$, $j \leftarrow j + 1$. Основной шаг может повторяться бесконечно. Однако, если в некоторый момент окажется, что введено *последнее* число x_j, алгоритм немедленно переключится на вывод цепной дроби для $(Ax_j + B)/(Cx_j + D)$, используя алгоритм Евклида, и остановит работу.

Решение требуемого примера приводится в следующей таблице, в которой матрица $\left(\begin{smallmatrix} B & A \\ D & C \end{smallmatrix}\right)$ начинается с верхнего левого угла. Затем происходит сдвиг на единицу вправо для ввода и на единицу вниз для вывода.

	x_j	-1	5	1	1	1	2	1	2	∞
X_k	39	97	-58	-193						
-2	-25	-62	37	123						
2			16	53						
3			5	17	22					
7			1	2	3	5				
1				3	1	4	5	14		
1					2	1	3	7		
1							2	7	9	25
12							1	0	1	2
2										1
∞										0

М. Мендес Франс (M. Mendès France) показал, что количество выводимых частных на одно вводимое частное асимптотически ограничено значениями $1/r$ и r, где $r = 2\lfloor L(|ad-bc|)/2 \rfloor + 1$ и L — функция, определенная в упр. 38; эта граница — наилучшая из возможных. [*Topics in Number Theory*, edited by P. Turán, *Colloquia Math. Soc. János Bolyai* **13** (1976), 183–194.]

Госпер (Gosper) показал также, что вышеприведенный алгоритм вычисления цепных дробей для x и y может быть обобщен на случай вычисления цепных дробей для $(axy + Bx + Cy + d)/(Axy + Bx + Cy + D)$ (в частности, для вычисления цепных дробей для сумм и произведений) [MIT AI Laboratory Memo 239 (February, 29, 1972), Hack 101].

16. По индукции нетрудно доказать, что $f_n(z) = z/(2n + 1) + O(z^3)$ — нечетная функция, представимая в некоторой окрестности начала координат в виде сходящегося степенного ряда, и что она удовлетворяет указанному дифференциальному уравнению. Поэтому

$$f_0(z) = /\!/z^{-1} + f_1(z)/\!/ = \cdots = /\!/z^{-1}, 3z^{-1}, \ldots, (2n + 1)z^{-1} + f_{n+1}(z)/\!/.$$

Остается доказать, что $\lim_{n\to\infty} /\!/z^{-1}, 3z^{-1}, \ldots, (2n + 1)z^{-1}/\!/ = f_0(z)$. [Фактически Эйлер в возрасте 24 лет получил разложения в цепные дроби для гораздо более общих диффе-

ренциальных уравнений вида $f_n'(z) = az^m + bf_n(z)z^{m-1} + cf_n(z)^2$, но он не потрудился доказать сходимость, поскольку в 18 веке было вполне достаточно формальных выкладок и интуиции.]

Для доказательства этого предельного уравнения имеется несколько способов. Прежде всего, полагая $f_n(z) = \sum_k a_{nk}z^k$, можно доказать из уравнения

$$(2n+1)a_{n1} + (2n+3)a_{n3}z^2 + (2n+5)a_{n5}z^4 + \cdots = 1 - (a_{n1}z + a_{n3}z^3 + a_{n5}z^5 + \cdots)^2,$$

что $(-1)^k a_{n(2k+1)}$ есть сумма членов вида $c_k/(2n+1)^{k+1}(2n+b_{k1})\ldots(2n+b_{kk})$, где c_k и b_{km} — положительные целые числа, не зависящие от n. Например, имеем $-a_{n7} = 4/(2n+1)^4(2n+3)(2n+5)(2n+7) + 1/(2n+1)^4(2n+3)^2(2n+7)$. Поэтому $|a_{(n+1)k}| \le |a_{nk}|$ и $|f_n(z)| \le \tan|z|$ для $|z| < \pi/2$. Такая равномерная оценка для $f_n(z)$ делает доказательство сходимости очень простым. При внимательном анализе этого доказательства обнаруживается, что степенной ряд для $f_n(z)$ фактически сходится при $|z| < \pi\sqrt{2n+1}/2$; поэтому вместе с ростом n особые точки функции $f_n(z)$ все больше и больше удаляются от начала координат и цепная дробь фактически представляет разложение функции $\tanh z$ на *всей* комплексной плоскости.

Другое доказательство дает дополнительную информацию иного плана. Если положить

$$A_n(z) = n!\sum_{k=0}^{n}\binom{2n-k}{n}\frac{z^k}{k!} = \sum_{k\ge0}\frac{(n+k)!\,z^{n-k}}{k!\,(n-k)!} = n!\,z^n\,{}_2F_0(n+1,-n;;-1/z),$$

то

$$\begin{aligned} A_{n+1}(z) &= \sum_{k\ge0}\frac{(n+k-1)!\,((4n+2)k + (n+1-k)(n-k))}{k!\,(n+1-k)!}\,z^{n+1-k} \\ &= (4n+2)A_n(z) + z^2A_{n-1}(z). \end{aligned}$$

По индукции получаем, что

$$K_n\Big(\frac{1}{z},\frac{3}{z},\ldots,\frac{2n-1}{z}\Big) = \frac{A_n(2z)+A_n(-2z)}{2^{n+1}z^n},$$
$$K_{n-1}\Big(\frac{3}{z},\ldots,\frac{2n-1}{z}\Big) = \frac{A_n(2z)-A_n(-2z)}{2^{n+1}z^n}.$$

Следовательно,

$$//z^{-1},3z^{-1},\ldots,(2n-1)z^{-1}// = \frac{A_n(2z)-A_n(-2z)}{A_n(2z)+A_n(-2z)}$$

и требуется показать, что это отношение стремится к $\tanh z$. В силу уравнений 1.2.9–(11) и 1.2.6–(24) имеем

$$e^zA_n(-z) = n!\sum_{m\ge0}z^m\left(\sum_{k=0}^{n}\binom{m}{k}\binom{2n-k}{n}(-1)^k\right) = \sum_{m\ge0}\binom{2n-m}{n}z^m\,\frac{n!}{m!}.$$

Поэтому

$$e^zA_n(-z) - A_n(z) = R_n(z) = (-1)^nz^{2n+1}\sum_{k\ge0}\frac{(n+k)!\,z^k}{(2n+k+1)!\,k!}.$$

Далее, имеем $(e^{2z}-1)(A_n(2z)+A_n(-2z)) - (e^{2z}+1)(A_n(2z)-A_n(-2z)) = 2R_n(2z)$; отсюда

$$\tanh z - //z^{-1},3z^{-1},\ldots,(2n-1)z^{-1}// = \frac{2R_n(2z)}{(A_n(2z)+A_n(-2z))(e^{2z}+1)}.$$

Итак, для рассматриваемой разности получена точная формула. При $|2z| \le 1$ множитель $e^{2z}+1$ отличен от нуля, $|R_n(2z)| \le e\,n!/(2n+1)!$ и

$$\frac{1}{2}|A_n(2z)+A_n(-2z)| \ge n!\left(\binom{2n}{n}-\binom{2n-2}{n}-\binom{2n-4}{n}-\binom{2n-6}{n}-\cdots\right)$$
$$\ge \frac{(2n)!}{n!}\left(1-\frac{1}{4}-\frac{1}{16}-\frac{1}{64}-\cdots\right)=\frac{2}{3}\frac{(2n)!}{n!}.$$

Таким образом, сходимость очень быстрая даже для комплексных значений z.

Для перехода от этой цепной дроби к цепной дроби для e^z можно воспользоваться равенством $\tanh z = 1-2/(e^{2z}+1)$. После несложных выкладок получим представление $(e^{2z}+1)/2$ в виде цепной дроби. Правило Гурвица дает разложение в цепную дробь функции $e^{2z}+1$, из которого остается вычесть единицу. Для нечетных n

$$e^{-2/n} = //\overline{1,\ 3mn+\lfloor n/2\rfloor,\ (12m+6)n,\ (3m+2)n+\lfloor n/2\rfloor,\ 1}//, \qquad m \ge 0.$$

Еще одно доказательство было предложено К. С. Дэвисом (C. S. Davis) и опубликовано в журнале *J. London Math. Soc.* **20** (1945), 194–198. Впервые разложение числа e в цепную дробь было получено эмпирически Роджером Коутсом (Roger Cotes), *Philosophical Transactions* **29** (1714), 5–45, Proposition 1, Scholium 3. Эйлер прокомментировал эти результаты в письме Гольдбаху (Goldbach) от 25 ноября 1731 года [*Correspondance Mathématique et Physique*, edited by P. H. Fuss, **1** (St. Petersburg, 1843), 56–60], а также опубликовал более подробное описание этих результатов в *Commentarii Acad. Sci. Petropolitanæ* **9** (1737), 98–137; **11** (1739), 32–81.

17. (b) $//x_1-1,\ 1,\ x_2-2,\ 1,\ x_3-2,\ 1,\ \ldots,\ 1,\ x_{2n-1}-2,\ 1,\ x_{2n}-1//$. [*Примечание.* Отрицательные параметры могут быть исключены из континуантов при помощи тождества

$$K_{m+n+1}(x_1,\ldots,x_m,-x,y_n,\ldots,y_1)$$
$$=(-1)^{n-1}K_{m+n+2}(x_1,\ldots,x_{m-1},x_m-1,1,x-1,-y_n,\ldots,-y_1),$$

из которого после повторного применения можно получить

$$K_{m+n+1}(x_1,\ldots,x_m,-x,y_n,\ldots,y_1)$$
$$=-K_{m+n+3}(x_1,\ldots,x_{m-1},x_m-1,1,x-2,1,y_n-1,y_{n-1},\ldots,y_1).$$

Похожее тождество встречается в упр. 41.]

(c) $\underset{\cdot}{1}+//1,1,3,1,5,1,\ldots//=1+//\overline{2m+1,1}//,\ m\ge 0$.

18. Поскольку в силу тождеств (5) и (8) выполняется

$$K_m(a_1,a_2,\ldots,a_m)\,//a_1,a_2,\ldots,a_m,x//$$
$$=K_{m-1}(a_2,\ldots,a_m)+(-1)^m/(K_{m-1}(a_1,\ldots,a_{m-1})+K_m(a_1,a_2,\ldots,a_m)x),$$

будет выполняться и

$$K_m(a_1,a_2,\ldots,a_m)\,//a_1,a_2,\ldots,a_m,x_1,a_1,a_2,\ldots,a_m,x_2,a_1,a_2,\ldots,a_m,x_3,a_1,\ldots//$$
$$=K_{m-1}(a_2,\ldots,a_m)+//(-1)^m(C+Ax_1),C+Ax_2,(-1)^m(C+Ax_3),\ldots//,$$

где $A=K_m(a_1,a_2,\ldots,a_m)$ и $C=K_{m-1}(a_2,\ldots,a_m)+K_{m-1}(a_1,\ldots,a_{m-1})$. Соответственно в силу (6) искомая разность равна

$$(K_{m-1}(a_2,\ldots,a_m)-K_{m-1}(a_1,\ldots,a_{m-1}))/K_m(a_1,a_2,\ldots,a_m).$$

[Случай, когда $m=2$, рассматривался Эйлером в *Commentarii Acad. Sci. Petropolitanæ* **9** (1737), 98–137, §24–26.]

19. Сумма для $1\le k\le N$ равна $\log_b((1+x)(N+1)/(N+1+x))$.

20. Пусть $H = SG$, $g(x) = (1+x)G'(x)$, $h(x) = (1+x)H'(x)$. Тогда из (37) следует, что $h(x+1)/(x+2) - h(x)/(x+1) = -(1+x)^{-2}g(1/(1+x))/(1+1/(1+x))$.

21. $\varphi(x) = c/(cx+1)^2 + (2-c)/((c-1)x+1)^2$, $U\varphi(x) = 1/(x+c)^2$. При $c \le 1$ минимум функции $\varphi(x)/U\varphi(x)$ достигается в точке $x = 0$ и $2c^2 \le 2$. Если $c \ge \phi$, минимум достигается в точке $x = 1$ и $\le \phi^2$. Когда $c \approx 1.31266$, значения функции при $x = 0$ и $x = 1$ почти одинаковы и минимум > 3.2; получены границы: $(0.29)^n\varphi \le U^n\varphi \le (0.31)^n\varphi$. Улучшение оценок границ произошло за счет хорошо подобранных линейных комбинаций в формуле $Tg(x) = \sum a_j/(x+c_j)$.

23. Основное тождество $R_n'(x) = (R_n(x) - R_n(0))/x + \frac{1}{2}xR_n''(\theta_n(x))$ для $\theta_n(x)$, значения которого изменяются от 0 до x, получим, используя интерполяционную формулу из упр. 4.6.4–15 для $x_0 = 0$, $x_1 = x$, $x_2 = x + \epsilon$ и полагая, что $\epsilon \to 0$. Функция R_n в этом тождестве имеет непрерывную вторую производную. Следовательно, в этом случае $R_n'(x) = O(2^{-n})$.

24. ∞. [А. Хинчин, в *Compos. Math.* **1** (1935), 361–382, доказал, что эта сумма $A_1 + \cdots + A_n$ первых n частичных отношений вещественного числа X будет равна примерно $n \lg n$ почти для всех X. В упр. 35 показывается, что для рациональных чисел X ситуация будет иной.]

25. Всякое объединение интервалов можно записать как объединение непересекающихся интервалов, поскольку имеем $\bigcup_{k\ge1} I_k = \bigcup_{k\ge1}(I_k \setminus \bigcup_{1\le j<k} I_j)$, а это есть объединение непересекающихся множеств $I_k \setminus \bigcup_{1\le j<k} I_j$, каждое из которых может быть выражено в виде конечного числа непересекающихся интервалов. Поэтому можно, пронумеровав каким-либо образом функциональные числа отрезка $[0\,..\,1]$, взять $\mathcal{I} = \bigcup I_k$, где I_k — некоторый интервал длиной $\epsilon/2^k$, содержащий k-е рациональное число. В этом случае $\mu(\mathcal{I}) \le \epsilon$, но $|\mathcal{I} \cap P_n| = n$ для всех n.

26. Цепные дроби $//A_1, \ldots, A_t//$, которые появляются при представлении наших чисел, — это именно те дроби, для которых $A_1 > 1$, $A_t > 1$, а $K_t(A_1, A_2, \ldots, A_t)$ есть делитель числа n. Поэтому (6) завершает доказательство. [*Замечание.* Если $m_1/n = //A_1, \ldots, A_t//$ и $m_2/n = //A_t, \ldots, A_1//$, где m_1 и m_2 взаимно просты с n, то $m_1m_2 \equiv \pm1$ (по модулю n). Это и есть условие, определяющее рассматриваемое соответствие. В случае, когда $A_1 = 1$, согласно (46) имеет место аналогичная симметрия.]

27. Сначала докажем результат для $n = p^e$, а затем — для $n = rs$, где числа r и s взаимно просты. Альтернативный путь — использовать формулу из следующего упражнения.

28. (a) Левая часть — мультипликативная функция (см. упр. 1.2.4–31). Она легко вычисляется, когда n является степенью простого числа. (c) С учетом (a) имеем *формулу обращения Мёбиуса (Möbius)*: если $f(n) = \sum_{d\backslash n} g(d)$, то $g(n) = \sum_{d\backslash n} \mu(n/d)f(d)$.

29. Сумма приближенно равна

$$(((12\ln 2)/\pi^2)\ln N!)/N - \textstyle\sum_{d\ge1}\Lambda(d)/d^2 + 1.47;$$

здесь $\sum_{d\ge1}\Lambda(d)/d^2$ сходится к постоянному значению $-\zeta'(2)/\zeta(2)$, в то время как согласно приближению Стирлинга $\ln N! = N\ln N - N + O(\log N)$.

30. Предлагаемая модификация алгоритма влияет на вычисления только в том случае, когда при выполнении следующего шага деления немодифицированным алгоритмом получается частное, равное 1. В таком случае этот следующий шаг деления будет пропущен модифицированным алгоритмом. Вероятность того, что данный шаг деления будет пропущен, равна вероятности того, что $A_k = 1$ и что этому частному предшествует четное число частных, равных 1. Учитывая условия симметрии, получаем, что это есть вероятность того, что $A_k = 1$ и что *за ним следует* четное число частных, равных 1. Последнее имеет место тогда и только тогда, когда $X_{k-1} > \phi - 1 = 0.618\ldots$, где ϕ —

отношение золотого сечения: действительно, $A_k = 1$ и $A_{k+1} > 1$ тогда и только тогда, когда $\frac{2}{3} \le X_{k-1} < 1$; $A_k = A_{k+1} = A_{k+2} = 1$ и $A_{k+3} > 1$ тогда и только тогда, когда $\frac{5}{8} \le X_{k-1} < \frac{2}{3}$, и т. д. Таким образом, экономится приблизительно $F_{k-1}(1) - F_{k-1}(\phi - 1) \approx 1 - \lg\phi \approx 0.306$ шагов деления. Среднее число шагов деления в случае, когда $v = n$ и u взаимно просто с n, приблизительно равно $((12\ln\phi)/\pi^2)\ln n$.

K. Вален (K. Vahlen) в работе *Crelle* **115** (1895), 221–233, рассматривал все алгоритмы, которые при $u \bmod v \neq 0$ на каждой итерации заменяют значения (u, v) значениями $(v, (\pm u) \bmod v)$. Для $u \perp v$ существует ровно v таких алгоритмов, и они могут быть представлены в виде бинарного дерева с v ветвями. Самые мелкие ветви дерева будут формироваться, если на каждой итерации выбирать самые маленькие остатки — это соответствует наименьшему возможному числу итераций выполнения всех таких алгоритмов определения наибольшего общего делителя. В случае выбора наибольших остатков будут формироваться самые глубокие ветви дерева. [Похожие идеи в *Hist. Acad. Sci.* **23** (Berlin, 1768), 111–180, §58, рассматривал Лагранж.] Дальнейшие результаты, относящиеся к рассматриваемому вопросу, изложены в работе Н. Г. де Брейна (N. G. de Bruijn) и У. М. Заринга (W. M. Zaring), опубликованной в *Nieuw Archief voor Wiskunde* (3) **1** (1953), 105–112, а также в работе Г. И. Ригера (G. J. Rieger), опубликованной в *Math. Nachr.* **82** (1978), 157–180.

На многих компьютерах применение модифицированного алгоритма приводит к удлинению каждого шага деления. В таких случаях предпочтительнее воспользоваться идеей, изложенной в упр. 1. Применив ее, можно избежать выполнения *всех* шагов деления, когда частное равно единице.

31. Пусть $a_0 = 0$, $a_1 = 1$, $a_{n+1} = 2a_n + a_{n-1}$; тогда $a_n = ((1+\sqrt{2})^n - (1-\sqrt{2})^n)/2\sqrt{2}$. Наихудший случай (в смысле теоремы F) имеет место, когда $u = a_n + a_{n-1}$, $v = a_n$, $n \ge 2$. Этот результат получен А. Дюпре (A. Dupré) и опубликован в журнале *J. de Math.* **11** (1846), 41–64. В этой работе А. Дюпре исследовал также более общие процедуры "с предварительным просмотром", предложенные Ж. Бине (J. Binet).

32. (b) Член $K_{m-1}(x_1, \ldots, x_{m-1})K_{n-1}(x_{m+2}, \ldots, x_{m+n})$ соответствует тем словам длиной $m+n$ в азбуке Морзе, для которых тире находится в m- и $(m+1)$-й позициях; другой член соответствует противоположному случаю. (Можно также воспользоваться упр. 2. Более общее тождество

$$\begin{aligned} K_{m+n}(x_1, \ldots, x_{m+n})K_k(x_{m+1}, \ldots, x_{m+k}) \\ \quad = K_{m+k}(x_1, \ldots, x_{m+k})K_n(x_{m+1}, \ldots, x_{m+n}) \\ \qquad + (-1)^k K_{m-1}(x_1, \ldots, x_{m-1})K_{n-k-1}(x_{m+k+2}, \ldots, x_{m+n}) \end{aligned}$$

также опубликовано в работе Л. Эйлера.)

33. (a) Новыми представлениями являются $x = m/d$, $y = (n-m)/d$, $x' = y' = d = \gcd(m, n-m)$ для $\frac{1}{2}n < m < n$. (b) Отношение $(n/x') - y \le x < n/x'$ определяет x. (c) Подсчитаем все элементы x, которые удовлетворяют условиям (b). (d) Пару целых чисел $x > y > 0$ с $x \perp y$ можно записать единственным образом в виде $x = K_m(x_1, \ldots, x_m)$, $y = K_{m-1}(x_1, \ldots, x_{m-1})$, где $x_1 \ge 2$ и $m \ge 1$; здесь $y/x = //x_m, \ldots, x_1//$. (e) Достаточно показать, что $\sum_{1 \le k \le n/2} T(k, n) = 2\lfloor n/2 \rfloor + h(n)$; это следует из упр. 26.

34. (a) Деление x и y на $\gcd(x, y)$ дает $g(n) = \sum_{d \backslash n} h(n/d)$. Применим результат упр. 28, (c) и используем условие симметрии между переменными со штрихом и без него. (b) Для фиксированных y и t представления с $xd \ge x'$ обладают свойством $x' < \sqrt{nd}$; следовательно, имеется $O(\sqrt{nd}/y)$ таких представлений. Суммируем теперь при условии $0 < t \le y < \sqrt{n/d}$. (c) Если $s(y)$ есть данная сумма, то, к примеру, $\sum_{d \backslash y} s(d) = y(H_{2y} - H_y) = k(y)$; отсюда $s(y) = \sum_{d \backslash y} k(y/d)$. Далее $k(y) = y\ln 2 - \frac{1}{4} + O(1/y)$.

(d) $\sum_{y=1}^{n}\varphi(y)/y^2=\sum_{y=1}^{n}[d\backslash y]\,\mu(d)/yd=\sum_{cd\le n}\mu(d)/cd^2$. (Аналогично $\sum_{y=1}^{n}\sigma_{-1}(y)/y^2=O(1)$.) (e) $\sum_{k=1}^{n}\mu(k)/k^2=6/\pi^2+O(1/n)$ (см. упр. 4.5.2–10(d)), а $\sum_{k=1}^{n}\mu(k)\log k/k^2=O(1)$. Следовательно, при $d\ge 1$ имеем $h_d(n)=n((3\ln 2)/\pi^2)\ln(n/d)+O(n)$ для $d\ge 1$. В итоге

$$h(n)=2\sum_{cd\backslash n}\mu(d)h_c(n/cd)=((6\ln 2)/\pi^2)n(\ln n-\textstyle\sum-\sum')+O(n\sigma_{-1}(n)^2),$$

где остаточные суммы равны $\sum=\sum_{cd\backslash n}\mu(d)\ln(cd)/cd=0$ и $\sum'=\sum_{cd\backslash n}\mu(d)\ln c/cd=\sum_{d\backslash n}\Lambda(d)/d$. [Известно, что $\sigma_{-1}(n)=O(\log\log n)$; см. работу Харди (Hardy) и Райт (Wright) *An Introduction to the Theory of Numbers*, §22.9.]

35. См. *Proc. Nat. Acad. Sci.* **72** (1975), 4720–4722. М. Л. В. Пайтвэй (M. L. V. Pitteway) и К. М. А. Кэстл (C. M. A. Castle) [*Bull. Inst. Math. and its Applications* **24** (1988), 17–20] нашли убедительное, но требующее больших усилий эмпирическое свидетельство того, что сумма частичных отношений равна

$$\frac{\pi^2}{24(\ln 2)^2}\left(T_n+\frac12-\frac{18(\ln 2)^2}{\pi^2}\right)^2+\frac{6}{\pi^2}\sum_{\substack{p\text{ простое}\\ p^r\backslash n}}\left(\frac{4r}{p^r}-\frac{p+1}{p^{2r}}\,\frac{p^r-1}{p-1}\right)(\ln p)^2$$
$$-2.542875+O(n^{-1/2}).$$

36. Выполняя алгоритм в обратном порядке и полагая, что на шаге C2 для заданного значения k выполняется t_k-1 делений, получаем минимальное число u_n, для которого $\gcd(u_{k+1},\ldots,u_n)=F_{t_1}\ldots F_{t_k}$ и $u_k\equiv F_{t_1}\ldots F_{t_{k-1}}F_{t_k-1}$ (по модулю $\gcd(u_{k+1},\ldots,u_n)$); здесь все $t_i\ge 2$, $t_1\ge 3$ и $t_1+\cdots+t_{n-1}=N+n-1$. При этих условиях можно минимизировать $u_n=F_{t_1}\ldots F_{t_{n-1}}$, приняв $t_1=3$, $t_2=\cdots=t_{n-2}=2$, $u_n=2F_{N-n+2}$. Если условиться также, что $u_1\ge u_2\ge\cdots\ge u_n$, решение $u_1=2F_{N-n+3}+1$, $u_2=\cdots=u_{n-1}=2F_{N-n+3}$, $u_n=2F_{N-n+2}$ имеет минимум u_1. [См. *CACM* **13** (1970), 433–436, 447–448.]

37. См. *Proc. Amer. Math. Soc.* **7** (1956), 1014–1021; см. также упр. 6.1–18.

38. Положим $m=\lceil n/\phi\rceil$, так что $m/n=\phi^{-1}+\epsilon=//a_1,a_2,\ldots//$, где $0<\epsilon<1/n$. Пусть k — минимальное значение, такое, что $a_k\ge 2$; тогда $(\phi^{1-k}+(-1)^kF_{k-1}\epsilon)/(\phi^{-k}-(-1)^kF_k\epsilon)\ge 2$, отсюда k четно и $\phi^{-2}=2-\phi\le\phi^kF_{k+2}\epsilon=(\phi^{2k+2}-\phi^{-2})\epsilon/\sqrt5$. [*Ann. Polon. Math.* **1** (1954), 203–206.]

39. Минимум 287; ни одна дробь со знаменателем <287 не принадлежит $//2,1,95//=96/287\approx .33449477$, как и интервалу

$$[.3335\ldots.3345]=[//2,1,666//\ldots//2,1,94,1,1,3//].$$

Чтобы решить основной вопрос для дроби с меньшим знаменателем в промежутке $[a\,..\,b]$ в случае, когда $0<a<b<1$, учтем, что в обозначениях, принятых для представления правильных цепных дробей, будет иметь место соответствие $//x_1,x_2,\ldots//<//y_1,y_2,\ldots//$ тогда и только тогда, когда $(-1)^jx_j<(-1)^jy_j$ для наименьших j, таких, что $x_j\ne y_j$, причем символ "∞" помещается за последним частичным отношением рационального числа. Таким образом, если $a=//x_1,x_2,\ldots//$ и $b=//y_1,y_2,\ldots//$ и если j минимально при $x_j\ne y_j$, в промежутке $[a\,..\,b]$ дробь принимает вид $c=//x_1,\ldots,x_{j-1},z_j,\ldots,z_m//$, где $//z_j,\ldots,z_m//$ лежит между $//x_j,x_{j+1},\ldots//$ и $//y_j,y_{j+1},\ldots//$ включительно. Положим, что $K_{-1}=0$. Знаменатель

$$K_{j-1}(x_1,\ldots,x_{j-1})K_{m-j+1}(z_j,\ldots,z_m)+K_{j-2}(x_1,\ldots,x_{j-2})K_{m-j}(z_{j+1},\ldots,z_m)$$

числа c будет минимальным при $m=j$ и $z_j=(j$ нечетно. $\Rightarrow y_j+[y_{j+1}\ne\infty]$; $x_j+[x_{j+1}\ne\infty])$. [Другой способ вывода этого метода основан на теории, изложенной в следующем упражнении.]

40. Можно доказать по индукции, что в каждом узле выполняется $p_r q_l - p_l q_r = 1$, т. е. числа p_l и q_l являются взаимно простыми. Поскольку $p/q < p'/q'$, то $p/q < (p+p')/(q+q') < p'/q'$. Кроме того, понятно, что метки на всех левых вершинах ветвей для чисел p/q меньше этих значений p/q, в то время как метки на всех правых вершинах больше этих значений. Поэтому каждое рациональное число может появиться в качестве метки не более одного раза.

Теперь остается показать, что каждое рациональное число должно обязательно появиться. Если $p/q = //a_1, \ldots, a_r, 1//$, где каждое из a_i — положительное целое число, то по индукции можно показать, что узел с меткой p/q находится посредством перемещения a_1 раз влево, затем — перемещения a_2 раз вправо, a_3 раз влево и т. д.

[Впервые последовательность меток на сформированных уровнях исследовалась М. А. Штерном (M. A. Stern) в *Crelle* **55** (1858), 193–220, хотя в этой работе связь с бинарными деревьями не прослеживается. На получение всех возможных дробей при помощи интерполирования дроби $(p+p')/(q+q')$ между соседними элементами p/q и p'/q' обратили внимание позже. Относящиеся к решению этой задачи важные идеи были опубликованы Даниэлем Швентером (Daniel Schwenter) [*Deliciæ Physico-Mathematicæ* (Nürnberg, 1636), Part 1, Problem 87; *Geometria Practica*, 3rd edition (1641), 68; см. M. Cantor, *Geschichte der Math.* **2** (1900), 763–765] и Джоном Уоллисом (John Wallis) в его книге *Treatise of Algebra* (1685), Chapters 10–11. К. Гюйгенс (C. Huygens) использовал эти идеи при конструировании приводов для своего планетария [см. *Descriptio Automati Planetarii* (1703), опубликовано после его смерти]. Лагранж в работе *Hist. Acad. Sci.* **23** (Berlin, 1767), 311–352, §24, и в дополнении к переводу на французский язык алгебры Эйлера (1774), §18–§20, привел полное описание этой идеи. См. также упр. 1.3.2–19; A. Brocot, *Revue Chronométrique* **6** (1860), 186–194; D. H. Lehmer, *AMM* **36** (1929), 59–67.]

41. Действительно, правильные цепные дроби для чисел, записываемых в общем виде

$$\frac{1}{l_1} + \frac{(-1)^{e_1}}{l_1^2 l_2} + \frac{(-1)^{e_2}}{l_1^4 l_2^2 l_3} + \cdots,$$

обладают интересной закономерностью, основанной на тождестве

$$\begin{aligned} K_{m+n+1}(x_1, \ldots, x_{m-1}, x_m - 1, 1, y_n - 1, y_{n-1}, \ldots, y_1) \\ = x_m K_{m-1}(x_1, \ldots, x_{m-1}) K_n(y_n, \ldots, y_1) \\ + (-1)^n K_{m+n}(x_1, \ldots, x_{m-1}, 0, -y_n, -y_{n-1}, \ldots, -y_1). \end{aligned}$$

Наибольший интерес это тождество представляет для случая, когда $y_n = x_{m-1}$, $y_{n-1} = x_{m-2}$ и т. д., так как

$$K_{n+1}(z_1, \ldots, z_k, 0, z_{k+1}, \ldots, z_n) = K_{n-1}(z_1, \ldots, z_{k-1}, z_k + z_{k+1}, z_{k+2}, \ldots, z_n).$$

В частности, если $p_n/q_n = K_{n-1}(x_2, \ldots, x_n)/K_n(x_1, \ldots, x_n) = //x_1, \ldots, x_n//$, то

$$p_n/q_n + (-1)^n/q_n^2 r = //x_1, \ldots, x_n, r-1, 1, x_n - 1, x_{n-1}, \ldots, x_1//.$$

Заменяя последовательность $//x_1, \ldots, x_n//$ последовательностью $//x_1, \ldots, x_{n-1}, x_n - 1, 1//$, можно управлять формированием знака $(-1)^n$.

Например, для частичных сумм первого ряда получаем следующие цепные дроби четной длины: $//1,1//$; $//1,1,1,1,0,1// = //1,1,1,2//$; $//1,1,1,2,1,1,1,1,1,1//$; $//1,1,1,2,1,1,1,1,1,1,1,0,1,1,1,1,1,1,2,1,1,1// = //1,1,1,2,1,1,1,1,1,1,1,1,2,1,1,1,1,2,1,1,1//$. Далее последовательность упорядочивается и подчиняется простой закономерности. Для случая,

когда $n-1=20q+r$ при $0\le r<20$, найдем, что частичное отношение a_n может быть легко вычислено:

$$a_n=\begin{cases}1, & \text{если } r=0,2,4,5,6,7,9,10,12,13,14,15,17 \text{ или } 19;\\ 2, & \text{если } r=3 \text{ или } 16;\\ 1+(q+r) \bmod 2, & \text{если } r=8 \text{ или } 11;\\ 2-d_q, & \text{если } r=1;\\ 1+d_{q+1}, & \text{если } r=18.\end{cases}$$

Здесь d_n — "последовательность дракона", определяемая правилами $d_0=1$, $d_{2n}=d_n$, $d_{4n+1}=0$, $d_{4n+3}=1$. Кривая дракона (рассматривалась в упр. 4.1–18) поворачивается вправо на n-м шаге тогда и только тогда, когда $d_n=1$.

Числа Лиувилля при $l\ge 3$ равны $//l-1, l+1, l^2-1, 1, l, l-1, l^{12}-1, 1, l-2, l, 1, l^2-1, l+1, l-1, l^{72}-1, \ldots //$. n-е частичное отношение a_n зависит от последовательности дракона, элементы которой представимы по $n \bmod 4$ в следующем виде: если $n \bmod 4=1$, то частное равно $l-2+d_{n-1}+(\lfloor n/2\rfloor \bmod 4)$, если $n \bmod 4=2$, то оно равно $l+2-d_{n+2}-(\lfloor n/2\rfloor \bmod 4)$; если $n \bmod 4=0$, то оно равно 1 или $l^{k!(k-1)}-1$ в зависимости от того, будет ли $d_n=0$ или 1, где число k — наибольшая степень 2, которая делит n; если $n \bmod 4=3$, то оно равно $l^{k!(k-1)}-1$ или 1 в зависимости от того, равно ли $d_{n+1}=0$ или 1, где число k — наибольшая степень 2, которая делит $n+1$. Для $l=2$ применяются те же правила, но из рассмотрения исключаются нули; поэтому в зависимости от $n \bmod 24$ получаются более сложные закономерности..

[*См.* J. O. Shallit, *J. Number Theory* **11** (1979), 209–217; Allouche, Lubiw, Mendès France, van der Poorten, and Shallit, *Acta Arithmetica* **77** (1996), 77–96.]

42. Предположим, что $\|qX\|=|qX-p|$. Можно всегда найти такие целые числа u и v, что $q=uq_{n-1}+vq_n$ и $p=up_{n-1}+vp_n$, где $p_n=K_{n-1}(A_2,\ldots,A_n)$, так как $q_np_{n-1}-q_{n-1}p_n=\pm1$. При $v=0$ результат очевиден. В противном случае должно быть $uv<0$, т. е. знак числа $u(q_{n-1}X-p_{n-1})$ совпадает со знаком числа $v(q_nX-p_n)$, а $|qX-p|$ равно $|u|\,|q_{n-1}X-p_{n-1}|+|v|\,|q_nX-p_n|$. Поскольку $u\ne0$, доказательство на этом завершается. Обобщение данного результата дается теоремой 6.4S.

43. Если число x представимо, оно родственно числу x в дереве Штерна-Брокота из упр. 40, поэтому представимые числа образуют поддерево бинарного дерева. Положим, что числа (u/u') и (v/v') — соседние представимые числа. Тогда одно из них является предшественником другого. Пусть, например, число (u/u') является предшественником числа (v/v'), поскольку другой вариант аналогичен. Тогда (u/u') — ближайший левый предшественник для (v/v'), так что все числа между u/u' и v/v' будут для числа (v/v') его потомками, а этим числом порождается медианное число $((u+v)/(u'+v'))$. Учитывая зависимость между правильной цепной дробью и бинарным деревом, медианное число и все его левые потомки будут иметь в качестве последнего представимого числа p_i/q_i число (u/u'), в то время как все потомки справа от медианного числа будут иметь в качестве последнего представимого числа p_i/q_i число (v/v'). (Числа p_i/q_i помечают *родителей* узлов "точек превращения" на пути к x.)

44. Контрпример для $M=N=100$ выглядит так: $(u/u')=\frac{1}{3}$, $(v/v')=\frac{67}{99}$. Тем не менее в силу уравнения (12) тождество почти всегда справедливо; оно нарушается только тогда, когда $u/u'+v/v'$ очень близко к дроби, более простой, чем (u/u').

45. При использовании обычного длинного деления для определения таких A и r, чтобы выполнялось равенство $u=Av+r$ при $0\le r<v$, требуется $O((1+\log A)(\log u))$ единиц

времени. Если частными во время выполнения алгоритма являются A_1, A_2, ..., A_m, то $A_1A_2\ldots A_m \le u$, так что $\log A_1 + \cdots + \log A_m \le \log u$. В силу теоремы L имеем также $m = O(\log u)$.

46. Да, эта граница может быть уменьшена до величины $O(n(\log n)^2(\log\log n))$, даже если придется вычислять последовательность частичных отношений, которые можно вычислить по алгоритму Евклида. (См. A. Schönhage, *Acta Informatica* **1** (1971), 139–144.) Более того, алгоритм Шёнхаге (Schönhage) является асимптотически оптимальным по отношению к выполняемым им операциям умножения и деления [V. Strassen, *SICOMP* **12** (1983), 1–27]. При не очень больших n на практике лучше применять алгоритм 4.5.2L, однако в книге A. Schönhage, A. F. W. Grotefeld, and E. Vetter *Fast Algorithms* (Heidelberg: Spektrum Akademischer Verlag, 1994), §7.2, приводятся идеи эффективного использования алгоритма для чисел длиной до 1 800 бит.

48. $T_j = (K_{j-2}(-a_2,\ldots,-a_{j-1}),\ K_{j-1}(-a_1,\ldots,-a_{j-1}),\ K_{n-j}(a_{j+1},\ldots,a_n)d)$
$= ((-1)^jK_{j-2}(a_2,\ldots,a_{j-1}),\ (-1)^{j-1}K_{j-1}(a_1,\ldots,a_{j-1}),\ K_{n-j}(a_{j+1},\ldots,a_n)d)$.

49. Поскольку $\lambda x_1 + \mu z_1 = \mu v$ и $\lambda x_{n+1} + \mu z_{n+1} = -\lambda v/d$, существует такое нечетное значение j, что $\lambda x_j + \mu z_j \ge 0$ и $\lambda x_{j+2} + \mu z_{j+2} \le 0$. Если $\lambda x_j + \mu z_j > \theta$ и $\lambda x_{j+2} + \mu z_{j+2} < -\theta$, то выполняется $\mu > \theta/z_j$ и $\lambda > -\theta/x_{j+2}$. Отсюда следует, что $0 < \lambda x_{j+1} + \mu z_{j+1} < \lambda\mu x_{j+1}z_j/\theta - \lambda\mu z_{j+1}x_{j+2}/\theta \le 2\lambda\mu v/\theta = 2\theta$, так как для всех k выполняется $|x_{k+1}z_k| = K_{k-1}(a_2,\ldots,a_k)K_{n-k}(a_{k+1},\ldots,a_n) \le K_{n-1}(a_2,\ldots,a_n) = v/d$. [H. W. Lenstra, Jr., *Math. Comp.* **42** (1984), 331–340.]

50. Положим $k = \lceil \beta/\alpha \rceil$. Если $k\alpha < \gamma$, то результат равен k; в противном случае результат равен

$$k - 1 + \left\lceil \frac{f((1/\alpha) \bmod 1, k - \gamma/\alpha, k - \beta/\alpha)}{\alpha} \right\rceil.$$

51. Если $ax - mz = y$ и $x \perp y$, то имеем $x \perp mz$. Рассмотрим дерево Стерна-Брокота из упр. 40 с заданным дополнительным узлом с меткой $0/1$. Объединим помеченное значение $y = ax - mz$ с каждым узлом с меткой z/x. Требуется найти все узлы z/x, для которых y по абсолютной величине не превышает $\theta = \sqrt{m/2}$ и для которых знаменатель x тоже $\le \theta$. Единственный возможный путь к таким узлам поддерживает положительный маркер влево и отрицательный — вправо. Это правило определяет единственный путь, который поворачивает вправо, когда маркер положительный, влево — когда маркер отрицательный, и останавливается, когда маркер становится равным нулю. Этот путь неявно поддерживается при выполнении алгоритма 4.5.2X, когда $u = m$ и $v = a$, исключая случай, когда алгоритм "прыгает" вперед — он просматривает узлы только перед тем, как маркер меняет знак (родители узлов "точки превращения", как в упр. 43).

Пусть z/x — первый узел пути, маркер которого y удовлетворяет условию $|y| \le \theta$. Если $x > \theta$, то решения нет, так как соответствующие значения на пути имеют даже бóльшие знаменатели. В противном случае $(\pm x, \mp y)$ является решением, полученным при $x \perp y$.

Легко видеть, что если $y = 0$, то решения не существует, и что если $y \ne 0$, то знак маркера на следующем узле пути не будет совпадать со знаком y. Поэтому узел z/x будет обработан алгоритмом 4.5.2X и для некоторого j будет выполняться $x = x_j = K_{j-1}(a_1,\ldots,a_{j-1})$, $y = y_j = (-1)^{(j-1)}K_{n-j}(a_{j+1},\ldots,a_n)d$, $z = z_j = K_{j-2}(a_2,\ldots,a_{j-1})$ (см. упр. 48). Следующим подходящим для решения узлом будет узел с меткой $z'/x' = (z_{j-1} + kz_j)/(x_{j-1} + kx_j)$ с маркером $y' = y_{j-1} + ky_j$, где k настолько мало́, что $|y'| \le \theta$; отсюда $y'y < 0$. Однако теперь нужно увеличить θ, иначе будем иметь $m = K_n(a_1,\ldots,a_n)d = x'|y| + x|y'| \le \theta^2 + \theta^2 = m$ и неравенство не будет удовлетворяться.

Эти рассуждения доказывают, что задача может быть эффективно решена путем применения алгоритма 4.5.2X для случая, когда $u = m$ и $v = a$, но при следующей замене операции шага X2: "Если $v_3 \le \sqrt{m/2}$, то выполнение алгоритма завершается. Пара $(x, y) =$

$(|v_2|, v_3 \operatorname{sign}(v_2))$ является, следовательно, единственным решением, обеспечивающим $x \perp y$ и $x \le \sqrt{m/2}$; в противном случае решения нет". [P. S. Wang, *Lecture Notes in Comp. Sci.* **162** (1983), 225–235; P. Kornerup, R. T. Gregory, *BIT* **23** (1983), 9–20.]

Подобный метод будет работать, если потребовать, чтобы $0 < x \le \theta_1$ и $|y| \le \theta_2$, когда $2\theta_1\theta_2 \le m$.

РАЗДЕЛ 4.5.4

1. Если d_k — не простое число, то его простые множители выделяются перед использованием пробного делителя d_k.

2. Нет; при такой модификации в случае, когда $p_{t-1} = p_t$, алгоритм сделает ошибку, выдав в качестве простого множителя единицу.

3. Можно взять P равным произведению первых 108 простых чисел. [*Замечание.* Для того чтобы только проверить, является ли число n простым, наименьший общий делитель для 416-разрядного числа $P = 19\,590\ldots5\,910$ может быть вычислен значительно быстрее, чем потребуется для выполнения 168 операций деления.]

4. В обозначениях упр. 3.1–11 имеем

$$\sum_{\mu,\lambda} 2^{\lceil \lg \max(\mu+1,\lambda)\rceil} P(\mu,\lambda) = \frac{1}{m} \sum_{l\ge 1} f(l) \prod_{k=1}^{l-1} \Big(1 - \frac{k}{m}\Big),$$

где $f(l) = \sum_{1\le\lambda\le l} 2^{\lceil \lg \max(l+1-\lambda,\lambda)\rceil}$. Если $l = 2^{k+\theta}$ при $0 < \theta \le 1$, то

$$f(l) = l^2(3\cdot 2^{-\theta} - 2\cdot 2^{-2\theta}),$$

где функция $3\cdot 2^{-\theta} - 2\cdot 2^{-2\theta}$ достигает максимума $\frac{9}{8}$ в точке $\theta = \lg(4/3)$ и имеет минимум, равный 1, при $\theta = 0$ и 1. Поэтому среднее значение величины $2^{\lceil \lg \max(\mu+1,\lambda)\rceil}$ находится между средними значениями величин $\mu + \lambda$, умноженными на постоянную в интервале от 1.0 до 1.125, откуда и следует результат.

Замечание. Ричард Брент (Richard Brent) заметил, что при $m \to \infty$ плотность

$$\textstyle\prod_{k=1}^{l-1}(1 - k/m) = \exp(-l(l-1)/2m + O(l^3/m^2))$$

стремится к нормальному распределению, поэтому можно положить, что значения θ распределены равномерно. Тогда функция $3\cdot 2^{-\theta} - 2\cdot 2^{-2\theta}$ имеет среднее значение $3/(4\ln 2)$, а среднее число итераций, необходимых для выполнения алгоритма B, стремится к значению $(3/(4\ln 2) + \frac{1}{2})\sqrt{\pi m/2} = 1.98277\sqrt{m}$. В результате подобного анализа более общего метода, который выполнен в упр. 3.1–7, получен следующий результат: $\sim 1.92600\sqrt{m}$, где $p \approx 2.4771366$ выбрано "оптимально" как корень из $(p^2-1)\ln p = p^2 - p + 1$ (см. *BIT* **20** (1980), 176–184).

Алгоритм B представляет собой уточненный алгоритм Полларда (Pollard), на базе которого было получено решение упр. 3.1–6(b) вместо еще не найденного решения упр. 3.1–7. Поллард показал, что минимальное число n, такое, что $X_{2n} = X_n$, равно среднему значению $\sim (\pi^2/12)Q(m) \approx 1.0308\sqrt{m}$; эта константа $\pi^2/12$ следует из уравнения 4.5.3–(21). Таким образом, общий средний объем вычислений оригинального алгоритма Полларда равен приблизительно $1.03081\sqrt{m}$ — числу операций вычисления наибольших общих делителей (или умножений по модулю m) и $3.09243\sqrt{m}$ операций вычисления квадрата. Это действительно *лучше*, чем при выполнении алгоритма B в случае, когда затраты на вычисление наибольшего общего делителя больше затрат на вычисление квадрата, умноженных на константу 1.17, как это обычно и случается для больших чисел.

Однако Брент обратил внимание на то, что алгоритм B может быть усовершенствован, если при $k > l/2$ из него исключить поиск наибольшего общего делителя; если выполнение шага B4 повторяется до тех пор, пока не станет удовлетворяться неравенство $k \le l/2$, то цикл можно обнаружить и после выполнения последующих $\lambda\lfloor \ell(\mu)/\lambda \rfloor = \ell(\mu) - (\ell(\mu) \bmod \lambda)$ итераций. В этом случае средние затраты приблизительно равны $(3/(4\ln 2))\sqrt{\pi m/2} \approx 1.35611\sqrt{m}$ — числу итераций, на которых выполняется вычисление квадрата без вычисления наибольшего общего делителя, плюс $((\ln\pi - \gamma)/(4\ln 2) + \frac{1}{2})\sqrt{\pi m/2} \approx .88\,319\sqrt{m}$ — число итераций, на которых выполняются обе операции. [См. анализ Генри Кохена (Henri Cohen) в *A Course in Computational Algebraic Number Theory* (Berlin: Springer, 1993), §8.5.]

5. Примечательно, что $11\,111 \equiv 8\,616\,460\,799$ (по модулю $3\cdot 7\cdot 8\cdot 11$), поэтому уравнение (14) справедливо и для $N = 11\,111$ за исключением случая, когда модуль равен 5. Поскольку при вычислении $(x^2 - N) \bmod 5$ остатки равны 4, 0, 3, 3, 0, должно быть $x \bmod 5 = 0$, 1 или 4. Первая проба $x \ge \lceil\sqrt{N}\,\rceil = 106$, при которой удовлетворяются все условия, дает $x = 144$; но вычисление квадратного корня из числа $144^2 - 11\,111 = 9\,625$ не дает в результате целое число. Однако следующая проба дает $156^2 - 11\,111 = 13\,225 = 115^2$, и в результате получаем $11\,111 = (156 - 115)\cdot(156 + 115) = 41\cdot 271$.

6. Подсчитаем число решений (x, y) конгруэнтных уравнений $N \equiv (x - y)(x + y)$ (по модулю p), где $0 \le x, y < p$. Поскольку $N \not\equiv 0$, а p — простое число, то $x + y \not\equiv 0$. Для каждого $v \not\equiv 0$ существует единственное u (по модулю p), такое, что $N \equiv uv$. Далее, так как p — простое число, конгруэнтные уравнения $x - y \equiv u$, $x + y \equiv v$ однозначно определяют $x \bmod p$ и $y \bmod p$. Таким образом, указанное выше уравнение имеет точно $p - 1$ решений (x, y). Если (x, y) — решение, то $(x, p - y)$ тоже является решением при $y \not\equiv 0$, так как $(p - y)^2 \equiv y^2$; и, если (x, y_1) и (x, y_2) — решения, для которых $y_1 \not\equiv y_2$, то $y_1^2 \equiv y_2^2$, откуда $y_1 = p - y_2$. Таким образом, количество различных значений x среди решений (x, y) равно $(p - 1)/2$, если уравнение $N \equiv x^2$ не имеет решений, или $(p + 1)/2$, если $N \equiv x^2$ имеет решения.

7. Одно из возможных решений состоит в том, чтобы для каждого модуля иметь два индекса: один — для адресации текущего слова, другой — для адресации текущего бита; загрузка двух слов таблицы и выполнение индексированной команды сдвига подравняет элементы таблицы. (Такими операциями манипулирования битами оснащены многие компьютеры.)

8. (Можно положить, что $N = 2M$, т. е. четно.) В следующем алгоритме используется вспомогательная таблица $X[1], X[2], \ldots, X[M]$, где в $X[k]$ отражен признак принадлежности числа $2k + 1$ к простым числам.

S1. Присвоить $X[k] \leftarrow 1$ для $1 \le k \le M$. Присвоить также $j \leftarrow 1$, $p \leftarrow 3$, $q \leftarrow 4$. (В ходе выполнения этого алгоритма $p = 2j + 1$ и $q = 2j + 2j^2$.)

S2. Если $X[j] = 0$, то перейти к шагу S4. В противном случае вывести p, которое является простым, и присвоить $k \leftarrow q$.

S3. Если $k \le M$, то присвоить $X[k] \leftarrow 0$, $k \leftarrow k + p$ и повторить этот шаг.

S4. Присвоить $j \leftarrow j + 1$, $p \leftarrow p + 2$, $q \leftarrow q + 2p - 2$. Если $j \le M$, то возвратиться к шагу S2. ▮

Можно заметно ускорить бóльшую часть вычислений, если на шаге S4 сравнить с M не j, а q, и добавить новый цикл, который, подавляя манипуляции p и q, выводит $2j + 1$ для всех оставшихся $X[j]$, равных 1.

Замечание. Оригинальное решето Эратосфена было описано в книге 1, главе 13 сочинений Никомаха (Nicomachus) *Introduction to Arithmetic*. Хорошо известно, что

$$\sum_{p \text{ простое}} [p \le N]/p = \ln\ln N + M + O((\log N)^{-10000}),$$

где $M = \gamma + \sum_{k\geq 2} \mu(k) \ln \zeta(k)/k$ — константа Мертенса, равная

0.26149 72128 47642 78375 54268 38608 69585 90516;

см. F. Mertens, *Crelle* **76** (1874), 46–62; Greene, Knuth, *Mathematics for the Analysis of Algorithms* (Boston: Birkhäuser, 1981), §4.2.3. В частности, число операций, выполняемых оригинальным алгоритмом Никомаха, равно $N \ln \ln N + O(N)$. В упр. 5.2.3–15 и разделе 7.1 рассматриваются пути повышения эффективности методов просеивания для генерирования простых чисел.

9. Если p^2 — делитель числа n для некоторого простого числа p, то p есть делитель числа $\lambda(n)$, но не числа $n-1$. Если $n = p_1 p_2$, где $p_1 < p_2$ — все простые числа, то $p_2 - 1$ является делителем числа $\lambda(n)$, и поэтому $p_1 p_2 - 1 \equiv 0$ (по модулю $p_2 - 1$). Поскольку $p_2 \equiv 1$, то $p_1 - 1$ кратно $p_2 - 1$, но это противоречит предположению, что $p_1 < p_2$. [Значения n, для которых $\lambda(n)$ есть собственный делитель числа $n-1$, называются *числами Кармайкла* (Carmichael). Например, приведем несколько малых чисел Кармайкла, содержащих более шести множителей: $3\cdot11\cdot17$, $5\cdot13\cdot17$, $7\cdot11\cdot13\cdot41$, $5\cdot7\cdot17\cdot19\cdot73$, $5\cdot7\cdot17\cdot73\cdot89\cdot107$. Имеется 8 241 число Кармайкла, меньшее 10^{12}, и существует хотя бы $\Omega(N^{2/7})$ чисел Кармайкла, меньших N [см. W. R. Alford, A. Granville, C. Pomerance, *Annals of Math.* (2) **139** (1994), 703–722].

10. Пусть k_p — порядок числа x_p по модулю n и λ — наименьшее общее кратное всех таких k_p. Тогда λ является делителем числа $n-1$, но не делителем любого $(n-1)/p$, поэтому $\lambda = n-1$. Поскольку $x_p^{\varphi(n)} \bmod n = 1$, то $\varphi(n)$ кратно k_p для всех p; следовательно. $\varphi(n) \geq \lambda$. Но $\varphi(n) < n-1$, если n — не простое число. (Другой способ доказательства заключается в том, что при помощи метода, рассмотренного в упр. 3.2.1.2–15, из элементов x_p строится элемент x, имеющий порядок $n-1$.)

11.

U	V	A	P	S	T	Выход
1984	1	0	992	0	—	
1981	1981	1	992	1	1981	
1983	4	495	993	0	1	$993^2 \equiv +2^2$
1983	991	2	98109	1	991	
1981	4	495	2	0	1	$2^2 \equiv +2^2$
1984	1981	1	99099	1	1981	
1984	1	1984	99101	0	1	$99101^2 \equiv +2^0$

Разложение $199 \cdot 991$ получается из первых или последних выходных данных. Краткость цикла и появление хорошо известного числа 1984 — это, вероятно, просто совпадение.

12. В следующем алгоритме используется вспомогательная $(m+1) \times (m+1)$-матрица с целочисленными элементами E_{jk}, $0 \leq j, k \leq m$, входной вектор $(b_0, b_1, \ldots, b_m)$ с элементами однократной точности и вектор $(x_0, x_1, \ldots, x_m)$ с элементами многократной точности, заданными в интервале $0 \leq x_k < N$.

F1. [Начальная установка.] Присвоить $b_i \leftarrow -1$ для $0 \leq i \leq m$; затем присвоить $j \leftarrow 0$.

F2. [Очередное решение.] Из алгоритма E взять очередное решение $(x, e_0, e_1, \ldots, e_m)$. (Алгоритмы E и F удобно рассматривать как сопрограммы.) Присвоить $k \leftarrow m$.

F3. [Найти нечетное число.] Если $k < 0$, перейти к шагу F5. В противном случае, если e_k четно, присвоить $k \leftarrow k-1$ и повторить этот шаг.

F4. [Векторы линейно зависимы?] Если $b_k \geq 0$, присвоить $i \leftarrow b_k$, $x \leftarrow (x_i x) \bmod N$, $e_r \leftarrow e_r + E_{ir}$ для $0 \leq r \leq m$; присвоить $k \leftarrow k-1$ и возвратиться к шагу F3. В противном случае присвоить $b_k \leftarrow j$, $x_j \leftarrow x$, $E_{jr} \leftarrow e_r$ для $0 \leq r \leq m$; присвоить $j \leftarrow j+1$ и возвратиться к шагу F2. (В последнем случае получаем новое линейно независимое решение по модулю 2, первым нечетным компонентом

которого является e_k. Значения E_{jr} могут и не быть значениями однократной точности, но они, скорее всего, будут оставаться малыми при уменьшении k от m до 1 согласно предположениям Моррисона (Morrison) и Бриллхарта (Brillhart).)

F5. [Попытаться выполнить разложение.] (Теперь e_0, e_1, ..., e_m четные.) Присвоить

$$y \leftarrow ((-1)^{e_0/2} p_1^{e_1/2} \dots p_m^{e_m/2}) \bmod N.$$

Если $x = y$ или $x+y = N$, возвратиться к шагу F2. В противном случае вычислить $\gcd(x-y, N)$, который является собственным делителем числа N, и завершить выполнение алгоритма. ▮

Этот алгоритм находит простые множители, когда есть возможность найти множитель из данного набора результатов алгоритма E. [*Доказательство.* Пусть результатами выполнения алгоритма E будут $(X_i, E_{i0}, \dots, E_{im})$ при $1 \le i \le t$, и положим, что удалось найти разложение на простые множители числа $N = N_1 N_2$, когда выполняются соотношения $x \equiv X_1^{a_1} \dots X_t^{a_t}$ и $y \equiv (-1)^{e_0/2} p_1^{e_1/2} \dots p_m^{e_m/2}$ (по модулю N), где $e_j = a_1 E_{1j} + \dots + a_t E_{tj}$ четно для всех j. Тогда $x \equiv \pm y$ (по модулю N_1) и $x \equiv \mp y$ (по модулю N_2). Нетрудно увидеть, что это решение можно преобразовать в пару (x, y), которая появляется при выполнении шага F5, путем выполнения ряда операций, на которых пары (x, y) последовательно заменяются парами (xx', yy'), где $x' \equiv \pm y'$ (по модулю N).]

13. Имеется 2^d значений величины x, имеющих одинаковые показатели степени $(e_0, \dots, e_m)$, поскольку, если $N = q_1^{f_1} \dots q_d^{f_d}$, знак величины x по модулю $q_i^{f_i}$ можно выбирать произвольно. Множители отсутствуют точно для двух из этих 2^d значений.

14. Поскольку $P^2 \equiv kNQ^2$ (по модулю p) для любого простого делителя p числа V, получим $1 \equiv P^{2(p-1)/2} \equiv (kNQ^2)^{(p-1)/2} \equiv (kN)^{(p-1)/2}$ (по модулю p) при $P \not\equiv 0$.

15. $U_n = (a^n - b^n)/\sqrt{D}$, где $a = \frac{1}{2}(P + \sqrt{D})$, $b = \frac{1}{2}(P - \sqrt{D})$, $D = P^2 - 4Q$. Тогда $2^{n-1} U_n = \sum_k \binom{n}{2k+1} P^{n-2k-1} D^k$; поэтому $U_p \equiv D^{(p-1)/2}$ (по модулю p), если p — нечетное простое число. Аналогично, если $V_n = a^n + b^n = U_{n+1} - QU_{n-1}$, то $2^{n-1} V_n = \sum_k \binom{n}{2k} P^{n-2k} D^k$ и $V_p \equiv P^p \equiv P$. Таким образом, если $U_p \equiv -1$, получаем, что $U_{p+1} \bmod p = 0$. Если $U_p \equiv 1$, то $(QU_{p-1}) \bmod p = 0$. Здесь, если Q кратно p, то $U_n \equiv P^{n-1}$ (по модулю p) для $n > 0$, поэтому U_n никогда не будет кратно p; если Q не кратно p, то $U_{p-1} \bmod p = 0$. Поэтому, как и в теореме L, $U_t \bmod N = 0$, если $N = p_1^{e_1} \dots p_r^{e_r}$, $N \perp Q$ и $t = \operatorname{lcm}_{1 \le j \le r} (p_j^{e_j - 1} (p_j + \epsilon_j))$. При предположениях из этого упражнения ранг появления числа N равен $N+1$; значит, N взаимно просто с Q, а t кратно $N+1$. Кроме того, из предположений этого упражнения следует, что каждое p_j является нечетным и каждое ϵ_j равно ± 1, поэтому $t \le 2^{1-r} \prod p_j^{e_j - 1} (p_j + \frac{1}{3} p_j) = 2(\frac{2}{3})^r N$; следовательно, $r = 1$ и $t = p_1^{e_1} + \epsilon_1 p_1^{e_1 - 1}$. Поэтому $e_1 = 1$ и $\epsilon_1 = 1$.

Замечание. Для того чтобы выполняемая проверка оказалась эффективной, следует подбирать P и Q таким образом, чтобы проверка выполнялась с некоторой вероятностью. Лемер (Lehmer) предложил выбрать $P = 1$, чтобы $D = 1 - 4Q$, а Q выбрать таким образом, чтобы $N \perp QD$. (Если последнее условие не соблюдается, значит, число N не простое, если только выполняется условие $|QD| < N$.) Далее, из приведенного выше рассуждения видно, что желательно иметь $\epsilon_1 = 1$, т. е. $D^{(N-1)/2} \equiv -1$ (по модулю N). Это еще одно условие, налагаемое на выбор Q. Если D удовлетворяет этому условию и если $U_{N+1} \bmod N \neq 0$, то известно, что N — *не* простое.

Пример. Если $P = 1$ и $Q = -1$, то получаем последовательность Фибоначчи и $D = 5$. Поскольку $5^{11} \equiv -1$ (по модулю 23), можно было бы попытаться доказать, что 23 — простое число, используя последовательность Фибоначчи:

$$\langle F_n \bmod 23 \rangle = 0, 1, 1, 2, 3, 5, 8, 13, 21, 11, 9, 20, 6, 3, 9, 12, 21, 10, 8, 18, 3, 21, 1, 22, 0, \dots.$$

Поэтому 24 есть ранг появления 23, и проверка сработала. Однако при помощи последовательности Фибоначчи не удается установить подобным образом тот факт, что числа 13 и 17 — простые, так как $F_7 \bmod 13 = 0$ и $F_9 \bmod 17 = 0$. Если $p \equiv \pm 1$ (по модулю 10), то $5^{(p-1)/2} \bmod p = 1$ и, следовательно, F_{p-1} (но не F_{p+1}) делится на p.

17. Пусть $f(q) = 2\lg q - 1$. Если $q = 2$ или 3, то дерево содержит не более $f(q)$ узлов. Для простого числа $q > 3$ положим $q = 1 + q_1 \ldots q_t$, где $t \geq 2$, а $q_1, \ldots, q_t$ — простые числа. Размер дерева $\leq 1 + \sum f(q_k) = 2 + f(q-1) - t < f(q)$. [*SICOMP* **7** (1975), 214–220.]

18. Величина $x(G(\alpha) - F(\alpha))$ равна количеству таких $n \leq x$, для которых второй по величине простой множитель не превышает x^α, а наибольший по величине простой множитель $> x^\alpha$. Следовательно,

$$xG'(t)\,dt = (\pi(x^{t+dt}) - \pi(x^t)) \cdot x^{1-t}(G(t/(1-t)) - F(t/(1-t))).$$

Вероятность того, что $p_{t-1} \leq \sqrt{p_t}$, равна $\int_0^1 F(t/2(1-t))t^{-1}\,dt$. [Кстати, можно показать, что она равна $\int_0^1 F(t/(1-t))\,dt$, т. е. среднему значению величины $\log p_t / \log x$; она равна и константе Дикмана-Голомба (Dickman-Golomb) .62433, приведенной в упр. 1.3.3–23 и 3.1–13. Можно показать также, что производная $G''(0)$ равна

$$\int_0^1 F(t/(1-t))t^{-2}\,dt = F(1) + 2F(\tfrac{1}{2}) + 3F(\tfrac{1}{3}) + \cdots = e^\gamma.$$

Для третьего по величине простого множителя $H(\alpha) = \int_0^\alpha (H(t/(1-t)) - G(t/(1-t)))t^{-1}\,dt$, а $H''(0) = \infty$. См. P. Billingsley, *Period. Math. Hungar.* **2** (1972), 283–289; J. Galambos, *Acta Arith.* **31** (1976), 213–218; D. E. Knuth, L. Trabb Pardo, *Theoretical Comp. Sci.* **3** (1976), 321–348; J. L. Hafner, K. S. McCurley, *J. Algorithms* **10** (1989), 531–556.]

19. Число $M = 2^D - 1$ кратно всем числам p, для которых порядок 2 по модулю p делит число D. Разовьем эту идею, положив $a_1 = 2$ и $a_{j+1} = a_j^{q_j} \bmod N$, где $q_j = p_j^{e_j}$, p_j есть j-е простое число и $e_j = \lfloor \log 1000 / \log p_j \rfloor$. Пусть $A = a_{169}$. Вычислим теперь $b_q = \gcd(A^q - 1, N)$ для всех простых чисел q, расположенных между 10^3 и 10^5. Это можно сделать, если начать вычисления с числа $A^{1009} \bmod N$, а затем умножить его на $A^4 \bmod N$ или $A^2 \bmod N$. (Похожий метод в 1920 году применил Д. Н. Лемер (D. N. Lehmer), но он не опубликовал результатов.) Так же, как и в алгоритме B, можно путем группирования исключить из рассмотрения почти все наибольшие общие делители; например, поскольку $b_{30r-k} = \gcd(A^{30r} - A^k, N)$, можно использовать группы из 8, вычисляя сначала $c_r = (A^{30r} - A^{29})(A^{30r} - A^{23}) \ldots (A^{30r} - A) \bmod N$, а затем — $\gcd(c_r, N)$ для $33 < r \leq 3334$.

20. См. H. C. Williams, *Math. Comp.* **39** (1982), 225–234.

21. Интересная теория, имеющая отношение к условиям этой задачи, была предложена Эриком Бахом (Eric Bach), *Information and Computation* **90** (1991), 139–155.

22. Алгоритм P не достигает цели только тогда, когда для случайного числа x не подтверждается тот факт, что число n не простое. Будем называть число x *плохим*, если $x^q \bmod n = 1$ или одно из чисел $x^{2^j q}$ удовлетворяет условию $\equiv -1$ (по модулю n) для $0 \leq j < k$. Поскольку число 1 плохое, получаем $p_n = (b_n - 1)/(n-2) < b_n/(n-1)$, где b_n — количество плохих чисел x_i, таких, что $1 \leq x_i < n$, если число n не является простым.

Каждое плохое число x удовлетворяет условию $x^{n-1} \equiv 1$ (по модулю n). Если число p простое, число решений уравнения $x^q \equiv 1$ (по модулю p^e) для $1 \leq x < p^e$ равно числу решений уравнения $qy \equiv 0$ (по модулю $p^{e-1}(p-1)$) для $0 \leq y < p^{e-1}(p-1)$, т. е. $\gcd(q, p^{e-1}(p-1))$, поскольку можно заменить число x числом a^y, где a — простой корень.

Пусть $n = n_1^{e_1} \ldots n_r^{e_r}$, где все n_i — различные простые числа. Согласно китайской теореме об остатках число решений уравнения $x^{n-1} \equiv 1$ (по модулю n) равно

$$\prod_{i=1}^r \gcd(n-1, n_i^{e_i-1}(n_i - 1)),$$

и таких решений не более чем $\prod_{i=1}^{r}(n_i-1)$, поскольку число n_i взаимно просто с $n-1$. Если некоторое $e_i>1$, то $n_i-1\le\frac{2}{9}n_i^{e_i}$ и, следовательно, число решений не превышает $\frac{2}{9}n$; в этом случае $b_n\le\frac{2}{9}n\le\frac{1}{4}(n-1)$, ибо $n\ge 9$.

В связи с этим можно положить, что число n равно произведению $n_1\dots n_r$ различных простых чисел. Пусть $n_i=1+2^{k_i}q_i$, где $k_1\le\dots\le k_r$. Отсюда получаем $\gcd(n-1,n_i-1)=2^{k_i'}q_i'$, где $k_i'=\min(k,k_i)$ и $q_i'=\gcd(q,q_i)$. При модуле n_i количество таких x, для которых выполняется условие $x^q\equiv 1$, равно q_i', а количество чисел x, для которых $x^{2^jq}\equiv -1$, равно $2^jq_i'$ при $0\le j<k_i'$ и 0 в противном случае. Поскольку $k\ge k_1$, то

$$b_n=q_1'\dots q_r'\,(1+\textstyle\sum_{0\le j<k_1}2^{jr}).$$

Для завершения доказательства достаточно показать, что $b_n\le\frac{1}{4}q_1\dots q_r2^{k_1+\dots+k_r}=\frac{1}{4}\varphi(n)$, так как $\varphi(n)<n-1$. Отсюда получаем

$$\begin{aligned}(1+\textstyle\sum_{0\le j<k_1}2^{jr})/2^{k_1+\dots+k_r}&\le(1+\textstyle\sum_{0\le j<k_1}2^{jr})/2^{k_1r}\\&=1/(2^r-1)+(2^r-2)/(2^{k_1r}(2^r-1))\le 1/2^{r-1}.\end{aligned}$$

Следовательно, если не выполняются равенства $r=2$ и $k_1=k_2$, получаем результат. Для $r=2$ в упр. 9 показано, что число $n-1$ не кратно как числу n_1-1, так и числу n_2-1. Значит, если $k_1=k_2$, то нельзя получить равенства $q_1'=q_1$ и $q_2'=q_2$; отсюда следует, что в этом случае $q_1'q_2'\le\frac{1}{3}q_1q_2$ и $b_n\le\frac{1}{6}\varphi(n)$.

[*См. J. Number Theory* **12** (1980), 128–138.] Из этого доказательства следует, что число p_n близко к $\frac{1}{4}$ только в двух случаях: когда число n равно $(1+2q_1)(1+4q_1)$ и когда оно является числом Кармайкла специального вида $(1+2q_1)(1+2q_2)(1+2q_3)$. Например, для $n=49\,939\cdot 99\,877$ получаем $b_n=\frac{1}{4}(49\,938\cdot 99\,876)$ и $p_n\approx .24999$; если $n=1667\cdot 2143\cdot 4523$, то $b_n=\frac{1}{4}(1\,666\cdot 2\,142\cdot 4522)$, $p_n\approx .24968$. Дополнительные примечания приведены в ответе к следующему упражнению.

23. (a) Доказательства для всех случаев, кроме, может быть, случая закона обратимости, выполняются просто. Пусть $p=p_1\dots p_s$ и $q=q_1\dots q_r$, где все p_i и q_j — простые числа. Тогда

$$\left(\frac{p}{q}\right)=\prod_{i,j}\left(\frac{p_i}{q_j}\right)=\prod_{i,j}(-1)^{(p_i-1)(q_j-1)/4}\left(\frac{q_j}{p_i}\right)=(-1)^{\sum_{i,j}(p_i-1)(q_j-1)/4}\left(\frac{q}{p}\right),$$

так что остается только убедиться в том, что $\sum_{i,j}(p_i-1)(q_j-1)/4\equiv(p-1)(q-1)/4$ (по модулю 2). Но $\sum_{i,j}(p_i-1)(q_j-1)/4=(\sum_i(p_i-1)/2)(\sum_j(q_j-1)/2)$ будет нечетной тогда и только тогда, когда для нечетного количества чисел p_i и q_j выполняется $\equiv 3$ (по модулю 4), а это происходит тогда и только тогда, когда число $(p-1)(q-1)/4$ нечетно. [C. G. J. Jacobi, *Bericht Königl. Preuß. Akad. Wiss. Berlin* **2** (1837), 127–136; В. А. Лебег (V. A. Lebesgue) в *J. Math. Pures Appl.* **12** (1847), 497–520, исследовал действенность этого метода.]

(b) Так же, как и в упр. 22, можно положить, что $n=n_1\dots n_r$, где все $n_i=1+2^{k_i}q_i$ — различные простые числа и $k_1\le\dots\le k_r$; положим также, что $\gcd(n-1,n_i-1)=2^{k_i'}q_i'$. Будем называть число x *плохим*, если оно приводит к ошибочной классификации числа n как похожего на простое. Пусть $\Pi_n=\prod_{i=1}^{r}q_i'\,2^{\min(k_i,k-1)}$ — число решений уравнения $x^{(n-1)/2}\equiv 1$. Количество плохих чисел x, для которых выполняется равенство $(\frac{x}{n})=1$, равно Π_n, умноженному на дополнительный множитель $\frac{1}{2}$ при $k_1<k$. (Этот множитель $\frac{1}{2}$ необходим, чтобы гарантировать выполнение равенства $(\frac{x}{n_i})=-1$ для четного количества чисел n_i при $k_i<k$.) Количество плохих чисел x, для которых $(\frac{x}{n})=-1$, равно Π_n, если $k_1=k$, и 0 в противном случае. [Если выполняется условие $x^{(n-1)/2}\equiv -1$ (по модулю n_i),

то получим в результате $(\frac{x}{n_i}) = -1$ при $k_i = k$, $(\frac{x}{n_i}) = +1$ и $k_i > k$ и противоречие, если $k_i < k$. Если $k_1 = k$, то существуют нечетные количества чисел k_i, равных k.]

Замечание. Вероятность того, что выбор плохой, будет $> \frac{1}{4}$ только тогда, когда n — число Кармайкла, для которого $k_r < k$; например, $n = 7 \cdot 13 \cdot 19 = 1729$, число, ставшее известным благодаря Рамануджану (Ramanujan) в другом контексте. Этот анализ был продолжен Луисом Моньером (Louis Monier) при получении следующих формул в замкнутом виде для количества плохих чисел x в общем случае:

$$b_n = \Big(1 + \frac{2^{rk_1} - 1}{2^r - 1}\Big) \prod_{i=1}^{r} q_i'; \qquad b_n' = \delta_n \prod_{i=1}^{r} \gcd\Big(\frac{n-1}{2}, n_i - 1\Big).$$

Здесь b_n' — количество плохих чисел x в этом упражнении, а δ_n равно либо 2 (если $k_1 = k$), либо $\frac{1}{2}$ (если $k_i < k$ и e_i является нечетным для некоторого i), либо 1 (в остальных случаях).

(c) Если $x^q \bmod n = 1$, то $1 = (\frac{x^q}{n}) = (\frac{x}{n})^q = (\frac{x}{n})$. Если $x^{2^j q} \equiv -1$ (по модулю n), то порядок x по модулю n_i должен быть нечетным кратным числа 2^{j+1} для всех простых делителей n_i числа n. Пусть $n = n_1^{e_1} \ldots n_r^{e_r}$ и $n_i = 1 + 2^{j+1} q_i'$; тогда $(\frac{x^*}{n_i}) = (-1)^{q_i''}$, так что $(\frac{x}{n}) = +1$ или -1 в зависимости от того, какой будет $\sum e_i q_i'$ — четной или нечетной. Поскольку $n \equiv (1 + 2^{j+1} \sum e_i q_i')$ (по модулю 2^{j+2}), сумма $\sum e_i q_i'$ будет нечетной тогда и только тогда, когда $j + 1 = k$. [*Theoretical Comp. Sci.* **12** (1980), 97–108.]

24. Пусть M_1 — матрица, имеющая по одной строке на каждое нечетное непростое число n в интервале $1 \le n \le N$, и $N - 1$ столбцов, пронумерованных от 2 до N. Значение элемента, расположенного в строке n и столбце x, равно 1, если проверка числа n алгоритмом P посредством x дает отрицательный результат, и равно 0 в остальных случаях. Известно, что если $N = qn + r$ и $0 \le r < n$, то в строке n содержится не более $-1 + q(b_n + 1) + \min(b_n + 1, r) < q(\frac{1}{4}(n-1)+1) + \min(b_n+1, r) \le \frac{1}{3}qn + \min(\frac{1}{4}n, r) = \frac{1}{3}N + \min(\frac{1}{4}n - \frac{1}{3}r, \frac{2}{3}r) \le \frac{1}{3}N + \frac{1}{6}n \le \frac{1}{2}N$ элементов, равных 0; поэтому по меньшей мере половина элементов матрицы равна 1. Тогда в некотором столбце x_1 матрицы M_1 хотя бы половина элементов равна 1. Если вычеркнуть столбец x_1 и все строки, элементы которых в этом столбце равны 1, то получится матрица M_2 со свойствами, подобными свойствам матрицы M_1. Повторно выполнив описанную процедуру, можно сформировать матрицу M_r, которая имеет $N - r$ столбцов и число строк, меньшее $N/2^r$, и которая содержит не менее $\frac{1}{2}(N-1)$ элементов, равных 1. [См. *FOCS* **19** (1978), 78.]

[Подобным образом может быть доказано существование *единственной* бесконечной последовательности $x_1 < x_2 < \cdots$, такой, что число $n > 1$ будет простым в том и только в том случае, если его проверка выполнена алгоритмом P посредством x для $x = x_1, \ldots, x = x_m$, где $m = \frac{1}{2}\lfloor \lg n \rfloor (\lfloor \lg n \rfloor - 1)$. Существует ли последовательность, обладающая такими же свойствами, но в случае, когда $m = O(\log n)$?]

25. Впервые эта теорема была строго доказана фон Мангольдтом (von Mangoldt) [*Crelle* **114** (1895), 255–305], который на самом деле показал, что остаточный член $O(1)$ равен

$$C + \textstyle\int_x^\infty dt/((t^2-1)t \ln t) \quad \text{минус} \quad 1/2k,$$

при условии, что число n равно k-й степени простого числа. Постоянная C равна $\operatorname{li} 2 - \ln 2 = \gamma + \ln\ln 2 + \sum_{n\ge 2} (\ln 2)^n / nn! = 0.35201\,65995\,57547\,47542\,73567\,67736\,43656\,84471+$.

[Итоги исследований этой задачи за 100 лет, прошедших после публикации работы фон Мангольдта, подвел А. А. Карацуба (A. A. Karatsuba) в книге *Complex Analysis in Number Theory* (CRC Press, 1995). В книге Эрика Баха (Eric Bach) и Джеффри Шэллита (Jeffrey Shallit) *Algorithmic Number Theory* **1** (MIT Press, 1996), глава 8, проанализирована связь гипотезы Римана с конкретными задачами теории чисел.]

26. Если число N не является простым, то оно содержит простой множитель $q \le \sqrt{N}$. Согласно условиям задачи каждый простой делитель p числа f содержит целое

число x_p, такое, что порядок числа x_p по модулю q является делителем числа $N-1$, но не $(N-1)/p$. Поэтому, если число p^k делит f, порядок числа x_p по модулю q будет кратным p^k. В упр. 3.2.1.2–15 показано, что существует элемент x порядка f по модулю q. Однако это невозможно, поскольку тогда должно быть $q^2 \geq (f+1)^2 \geq (f+1)r \geq N$, и равенство не может быть выполнено. [*Proc. Camb. Phil. Soc.* **18** (1914), 29–30.]

27. Если число k не делится на 3 и если $k \leq 2^n+1$, то число $k\cdot 2^n+1$ будет простым тогда и только тогда, когда $3^{2^{n-1}k} \equiv -1$ (по модулю $k\cdot 2^n+1$) (согласно упр. 26). Если же $k\cdot 2^n+1$ — простое число, то согласно закону взаимности квадратичных вычетов число 3 не является квадратичным вычетом по модулю $k\cdot 2^n+1$, поскольку $(k\cdot 2^n+1) \bmod 12 = 5$. [Этот способ проверки был предложен без доказательства Протом (Proth) в *Comptes Rendus Acad. Sci. Paris* **87** (1878), 926.]

Чтобы применять способ проверки Прота с достаточной эффективностью, необходимо обеспечить вычисление значения $x^2 \bmod (k\cdot 2^n+1)$ с почти такой же скоростью, как и вычисление значения $x^2 \bmod (2^n-1)$. Положим, что $x^2 = A\cdot 2^n + B$; тогда $x^2 \equiv B - \lfloor A/k \rfloor + 2^n(A \bmod k)$, и в случае, если k представляется с однократной точностью, остаток вычисляется легко.

[Несколько сложнее проверить "простоту" чисел вида $3\cdot 2^n+1$; для этого необходимо сначала применить случайные числа однократной точности, пока одно из них в соответствии с законом квадратичной взаимности не окажется квадратичным без остатка $\bmod 3\cdot 2^n+1$. После этого в способе проверки, описанном выше, заменяем "3" этим числом. Если окажется, что $n \bmod 4 \neq 0$, можно использовать число 5. Получается, что число $3\cdot 2^n+1$ будет простым, когда $n = 1, 2, 5, 6, 8, 12, 18, 30, 36, 41, 66, 189, 201, 209, 276, 353, 408, 438, 534, 2\,208, 2\,816, 3\,168, 3\,189, 3\,912, 20\,909, 34\,350, 42\,294, 42\,665, 44\,685, 48\,150, 55\,182, 59\,973, 80190, 157169, 213\,321$; других таких чисел вплоть до $n \leq 300\,000$ нет. Число $5\cdot 2^n+1$ будет простым, когда $n = 1, 3, 7, 13, 15, 25, 39, 55, 75, 85, 127, 1\,947, 3\,313, 4\,687, 5\,947, 13\,165, 23\,473, 26\,607, 125\,413, 209\,787, 240\,937$ ($n \leq 300000$). См. R. M. Robinson, *Proc. Amer. Math. Soc.* **9** (1958), 673–681; G. V. Cormack, H. C. Williams, *Math. Comp.* **35** (1980), 1419–1421; H. Dubner, W. Keller, *Math. Comp.* **64** (1995), 397–405; J. S. Young, *Math. Comp.* **67** (1998), 1735–1738.]

28. Имеем $f(p,p^2d) = 2/(p+1) + f(p,d)/p$, поскольку $1/(p+1)$ — вероятность того, что число A кратно числу p; если $d \bmod p \neq 0$, то $f(p,pd) = 1/(p+1)$. Так как $A^2-(4k+3)B^2$ не может быть кратно 4, то $f(2,4k+3) = \frac{1}{3}$. Так как $A^2-(8k+5)B^2$ не может быть кратно 8, то $f(2,8k+5) = \frac{2}{3}$. $f(2,8k+1) = \frac{1}{3}+\frac{1}{3}+\frac{1}{3}+\frac{1}{6}+\frac{1}{12}+\cdots = \frac{4}{3}$. Если $d^{(p-1)/2} \bmod p = (1, p-1)$, то соответственно для нечетных p получим $f(p,d) = (2p/(p^2-1), 0)$.

29. Количество неотрицательных целых решений x_i неравенства $x_1+\cdots+x_m \leq r$ равно $\binom{m+r}{r} \geq m^r/r!$; каждое из этих решений соответствует единственному целому числу $p_1^{x_1}\ldots p_m^{x_m} \leq n$. [Более точные оценки для специального случая, когда числа p_j являются j-ми простыми числами при всех j, приведены в работах N. G. de Bruijn, *Indag. Math.* **28** (1966), 240–247; H. Halberstam, *Proc. London Math. Soc.* (3) **21** (1970), 102–107.]

30. Если $p_1^{e_1}\ldots p_m^{e_m} \equiv x_i^2$ (по модулю q_i), можно найти такие y_i, что $p_1^{e_1}\ldots p_m^{e_m} \equiv (\pm y_i)^2$ (по модулю $q_i^{d_i}$); поэтому согласно китайской теореме об остатках находим 2^d значений величины X, таких, что $X^2 \equiv p_1^{e_1}\ldots p_m^{e_m}$ (по модулю N). Количество пар $(e'_1,\ldots,e'_m; e''_1,\ldots,e''_m)$, для которых соблюдаются указанные свойства и которым соответствуют такие последовательности $(e_1,\ldots,e_m)$, не превышает величины $\binom{r}{r/2}$. Теперь для каждого из 2^d двоичных чисел $a = (a_1\ldots a_d)_2$ положим, что n_a — количество показателей $(e'_1,\ldots,e'_m)$, таких, что $(p_1^{e'_1}\ldots p_m^{e'_m})^{(q_i-1)/2} \equiv (-1)^{a_i}$ (по модулю q_i). Следовательно, доказано, что требуемое количество целых чисел X не меньше $2^d(\sum_a n_a^2)/\binom{r}{r/2}$. Поскольку $\sum_a n_a$ — количество способов замены путем перестановок не более $r/2$ объектов из множества m объектов, т. е. $\binom{m+r/2}{r/2}$, получаем $\sum_a n_a^2 \geq \binom{m+r/2}{r/2}^2/2^d \geq m^r/(2^d(r/2)!^2)$. [Дополнительные сведения,

касающиеся тонкостей применения теоремы D, приведены Шнорром в *J. Algorithms* **3** (1982), 101–127.]

31. Чтобы показать, что $\Pr(X \le 2m) \le e^{-m/2}$, присвоим $n = M$, $pM = 4m$ и $\epsilon M = 2m$.

32. Пусть $M = \lfloor \sqrt[3]{N} \rfloor$ и пусть все x_i каждого из сообщений ограничены интервалом $0 \le x < M^3 - M^2$. Если $x \ge M$, кодируем его в виде $x^3 \bmod N$, как и ранее, но при $x < M$ изменяем кодирование на $(x+yM)^3 \bmod N$, где y — случайное число, принадлежащее интервалу $M^2 - M \le y \le M^2$. При декодировании сначала вычисляем кубический корень и, если в результате получаем значение $M^3 - M^2$ или большее, берем остаток $\bmod M$.

34. Пусть P — вероятность того, что выполняется равенство $x^m \bmod p = 1$; пусть также Q — вероятность того, что выполняется равенство $x^m \bmod q = 1$. Вероятность того, что $\gcd(x^m - 1, N) = p$ или q, равна $P(1-Q) + Q(1-P) = P + Q - 2PQ$. Если $P \le \frac{1}{2}$ или $Q \le \frac{1}{2}$, данная вероятность $\ge 2(10^{-6} - 10^{-12})$, поэтому есть хорошие шансы найти простой множитель после выполнения примерно $10^6 \log m$ арифметических операций по модулю N. С другой стороны, если $P > \frac{1}{2}$ и $Q > \frac{1}{2}$, то $P \approx Q \approx 1$, поскольку есть основная формула $P = \gcd(m, p-1)/p$; поэтому в подобном случае m кратно $\operatorname{lcm}(p-1, q-1)$. Положим, что $m = 2^k r$, где r нечетно, и построим последовательность $x^r \bmod N$, $x^{2r} \bmod N$, ..., $x^{2^k r} \bmod N$; так же, как и в результате выполнения алгоритма P, получаем, что впервые 1 появится после значения y, не равного $N-1$, с вероятностью, не меньшей $\frac{1}{2}$; следовательно, $\gcd(y-1, N) = p$ или q.

35. Пусть $f = (p^{q-1} - q^{p-1}) \bmod N$. Поскольку $p \bmod 4 = q \bmod 4 = 3$, то $(\frac{-1}{p}) = (\frac{-1}{q}) = (\frac{f}{p}) = -(\frac{f}{q}) = -1$ и, кроме того, $(\frac{2}{p}) = -(\frac{2}{q}) = -1$. Пусть для данного сообщения x в интервале $0 \le x \le \frac{1}{8}(N-5)$ имеем $\bar{x} = 4x + 2$ или $8x + 4$, любое из которых удовлетворяет условию $(\frac{\bar{x}}{N}) = 1$. Тогда передаем сообщение $\bar{x}^2 \bmod N$.

Чтобы закодировать это сообщение, сначала используем SQRT-блок для нахождения единственного числа y, такого, чтобы выполнялись условия $y^2 \equiv \bar{x}^2 \bmod N$ и $(\frac{y}{N}) = 1$ и y было четным. Тогда имеем $y = \bar{x}$, поскольку три остальных квадратных корня из числа $\bar{x}^2$ равны $N - \bar{x}$ и $(\pm f\bar{x}) \bmod N$; первый из этих корней нечетный, два других имеют отрицательные символы Якоби. Декодирование на этом завершается присвоением $x \leftarrow \lfloor y/4 \rfloor$, если $y \bmod 4 = 2$, и $x \leftarrow \lfloor y/8 \rfloor$ — в противном случае.

Каждый, кто сможет декодировать такое закодированное сообщение, сможет найти множители числа N, поскольку декодирование ложного сообщения $\bar{x}^2 \bmod N$ в случае, когда $(\frac{\bar{x}}{N}) = -1$, позволяет обнаружить $(\pm f) \bmod N$ и $((\pm f) \bmod N) - 1$ имеет нетривиальный наибольший общий делитель с числом N. [См. *IEEE Transactions* **IT-26** (1980), 726–729.]

36. Согласно выражению (4) m-е простое число равно

$$m \ln m + m \ln\ln m - m + m \ln\ln m / \ln m - 2m/\ln m + O(m(\log\log m)^2 (\log m)^{-2}),$$

хотя для решения поставленной задачи достаточно более слабой оценки $p_m = m \ln m + O(m \log\log m)$. (Полагаем, что p_m является m-м простым числом, учитывая предположение, что значения V_i распределены равномерно.) Если выбрать $\ln m = \frac{1}{2} c \sqrt{\ln N \ln\ln N}$, где $c = O(1)$, получим

$$r = c^{-1}\sqrt{\ln N/\ln\ln N} - c^{-2} - c^{-2}(\ln\ln\ln N/\ln\ln N) - 2c^{-2}(\ln \tfrac{1}{2}c)/\ln\ln N + O(\sqrt{\ln\ln N/\ln N}).$$

Оценка (22) времени выполнения алгоритма E несколько неожиданно упрощается:

$$\exp(f(c,N)\sqrt{\ln N \ln\ln N} + O(\log\log N)),$$

где функция $f(c,N) = c + (1 - (1+\ln 2)/\ln\ln N)c^{-1}$. Значение числа c, минимизирующего функцию $f(c,N)$, равно $\sqrt{1 - (1+\ln 2)/\ln\ln N}$. Таким образом, получаем оценку

$$\exp(2\sqrt{\ln N \ln\ln N}\sqrt{1 - (1+\ln 2)/\ln\ln N} + O(\log\log N)).$$

Для $N = 10^{50}$ эта оценка дает $\epsilon(N) \approx .33$, что по-прежнему существенно превышает результаты наблюдений за поведением процесса.

Замечание. Поведение частичных отношений числа $\sqrt{D}$ подчиняется распределению, полученному в разделе 4.5.3 для случайных вещественных чисел. Например, первый миллион частичных отношений для числа $10^{18}+314159$ содержит точно (415 236, 169 719, 93 180, 58 606) случаев, когда A_n соответствует $(1, 2, 3, 4)$. Более того, в соответствии с результатом упр. 4.5.3–12(c) и уравнением 4.5.3–(12) имеем $V_{n+1} = |p_n^2 - Dq_n^2| = 2\sqrt{D}q_n|p_n - \sqrt{D}q_n| + O(q_n^{-2})$. Поэтому следует ожидать, что поведение величины $V_n/2\sqrt{D}$ будет, по существу, аналогично поведению величины $\theta_n(x) = q_n|p_n - xq_n|$, где x — случайное вещественное число. Для случайной переменной θ_n известна приближенная плотность распределения $\min(1, \theta^{-1}-1)/\ln 2$ для $0 \le \theta \le 1$, равномерная при $\theta \le 1/2$ [см. Bosma, Jager, Wiedijk, *Indag. Math.* **45** (1983), 281–299]. Таким образом, при обнаружении неприемлемой эффективности алгоритма E необходимо, кроме размера величин V_n, принимать во внимание какие-то дополнительные условия.

37. Примените к числу $\sqrt{D}+R$ результат упр. 4.5.3–12, чтобы увидеть, начинается ли сразу же периодическая часть дроби, и затем проверьте свойство палиндромности, вычисляя период в обратном порядке. [Отсюда следует, что вторая часть периода дает те же значения V, что и первая, и выполнение алгоритма E может прекратиться раньше, при выполнении шага E5, когда $U = U'$ или $V = V'$. Однако в общем случае этот период настолько длинен, что не удается добраться до его половины; поэтому для дополнительного усложнения алгоритма оснований нет.]

38. Пусть $r = (10^{50} - 1)/9$. Тогда $P_0 = 10^{49} + 9$; $P_1 = r + 3 \cdot 10^{46}$; $P_2 = 2r + 3 \cdot 10^{47} + 7$; $P_3 = 3r + 2 \cdot 10^{49}$; $P_4 = 4r + 2 \cdot 10^{49} - 3$; $P_5 = 5r + 3 \cdot 10^{49} + 4$; $P_6 = 6r + 2 \cdot 10^{48} + 3$; $P_7 = 7r + 2 \cdot 10^{25}$ (прекрасно!); $P_8 = 8r + 10^{38} - 7$; $P_9 = 9r - 8000$.

39. Заметим, что в случае, когда число $q - 1$ имеет 2 и p в качестве простых множителей, легко доказать, что q есть простое число. Преемниками числа 2 являются числа Ферма, а одной из наиболее известных нерешенных задач теории чисел для этого случая является существование или отсутствие шестого простого числа Ферма. Поэтому мы, возможно, никогда не узнаем, как определить, имеет ли произвольное целое число каких-либо преемников. Тем не менее в некоторых случаях это возможно; например, в 1962 году Джон Селфридж (John Selfridge) доказал, что числа 78 557 и 271 129 таких преемников не имеют [см. *AMM* **70** (1963), 101–102]. Позже В. Серпиньский (W. Sierpiński) доказал существование бесконечного количества нечетных чисел без преемников [*Elemente der Math.* **15** (1960), 73–74]. Возможно, число 78 557 является самым маленьким из них, хотя в настоящее время уже известно 69 претендентов на эту роль благодаря исследованиям, выполненным в 1983 году Г. Йешке (G. Jaeschke) и У. Келлером (W. Keller) [*Math. Comp.* **40** (1983), 381–384, 661–673; **45** (1985), 637].

Сведения о более традиционной форме цепочек простых чисел (форме Каннингэма (Cunningham)), в которых переходами являются $p \to 2p \pm 1$, приведены в статье Гюнтера Лоха (Günter Löh) *Math. Comp.* **53** (1989), 751–759. В частности, он нашел, что число $554\,688\,278\,430 \cdot 2^k - 1$ является простым при $0 \le k < 12$.

40. [*Inf. Proc. Letters* **8** (1979), 28–31.] Заметим, что в таком абстрактном автомате $x \bmod y = x - y\lfloor x/y \rfloor$ может быть легко вычислено, в результате чего получим просто константы вида $0 = x - x$, $1 = \lfloor x/x \rfloor$, $2 = 1 + 1$. Убедиться в выполнении условия $x > 0$ можно, проверив, будет ли $x = 1$ или $\lfloor x/(x-1) \rfloor \ne 0$.

(a) Сначала за $O(\log n)$ шагов вычислим $l = \lfloor \lg n \rfloor$ путем повторного деления на 2; одновременно вычислим $k = 2^l$ и $A = 2^{2^{l+1}}$ за $O(\log n)$ циклов путем повторного присвоения $k \leftarrow 2k$, $A \leftarrow A^2$. Прежде чем выполнять основные вычисления, предположим, что известны значения $t = A^m$, $u = (A+1)^m$ и $v = m!$. Теперь можно выполнить присвоения

$m \leftarrow m+1$, $t \leftarrow At$, $u \leftarrow (A+1)u$, $v \leftarrow vm$, увеличив таким образом значение m на 1; кроме того, можно *удвоить* значение m путем присвоения $m \leftarrow 2m$, $u \leftarrow u^2$, $v \leftarrow (\lfloor u/t \rfloor \bmod A)v^2$, $t \leftarrow t^2$, учитывая, что число A является достаточно большим. (Подумайте над вариантом, когда число u представлено в системе счисления по основанию A; при этом A должно быть больше, чем $\binom{2m}{m}$.) Далее, если $n = (a_l \dots a_0)_2$, положим $n_j = (a_l \dots a_j)_2$; если $m = n_j$ и $k = 2^j$ при $j > 0$, то можно уменьшить j на 1, присвоив $k \leftarrow \lfloor k/2 \rfloor$, $m \leftarrow 2m + (\lfloor n/k \rfloor \bmod 2)$. Следовательно, можно вычислить число $n_j!$ для $j = l$, $l-1$, $\dots$, 0 за $O(\log n)$ циклов. [Джулия Робинсон (Julia Robinson) предложила другое решение, а именно: вычислять $n! = \lfloor B^n / \binom{B}{n} \rfloor$ тогда, когда $B > (2n)^{n+1}$ (см. *AMM* **80** (1973), 250–251, 266).]

(b) Сначала, как и в (a), вычисляем $A = 2^{2^{l+2}}$, затем находим наименьшее $k \geq 0$, такое, что $2^{k+1}! \bmod n = 0$. Если $\gcd(n, 2^k!) \neq 1$, полагаем $f(n)$ равным этому значению; обратите внимание, что этот наибольший общий делитель может быть вычислен при помощи алгоритма Евклида за $O(\log n)$ циклов. Если $\gcd(n, 2^k!) = 1$, можно найти наименьшее целое число m, такое, что $\binom{m}{\lfloor m/2 \rfloor} \bmod n = 0$, и положить $f(n) = \gcd(m, n)$. (Учтите, что в этом случае $2^k < m \leq 2^{k+1}$; следовательно, $\lceil m/2 \rceil \leq 2^k$ и $\lceil m/2 \rceil!$ взаимно просто с числом n, поэтому $\binom{m}{\lfloor m/2 \rfloor} \bmod n = 0$ тогда и только тогда, когда $m! \bmod n = 0$. В дальнейшем $n \neq 4$.)

При ограниченном количестве регистров для вычисления m можно использовать числа Фибоначчи (см. алгоритм 6.2.1F). Предположим, известно, что $s = F_j$, $s' = F_{j+1}$, $t = A^{F_j}$, $t' = A^{F_{j+1}}$, $u = (A+1)^{2F_j}$, $u' = (A+1)^{2F_{j+1}}$, $v = A^m$, $w = (A+1)^{2m}$, $\binom{2m}{m} \bmod n \neq 0$ и $\binom{2(m+s)}{m+s} \bmod n = 0$. Этого можно легко достичь при $m = F_{j+1}$ для достаточно больших j за $O(\log n)$ циклов; более того, A будет больше, чем $2^{2(m+s)}$. Если $s = 1$, присваиваем $f(n) = \gcd(2m+1, n)$ или $\gcd(2m+2, n)$, выбирая то из них, которое $\neq 1$. Выполнение алгоритма на этом завершается. В противном случае уменьшаем j на 1 следующим образом: сначала присваиваем $r \leftarrow s$, $s \leftarrow s' - s$, $s \leftarrow r$, $r \leftarrow t$, $t \leftarrow \lfloor t'/t \rfloor$, $t' \leftarrow r$, $r \leftarrow u$, $u \leftarrow \lfloor u'/u \rfloor$, $u' \leftarrow r$, затем, если $(\lfloor wu/vt \rfloor \bmod A) \bmod n \neq 0$, присваиваем $m \leftarrow m + s$, $w \leftarrow wu$, $v \leftarrow vt$.

[Можно ли решить эту задачу за менее чем $O(\log n)$ операций? Можно ли вычислить наименьший или наибольший простой множитель числа n за $O(\log n)$ операций?]

41. (a) Очевидно, что $\pi(x) = \pi(m) + f_1(x,m) = \pi(m) + f(x,m) - f_0(x,m) - f_2(x,m) - f_3(x,m) - \cdots$ при $1 \leq m \leq x$. Присвоим $x = N^3$, $m = N$ и учтем, что $f_k(N^3, N) = 0$ при $k > 2$.

(b) Получаем $f_2(N^3, N) = \sum_{N<p\leq q} [pq \leq N^3] = \sum_{N<p\leq N^{3/2}} (\pi(N^3/p) - \pi(p) + 1) = \sum_{N<p\leq N^{3/2}} \pi(N^3/p) - \binom{\pi(N^{3/2})}{2} + \binom{\pi(N)}{2}$, где p и q определены на множестве простых чисел. Следовательно, $f_2(1\,000, 10) = \pi(\frac{1\,000}{11}) + \pi(\frac{1\,000}{13}) + \pi(\frac{1\,000}{17}) + \pi(\frac{1\,000}{19}) + \pi(\frac{1\,000}{23}) + \pi(\frac{1\,000}{29}) + \pi(\frac{1\,000}{31}) - \binom{\pi(31)}{2} + \binom{\pi(10)}{2} = 24 + 21 + 16 + 15 + 14 + 11 + 11 - 55 + 6 = 63$.

(c) Приведенное в указании тождество просто означает, что p_j-долгожитель есть p_{j-1}-долгожитель, который не кратен p_j. Очевидно, что $f(N^3, N) = f(N^3, p_{\pi(N)})$. Применяем это тождество до тех пор, пока не получим значения $f(x, p_j)$, где либо $j = 0$, либо $x \leq N^2$. Найдем результат:

$$f(N^3, N) = \sum_{k=1}^{N-1} \mu(k) f\Big(\frac{N^3}{k}, 1\Big) - \sum_{j=1}^{\pi(N)} \sum_{N/p_j \leq k < N} \mu(k) f\Big(\frac{N^3}{kp_j}, p_{j-1}\Big) [k \text{ — } p_j\text{-долгожитель}].$$

Далее, $f(x, 1) = \lfloor x \rfloor$, поэтому первая сумма равна $1\,000 - 500 - 333 - 200 + 166 - 142 = -9$ (при $N = 10$). Вторая сумма такова: $-f(\frac{1\,000}{10}, 1) - f(\frac{1\,000}{14}, 1) - f(\frac{1\,000}{15}, 2) - f(\frac{1\,000}{21}, 2) - f(\frac{1\,000}{35}, 3) = -100 - 71 - 33 - 24 - 9 = -237$. Отсюда $f(1\,000, 10) = -9 + 237 = 228$ и $\pi(1\,000) = 4 + 228 - 1 - 63 = 168$.

(d) Если $N^2 \leq 2^m$, можно сформировать массив, в котором элементы $a_{2^m-1+n} = [n+1$ есть p_j-долгожитель] для $1 \leq n \leq N^2$ представляют решето после j проходов, а $a_n =$

$a_{2n} + a_{2n+1}$ для $1 \le n < 2^m$. Затем, если $x \le N^2$, легко вычислить функцию $f(x, p_j)$ за $O(m)$ шагов и удалить из решета числа, кратные p, за $O(N^2 m/p)$ шагов. Общее время вычисления функции $f(N^3, N)$ будет равно $O(N^2 \log N \log\log N)$, так как $\sum_{j=1}^{\pi(N)} 1/p_j = O(\log\log N)$.

Требования к объему памяти могут быть снижены от значения $2N^2 m$ до $2Nm$, если разбить решето на N фрагментов размером N каждое и работать с каждым из них отдельно. Могут пригодиться вспомогательные таблицы значений p_j для $1 \le j \le \pi(N)$ и $\mu(k)$, а также наименьших простых множителей чисел k, $1 \le k \le N$, которые можно просто сформировать перед выполнением основных вычислений.

[См. *Math. Comp.* **44** (1985), 537–560. Впервые подобный метод был предложен Д. Ф. Э. Мейселем (D. F. E. Meissel, *Math. Annalen* **2** (1870), 636–642; **3** (1871), 523–525; **21** (1883), 304; **25** (1885), 251–257. Д. Г. Лемер (D. H. Lehmer) опубликовал в журнале *Illinois J. Math.* **3** (1959), 381–388, ряд уточнений этого метода. Но ни Мейсель, ни Лемер не нашли правила, по которому рекуррентная процедура останавливалась бы с той же эффективностью, что в описанном выше методе. Кроме того, Лагариас (Lagarias) и Одлыжко (Odlyzko), используя принципы аналитической теории чисел, разработали совершенно иное приближение, посредством которого величина $\pi(N)$ может быть вычислена за $O(N^{1/2+\epsilon})$ шагов (см. *J. Algorithms* **8** (1987), 173–191). При помощи уточненного метода, выполняющего вычисления за $O(N^{2/3+\epsilon})$ шагов, Делеглиз (Deléglise) и Рива (Rivat) [*Math. Comp.* **65** (1996), 235–245] установили мировой рекорд вычисления простых чисел: $\pi(10^{20}) = 2\,220\,819\,602\,560\,918\,840$.]

42. **L1.** [Начальная установка.] Найти $\bar{r}$, такое, что $r\bar{r} \equiv 1$ (по модулю s); затем присвоить $r' \leftarrow n\bar{r} \bmod s$, $u \leftarrow r'\bar{r} \bmod s$, $v \leftarrow s$, $w \leftarrow (n - rr')\bar{r}/s \bmod s$, $\theta \leftarrow \lfloor\sqrt{N/s}\rfloor$, $(u_1, u_3) \leftarrow (1, u)$, $(v_1, v_3) \leftarrow (0, v)$. (Задача заключается в поиске всех пар целых чисел (λ, μ), таких, что $(\lambda s + r)(\mu s + r') = N$; отсюда следует, что $\lambda u + \mu \equiv w$ (по модулю s) и $\sqrt{\lambda\mu v} \le \theta$. Алгоритм 4.5.2X будет выполняться без учета величин t_2, u_2, v_2; отношения

$$\lambda t_3 + \mu t_1 \equiv w t_1, \quad \lambda u_3 + \mu u_1 \equiv w u_1, \quad \lambda v_3 + \mu v_1 \equiv w v_1 \qquad \text{(по модулю } s\text{)}$$

остаются инвариантными.)

L2. [Попытка для делителей.] Если $v_1 = 0$, то вывести значение $\lambda s + r$ во всех случаях, когда $\lambda s + r$ делит N и $0 \le \lambda \le \theta/s$. Если $v_3 = 0$, вывести значение $N/(\mu s + r')$ во всех случаях, когда $\mu s + r'$ делит N и $0 \le \mu \le \theta/s$. В остальных случаях для всех k, таких, что $|wv_1 + ks| \le \theta$, если $v_1 < 0$, или $0 < wv_1 + ks \le 2\theta$, если $v_1 > 0$, и для $\sigma = +1$ и -1 вывести значение $\lambda s + r$, если $d = (wv_1 s + ks^2 + v_3 r + v_1 r')^2 - 4v_1 v_3 N$ — точный квадрат и если числа

$$\lambda = \frac{wv_1 s + ks^2 - v_3 r + v_1 r' + \sigma\sqrt{d}}{2v_3 s}, \quad \mu = \frac{wv_1 s + ks^2 + v_3 r - v_1 r' - \sigma\sqrt{d}}{2v_3 s}$$

являются положительными целыми числами. (Решения для $\lambda v_3 + \mu v_1 = wv_1 + ks$, $(\lambda s + r)(\mu s + r') = N$ существуют.)

L3. [Выполнено?] Если $v_3 = 0$, то выполнение алгоритма завершается.

L4. [Разделить и вычесть.] Присвоить $q \leftarrow \lfloor u_3/v_3 \rfloor$. Если $u_3 = qv_3$ и $v_1 < 0$, то уменьшить q на 1. Затем присвоить

$$(t_1, t_3) \leftarrow (u_1, u_3) - (v_1, v_3)q, \quad (u_1, u_3) \leftarrow (v_1, v_3), \quad (v_1, v_3) \leftarrow (t_1, t_3)$$

и возвратиться к шагу L2. ▮

[См. *Math. Comp.* **42** (1984), 331–340. Оценки для шага L2 можно уточнить, например, для того, чтобы обеспечить $d \geq 0$. Некоторые множители могут выводиться более одного раза.]

43. (a) Прежде всего убедимся в том, что символ Якоби $\left(\frac{y}{m}\right)$ равен $+1$. (Если он равен 0, задача упрощается; если он равен -1, то $y \notin Q_m$.) Затем выберем случайные целые числа $x_1, \ldots, x_n$ в интервале $[0\,..\,m)$ и положим $X_j = [G(y^2 x_j^4 \bmod m) = (y x_j^2 \bmod m) \bmod 2]$. Если $y \in Q_m$, то $\mathrm{E}\,X_j \geq \frac{1}{2} + \epsilon$; в противном случае $m - y \in Q_m$ и $\mathrm{E}\,X_j \leq \frac{1}{2} - \epsilon$. Сообщить, что $y \in Q_m$, если $X_1 \cdots + X_n \geq \frac{1}{2}n$. В силу результата упр. 1.2.10–21 вероятность неудачи не превышает величины $e^{-2\epsilon^2 n}$. Поэтому выбираем $n = \lceil \frac{1}{2}\epsilon^{-2} \ln \delta^{-1} \rceil$.

(b) Находим x с символом Якоби $\left(\frac{x}{m}\right) = -1$ и присваиваем $y \leftarrow x^2 \bmod m$. В этом случае множителями числа m будут $\gcd(x+\sqrt{y}, m)$ и $\gcd(x-\sqrt{y}, m)$, так что задача теперь состоит в том, чтобы найти $\pm\sqrt{y}$ для заданного $y \in Q_m$. Если найти τv для любого ненулевого значения v, то поставленная задача будет решена, поскольку $\sqrt{y} = (v^{-1}\tau v) \bmod m$, если $\gcd(v, m)$ не является множителем числа m.

Предположим, что для некоторого $e \geq 1$ выполняется равенство $\epsilon = 2^{-e}$. Выберем в интервале $[0\,..\,m)$ случайные целые числа a и b и предположим, что известны двоичные функции α_0 и β_0, такие, что выполняются неравенства

$$\left|\frac{\tau a}{m} - \alpha_0\right| < \frac{\epsilon}{64}, \qquad \left|\frac{\tau b}{m} - \beta_0\right| < \frac{\epsilon^3}{64}.$$

Здесь α_0 — нечетное число, кратное $\epsilon/64$, а β_0 — нечетное число, кратное $\epsilon^3/64$. Положим также, что известны λa и λb. Конечно, на самом деле нам не известны значения α_0, β_0, λa и λb, но мы попробуем применить все возможные значения $32\epsilon^{-1} \times 32\epsilon^{-3} \times 2 \times 2$. Ложные ответвления программы, которые оперируют некорректными предположениями, не нанесут вреда.

Определим числа в виде $u_{tj} = 2^{-t}(a + (j+\frac{1}{2})b) \bmod m$ и $v_{tj} = 2^{-t-1}(a + jb) \bmod m$. Оба эти числа u_{tj} и v_{tj} равномерно распределены в интервале $[0\,..\,m)$, так как числа a и b были выбраны случайными. Далее, для фиксированного t числа u_{tj} при $j_0 \leq j < j_0 + l$ являются *попарно независимыми*, то же самое относится к числам v_{tj} при $j_0 \leq j < j_0 + l$ до тех пор, пока значение l не достигнет наименьшего простого множителя числа m. Числа u_{tj} и v_{tj} можно использовать только для неравенства $-2r\epsilon^{-2} \leq j < 2r\epsilon^2$. Если для любого из этих чисел имеется множитель, общий с множителем для m, то задача решена.

Для всех $v \perp m$ определим $\chi v = +1$, если $v \in Q_m$, $\chi v = -1$, если $-v \in Q_m$, и $\chi v = 0$, если $\left(\frac{v}{m}\right) = -1$. Заметим, что $\chi u_{(t+2)j} = \chi u_{tj}$, поскольку $u_{tj} = (2^2 u_{(t+2)j}) \bmod m$. Поэтому можно определить χu_{tj} и χv_{tj} для всех t и j при помощи алгоритма A, примененного к u_{tj} и v_{tj} для $0 \leq t \leq 1$ и $-2r\epsilon^{-2} \leq j < 2r\epsilon^{-2}$. Установка $\delta = \frac{1}{1440}\epsilon^2 r^{-1}$ в этом алгоритме гарантирует, что все значения величины χ верны с вероятностью $\geq 1 - \frac{1}{90}$.

Выполнение алгоритма осуществляется не более чем за r этапов. В начале этапа t для $0 \leq t < r$ полагаем, что известны значения величин $\lambda 2^{-t} a$, $\lambda 2^{-t} b$ и дробей α_t, β_t, такие, что

$$\left|\frac{\tau 2^{-t} a}{m} - \alpha_t\right| < \frac{\epsilon}{2^{t+6}}, \qquad \left|\frac{\tau 2^{-t} b}{m} - \beta_t\right| < \frac{\epsilon^3}{2^{t+6}}.$$

Определим $\alpha_{t+1} = \frac{1}{2}(\alpha_t + \lambda 2^{-t} a)$ и $\beta_{t+1} = \frac{1}{2}(\beta_t + \lambda 2^{-t} b)$; это обеспечивает выполнение неравенств. На следующем шаге находим $\lambda 2^{-t-1} b$, которое удовлетворяет отношению

$$\lambda u_{tj} + \lambda 2^{-t} a + j\lambda 2^{-t} b + \lambda 2^{-t-1} b + \left\lfloor \frac{\tau 2^{-t} a + j\tau 2^{-t} b + \tau 2^{-t-1} b}{m} \right\rfloor \equiv 0 \quad (\text{по модулю } 2).$$

Пусть $n = 4\min(r, 2^t)\epsilon^{-2}$; тогда, если $|j| \le \frac{n}{2}$, получаем

$$\left|\frac{\tau 2^{-t}a}{m} + j\frac{\tau 2^{-t}b}{m} + \frac{\tau 2^{-t-1}b}{m} - (\alpha_t + j\beta_t + \beta_{t+1})\right| < \frac{\epsilon}{16}.$$

Поэтому, если $\chi u_{tj} = 1$, вероятно, что $\lambda 2^{-t-1}b = G_j$, где $G_j = (G(u_{tj}^2 y \bmod m) + \lambda 2^{-t}a + j\lambda 2^{-t}b + \lfloor \alpha_t j\beta_t + \beta_{t+1}\rfloor) \bmod 2$. Более точно, получим

$$\lfloor(\tau 2^{-t}a + j\tau 2^{-t}b + \tau 2^{-t-1}b)/m\rfloor = \lfloor \alpha_t + j\beta_t + \beta_{t+1}\rfloor,$$

если только не выполняются неравенства $\tau u_{tj} < \frac{\epsilon}{16}m$ и $\tau u_{tj} > (1 - \frac{\epsilon}{16})m$. Пусть $Y_j = (2G_j - 1)\chi u_{tj}$. Если $Y_j = +1$, это довод в пользу $\lambda 2^{-t-1}b = 1$; если $Y_j = -1$, то в пользу $\lambda 2^{-t-1}b = 0$, если $Y_j = 0$, то никаких действий не предпринимаем. Будем демократичны и установим $\lambda 2^{-t-1}b = [\sum_{j=-n/2}^{n/2-1} Y_j \ge 0]$.

Какова вероятность того, что $\lambda 2^{-t-1}b$ — корректное значение? Пусть $Z_j = -1$, если $\chi u_{tj} \ne 0$ и ($\tau u_{tj} < \frac{\epsilon}{16}m$ или $\tau u_{tj} > (1 - \frac{\epsilon}{16})m$, или $G(u_{tj}^2 y \bmod m) \ne \lambda u_{tj}$). В противном случае полагаем, что $Z_j = |\chi u_{tj}|$. Поскольку Z_j — функция u_{tj}, случайные переменные Z_j попарно независимы и одинаково распределены. Пусть $Z = \sum_{j=-n/2}^{n/2-1} Z_j$; если $Z > 0$, значение $\lambda 2^{-t-1}b$ будет верным. Вероятность того, что $Z_j = 0$, равна $\frac{1}{2}$, а вероятность того, что $Z_j = +1$, равна $\ge \frac{1}{4} + \frac{\epsilon}{2} - \frac{\epsilon}{8}$; поэтому $\mathrm{E}\, Z_j \ge \frac{3}{4}\epsilon$. Очевидно, что $\mathrm{var}(Z_j) \le \frac{1}{2}$. Таким образом, возможность появления ошибки в ветви программы, основанной на правильных предпосылках, согласно неравенству Чебышева не превышает величины

$$\Pr(Z \le 0) \le \Pr((Z - n\,\mathrm{E}\, Z_j)^2 \ge \tfrac{9}{16}n^2\epsilon^2) \le \tfrac{8}{9}n^{-1}\epsilon^2 = \tfrac{2}{9}\min(r, 2^t)^{-1}$$

(см. упр. 3.5–42).

Аналогичный метод может быть использован для определения величины $\lambda 2^{-t-1}a$ с погрешностью $\le \frac{2}{9}\min(r, 2^t)^{-1}$, если заменить величину u_{tj} величиной v_{tj}. Возможно, окажется, что $\epsilon^3/2^{t+6} < 1/(2m)$, так что число $\tau 2^{-t}b$ будет ближайшим целым к $m\beta_t$. В этом случае можно вычислить значение $\sqrt{y} = (2^t b^{-1}\tau 2^{-t}b) \bmod m$. Выполнив возведение этого числа в квадрат, можно узнать, были ли мы правы.

Суммарный шанс допустить ошибку ограничен величиной $\frac{4}{9}\sum_{t\ge 1} 2^{-t} = \frac{4}{9}$ на стадии $t < \lg n$, а на последующих стадиях — величиной $\frac{4}{9}\sum_{t\le r} r^{-1} = \frac{4}{9}$. Таким образом, общая вероятность возникновения ошибки, включая возможность того, что не все значения величины χ были определены верно, не превышает $\frac{4}{9} + \frac{4}{9} + \frac{1}{90} = \frac{9}{10}$. Выполнение программы завершится успешным вычислением значения $\sqrt{y}$ не менее чем в $\frac{1}{10}$ случаев; следовательно, множители числа m будут получены после повторения процесса в среднем не более десяти раз.

В общем времени выполнения программы доминирует величина $O(r\epsilon^{-4}\log(r\epsilon^{-2})T(G))$, соответствующая времени вычисления χ. К ней следует добавить $O(r^2\epsilon^{-2}T(G))$ — время, затрачиваемое на последующие прогнозы, и $O(r^2\epsilon^{-6})$ — время на вычисление значений α_t, β_t, $\lambda 2^{-t}a$ и $\lambda 2^{-t}b$ во всех ответвлениях программы.

Эта процедура, которая ясно иллюстрирует основные парадоксы вероятностных алгоритмов, разработана Р. Фишлином (R. Fischlin) и К.-П. Шнорром (C.-P. Schnorr) [*Lecture Notes in Comp. Sci.* **1233** (1997), 267–279] на базе более ранних исследований, выполненных Алекси (Alexi), Чором (Chor), Голдрихом (Goldreich) и Шнорром [*SICOMP* **17** (1988), 194–209], а также Бен-Ором (Ben-Or), Чором и Шамиром (Shamir) [*STOC* **15** (1983), 421–430]. Если объединить эту процедуру с леммой 3.5P4, получится теорема, аналогичная теореме 3.5P, в которой последовательность 3.2.2–(17) заменяется последовательностью 3.2.2–(16). Фишлин и Шнорр показали, как упорядо-

чить вычисления, чтобы алгоритм разложения на простые множители выполнялся за $O(r\epsilon^{-4}\log(r\epsilon^{-1})T(G))$ шагов; результирующая оценка времени "взлома" 3.2.2–(16) есть $T(F) = O(RN^4\epsilon^{-4}\log(RN\epsilon^{-1})(T(G)+R^2))$. Постоянный множитель O в этом равенстве достаточно велик, но в обозримых пределах. Подобным методом можно получить x из RSA-функции $y = x^a \bmod m$ для случая, когда $a \perp \varphi(m)$, если предсказывать $y^{1/a} \bmod 2$ с вероятностью $\geq \frac{1}{2} + \epsilon$.

44. Предположим, что $\sum_{j=0}^{d-1} a_{ij}x^j \equiv 0$ (по модулю m_i), $\gcd(a_{i0}, a_{i1}, \ldots, a_{i(d-1)}, m_i) = 1$ и $|x| < m_i$ для $1 \leq i \leq k = d(d-1)/2 + 1$, где $m_i \perp m_j$ при $1 \leq i < j \leq k$. Положим также, что $m = \min\{m_1, \ldots, m_k\} > n^{n/2}2^{n^2/2}d^d$, где $n = d + k$. Сначала найдем такую последовательность значений $u_1, \ldots, u_k$, чтобы выполнялись равенства $u_j \bmod m_i = \delta_{ij}$. Теперь сформируем матрицу размера $n \times n$

$$L = \begin{pmatrix} M & & & & & & & \\ 0 & mM & & & & & & \\ \vdots & \vdots & \ddots & & & & & \\ 0 & 0 & \ldots & m^{d-1}M & & & & \\ a_{10}u_1 & ma_{11}u_1 & \ldots & m^{d-1}a_{1(d-1)}u_1 & M/m_1d & & & \\ a_{20}u_2 & ma_{21}u_2 & \ldots & m^{d-1}a_{2(d-1)}u_2 & 0 & M/m_2d & & \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \ddots & \\ a_{k0}u_k & ma_{k1}u_k & \ldots & m^{d-1}a_{k(d-1)}u_k & 0 & 0 & \ldots & M/m_kd \end{pmatrix},$$

где $M = m_1m_2\ldots m_k$. Все наддиагональные элементы этой матрицы равны нулю, следовательно, $\det L = M^{n-1}m^{k-1}d^{-k}$. Пусть теперь $v = (t_0, \ldots, t_{d-1}, v_1, \ldots, v_k)$ — ненулевой вектор с целочисленными компонентами длиной $(vL) \leq \sqrt{n2^n}M^{(n-1)/n}m^{(k-1)/n}d^{-k/n}$. Поскольку $M^{(n-1)/n} < M/m^{k/n}$, длина $(vL) < M/d$. Положим еще, что $c_j = t_jM + \sum_{i=1}^{k} a_{ij}u_iv_i$ и $P(x) = c_0 + c_1x + \cdots + c_{d-1}x^{d-1}$. Тогда при $1 \leq i \leq k$ имеем $P(x) \equiv v_i(a_{i0}+a_{i1}x+\cdots+a_{i(d-1)}x^{d-1}) \equiv 0$ (по модулю m_i); так что $P(x) \equiv 0$ (по модулю M). Кроме того, $|m^jc_j| < M/d$, откуда следует, что $P(x) = 0$. Но $P(x)$ не равно тождественно нулю, так как из условий $v_ia_{ij} \equiv 0$ (по модулю m_i) и $\gcd(a_{i0}, \ldots, a_{i(d-1)}, m_i) = 1$ следует $v_i \equiv 0$ (по модулю m_i), в то время как из неравенства $|v_iM/m_id| < M/d$ следует неравенство $|v_i| < m_i$; поэтому не может быть, чтобы одновременно $v_1 = \cdots = v_k = 0$. Таким образом, можно найти x (точнее, не более $d-1$ возможных значений x). Общее время выполнения алгоритма выражается в виде полинома по $\lg M$. [*Lecture Notes in Comp. Sci.* **218** (1985), 403–408.]

45. Следствие 1. Решение всегда существует. Предположим сначала, что n — простое число. Если $(\frac{b}{n}) = 1$, то при $y = 0$ решение существует. Если $(\frac{b}{n}) = -1$, то пусть $j > 0$ принимает минимальное значение, при котором выполняется равенство $(\frac{-ja}{n}) = -1$; тогда $x_0^2 - a \equiv -ja$ и $b \equiv -ja(y_0)^2$ для некоторых x_0 и y_0 (по модулю n). Следовательно, $(x_0y_0)^2 - ay_0^2 \equiv b$. Предположим теперь, что найдено решение уравнения $x^2 - ay^2 \equiv b$ (по модулю n) и его нужно распространить на решение по модулю n^2. Можно всегда найти такие значения параметров c и d, что $(x+cn)^2 - a(y+dn)^2 \equiv b$ (по модулю n^2), поскольку $(x+cn)^2 - a(y+dn)^2 \equiv x^2 - ay^2 + (2cx - 2ayd)n$ и $\gcd(2x, 2ay) \perp n$. Следовательно, в случае, когда n является степенью нечетного простого числа, решение всегда существует. (Необходимо еще положить, что число n также нечетно, потому что, например, уравнение $x^2 \pm y^2 \equiv 3$ (по модулю 8) решений не имеет.) Таким образом, в соответствии с китайской теоремой об остатках решение существует для всех нечетных n.

Следствие 2. Для заданных a и n, для которых $a \perp n$, число решений одинаково для всех $b \perp n$. Это следует из тождества, приведенного в указании, и факта 1, так как если $x_1^2 - ay_1^2 \equiv b$, то $(x_1x_2 - ay_1y_2, x_1y_2 + x_2y_1)$ удовлетворяет всем решениям уравнения

$x^2 - ay^2 \equiv b$, как только (x_2, y_2) удовлетворяет всем решениям уравнения $x^2 - y^2 \equiv 1$. Другими словами, пара решений (x_2, y_2) однозначно определяется парами решений (x_1, y_1) и (x, y), как только получим $x_1^2 - ay_1^2 \perp n$.

Следствие 3. По заданным целым числам (a, s, z), таким, что $z^2 \equiv a$ (по модулю s), можно найти целые числа (x, y, m, t), для которых выполняется равенство $x^2 - ay^2 = m^2 st$, где $(x, y) \neq (0, 0)$ и $t^2 \leq \frac{4}{3}|a|$. Действительно, если $z^2 = a + ms$, то (u, v) будет парой ненулевых целых чисел, минимизирующих функцию $(zu+mv)^2+|a|u^2$. Значение этой пары можно найти, используя методы из раздела 3.3.4 и неравенство $(zu+mv)^2+|a|u^2 \leq (\frac{4}{3}|a|)^{1/2}$ из упр. 3.3.4–9. Поэтому $(zu+mv)^2-au^2 = mt$, где $t^2 \leq \frac{4}{3}|a|$. Уравнение $x^2-ay^2 = (ms)(mt)$ решается на основании тождества из указания.

Следствие 4. Уравнение $x^2 - y^2 \equiv b$ (по модулю n) решается просто, так как можно положить, что $x = (b+1)/2$, $y = (b-1)/2$.

Следствие 5. Нетрудно решить и уравнение $x^2 + y^2 \equiv b$ (по модулю n), потому что при помощи рассмотренного в упр. 3.3.4–11 метода можно решить уравнение $x^2 + y^2 = p$ для p простого и $p \bmod 4 = 1$; таким простым числом будет одно из чисел $b, b+n, b+2n, \ldots$.

Теперь решать сформулированную задачу для случая, когда $|a| > 1$, можно следующим образом. Сначала выберем числа u и v как случайные в интервале между 1 и $n-1$, затем вычислим значения $w = (u^2 - av^2) \bmod n$ и $d = \gcd(w, n)$. Если $1 < d < n$ или $\gcd(v, n) > 1$, то можно уменьшить n; методы, используемые для доказательства следствия 1, переведут решения, полученные для множителей числа n, в решения для самих чисел n. Если $d = n$ и $v \perp n$, то $(u/v)^2 \equiv a$ (по модулю n); значит, можно уменьшить a на 1. В противном случае $d = 1$; положим, что $s = bw \bmod n$. Это число s согласно следствию 2 равномерно распределено между простыми числами до n. Если $(\frac{a}{s}) = 1$, попытаемся найти решение уравнения $z^2 \equiv a$ (по модулю s), полагая, что s — простое число (упр. 4.6.2–15). Если решение не удалось найти, предпринимаем вторую попытку при другом выборе случайных чисел u и v. Если же решение найдено, полагаем, что $z^2 = a + ms$, и вычисляем $d = \gcd(ms, n)$. Если $d > 1$, то упрощаем задачу в соответствии с описанной выше процедурой. В противном случае используем следствие 3 для того, чтобы найти $x^2 - ay^2 = m^2 st$ при $t^2 \leq \frac{4}{3}|a|$; это приводит к уравнению $(x/m)^2 - a(y/m)^2 \equiv st$ (по модулю n). Если $t = 0$, то уменьшаем a на 1. В противном случае для решения уравнения $X^2 - tY^2 \equiv a$ (по модулю n) применяем этот алгоритм рекурсивно. (Так как t значительно меньше, чем a, то потребуется только $O(\log \log n)$ уровней рекурсии.) Если $\gcd(Y, n) > 1$, можно уменьшить параметры n и a; в противном случае получим $(X/Y)^2 - a(1/Y)^2 \equiv t$ (по модулю n). В итоге указанное тождество дает решение уравнения $x'^2 - ay'^2 \equiv s$ (см. следствие 2), которое приводит к искомому решению, так как $u^2 - av^2 \equiv s/b$.

На практике, как правило, требуется только $O(\log n)$ случайных пробных чисел для того, чтобы гарантировать успешное выполнение этого алгоритма в соответствии со сделанным предположением. Но для формального доказательства потребуется принять во внимание расширенную гипотезу Римана [*IEEE Trans.* **IT-33** (1987), 702–709]. Более сложный и медленный алгоритм, который не основывается ни на одном из недоказанных предположений, предложили Эдлеман (Adleman), Истес (Estes) и Мак-Керли (McCurley) [*Math. Comp.* **48** (1987), 17–28].

46. [*FOCS* **20** (1979), 55–60.] После того как числа $a^{n_i} \bmod p = \prod_{j=1}^m p_j^{e_{ij}}$ получены для достаточного количества n_i, можно найти целочисленное решение x_{ijk}, t_{jk}, $1 \leq j, k \leq m$ уравнения $\sum_i x_{ijk} e_{ij} + (p-1)t_{jk} = \delta_{jk}$ (например, как для уравнения 4.5.2–(23)), подставляя известные решения $N_j = (\sum_i x_{ijk} e_{jk}) \bmod (p-1)$ в $a^{N_j} \bmod p = p_j$. Тогда, если $ba^{n'} \bmod p = \prod_{j=1}^m p_j^{e'_j}$, получим $n + n' \equiv \sum_{j=1}^m e'_j N_j$ (по модулю $p-1$). [Известны усовершенствованные алгоритмы (см., например, Coppersmith, Odlyzko, Schroeppel, *Algorithmica* **1** (1986), 1–15).]

РАЗДЕЛ 4.6

1. $9x^2 + 7x + 7$; $5x^3 + 7x^2 + 2x + 6$.

2. (a) Истинно. (b) Ложно, если алгебраическая система S содержит *делители нуля*, т. е. ненулевые числа, произведение которых равно нулю, как в упр. 1; в противном случае — истинно. (c) Истинно при $m \neq n$, но, вообще говоря, ложно при $m = n$, так как старшие коэффициенты могут сократиться.

3. Положим, что $r \leq s$. При $0 \leq k \leq r$ максимум равен $m_1 m_2(k+1)$, при $r \leq k \leq s$ он равен $m_1 m_2(r+1)$ и при $s \leq k \leq r+s$ равен $m_1 m_2(r+s+1-k)$. Наименьшая верхняя граница, справедливая для всех k, равна $m_1 m_2(r+1)$. (Тот, кто решил эту задачу, теперь знает, как разделить на множители полином $x^7 + 2x^6 + 3x^5 + 3x^4 + 3x^3 + 3x^2 + 2x + 1$.)

4. Если один из полиномов имеет меньше 2^t ненулевых коэффициентов, произведение можно найти, поместив ровно $t-1$ нулей между всеми коэффициентами, а затем выполнив умножение в двоичной системе счисления и использовав побитовую операцию AND (которая имеется в большинстве двоичных компьютеров; см. алгоритм 4.5.4D) для обнуления лишних битов. Например, при $t = 3$ умножение, приведенное в тексте раздела, будет иметь вид $(1001000001)_2 \times (1000001001)_2 = (1001001011001001001)_2$. Искомый ответ может быть получен с помощью побитовой операции AND с константой $(1001001\ldots1001)_2$. Подобная методика применима для умножения полиномов с не очень большими неотрицательными коэффициентами.

5. Полиномы степени $\leq 2n$ могут быть записаны как $U_1(x)x^n + U_0(x)$, где $\deg(U_1) \leq n$ и $\deg(U_0) \leq n$, и $(U_1(x)x^n + U_0(x))(V_1(x)x^n + V_0(x)) = U_1(x)V_1(x)(x^{2n} + x^n) + (U_1(x) + U_0(x))(V_1(x) + V_0(x))x^n + U_0(x)V_0(x)(x^n + 1)$. (В уравнении предполагается, что арифметические действия выполняются по модулю 2.) Таким образом, выполняются соотношения 4.3.3–(3) и 4.3.3–(5).

Примечание. С. А. Кук (S. A. Cook) показал, что алгоритм 4.3.3Т можно расширить таким же способом. А. Шёнхаге (A. Schönhage) [*Acta Informatica* **7** (1977), 395–398] показал, как можно умножить полиномы по модулю 2 при помощи $O(n \log n \log\log n)$ битовых операций. В действительности полиномы над любым кольцом S могут быть умножены с помощью $O(n \log n \log\log n)$ алгебраических операций, даже когда S представляет собой алгебраическую систему, в которой умножение может не быть коммутативным или ассоциативным [D. G. Cantor and E. Kaltofen, *Acta Informatica* **28** (1991), 693–701]. См. также упр. 4.6.4–57 и 4.6.4–58. Однако эти идеи практически бесполезны для разреженных полиномов, большинство коэффициентов которых нулевые.

РАЗДЕЛ 4.6.1

1. $q(x) = 1 \cdot 2^3 x^3 + 0 \cdot 2^2 x^2 - 2 \cdot 2x + 8 = 8x^3 - 4x + 8$; $r(x) = 28x^2 + 4x + 8$.

2. Последовательность нормированных полиномов, полученная при работе алгоритма Евклида, имеет коэффициенты $(1,5,6,6,1,6,3)$, $(1,2,5,2,2,4,5)$, $(1,5,6,2,3,4)$, $(1,3,4,6)$, 0. Следовательно, наибольший общий делитель равен $x^3 + 3x^2 + 4x + 6$. (Наибольший общий делитель полинома и "обратного" к нему всегда симметричен в том смысле, что он равен своему "обратному", умноженному на обратимый элемент.)

3. Алгоритм 4.5.2X остается корректным при замене целых чисел полиномами над S. По завершении алгоритма мы получим $U(x) = u_2(x)$, $V(x) = u_1(x)$. Пусть $m = \deg(u)$, $n = \deg(v)$. По индукции легко доказать, что после шага X3 в процессе выполнения алгоритма $\deg(u_3) + \deg(v_1) = n$, $\deg(u_3) + \deg(v_2) = m$ при условии, что $m \geq n$. Следовательно, если m и n больше, чем $d = \deg(\gcd(u,v))$, то $\deg(U) < m-d$, $\deg(V) < n-d$; точные степени равны $m-d_1$ и $n-d_1$, где d_1 — степень предпоследнего ненулевого остатка. При $d = \min(m,n)$, скажем, $d = n$, имеем $U(x) = 0$ и $V(x) = 1$.

При $u(x) = x^m - 1$ и $v(x) = x^n - 1$ тождество $(x^m - 1) \bmod (x^n - 1) = x^{m \bmod n} - 1$ показывает, что все полиномы, образующиеся во время вычислений, нормированы и имеют целые коэффициенты. При $u(x) = x^{21} - 1$ и $v(x) = x^{13} - 1$ имеем $V(x) = x^{11} + x^8 + x^6 + x^3 + 1$ и $U(x) = -(x^{19} + x^{16} + x^{14} + x^{11} + x^8 + x^6 + x^3 + x)$. (См. также равенство 3.3.3–(29), дающее альтернативную формулу для $U(x)$ и $V(x)$, а также упр. 4.3.2–6, в котором 2 заменено на x.)

4. Поскольку частное $q(x)$ зависит только от $v(x)$ и первых $m - n$ коэффициентов $u(x)$, остаток $r(x) = u(x) - q(x)v(x)$ равномерно распределен и независим от $v(x)$. Следовательно, каждый шаг алгоритма может рассматриваться как независимый от других. Этот алгоритм ведет себя существенно лучше, чем алгоритм Евклида над целыми числами.

Вероятность того, что $n_1 = n - k$, равна $p^{1-k}(1 - 1/p)$, и $t = 0$ с вероятностью p^{-n}. Каждый последующий шаг, по существу, ведет себя так же, поэтому любая данная последовательность степеней n, n_1, ..., n_t, $-\infty$ появляется с вероятностью $(p-1)^t/p^n$. Чтобы найти среднее значение $f(n_1, \ldots, n_t)$, введем обозначение S_t для суммы $f(n_1, \ldots, n_t)$ по всем последовательностям $n > n_1 > \cdots > n_t \geq 0$, имеющим данное значение t; тогда среднее значение составляет $\sum_t S_t(p-1)^t/p^n$.

Пусть $f(n_1, \ldots, n_t) = t$; тогда $S_t = \binom{n}{t} t$, так что среднее равно $n(1 - 1/p)$. Точно так же при $f(n_1, \ldots, n_t) = n_1 + \cdots + n_t$ получим $S_t = \binom{n}{2}\binom{n-1}{t-1}$ и среднее значение $\binom{n}{2}(1 - 1/p)$. И наконец, для $f(n_1, \ldots, n_t) = (n - n_1)n_1 + \cdots + (n_{t-1} - n_t)n_t$

$$S_t = \binom{n+2}{t+2} - (n+1)\binom{n+1}{t+1} + \binom{n+1}{2}\binom{n}{t}$$

и среднее значение составляет $\binom{n+1}{2} - (n+1)p/(p-1) + (p/(p-1))^2(1 - 1/p^{n+1})$.

(Вероятность того, что $n_{j+1} = n_j - 1$ для $1 \leq j \leq t = n$ равна $(1 - 1/p)^n$, получается, если выбрать $S_t = [t = n]$; эта вероятность стремится к 1 при $p \to \infty$. Как следствие имеем дополнительные основания утверждать, что алгоритм C всегда дает $\delta_2 = \delta_3 = \cdots = 1$, поскольку любые полиномы, не удовлетворяющие последнему условию, не будут удовлетворять и прежнему условию по модулю p при любом p.)

5. Используя формулы из упр. 4, при $f(n_1, \ldots, n_t) = [n_t = 0]$ найдем, что вероятность равна $1 - 1/p$ при $n > 0$ и 1 при $n = 0$.

6. Полагая, что постоянные члены $u(0)$ и $v(0)$ ненулевые, представим себе алгоритм деления "справа налево", $u(x) = v(x)q(x) + x^{m-n}r(x)$, где $\deg(r) < \deg(v)$. Получим алгоритм поиска gcd, аналогичный алгоритму 4.5.2B, который, по сути, представляет собой алгоритм Евклида, приложимый к "обратному" полиному (в смысле упр. 2). Впоследствии ответ обращается и умножается на подходящую степень x.

Существует подобный алгоритм, аналогичный методу из упр. 4.5.2–40. Среднее количество итераций для обоих алгоритмов найдено в работах G. H. Norton, *SICOMP* **18** (1989), 608–624; K. Ma and J. von zur Gathen, *J. Symbolic Comp.* **9** (1990), 429–455.

7. Обратимым элементам S (в качестве полиномов нулевой степени).

8. Если $u(x) = v(x)w(x)$, где $u(x)$ имеет целые коэффициенты, а $v(x)$ и $w(x)$ — рациональные коэффициенты, существуют ненулевые целые m и n, такие, что $m \cdot v(x)$ и $n \cdot w(x)$ имеют целые коэффициенты. $u(x)$ примитивен, так что (4) означает следующее:

$$u(x) = \operatorname{pp}((m \cdot v(x))(n \cdot w(x))) = \pm \operatorname{pp}(m \cdot v(x)) \operatorname{pp}(n \cdot w(x)).$$

9. Алгоритм E можно расширить следующим образом. Пусть $(u_1(x), u_2(x), u_3, u_4(x))$ и $(v_1(x), v_2(x), v_3, v_4(x))$ представляют собой четверки, удовлетворяющие соотношениям $u_1(x)u(x) + u_2(x)v(x) = u_3u_4(x)$ и $v_1(x)u(x) + v_2(x)v(x) = v_3v_4(x)$. Расширенный алгоритм начинается с четверок $(1, 0, \operatorname{cont}(u), \operatorname{pp}(u(x)))$ и $(0, 1, \operatorname{cont}(v), \operatorname{pp}(v(x)))$ и работает с ними таким образом, чтобы соблюсти указанные выше условия, где $u_4(x)$ и $v_4(x)$ проходят по той же последовательности, что и $u(x)$ и $v(x)$ в алгоритме E. Если $au_4(x) = q(x)v_4(x) + br(x)$, то

$av_3(u_1(x),u_2(x))-q(x)u_3(v_1(x),v_2(x))=(r_1(x),r_2(x))$, где $r_1(x)u(x)+r_2(x)v(x)=bu_3v_3r(x)$, так что расширенный алгоритм может сохранить требуемые соотношения. Если $u(x)$ и $v(x)$ взаимно просты, то расширенный алгоритм в конечном счете находит $r(x)$ нулевой степени и мы получаем $U(x)=r_2(x)$, $V(x)=r_1(x)$, как и требовалось. (На практике можно разделить $r_1(x)$, $r_2(x)$ и bu_3v_3 на $\gcd(\mathrm{cont}(r_1),\mathrm{cont}(r_2))$.) Обратно, если такие $U(x)$ и $V(x)$ существуют, то $u(x)$ и $v(x)$ не имеют общих простых делителей, поскольку они примитивны и не имеют общих делителей положительной степени.

10. С помощью последовательного разложения приводимых полиномов на полиномы меньших степеней мы должны получить конечное разложение любого полинома на неприводимые. Разложение *содержимого* единственно. Чтобы показать, что существует не более одного разложения на примитивные части, необходимо доказать, что если $u(x)$ — неприводимый делитель произведения $v(x)w(x)$, но не произведение обратимого элемента и неприводимого полинома $v(x)$, то $u(x)$ является делителем $w(x)$. Это можно доказать, обратив внимание на то, что $u(x)$ представляет собой делитель $v(x)w(x)U(x)=rw(x)-w(x)u(x)V(x)$ согласно результату упр. 9, где r — ненулевая постоянная.

11. Потребовались бы только строки A_1, A_0, B_4, B_3, B_2, B_1, B_0, C_1, C_0, D_0. В общем, пусть $u_{j+2}(x)=0$. Тогда строки, необходимые для доказательства, — от $A_{n_2-n_j}$ до A_0, от $B_{n_1-n_j}$ до B_0, от $C_{n_2-n_j}$ до C_0, от $D_{n_3-n_j}$ до D_0 и т. д.

12. Если $n_k=0$, доказательство, приведенное в тексте для (24), показывает, что значение детерминанта составляет $\pm h_k$, что равно $\pm\ell_k^{n_{k-1}}/\prod_{1<j<k}\ell_j^{\delta_{j-1}(\delta_j-1)}$. Если полиномы имеют множитель положительной степени, можно искусственно положить, что нулевой полином имеет нулевую степень, и использовать ту же формулу с $\ell_k=0$.

Примечание. Значение детерминанта Сильвестра $R(u,v)$ называется *результантом* u и v, а величина $(-1)^{\deg(u)(\deg(u)-1)/2}\ell(u)^{-1}R(u,u')$ — *дискриминантом* u, где u' — производная u. Если $u(x)$ имеет разложение вида $a(x-\alpha_1)\dots(x-\alpha_m)$ и если $v(x)=b(x-\beta_1)\dots(x-\beta_n)$, то результант $R(u,v)$ равен $a^nv(\alpha_1)\dots v(\alpha_m)=(-1)^{mn}b^mu(\beta_1)\dots u(\beta_n)=a^nb^m\prod_{i=1}^m\prod_{j=1}^n(\alpha_i-\beta_j)$. Отсюда следует, что полиномы степени mn от y определены как соответствующие результанты $u(y-x)$, $u(y+x)$, $x^mu(y/x)$ и $u(yx)$ с $v(x)$, имеющие в качестве соответствующих корней суммы $\alpha_i+\beta_j$, разности $\alpha_i-\beta_j$, произведения $\alpha_i\beta_j$ и частные α_i/β_j (при $v(0)\neq 0$). Эта идея была использована Р. Г. К. Лоосом (R. G. K. Loos) для создания алгоритмов арифметики алгебраических чисел [*Computing*, Supplement **4** (1982), 173–187].

Если каждую строку A_i в матрице Сильвестра заменить строкой на

$$(b_0A_i+b_1A_{i+1}+\cdots+b_{n_2-1-i}A_{n_2-1})-(a_0B_i+a_1B_{i+1}+\cdots+a_{n_2-1-i}B_{n_2-1}),$$

а затем удалить строки с B_{n_2-1} по B_0 и последние n_2 столбцов, то получим детерминант размера $n_1\times n_1$ в качестве результанта вместо исходного определителя размера $(n_1+n_2)\times(n_1+n_2)$. В некоторых случаях результант может быть эффективно вычислен по значению этого детерминанта [см. *CACM* **12** (1969), 23–30, 302–303].

Я. Т. Шварц (J. T. Schwartz) показал, что можно вычислить результанты и последовательности Штурма для полиномов степени n с помощью всего $O(n(\log n)^2)$ арифметических операций при $n\to\infty$ [см. *JACM* **27** (1980), 701–717].

13. С помощью индукции по j можно показать, что значения $(u_{j+1}(x),g_{j+1},h_j)$ замещены соответственно значениями $(\ell^{1+p_j}w(x)u_j(x),\ell^{2+p_j}g_j,\ell^{p_j}h_j)$ при $j\geq 2$, где $p_j=n_1+n_2-2n_j$. [Несмотря на этот рост грани (26) остаются справедливыми.]

14. Пусть p — простой элемент из данной области и пусть j,k — максимум, такой, что $p^k\backslash v_n=\ell(v)$, $p^j\backslash v_{n-1}$. Пусть $P=p^k$. Согласно алгоритму R можем записать $q(x)=a_0+Pa_1x+\cdots+P^sa_sx^s$, где $s=m-n\geq 2$. Рассмотрим коэффициенты при x^{n+1}, x^n и x^{n-1} в $v(x)q(x)$, а именно — $Pa_1v_n+P^2a_2v_{n-1}+\cdots$, $a_0v_n+Pa_1v_{n-1}+\cdots$ и $a_0v_{n-1}+Pa_1v_{n-2}+\cdots$,

каждый из которых кратен P^3. Из первого делаем вывод, что $p^j \backslash a_1$, из второго — что $p^{\min(k,2j)} \backslash a_0$, а из третьего — что $P \backslash a_0$. Следовательно, $P \backslash r(x)$. [Если $m = n+1$, лучшее, что можно доказать, — это то, что $p^{\lceil k/2 \rceil}$ делит $r(x)$. Например, взгляните на $u(x) = x^3+1$, $v(x) = 4x^2+2x+1$, $r(x) = 18$. С другой стороны, можно воспользоваться аргументом, основанным на детерминантах матриц наподобие (21) и (22) для того, чтобы показать, что $\ell(r)^{\deg(v)-\deg(r)-1} r(x)$ всегда кратно $\ell(v)^{(\deg(u)-\deg(v))(\deg(v)-\deg(r)-1)}$.]

15. Пусть $c_{ij} = a_{i1}a_{j1} + \cdots + a_{in}a_{jn}$. Можно положить, что $c_{ii} > 0$ при всех i. Если $c_{ij} \neq 0$ для некоторого $i \neq j$, то можно заменить строку i и столбец i на $(c_{i1} - tc_{j1}, \ldots, c_{in} - tc_{jn})$, где $t = c_{ij}/c_{jj}$; эта операция не изменяет значение $\det C$ и уменьшает значение верхней грани, которое мы доказываем, поскольку c_{ii} заменяется на $c_{ii} - c_{ij}^2/c_{jj}$. Такое замещение можно производить систематично для увеличивающегося i и $j < i$, пока не будет достигнуто $c_{ij} = 0$ для всех $i \neq j$. [Этот алгоритм называется *ортогонализацией Грама-Шмидта* (см. *Crelle* **94** (1883), 41–73; *Math. Annalen* **63** (1907), 442).] Тогда $\det(A)^2 = \det(AA^T) = c_{11} \ldots c_{nn}$.

16. Полином от одной переменной степени d над любой областью единственного разложения имеет не более d корней (см. упр. 3.2.1.2–16(b)); так что если $n = 1$, то ясно, что $|r(S_1)| \leq d_1$. При $n > 1$ имеем $f(x_1, \ldots, x_n) = g_0(x_2, \ldots, x_n) + x_1 g_1(x_2, \ldots, x_n) + \cdots + x_1^{d_1} g_{d_1}(x_2, \ldots, x_n)$, где g_k ненулевое минимум для одного k. Для данных $(x_2, \ldots, x_n)$ следует, что $f(x_1, \ldots, x_n)$ равна нулю не более чем при d_1 значениях x_1, кроме случая, когда $g_k(x_2, \ldots, x_n) = 0$. Следовательно, $|r(S_1, \ldots, S_n)| \leq d_1(|S_2| - d_2) \ldots (|S_n| - d_n) + |S_1|(|S_2| \ldots |S_n| - (|S_2| - d_2) \ldots (|S_n| - d_n))$. [R. A. DeMillo and R. J. Lipton, *Inf. Proc. Letters* **7** (1978), 193–195.]

Примечание. Указанная верхняя грань — наилучшая из возможных, так как для полинома $f(x_1, \ldots, x_n) = \prod\{x_j - s_k \mid s_k \in S_j,\ 1 \leq k \leq d_j,\ 1 \leq j \leq n\}$ достигается равенство. Однако существует другой подход, при котором верхняя грань может быть значительно улучшена, хотя и в другом смысле. Пусть $f_1(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ и пусть $f_{j+1}(x_{j+1}, \ldots, x_n)$ — любой ненулевой коэффициент при степени x_j в $f_j(x_j, \ldots, x_n)$. Тогда можем положить d_j равным степени при x_j в f_j вместо (зачастую существенно большей) степени x_j в f. Например, можем положить $d_1 = 3$ и $d_2 = 1$ в полиноме $x_1^3 x_2^9 - 3x_1^2 x_2 + x_2^{100} + 5$. Это наблюдение гарантирует, что $d_1 + \cdots + d_n \leq d$, когда каждый член f имеет общую степень $\leq d$. Следовательно, вероятность в этих случаях равна

$$\frac{|r(S, \ldots, S)|}{|S|} \leq 1 - \left(1 - \frac{d_1}{|S|}\right) \ldots \left(1 - \frac{d_n}{|S|}\right) \leq \frac{d_1 + \cdots + d_n}{|S|} \leq \frac{d}{|S|}$$

при равенстве всех множеств S_j. Если она $\leq \frac{1}{2}$ и если $f(x_1, \ldots, x_n)$ становится равным нулю для 50 случайно выбранных векторов $(x_1, \ldots, x_n)$, то $f(x_1, \ldots, x_n)$ тождественно равна нулю с вероятностью как минимум $1 - 2^{-50}$.

Более того, если $f_j(x_j, \ldots, x_n)$ имеет специальный вид $x_j^{e_j} f_{j+1}(x_{j+1}, \ldots, x_n)$ с $e_j > 0$, можно получить $d_j = 1$, так как x_j должно быть равно 0, когда $f_{j+1}(x_{j+1}, \ldots, x_n) \neq 0$. Поэтому разреженные полиномы с только m ненулевыми членами будут иметь $d_j \leq 1$ для минимум $n - \lg m$ значений j.

Применения этого неравенства для вычисления gcd и других операций над разреженными полиномами от многих переменных были введены Р. Циппелем (R. Zippel) (*Lecture Notes in Comp. Sci.* **72** (1979), 216–226). Я. Т. Шварц (J. T. Schwartz) [*JACM* **27** (1980), 701–717] привел дальнейшие расширения, включая устранение больших чисел с помощью модулярной арифметики: если коэффициенты f целые, если P является множеством целых чисел $\geq q$ и если $|f(x_1, \ldots, x_n)| \leq L$ для каждого $x_j \in S_j$, то количество решений $f(x_1, \ldots, x_n) \equiv 0 \pmod{p}$ для $p \in P$ не превышает

$$|S_1| \ldots |S_n||P| - (|S_1| - d_1) \ldots (|S_n| - d_n)(|P| - \log_q L).$$

17. (a) Для удобства опишем алгоритм только для $A = \{a, b\}$. Из наших посылок следует, что $\deg(Q_1U) = \deg(Q_2V) \geq 0$, $\deg(Q_1) \leq \deg(Q_2)$. Если $\deg(Q_1) = 0$, то Q_1 представляет собой просто ненулевое рациональное число, так что мы считаем $Q = Q_2/Q_1$. В противном случае пусть $Q_1 = aQ_{11} + bQ_{12} + r_1$, $Q_2 = aQ_{21} + bQ_{22} + r_2$, где r_1 и r_2 — рациональные числа; отсюда следует, что

$$Q_1U - Q_2V = a(Q_{11}U - Q_{21}V) + b(Q_{12}U - Q_{22}V) + r_1U - r_2V.$$

У нас должно быть либо $\deg(Q_{11}) = \deg(Q_1) - 1$, либо $\deg(Q_{12}) = \deg(Q_1) - 1$. В первом случае, рассматривая члены высшей степени, которые начинаются с a, имеем $\deg(Q_{11}U - Q_{21}V) < \deg(Q_{11}U)$; так что можем заменить Q_1 на Q_{11}, Q_2 на Q_{21} и повторить процесс. Подобным образом в последнем случае можно заменить (Q_1, Q_2) на (Q_{12}, Q_{22}) и повторить процедуру.

(b) Положим, что $\deg(U) \geq \deg(V)$. Если $\deg(R) \geq \deg(V)$, то заметьте, что $Q_1U - Q_2V = Q_1R - (Q_2 - Q_1Q)V$ имеет степень, меньшую, чем $\deg(V) \leq \deg(Q_1R)$, так что можно повторить процесс с U, замененным на R. Получим $R = Q'V + R'$, $U = (Q + Q')V + R'$, где $\deg(R') < \deg(R)$, так что в конце концов решение будет получено.

(c) Алгоритм из (b) дает $V_1 = UV_2 + R$, $\deg(R) < \deg(V_2)$; в силу однородности $R = 0$ и U однородно.

(d) Мы можем положить, что $\deg(V) \leq \deg(U)$. Если $\deg(V) = 0$, установить $W \leftarrow U$; в противном случае воспользуемся результатом (c) для поиска $U = QV$, так что $QVV = VQV$, $(QV - VQ)V = 0$. Отсюда следует, что $QV = VQ$, так что можно установить $U \leftarrow V$, $V \leftarrow Q$ и повторить процесс.

Более подробная информация о рассматриваемом вопросе приводится в работе P. M. Cohn, *Proc. Cambridge Phil. Soc.* **57** (1961), 18–30. Существенно более сложная задача описания *всех* строковых полиномов, таких, что $UV = VU$, была решена Дж. М. Бергманом (G. M. Bergman) [Ph. D. thesis, Harvard University, 1967].

18. [P. M. Cohn, *Transactions of the Amer. Math. Soc.* **109** (1963), 332–356.]

C1. Установить $u_1 \leftarrow U_1$, $u_2 \leftarrow U_2$, $v_1 \leftarrow V_1$, $v_2 \leftarrow V_2$, $z_1 \leftarrow z_2' \leftarrow w_1 \leftarrow w_2' \leftarrow 1$, $z_1' \leftarrow z_2 \leftarrow w_1' \leftarrow w_2 \leftarrow 0$, $n \leftarrow 0$.

C2. (В этот момент выполняются данные в условии упражнения тождества и $u_1v_1 = u_1v_2$; $v_2 = 0$ тогда и только тогда, когда $u_1 = 0$.) Если $v_2 = 0$, алгоритм завершается с $\text{НОПД}(V_1, V_2) = v_1$, $\text{НОЛК}(V_1, V_2) = z_1'V_1 = -z_2'V_2$. (Кроме того, в силу симметрии имеем $\text{НОЛД}(U_1, U_2) = u_2$ и $\text{НОПК}(U_1, U_2) = U_1w_1 = -U_2w_2$.)

C3. Найти Q и R, такие, что $v_1 = Qv_2 + R$, где $\deg(R) < \deg(v_2)$. (Имеем $u_1(Qv_2 + R) = u_2v_2$, так что $u_1R = (u_2 - u_1Q)v_2 = R'v_2$.)

C4. Установить $(w_1, w_2, w_1', w_2', z_1, z_2, z_1', z_2', u_1, u_2, v_1, v_2) \leftarrow (w_1' - w_1Q, w_2' - w_2Q, w_1, w_2, z_1', z_2', z_1 - Qz_1', z_2 - Qz_2', u_2 - u_1Q, u_1, v_2, v_1 - Qv_2)$ и $n \leftarrow n + 1$. Вернуться к шагу C2. ▮

Это расширение алгоритма Евклида включает большинство особенностей, которые встречались в предыдущих расширениях алгоритма Евклида. Это приводит нас к новому пониманию уже рассмотренных частных случаев. Для доказательства корректности алгоритма сначала заметим, что $\deg(v_2)$ уменьшается на шаге C4, так что алгоритм обязательно завершается. По завершении алгоритма v_1 представляет собой общий правый делитель V_1 и V_2, поскольку $w_1v_1 = (-1)^nV_1$ и $-w_2v_1 = (-1)^nV_2$. Также, если d является любым общим правым делителем V_1 и V_2, он является и правым делителем $z_1V_1 + z_2V_2 = v_1$. Следовательно, $v_1 = \text{НОПД}(V_1, V_2)$. Кроме того, если m является любым общим левым кратным V_1 и V_2, без потери общности можно считать, что $m = U_1V_1 = U_2V_2$, поскольку последовательность значений Q не зависит от U_1 и U_2. Значит, $m = (-1)^n(-u_2z_1')V_1 = (-1)^n(u_2z_2')V_2$ кратно $z_1'V_1$.

На практике, чтобы вычислить только НОПД(V_1, V_2), можно опустить вычисление n, w_1, w_2, w_1', w_2', z_1, z_2, z_1', z_2'. Эти дополнительные величины были добавлены к алгоритму, в первую очередь, чтобы более просто установить его корректность.

Примечание. Нетривиальное разложение строковых полиномов, таких, как приведенные в качестве примера в этом упражнении, могут быть найдены из матричных тождеств наподобие

$$\begin{pmatrix} a & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} b & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} c & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -c \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -b \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -a \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

поскольку эти тождества выполняются даже при некоммутативности умножения, например

$$(abc + a + c)(1 + ba) = (ab + 1)(cba + a + c).$$

(Сравните это с континуантами из раздела 4.5.3.)

19. [См. Eugène Cahen, *Théorie des Nombres* **1** (Paris, 1914), 336–338.] Если такой алгоритм существует, то в соответствии с рассуждениями из упр. 18 D является НОПД. Рассмотрим A и B как единую матрицу C размера $2n \times n$, первые n строк которой представлены строками A, а следующие n строк — строками B. Точно так же можно комбинировать матрицы P и Q в матрицу R размера $2n \times n$, а X и Y — в матрицу Z размера $n \times 2n$. Требуемые условия теперь сводятся к двум уравнениям: $C = RD$, $D = ZC$. Если можно найти целочисленную матрицу U размера $2n \times 2n$ с детерминантом ± 1, такую, что все последние n строк $U^{-1}C$ нулевые, то решением будут матрицы $R =$ (первые n столбцов U), $D =$ (первые n строк $U^{-1}C$), $Z =$ (первые n строк U^{-1}). Поэтому можно использовать, например, следующий алгоритм (с $m = 2n$).

Алгоритм T (*Триангуляризация*). Пусть C — целочисленная матрица размера $m \times n$. Данный алгоритм находит целочисленные матрицы U и V размера $m \times m$ такие, что $UV = I$ и VC представляет собой *треугольную* матрицу (такую, что элемент на пересечении i-й строки и j-го столбца матрицы VC равен нулю, если $i > j$).

T1. [Инициализация.] Установить $U \leftarrow V \leftarrow I$ (I — единичная матрица размера $m \times m$) и $T \leftarrow C$. (На протяжении всего алгоритма будут выполняться соотношения $T = VC$ и $UV = 1$.)

T2. [Итерация по j.] Выполнить шаг T3 для $j = 1, 2, \ldots, \min(m, n)$; затем прекратить выполнение алгоритма.

T3. [Обнуление столбца j.] Не выполнять следующие действия или выполнять их до тех пор, пока T_{ij} не станет равным нулю для всех $i > j$. Пусть T_{kj} — ненулевой элемент $\{T_{ij}, T_{(j+1)j}, \ldots, T_{mj}\}$, имеющий наименьшее абсолютное значение. Переставим строки k и j матриц T и V и переставим столбцы k и j матрицы U. Затем вычтем строку j из строки i $\lfloor T_{ij}/T_{jj} \rfloor$ раз в матрицах T и V и прибавим столько же раз столбец i к столбцу j в матрице U для $j < i \le m$. ▮

Для приведенного в упражнении примера алгоритм дает $\left(\begin{smallmatrix} 1 & 2 \\ 3 & 4 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 1 & 0 \\ 3 & 2 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 2 \\ 0 & -1 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 4 & 3 \\ 2 & 1 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 4 & 5 \\ 2 & 3 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 2 \\ 0 & -1 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 1 & 2 \\ 0 & -1 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 1 & 0 \\ 2 & -2 \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & 2 \\ 3 & 4 \end{smallmatrix}\right) + \left(\begin{smallmatrix} 0 & 0 \\ 1 & 0 \end{smallmatrix}\right)\left(\begin{smallmatrix} 4 & 3 \\ 2 & 1 \end{smallmatrix}\right)$. (На самом деле в этом частном случае НОПД будет являться *любая* матрица с детерминантом ± 1.)

20. Рассмотрите построение из упр. 4.6.2–22, но такое, что p^m заменено малым числом ϵ.

21. Для получения верхней грани положим, что алгоритм R используется только при $m - n \le 1$ и коэффициенты ограничены согласно (26) с $m = n$. [Указанная формула дает практическое время вычисления, а не только верхнюю грань. Более подробную информацию можно найти в работе G. E. Collins, *Proc. 1968 Summer Inst. on Symbolic Mathematical Computation*, edited by Robert G. Tobey (IBM Federal Systems Center, June, 1969), 195–231.]

22. Последовательность знаков не может содержать два идущих подряд нуля, поскольку $u_{k+1}(x)$ — ненулевая константа в (29). Кроме того, в качестве подпоследовательности не может встретиться "+, 0, +" или "−, 0, −". Формула $V(u,a)-V(u,b)$, очевидно, верна при $b=a$, так что ее необходимо проверить только при увеличивающемся b. Полиномы $u_j(x)$ имеют конечное количество корней, и $V(u,b)$ изменяется только тогда, когда b встречается или проходит через такие корни. Пусть x — корень некоторого (возможно, нескольких) u_j. Когда b увеличивается от $x-\epsilon$ до x, знаковая последовательность около j переходит от "+, ±, −" к "+, 0, −" или от "−, ±, +" к "−, 0, +" при $j>0$; при $j=0$ — от "+, −" к "0, −" или от "−, +" к "0, +" (поскольку $u'(x)$ — производная, $u'(x)$ отрицательна при уменьшении $u(x)$). Таким образом, изменение V составляет $-\delta_{j0}$. При возрастании b от x до $x+\epsilon$ подобное рассмотрение показывает, что V остается неизменным.

[Л. Э. Хайндел (L. E. Heindel) в работе *JACM* **18** (1971), 533–548, применил эти идеи для построения алгоритма обособления действительных нулей данного полинома $u(x)$ за время, растущее как полином от $\deg(u)$ и $\log N$, где все коэффициенты u_j — целые числа $|u_j|\le N$, а все операции абсолютно точны.]

23. Если v имеет $n-1$ действительный корень между n действительными корнями u, то, рассмотрев изменения знаков, получим, что $u(x) \bmod v(x)$ имеет $n-2$ действительных корня, лежащих между $n-1$ корнем v.

24. Сначала покажите, что $h_j = g_j^{\delta_{j-1}} g_{j-1}^{\delta_{j-2}(1-\delta_{j-1})} \ldots g_2^{\delta_1(1-\delta_2)\ldots(1-\delta_{j-1})}$, а затем — что показатель степени g_2 в левой части (18) имеет вид $\delta_2+\delta_1 x$, где $x=\delta_2+\cdots+\delta_{j-1}+1-\delta_2(\delta_3+\cdots+\delta_{j-1}+1)-\delta_3(1-\delta_2)(\delta_4+\cdots+\delta_{j-1}+1)-\cdots-\delta_{j-1}(1-\delta_2)\ldots(1-\delta_{j-2})(1)$. Однако $x=1$, поскольку он не зависит от δ_{j-1}, и можно принять, что $\delta_{j-1}=0$, и т. д. Подобное доказательство работает и для g_3, g_4, ..., а упрощение — для (23).

25. Каждый коэффициент $u_j(x)$ может быть выражен как детерминант, в котором один столбец содержит только $\ell(u)$, $\ell(v)$ и нули. Чтобы использовать этот факт, модифицируем алгоритм C следующим образом. На шаге C1 установим $g\leftarrow\gcd(\ell(u),\ell(v))$ и $h\leftarrow 0$. На шаге C3, если $h=0$, установим $u(x)\leftarrow v(x)$, $v(x)\leftarrow r(x)/g$, $h\leftarrow \ell(u)^\delta/g$, $g\leftarrow\ell(u)$ и вернемся к шагу C2. В противном случае будем работать по немодифицированному алгоритму. Эффект этой новой инициализации заключается в простой замене $u_j(x)$ на $u_j(x)/\gcd(\ell(u),\ell(v))$ для всех $j\ge 3$. Таким образом, в (28) ℓ^{2j-4} становится ℓ^{2j-5}.

26. Фактически верно даже большее. Обратите внимание, что алгоритм из упр. 3 вычисляет $\pm p_n(x)$ и $\mp q_n(x)$ для $n\ge -1$. Пусть $e_n=\deg(q_n)$ и $d_n=\deg(p_nu-q_nv)$. В упр. 3 мы видели, что $d_{n-1}+e_n=\deg(u)$ для $n\ge 0$. Докажем, что условия $\deg(q)<e_n$ и $\deg(pu-qv)<d_{n-2}$ влекут за собой $p(x)=c(x)p_{n-1}(x)$ и $q(x)=c(x)q_{n-1}(x)$. По данным p и q можно найти $c(x)$ и $d(x)$, такие, что $p(x)=c(x)p_{n-1}(x)+d(x)p_n(x)$ и $q(x)=c(x)q_{n-1}(x)+d(x)q_n(x)$, поскольку $p_{n-1}(x)q_n(x)-p_n(x)q_{n-1}(x)=\pm 1$. Следовательно, $pu-qv=c(p_{n-1}u-q_{n-1}v)+d(p_nu-q_nv)$. Если $d(x)\ne 0$, мы должны иметь $\deg(c)+e_{n-1}=\deg(d)+e_n$, так как $\deg(q)<\deg(q_n)$. Отсюда следует, что $\deg(c)+d_{n-1}>\deg(d)+d_n$, поскольку это, несомненно, верно, если $d_n=-\infty$. В противном случае $d_{n-1}+e_n=d_n+e_{n+1}>d_n+e_{n-1}$. Таким образом, $\deg(pu-qv)=\deg(c)+d_{n-1}$. Но мы предположили, что $\deg(pu-qv)<d_{n-2}=d_{n-1}+e_n-e_{n-1}$, так что $\deg(c)<e_n-e_{n-1}$ и $\deg(d)<0$, что приводит к противоречию.

[Этот результат, по сути, получен в работе L. Kronecker, *Monatsberichte Königl. preuß. Akad. Wiss.* (Berlin, 1881), 535–600. Отсюда вытекает следующая теорема. "Пусть $u(x)$ и $v(x)$ — взаимно простые полиномы над полем и пусть $d\le\deg(v)<\deg(u)$. Если $q(x)$ является полиномом наименьшей степени, таким, что существуют полиномы $p(x)$ и $r(x)$, такие, что $p(x)u(x)-q(x)v(x)=r(x)$ и $\deg(r)=d$, то $p(x)/q(x)=p_n(x)/q_n(x)$ для некоторого n". Для $d_{n-2}>d\ge d_{n-1}$ существует решение $q(x)$, такое, что $\deg(q)=e_{n-1}+d-d_{n-1}<e_n$. Итак, все решения столь низкой степени имеют указанное свойство.]

27. Приложимы идеи упр. 4.3.1–40, но в более простой форме, потому что полиномиальная арифметика не оперирует переносами; деление справа налево использует 4.7–(3). Другой путь заключается в том, чтобы при больших значениях n делить преобразования Фурье коэффициентов с "обратным" использованием упр. 4.6.4–57.

РАЗДЕЛ 4.6.2

1. Для любого выбора $k \le n$ различных корней существует p^{n-k} нормированных полиномов, имеющих как минимум по одному из этих корней. Поэтому согласно принципу включения и исключения (см. раздел 1.3.3) количество полиномов без линейных множителей составляет $\sum_{k\le n} \binom{p}{k} p^{n-k}(-1)^k$. Частичные суммы этого ряда попеременно больше или меньше его суммы. Требуемые границы можно получить, положив $n = 2$ и $n = 3$. При $n \ge p$ вероятность наличия хотя бы одного линейного множителя составляет $1 - (1 - 1/p)^p$. Среднее количество линейных множителей в p раз превышает среднее количество случаев, когда величина x делит полином $u(x)$, так что оно составляет $1 + p^{-1} + \cdots + p^{1-n} = \frac{p}{p-1}(1 - p^{-n})$.

[Аналогично находим, что вероятность существования неприводимого множителя степени 2 равна $\sum_{k\le n/2} \binom{p(p-1)/2}{k}(-1)^k p^{-2k}$. Эта вероятность лежит между $\frac{3}{8} - \frac{1}{4}p^{-1}$ и $\frac{1}{2} - \frac{1}{2}p^{-1}$ при $n \ge 2$ и стремится к $1 - e^{-1/2}(1 + \frac{1}{2}p^{-1}) + O(p^{-2})$ при $n \to \infty$. Среднее количество таких множителей равно $\frac{1}{2} - \frac{1}{2}p^{-2\lfloor n/2 \rfloor}$.]

Примечание. Пусть $u(x)$ — фиксированный полином с целыми коэффициентами. Петер Вайнбергер (Peter Weinberger) обнаружил, что если $u(x)$ неприводим над кольцом целых чисел, то среднее количество линейных множителей $u(x)$ по модулю p стремится к 1 при $p \to \infty$, потому что группа Галуа $u(x)$ транзитивна и среднее количество единичных циклов в случайно выбранном элементе любой группы транзитивных перестановок равно 1. Следовательно, *среднее количество линейных множителей $u(x)$ по модулю p равно количеству неприводимых множителей $u(x)$ над кольцом целых чисел* при $p \to \infty$. [См. примечания в ответе к упр. 37, а также *Proc. Symp. Pure Math.* **24** (Amer. Math. Soc., 1972), 321–332.]

2. (a) Известно, что $u(x)$ может быть представлен как произведение неприводимых полиномов и что старшие коэффициенты этих полиномов должны быть обратимыми элементами, поскольку они делят старший коэффициент полинома $u(x)$. Поэтому можно считать, что $u(x)$ имеет представление в виде произведения нормированных неприводимых полиномов $p_1(x)^{e_1} \dots p_r(x)^{e_r}$, где $p_1(x), \dots, p_r(x)$ различны. Это представление единственно с точностью до порядка множителей, так что условия, налагаемые на $u(x)$, $v(x)$ и $w(x)$, удовлетворяются тогда и только тогда, когда

$$v(x) = p_1(x)^{\lfloor e_1/2 \rfloor} \dots p_r(x)^{\lfloor e_r/2 \rfloor}, \qquad w(x) = p_1(x)^{e_1 \bmod 2} \dots p_r(x)^{e_r \bmod 2}.$$

(b) Производящая функция для количества нормированных полиномов степени n представляет собой $1 + pz + p^2z^2 + \cdots = 1/(1 - pz)$. Производящая функция для количества полиномов степени n, имеющих вид $v(x)^2$, где $v(x)$ — нормированный полином, представляет собой $1 + pz^2 + p^2z^4 + \cdots = 1/(1 - pz^2)$. Если обозначить производящую функцию для количества нормированных свободных от квадратов полиномов степени n через $g(z)$, то согласно п. (a) этого упражнения $1/(1-pz) = g(z)/(1-pz^2)$. Следовательно, $g(z) = (1 - pz^2)/(1 - pz) = 1 + pz + (p^2 - p)z^2 + (p^3 - p^2)z^3 + \cdots$. Таким образом, ответ — $p^n - p^{n-1}$ для $n \ge 2$. [Любопытно, что это доказывает, что $u(x) \perp u'(x)$ с вероятностью $1 - 1/p$; это та же вероятность, что и вероятность того, что $u(x) \perp v(x)$ в случае *независимости* $u(x)$ и $v(x)$ согласно упр. 4.6.1–5.]

Примечание. Аналогично доказывается, что каждый полином $u(x)$ может быть единственным образом представлен в виде $v(x)w(x)^r$, где $v(x)$ не делится на r-ю степень

никакого неприводимого полинома; количество таких нормированных полиномов $v(x)$ составляет $p^n - p^{n-r+1}$ при $n \geq r$.

3. Пусть $u(x) = u_1(x) \dots u_r(x)$. Имеется *не более* одного такого полинома $v(x)$ согласно доказательству теоремы 4.3.2C. Существует *по меньшей мере один* такой полином, если для каждого j можно решить систему с $w_j(x) = 1$ и $w_k(x) = 0$ при $k \neq j$. Решением последней является $v_1(x) \prod_{k \neq j} u_k(x)$, где $v_1(x)$ и $v_2(x)$ удовлетворяют соотношению

$$v_1(x) \prod_{k \neq j} u_k(x) + v_2(x) u_j(x) = 1, \qquad \deg(v_1) < \deg(u_j)$$

согласно расширению алгоритма Евклида (упр. 4.6.1–3).

Над кольцом целых чисел нельзя сделать $v(x) \equiv 1$ (по модулю x) и $v(x) \equiv 0$ (по модулю $x - 2$) при $\deg(v) < 2$.

4. Исходя из единственности разложения, имеем $(1 - pz)^{-1} = \prod_{n \geq 1} (1 - z^n)^{-a_{np}}$; после взятия логарифмов это соотношение может быть переписано как

$$\ln(1/(1 - pz)) = \sum_{k,j \geq 1} a_{kp} z^{kj}/j = \sum_{j \geq 1} G_p(z^j)/j.$$

Из утверждения указания следует ответ $G_p(z) = \sum_{m \geq 1} \mu(m) m^{-1} \ln(1/(1 - pz^m))$, из которого получаем $a_{np} = \sum_{d \backslash n} \mu(n/d) p^d/n$. Таким образом, $\lim_{p \to \infty} a_{np}/p^n = 1/n$.

Для доказательства утверждения, приведенного в указании, заметим, что

$$\sum_{n,j \geq 1} \mu(n) g(z^{nj}) n^{-t} j^{-t} = \sum_{m \geq 1} g(z^m) m^{-t} \sum_{n \backslash m} \mu(n) = g(z).$$

[Числа a_{np} были впервые найдены Гауссом; см. *Werke* **2**, 219–222.]

5. Пусть a_{npr} — количество нормированных полиномов степени n по модулю p, имеющих в точности r неприводимых множителей. Тогда $\mathcal{G}_p(z, w) = \sum_{n,r \geq 0} a_{npr} z^n w^r = \exp(\sum_{k \geq 1} G_p(z^k) w^k/k) = \exp(\sum_{m \geq 1} a_{mw} \ln(1/(1 - pz^{-m}))$; см. формулу 1.2.9–(38). Имеем

$$\begin{aligned} \sum_{n \geq 0} A_{np} z^n = d\mathcal{G}_p(z/p, w)/dw \,|_{w=1} &= (\textstyle\sum_{k \geq 1} G_p(z^k/p^k))\, \mathcal{G}_p(z/p, 1) \\ &= (\textstyle\sum_{n \geq 1} \ln(1/(1 - p^{1-n} z^n)) \varphi(n)/n)/(1 - z), \end{aligned}$$

следовательно, $A_{np} = H_n + 1/2p + O(p^{-2})$ для $n \geq 2$. Среднее значение 2^r равно

$$[z^n]\, \mathcal{G}_p(z/p, 2) = n + 1 + (n - 1)/p + O(np^{-2}).$$

(Дисперсия, однако, есть величина порядка n^3: положите $w = 4$.)

6. Согласно теореме Ферма $x - s$ является множителем $x^p - x$ (по модулю p) для $0 \leq s < p$. Значит, $x^p - x$ кратно $\operatorname{lcm}(x - 0, x - 1, \dots, x - (p - 1)) = x^{\underline{p}}$. [*Примечание.* Следовательно, числа Стирлинга $\left[{p \atop k}\right]$ кратны p за исключением случаев, когда $k = 1$ или $k = p$. Формула 1.2.6–(45) показывает, что то же утверждение справедливо и для чисел Стирлинга $\left\{{p \atop k}\right\}$ второго рода.]

7. Множители в правой части взаимно просты, и каждый из них является делителем $u(x)$, так что их произведение делит $u(x)$. С другой стороны, $u(x)$ делит

$$v(x)^p - v(x) = \prod_{0 \leq s < p} (v(x) - s),$$

так что $u(x)$ делит правую часть согласно упр. 4.5.2–2.

8. Вектор (18) является единственным, k-я компонента которого не равна нулю.

9. Например, начав с $x \leftarrow 1$ и $y \leftarrow 1$. Затем следует сто раз установить $R[x] \leftarrow y$, $x \leftarrow 2x \bmod 101$, $y \leftarrow 51y \bmod 101$.

10. Матрица $Q - I$, приведенная ниже, имеет ядро, порожденное двумя векторами $v^{[1]} = (1,0,0,0,0,0,0,0)$, $v^{[2]} = (0,1,1,0,0,1,1,1)$. Разложение представляет собой

$$(x^6 + x^5 + x^4 + x + 1)(x^2 + x + 1).$$

$p = 2$

$$\begin{pmatrix} 0&0&0&0&0&0&0&0\\ 0&1&1&0&0&0&0&0\\ 0&0&1&0&1&0&0&0\\ 0&0&0&1&0&0&1&0\\ 1&0&0&1&0&0&1&0\\ 1&0&1&1&1&0&0&0\\ 0&0&1&0&1&1&0&1\\ 1&1&0&1&1&1&0&1 \end{pmatrix}$$

$p = 5$

$$\begin{pmatrix} 0&0&0&0&0&0&0\\ 0&4&0&0&0&1&0\\ 0&2&2&0&4&3&4\\ 0&1&4&4&4&2&1\\ 2&2&2&3&4&3&2\\ 0&0&4&0&1&3&2\\ 3&0&2&1&4&2&1 \end{pmatrix}$$

11. После удаления тривиального множителя x приведенная выше матрица $Q - I$ имеет ядро, порожденное $(1,0,0,0,0,0,0)$ и $(0,3,1,4,1,2,1)$. Полное разложение полинома таково:

$$x(x^2 + 3x + 4)(x^5 + 2x^4 + x^3 + 4x^2 + x + 3).$$

12. Если $p = 2$, $(x+1)^4 = x^4 + 1$. Если $p = 8k+1$, $Q - I$ представляет собой нулевую матрицу, а значит, существует четыре множителя. Для других значений p имеем:

$$Q - I = \begin{pmatrix} 0&0&0&0\\ 0&-1&0&1\\ 0&0&-2&0\\ 0&1&0&-1 \end{pmatrix} \quad p = 8k+3$$

$$\begin{pmatrix} 0&0&0&0\\ 0&-2&0&0\\ 0&0&0&0\\ 0&0&0&-2 \end{pmatrix} \quad p = 8k+5$$

$$\begin{pmatrix} 0&0&0&0\\ 0&-1&0&-1\\ 0&0&-2&0\\ 0&-1&0&-1 \end{pmatrix} \quad p = 8k+7.$$

Здесь $Q - I$ имеет ранг 2, поэтому есть $4 - 2 = 2$ множителя. [Однако легко доказать, что $x^4 + 1$ неприводим над кольцом целых чисел, поскольку у него нет линейных множителей и коэффициент при x в любом множителе степени 2 согласно упр. 20 должен быть не больше 2 по абсолютному значению (см. также упр. 32, поскольку $x^4 + 1 = \Psi_8(x)$). Г. П. Ф. Свиннертон-Дайер (H. P. F. Swinnerton-Dyer) показал, что для всех $k \geq 2$ полиномы степени 2^k, неприводимые над кольцом целых чисел, полностью разлагаются на линейные и квадратичные множители по модулю любого простого числа. Для степени 8 в качестве примера он привел полином $x^8 - 16x^6 + 88x^4 + 192x^2 + 144$, имеющий корни $\pm\sqrt{2} \pm \sqrt{3} \pm i$ [см. *Math. Comp.* **24** (1970), 733–734]. Согласно теореме Фробениуса (Frobenius) из упр. 37 любой неприводимый полином степени n, в группе Галуа которого не содержится n-циклов, будет иметь множители по модулю почти всех простых чисел.]

13. Случай, когда $p = 8k+1$: $(x + (1+\sqrt{-1})/\sqrt{2})(x + (1-\sqrt{-1})/\sqrt{2})(x - (1+\sqrt{-1})/\sqrt{2}) \times (x - (1-\sqrt{-1})/\sqrt{2})$. Случай, когда $p = 8k+3$: $(x^2 + \sqrt{-2}x - 1)(x^2 - \sqrt{-2}x - 1)$. Случай, когда $p = 8k+5$: $(x^2 + \sqrt{-1})(x^2 - \sqrt{-1})$. Случай, когда $p = 8k+7$: $(x^2 + \sqrt{2}x + 1) \times (x^2 - \sqrt{2}x + 1)$. Разложение для $p = 8k+7$ также справедливо над полем действительных чисел.

14. Алгоритм N может быть адаптирован для поиска коэффициентов w. Пусть A — матрица размера $(r+1) \times n$, в k-й строке которой содержатся коэффициенты $v(x)^k \bmod u(x)$ для $0 \leq k \leq r$. Будем применять метод алгоритма N до тех пор, пока не будет найдена первая зависимость на шаге N3; затем алгоритм завершается с $w(x) = v_0 + v_1 x + \cdots + v_k x^k$, где v_j определены в (18). В этот момент $2 \leq k \leq r$; заранее знать r не нужно, поскольку можно проверять зависимость после создания каждой строки A.

15. Можно считать, что $u \neq 0$ и что p нечетно. Из метода Берлекампа, примененного к полиному $x^2 - u$, следует, что квадратный корень существует тогда и только тогда,

когда $Q - I = O$, либо тогда и только тогда, когда $u^{(p-1)/2} \bmod p = 1$, но это нам уже известно. Из метода Кантора и Зассенхауза в (20) или, что еще лучше, из (21) с $d = 1$ следует, что $\gcd(x^2-u, (x+s)^{(p-1)/2}-1)$ будет нетривиальным множителем с вероятностью $> \frac{1}{2}$ при случайном выборе s. На практике выбор последовательных s работает так же, как и случайный выбор s, так что получается следующий алгоритм: "Вычислять $\gcd(x^2-u, x^{(p-1)/2}-1)$, $\gcd(x^2-u, (x+1)^{(p-1)/2}-1)$, $\gcd(x^2-u, (x+2)^{(p-1)/2}-1)$, ... до тех пор, пока не будет найден первый наибольший общий делитель вида $x+v$. Тогда $\sqrt{u} = \pm v$". Ожидаемое время работы алгоритма составляет $O(\log p)^3$ для больших p.

Подробное рассмотрение показывает, что первый шаг этого алгоритма успешен тогда и только тогда, когда $p \bmod 4 = 3$. Для $p = 2q+1$, где q нечетно, имеем $x^q \bmod (x^2-u) = u^{(q-1)/2}x$ и $\gcd(x^2-u, x^q-1) \equiv x - u^{(q+1)/2}$, поскольку $u^q \equiv 1 \bmod p$. В действительности формула $\sqrt{u} = \pm u^{(p+1)/4} \bmod p$, когда $p \bmod 4 = 3$, непосредственно дает квадратный корень.

Однако, если $p \bmod 4 = 1$, получим $x^{(p-1)/2} \bmod (x^2-u) = u^{(p-1)/4}$ и наибольший общий делитель будет равен 1. Поэтому приведенный выше алгоритм должен использоваться только при $p \bmod 4 = 1$ и первый наибольший общий делитель должен быть пропущен.

Прямой метод, хорошо работающий при $p \bmod 8 = 5$, был открыт в 90-х годах А. О. Л. Аткином (A. O. L. Atkin) на основе того факта, что в этом случае $2^{(p-1)/2} \equiv -1$. Установить сначала $v \leftarrow (2u)^{(p-5)/8} \bmod p$ и $i \leftarrow (2uv^2) \bmod p$, затем — $\sqrt{u} = \pm(uv(i-1)) \bmod p$, а также $\sqrt{-1} = \pm i$. [*Computational Perspectives on Number Theory* (Cambridge, Mass.: International Press, 1998), 1–11; см. также H. C. Pocklington, *Proc. Camb. Phil. Soc.* **19** (1917), 57–59.]

При $p \bmod 8 = 1$ необходимо прибегнуть к методу проб и ошибок. В этом случае другие известные процедуры зачастую превосходит метод Дэниэла Шенкса (Daniel Shanks): предположим, что $p = 2^e q + 1$, где $e \geq 3$.

S1. Выбрать x случайным образом из диапазона $1 < x < p$ и установить $z = x^q \bmod p$. Если $z^{2^{e-1}} \bmod p = 1$, повторить этот шаг (среднее количество построений будет менее 2; на шагах S2 и S3 случайные числа не потребуются). На практике можно сэкономить время, перебирая малые нечетные простые числа x и останавливаясь с $z = x^q \bmod p$, когда $p^{(x-1)/2} \bmod x = x - 1$; см. упр. 1.2.4–47.

S2. Установить $y \leftarrow z$, $r \leftarrow e$, $x \leftarrow u^{(q-1)/2} \bmod p$, $v \leftarrow ux \bmod p$, $w \leftarrow ux^2 \bmod p$.

S3. Если $w = 1$, остановиться; v при этом содержит искомый ответ. В противном случае найти наименьшее k, такое, что $w^{2^k} \bmod p$ равно 1. Если $k = r$, остановиться (ответ не существует); в противном случае установить

$$(y, r, v, w) \leftarrow (y^{2^{r-k}}, k, vy^{2^{r-k-1}}, wy^{2^{r-k}})$$

и повторить шаг S3. ▮

Корректность этого алгоритма следует из тождеств-инвариантов $uw \equiv v^2$, $y^{2^{r-1}} \equiv -1$, $w^{2^{r-1}} \equiv 1 \bmod p$. При $w \neq 1$ шаг S3 выполняет $r+2$ умножений по модулю p; следовательно, максимальное количество умножений на этом шаге меньше, чем $\binom{e+3}{2}$, а среднее количество меньше $\frac{1}{2}\binom{e+4}{2}$. Таким образом, время работы составляет $O(\log p)^3$ для шагов S1 и S2 плюс порядка $e^2(\log p)^2$ для шага S3, по сравнению со временем $O(\log p)^3$ для рандомизированного метода, основанного на (21). Однако постоянный множитель в методе Шенкса мал. [*Congressus Numerantium* **7** (1972), 58–62. Схожий, но менее эффективный метод был опубликован в A. Tonelli, *Göttinger Nachrichten* (1891), 344–346. Первым алгоритм вычисления квадратного корня с ожидаемым временем работы $O(\log p)^3$ открыл M. Чиполла (M. Cipolla), *Rendiconti Accad. Sci. Fis. Mat. Napoli* **9** (1903), 154–163.]

16. (a) В доказательстве для $n = 1$ вместо целых чисел подставьте полиномы по модулю p. (b) Доказательство для $n = 1$ распространяется на любое конечное поле. (c) Поскольку

$x = \xi^k$ для некоторого k, $x^{p^n} = x$ в поле, определенном $f(x)$. К тому же множество элементов y, удовлетворяющих уравнению $y^{p^m} = y$, в этом поле замкнуто по отношению к сложению и по отношению к умножению. Так что если $x^{p^m} = x$, то ξ (будучи полиномом от x с целыми коэффициентами) удовлетворяет уравнению $\xi^{p^m} = \xi$.

17. Если ξ — примитивный корень, то каждый ненулевой элемент является некоторой степенью ξ. Следовательно, порядок должен быть делителем $13^2 - 1 = 2^3 \cdot 3 \cdot 7$ и порядок f имеют $\varphi(f)$ элементов.

f	$\varphi(f)$	f	$\varphi(f)$	f	$\varphi(f)$	f	$\varphi(f)$
1	1	3	2	7	6	21	12
2	1	6	2	14	6	42	12
4	2	12	4	28	12	84	24
8	4	24	8	56	24	168	48

18. (a) Согласно лемме Гаусса $\operatorname{pp}(p_1(u_n x)) \ldots \operatorname{pp}(p_r(u_n x))$. Например, пусть

$$u(x) = 6x^3 - 3x^2 + 2x - 1, \qquad v(x) = x^3 - 3x^2 + 12x - 36 = (x^2 + 12)(x - 3);$$

тогда $\operatorname{pp}(36x^2 + 12) = 3x^2 + 1$, $\operatorname{pp}(6x - 3) = 2x - 1$. (Это современная версия использовавшегося многие годы для решения алгебраических уравнений трюка 14 века.)

(b) Пусть $\operatorname{pp}(w(u_n x)) = \bar{w}_m x^m + \cdots + \bar{w}_0 = w(u_n x)/c$, где c — содержание $w(u_n x)$ как полинома от x. Тогда $w(x) = (c\bar{w}/u_n^m)x^m + \cdots + c\bar{w}_0$. Следовательно, $c\bar{w}_m = u_n^m$; поскольку $\bar{w}_m$ является делителем u_n, c кратно u_n^{m-1}.

19. Если $u(x) = v(x)w(x)$ и при этом $\deg(v)\deg(w) \geq 1$, то $u_n x^n \equiv v(x)w(x) \bmod p$. В соответствии с единственностью разложения по модулю p все коэффициенты v и w, кроме старших, кратны p и p^2 делит $v_0 w_0 = u_0$.

20. (a) $\sum(\alpha u_j - u_{j-1})(\bar{\alpha}\bar{u}_j - \bar{u}_{j-1}) = \sum(u_j - \bar{\alpha}u_{j-1})(\bar{u}_j - \alpha\bar{u}_{j-1})$. (b) Можно считать, что $u_0 \neq 0$. Пусть $m(u) = \prod_{j=1}^n \min(1, |\alpha_j|) = |u_0|/M(u)$. При $|\alpha_j| < 1$ замените в $u(x)$ множитель $x - \alpha_j$ на $\bar{\alpha}_j x - 1$. Это не повлияет на $\|u\|$, но заменит $|u_0|$ на $M(u)$. (c) $u_j = u_m \sum \alpha_{i_1} \ldots \alpha_{i_{m-j}}$, представляя собой элементарную симметричную функцию. Следовательно, $|u_j| \leq |u_m| \sum \beta_{i_1} \ldots \beta_{i_{m-j}}$, где $\beta_i = \max(1, |\alpha_i|)$. Завершим доказательство, показав, что при $x_1 \geq 1, \ldots, x_n \geq 1$ и $x_1 \ldots x_n = M$ элементарная симметричная функция $\sigma_{nk} = \sum x_{i_1} \ldots x_{i_k} \leq \binom{n-1}{k-1}M + \binom{n-1}{k}$ — предполагаемому значению при $x_1 = \cdots = x_{n-1} = 1$ и $x_n = M$. (Если $x_1 \leq \cdots \leq x_n < M$, преобразование $x_n \leftarrow x_{n-1}x_n$, $x_{n-1} \leftarrow 1$ увеличивает σ_{nk} на $\sigma_{(n-2)(k-1)}(x_n - 1)(x_{n-1} - 1)$, являющуюся положительной величиной.) (d) $|v_j| \leq \binom{m-1}{j}M(v) + \binom{m-1}{j-1}|v_m| \leq \binom{m-1}{j}M(u) + \binom{m-1}{j-1}|u_n|$, поскольку $M(v) \leq M(u)$ и $|v_m| \leq |u_n|$. [M. Mignotte, *Math. Comp.* **28** (1974), 1153–1157.]

Примечания. Это решение показывает, что $\binom{m-1}{j}M(u) + \binom{m-1}{j-1}|u_n|$ является верхней гранью, так что хотелось бы иметь лучшую оценку $M(u)$. Известно несколько методов нахождения этих оценок [W. Specht, *Math. Zeit.* **53** (1950), 357–363; Cerlienco, Mignotte, and Piras, *J. Symbolic Comp.* **4** (1987), 21–33]. Простейшим и наиболее быстро сходящимся, возможно, является следующий метод [см. C. H. Graeffe, *Auflösung der höheren numerischen Gleichungen* (Zürich, 1837)]. Полагая, что $u(x) = u_n(x - \alpha_1)\ldots(x - \alpha_n)$, обозначим $\hat{u}(x) = u(\sqrt{x})u(-\sqrt{x}) = (-1)^n u_n^2(x - \alpha_1^2)\ldots(x - \alpha_n^2)$. Тогда $M(u)^2 = M(\hat{u}) \leq \|\hat{u}\|$. Следовательно, можно установить $c \leftarrow \|u\|$, $v \leftarrow u/c$, $t \leftarrow 0$, а затем несколько раз установить $t \leftarrow t + 1$, $c \leftarrow \|\hat{v}\|^{1/2^t} c$, $v \leftarrow \hat{v}/\|\hat{v}\|$. Инвариантные соотношения $M(u) = cM(v)^{1/2^t}$ и $\|v\| = 1$ гарантируют, что $M(u) \leq c$ на каждом шаге итерации. Заметьте, что, когда $v(x) = v_0(x^2) + xv_1(x^2)$, мы имеем $\hat{v}(x) = v_0(x)^2 - xv_1(x)^2$. Можно показать, что если каждое $|\alpha_j| \leq \rho$ или $\geq 1/\rho$, то $M(u) = \|u\|(1 + O(\rho))$; следовательно, c после t шагов будет равно $M(u)(1 + O(\rho^{2^t}))$.

Например, если $u(x)$ представляет собой полином из (22), последовательные значения c для $t = 0, 1, 2, \ldots$ оказываются равными 10.63, 12.42, 6.85, 6.64, 6.65, 6.6228, 6.62246, 6.62246, ... В этом примере $\rho \approx .90982$. Заметьте, что сходимость не монотонна. В конечном итоге $v(x)$ сходится к одночлену x^m, где m — количество корней с $|\alpha_j| < 1$, в предположении, что $|\alpha_j| \neq 1$ для всех j; в общем, если имеется k корней с $|\alpha_j| = 1$, коэффициенты при x^m и x^{m+k} не достигнут нуля, в то время как это произойдет с коэффициентами при высших и низших степенях x.

Известная формула Дженсена [*Acta Math.* **22** (1899), 359–364] доказывает, что $M(u)$ является геометрическим средним $|u(x)|$ в единичном круге, а именно —

$$\exp\left(\tfrac{1}{2\pi}\int_0^{2\pi} \ln|f(e^{i\theta})|\,d\theta\right).$$

В упр. 21, (a) будет аналогично показано, что $\|u\|$ является средним квадратичным $|u(x)|$ в единичном круге. Неравенство $M(u) \le \|u\|$, восходящее к Э. Ландау (E. Landau) [*Bull. Soc. Math. de France* **33** (1905), 251–261], может поэтому трактоваться как соотношение между средними значениями. Число $M(u)$ часто называется *мерой Махлера* полинома, потому что Курт Махлер (Kurt Mahler) использовал его в *Mathematika* **7** (1960), 98–100. Дженсен также доказал, что $\frac{1}{2\pi}\int_0^{2\pi} e^{im\theta}\ln|f(e^{i\theta})|\,d\theta = -\sum_{j=1}^{n}\alpha_j^m/(2m\max(|\alpha_j|,1)^{2m})$ при $m > 0$.

21. (a) Коэффициент при $a_{\mathbf{p}}b_{\mathbf{q}}c_{\mathbf{r}}d_{\mathbf{s}}$ равен нулю с обеих сторон, кроме случая $\mathbf{p}+\mathbf{s} = \mathbf{q}+\mathbf{r}$. Когда это условие выполняется, коэффициент справа равен $(\mathbf{p}+\mathbf{s})!$; слева он составляет

$$\sum_{\mathbf{j}}\binom{\mathbf{p}}{\mathbf{j}}\binom{\mathbf{s}}{\mathbf{r}-\mathbf{j}}\mathbf{q}!\,\mathbf{r}! = \binom{\mathbf{p}+\mathbf{s}}{\mathbf{r}}\mathbf{q}!\,\mathbf{r}! = (\mathbf{q}+\mathbf{r})!\,.$$

[B. Beauzamy and J. Dégot, *Trans. Amer. Math. Soc.* **345** (1995), 2607–2619; D. Zeilberger, *AMM* **101** (1994), 894–896.]

(b) Пусть $a_{\mathbf{p}} = v_{\mathbf{p}}$, $b_{\mathbf{q}} = w_{\mathbf{q}}$, $c_{\mathbf{r}} = \overline{v}_{\mathbf{r}}$, $d_{\mathbf{s}} = \overline{w}_{\mathbf{s}}$. Тогда правая сторона (a) равна $B(u)$, а левая сторона представляет собой сумму неотрицательных членов для каждого $\mathbf{j}$ и $\mathbf{k}$. Если рассмотреть только члены, где $\Sigma\,\mathbf{j}$ является степенью v, члены $v_{\mathbf{p}}/(\mathbf{p}-\mathbf{j})!$ исчезнут, за исключением $\mathbf{p} = \mathbf{j}$. Эти члены сводятся к

$$\sum_{\mathbf{j},\mathbf{k}}\frac{1}{\mathbf{j}!\,\mathbf{k}!}\left|v_{\mathbf{j}}w_{\mathbf{k}}\,\mathbf{j}!\,\mathbf{k}!\right|^2 = B(v)B(w)\,.$$

[B. Beauzamy, E. Bombieri, P. Enflo, and H. Montgomery, *J. Number Theory* **36** (1990), 219–245.]

(c) Добавление новой переменной при необходимости достичь однородности не меняет соотношения $u = vw$. Таким образом, если v и w имеют общие степени m и n соответственно, получаем $(m+n)!\,[u]^2 \ge m!\,[v]^2\,n!\,[w]^2$; другими словами, $[v][w] \le \binom{m+n}{m}^{1/2}[u]$.

Существует один изящный способ рассмотрения нормы Бомбьери, заключающийся в том, чтобы представить, что переменные некоммутативны. Например, вместо $3xy^3 - z^2w^2$ можно записать $\frac{3}{4}xyyy+\frac{3}{4}yxyy+\frac{3}{4}yyxy+\frac{3}{4}yyyx-\frac{1}{6}zzww-\frac{1}{6}zwzw-\frac{1}{6}zwwz-\frac{1}{6}wzzw-\frac{1}{6}wzwz-\frac{1}{6}wwzz$. Тогда норма Бомбьери будет представлять собой норму $\|\ \|$ новых коэффициентов. Другой интересной формулой для однородного полинома u степени n является

$$[u]^2 = \frac{1}{n!\,\pi^n}\int_{\mathbf{x}}\int_{\mathbf{y}} e^{-x_1^2-\cdots-x_t^2-y_1^2-\cdots-y_t^2}\,|u(\mathbf{x}+i\mathbf{y})|^2\,d\mathbf{x}\,d\mathbf{y}\,.$$

(d) Случай с одной переменной соответствует $t = 2$. Предположим, что $u = vw$, где v — однородный полином степени m от t переменных. Тогда $|v_{\mathbf{k}}|^2\,\mathbf{k}!/m! \le [v]^2$ для всех $\mathbf{k}$ и $\mathbf{k}! \ge (m/t)!^t$, поскольку $\log\Gamma(x)$ выпукла при $x > 0$; поэтому $|v_{\mathbf{k}}|^2 \le m!\,[v]^2/(m/t)!^t$. Можно положить, что $m!\,[v]^2/(m/t)!^t \le m'!\,[w]^2/(m'/t)!^t$, где $m' = n-m$ — степень w. Тогда

$$|v_{\mathbf{k}}|^2 \le m!\,[v]^2/(m/t)!^t \le m!^{1/2}m'!^{1/2}\,[v][w]/(m/t)!^{t/2}(m'/t)!^{t/2} \le n!^{1/2}\,[u]/(n/2t)!^t\,.$$

(Лучшая грань получится, если максимизировать предпоследнее выражение по всем степеням m для каждого неисключенного множителя.) Величина $n!^{1/4}/(n/2t)!^{t/2}$ равна $c_t(2t)^{n/4}n^{-(2t-1)/8}(1+O(\frac{1}{n}))$, где $c_t = 2^{1/8}\pi^{-(2t-1)/8}t^{t/4}$ (≈ 1.004 при $t=2$).

Заметьте, что существование *неприводимого* множителя с такими малыми коэффициентами здесь не доказывалось; может потребоваться дальнейшее разложение (см. упр. 41).

(e) $[u]^2 = \sum_k \binom{n}{k}^2/\binom{2n}{2k} = \sum_k \binom{2k}{k}\binom{2n-2k}{n-k}/\binom{2n}{n} = 4^n/\binom{2n}{n} = \sqrt{\pi n} + O(n^{-1/2})$. Если $v(x) = (x-1)^n$ и $w(x) = (x+1)^n$, имеем $[v] = [w] = 2^n$; следовательно, неравенство (c) становится в этом случае равенством.

(f) Пусть u и v — однородные полиномы степени m и n. Тогда

$$[uv]^2 \le \sum_{\mathbf{k}} \frac{(\sum_{\mathbf{j}} |u_{\mathbf{j}} v_{\mathbf{k}-\mathbf{j}}|)^2}{\binom{m+n}{\mathbf{k}}} \le \sum_{\mathbf{k}} \left(\sum_{\mathbf{j}} \frac{|u_{\mathbf{j}}|^2}{\binom{m}{\mathbf{j}}} \frac{|v_{\mathbf{k}-\mathbf{j}}|^2}{\binom{n}{\mathbf{k}-\mathbf{j}}}\right)\left(\sum_{\mathbf{j}} \frac{\binom{m}{\mathbf{j}}\binom{n}{\mathbf{k}-\mathbf{j}}}{\binom{m+n}{\mathbf{k}}}\right) = [u]^2[v]^2$$

согласно неравенству Коши. [B. Beauzamy, *J. Symbolic Comp.* **13** (1992), 465–472, Proposition 5.]

(g) В соответствии с упр. 20 $\binom{n}{\lfloor n/2\rfloor}^{-1}M(u)^2 \le \binom{n}{\lfloor n/2\rfloor}^{-1}\|u\|^2 = \binom{n}{\lfloor n/2\rfloor}^{-1}\sum_j |u_j|^2 \le [u]^2 = \sum_j \binom{n}{j}^{-1}|u_j|^2 \le \sum_j \binom{n}{j}M(u)^2 = 2^n M(u)^2$. Первое неравенство следует также из п. (f); если $u(x) = u_n\prod_{j=1}^n(x-\alpha_j)$, имеем $[u]^2 \le |u_n|^2\prod_{j=1}^n[x-\alpha_j]^2 = |u_n|^2\prod_{j=1}^n(1+|\alpha_j|^2) \le |u_n|^2\prod_{j=1}^n(2\max(1,|\alpha_j|)^2) = 2^nM(u)^2$.

22. Рассмотрим более общую ситуацию. Предположим, что $u(x) \equiv v(x)w(x) \bmod q$, $a(x)v(x)+b(x)w(x) \equiv 1 \bmod p$, и $c\ell(v) \equiv 1 \bmod r$, $\deg(a) < \deg(w)$, $\deg(b) < \deg(v)$, $\deg(u) = \deg(v)+\deg(w)$, где $r = \gcd(p,q)$ и p,q не обязательно должны быть простыми числами. Построим полиномы $V(x) \equiv v(x)$ и $W(x) \equiv w(x) \bmod q$ такие, что $u(x) \equiv V(x)W(x) \bmod qr$, $\ell(V) = \ell(v)$, $\deg(V) = \deg(v)$, $\deg(W) = \deg(w)$. Кроме того, если r — простое число, результат по модулю qr окажется единственным.

В задаче спрашивается, как найти $\bar v(x)$ и $\bar w(x)$, такие, что $V(x) = v(x)+q\bar v(x)$, $W(x) = w(x)+q\bar w(x)$, $\deg(\bar v) < \deg(v)$, $\deg(\bar w) \le \deg(w)$; другое условие,

$$(v(x)+q\bar v(x))(w(x)+q\bar w(x)) \equiv u(x) \mod qr,$$

эквивалентно $\bar w(x)v(x) + \bar v(x)w(x) \equiv f(x) \bmod r$, где $f(x)$ удовлетворяет соотношению $u(x) \equiv v(x)w(x) + qf(x) \bmod qr$. Для всех $t(x)$ имеем

$$(a(x)f(x)+t(x)w(x))v(x) + (b(x)f(x)-t(x)v(x))w(x) \equiv f(x) \mod r.$$

Поскольку для $\ell(v)$ существует обратный элемент по модулю r, можно с помощью алгоритма 4.6.1D найти частное $t(x)$, такое, что $\deg(bf-tv) < \deg(v)$; для этого $t(x)$, $\deg(af+tw) \le \deg(w)$, поскольку имеем $\deg(f) \le \deg(u) = \deg(v)+\deg(w)$. Таким образом, искомое решение — $\bar v(x) = b(x)f(x) - t(x)v(x) = b(x)f(x) \bmod v(x)$, $\bar w(x) = a(x)f(x)+t(x)w(x)$. Если $(\bar{\bar v}(x), \bar{\bar w}(x))$ представляет собой другое решение, имеем $(\bar w(x)-\bar{\bar w}(x))v(x) \equiv (\bar{\bar v}(x)-\bar v(x))w(x) \bmod r$. Следовательно, если r — простое число, $v(x)$ должно делить $\bar{\bar v}(x)-\bar v(x)$, но $\deg(\bar{\bar v}-\bar v) < \deg(v)$, так что $\bar{\bar v}(x) = \bar v(x)$ и $\bar{\bar w}(x) = \bar w(x)$.

Если p делит q, так что $r = p$, выбор $V(x)$ и $W(x)$ удовлетворяет также соотношению $a(x)V(x)+b(x)W(x) \equiv 1$ (по модулю p), что и требуется по лемме Хенселя.

Для $p = 2$ процесс разложения протекает следующим образом (далее записываются только коэффициенты с надчеркиванием для обозначения отрицательных цифр). В упр. 10 утверждается, что $v_1(x) = (\bar 1\,\bar 1\,\bar 1)$, $w_1(x) = (\bar 1\,\bar 1\,\bar 1\,0\,0\,\bar 1\,\bar 1)$ в однобитовой комплементарной к 2 записи. Расширенный алгоритм Евклида приводит к $a(x) = (1\,0\,0\,0\,0\,1), b(x) = (1\,0)$. Множитель $v(x) = x^2+c_1x+c_0$ должен иметь $|c_1| \le \lfloor 1+\sqrt{113}\rfloor = 11$, $|c_0| \le 10$ согласно упр. 20. Три применения леммы Хенселя дают $v_4(x) = (1\,3\,\bar 1)$, $w_4(x) = (1\,\bar 3\,\bar 5\,\bar 4\,4\,\bar 3\,5)$. Таким образом, $c_1 \equiv 3$ и $c_0 \equiv -1 \bmod 16$; единственный возможный квадратичный множитель

$u(x) = x^2 + 3x - 1$. Деление ошибочно, поэтому $u(x)$ — неприводимый полином. (Поскольку "неприводимость" этого "ненаглядного" полинома доказана четырьмя различными методами, вряд ли кто-то сможет найти хоть один его множитель...)

Ганс Зассенхауз (Hans Zassenhaus) обнаружил, что зачастую можно ускорить вычисления, увеличив p так же, как и q: если в приведенных выше обозначениях $r = p$, можно найти $A(x)$, $B(x)$, такие, что $A(x)V(x) + B(x)W(x) \equiv 1 \bmod p^2$, а именно — взяв $A(x) = a(x) + p\bar{a}(x)$, $B(x) = b(x) + p\bar{b}(x)$, где $\bar{a}(x)V(x) + \bar{b}(x)W(x) \equiv g(x) \bmod p$, $a(x)V(x) + b(x)W(x) \equiv 1 - pg(x) \bmod p^2$. Также можно найти C с $\ell(V)C \equiv 1 \bmod p^2$. Так можно свести разложение, свободное от квадратов, $u(x) \equiv v(x)w(x) \bmod p$, к их единственным расширениям по модулю p^2, p^4, p^8, p^{16} и т. д. Однако такая "ускоренная" процедура на практике достигает точки замедления, как только мы достигаем двойной точности, поскольку время для умножения чисел с многократной точностью в реальном диапазоне значений перевешивает преимущества от возведения модулей в квадрат. С вычислительной точки зрения представляется, что лучше работать с последовательными модулями p, p^2, p^4, p^8, ..., p^E, p^{E+e}, p^{E+2e}, p^{E+3e}, ..., где E — наименьшая степень 2 с p^E, бóльшим однократной точности, и e — наибольшим целым числом, таким, что p^e имеет однократную точность.

"Лемма Хенселя" на самом деле была открыта К. Ф. Гауссом (C. F. Gauss) около 1799 года, в черновике неоконченной книги *Analysis Residuorum*, §373–374. Гаусс внес большую часть материала из этой рукописи в свою *Disquisitiones Arithmeticæ* (1801), но его идеи о разложении полиномов были опубликованы лишь после его смерти [см. *Werke* **2** (Göttingen, 1876), 238]. Имя Хенселя оказалось связанным с методом, потому что он стал основой теории p-ичных чисел (см. упр. 4.1–31). Лемма может быть обобщена несколькими способами. Во-первых, если имеется большее количество множителей, например $u(x) \equiv v_1(x)v_2(x)v_3(x) \bmod p$, можно найти $a_1(x)$, $a_2(x)$, $a_3(x)$, такие, что $a_1(x)v_2(x)v_3(x) + a_2(x)v_1(x)v_3(x) + a_3(x)v_1(x)v_2(x) \equiv 1 \bmod p$ и $\deg(a_i) < \deg(v_i)$. (По существу, $1/u(x)$ расширяется в отдельных частях до $\sum a_i(x)/v_i(x)$.) Точный аналог построения теперь позволяет провести разложение, не изменяя старшие коэффициенты v_1 и v_2; мы получаем $\bar{v}_1(x) = a_1(x)f(x) \bmod v_1(x)$, $\bar{v}_2(x) = a_2(x)f(x) \bmod v_2(x)$ и т. д. Другое важное обобщение состоит в различных одновременных модулях соответствующего вида $p^e, (x_2 - a_2)^{n_2}, \ldots, \ldots, (x_t - a_t)^{n_t}$ при выполнении поиска наибольших общих делителей и разложении полиномов от нескольких переменных. [См. D. Y. Y. Yun, Ph. D. Thesis (M. I. T., 1974).]

23. Дискриминант $\operatorname{pp}(u(x))$ представляет собой ненулевое целое число (см. упр. 4.6.1–12), и кратные множители по модулю p существуют тогда и только тогда, когда p делит этот дискриминант. [Разложение (22) по модулю 3 равно $(x+1)(x^2-x-1)^2(x^3+x^2-x+1)$; квадратные множители для этого полинома имеются только для $p = 3$, 23, 233 и 121702457. Нетрудно доказать, что наименьшее простое число, не являющееся "неудачным", не превышает $O(n \log Nn)$, если $n = \deg(u)$, а N является гранью по абсолютной величине коэффициентов $u(x)$.]

24. Умножьте нормированный полином с рациональными коэффициентами на подходящее ненулевое целое число для получения примитивного полинома над кольцом целых чисел. Разложите полином над кольцом целых чисел, а затем конвертируйте множители в нормированные полиномы (таким образом, не будет потеряно ни одно разложение; см. упр. 4.6.1–8).

25. Рассмотрение постоянного члена показывает, что у полинома нет множителей степени 1, так что если полином приводим, то он должен иметь один множитель степени 2 и один — степени 3. Разложение по модулю 2 представляет собой $x(x+1)^2(x^2+x+1)$ и для

нас бесполезно. Разложение по модулю 3 представляет собой $(x+2)^2(x^3+2x+2)$, а по модулю 5 — $(x^2+x+1)(x^3+4x+2)$. Таким образом, искомый ответ — $(x^2+x+1)(x^3-x+2)$.

26. Начнем с $D \leftarrow (0\ldots01)$, представляющего множество $\{0\}$. Затем для $1 \le j \le r$ установим $D \leftarrow D \vee (D \ll d_j)$, где $\vee$ означает побитовое "или", а $D \ll d$ — побитовый сдвиг D влево на d позиций. (В действительности достаточно работать только с битовым вектором размерности $\lceil (n+1)/2 \rceil$, поскольку $n-m$ содержится в множестве тогда и только тогда, когда в нем содержится m.)

27. В упр. 4 утверждается, что случайный полином степени n неприводим по модулю p с весьма малой вероятностью, около $1/n$. Но из китайской теоремы об остатках следует, что случайный нормированный полином степени n над кольцом целых чисел будет приводим для каждого из k различных простых чисел с вероятностью около $(1-1/n)^k$, которая будет стремиться к нулю при $k \to \infty$. Следовательно, почти все полиномы над кольцом целых чисел являются неприводимыми для бесконечно большого количества простых чисел и почти все примитивные полиномы над кольцом простых чисел являются неприводимыми полиномами (другое доказательство дано в работе W. S. Brown, *AMM* **70** (1963), 965–969).

28. См. упр. 4; вероятность составляет $[z^n](1+a_{1p}z/p)(1+a_{2p}z^2/p^2)(1+a_{3p}z^3/p^3)\ldots$, предельное значение при $n \to \infty$ составляет $g(z) = (1+z)(1+\frac{1}{2}z^2)(1+\frac{1}{3}z^3)\ldots$. Для $1 \le n \le 10$ искомые значения равны $\frac{1}{2}$, $\frac{5}{6}$, $\frac{7}{12}$, $\frac{37}{60}$, $\frac{79}{120}$, $\frac{173}{280}$, $\frac{101}{168}$, $\frac{127}{210}$, $\frac{1033}{1680}$. [Пусть $f(y) = \ln(1+y) - y = O(y^2)$. Имеем

$$g(z) = \exp(\textstyle\sum_{n\ge1} z^n/n + \sum_{n\ge1} f(z^n/n)) = h(z)/(1-z),$$

и можно показать, что предельная вероятность равна $h(1) = \exp(\sum_{n\ge1} f(1/n)) = e^{-\gamma} \approx .56146$. В действительности Н. Г. де Брейн (N. G. de Bruijn) установил, что асимптотическая формула имеет следующий вид: $\lim_{p\to\infty} a_{np} = e^{-\gamma} + e^{-\gamma}/n + O(n^{-2}\log n)$. [См. D. H. Lehmer, *Acta Arith.* **21** (1972), 379–388; D. H. Greene and D. E. Knuth, *Math. for the Analysis of Algorithms* (Boston: Birkhäuser, 1981), §4.1.6.] С другой стороны, ответ для $1 \le n \le 10$ при $p = 2$ имеет меньшие значения: 1, $\frac{1}{4}$, $\frac{1}{2}$, $\frac{7}{16}$, $\frac{7}{16}$, $\frac{7}{16}$, $\frac{27}{64}$, $\frac{111}{256}$, $\frac{109}{256}$, $\frac{109}{256}$. В работе A. Knopfmacher and R. Warlimont, *Trans. Amer. Math. Soc.* **347** (1995), 2235–2243, показано, что для фиксированного p вероятность составляет $c_p + O(1/n)$, где $c_p = \prod_{m\ge1} e^{-1/m}(1+a_{mp}/p^m)$ и $c_2 \approx .397$.]

29. Пусть $q_1(x)$ и $q_2(x)$ — любые неприводимые делители $g(x)$. По китайской теореме об остатках (упр. 3) выбор случайного полинома $t(x)$ степени $< 2d$ эквивалентен выбору двух случайных полиномов $t_1(x)$ и $t_2(x)$ степени $< d$ каждый, где $t_i(x) = t(x) \bmod q_i(x)$. Наибольший общий делитель будет корректным множителем, если $t_1(x)^{(p^d-1)/2} \bmod q_1(x) = 1$ и $t_2(x)^{(p^d-1)/2} \bmod q_1(x) \ne 1$ или наоборот, и это условие выполняется в точности для $2((p^d-1)/2)((p^d+1)/2) = (p^{2d}-1)/2$ выборов $t_1(x)$ и $t_2(x)$.

Примечания. Здесь рассматривается только поведение с учетом двух неприводимых множителей, но истинное поведение, вероятно, гораздо лучше. Предположим, что каждый неприводимый множитель $q_i(x)$ имеет вероятность $\frac{1}{2}$ деления полинома $t(x)^{(p^d-1)/2} - 1$ для каждого $t(x)$ независимо от поведения других $q_j(x)$ и $t(x)$; предположим, что $g(x)$ имеет всего r неприводимых множителей. Тогда, если закодировать каждый $q_i(x)$ последовательностью нулей и единиц в соответствии с тем, делит $q_i(x)$ или нет $t(x)^{(p^d-1)/2} - 1$ для последовательных проверяемых t, можно получить случайный бинарный луч с r "листьями" (см. раздел 6.3). Цена, связанная с внутренним узлом этого луча, который имеет m листьев в качестве потомков, равна $O(m^2(\log p))$, а решением рекуррентного соотношения $A_n = \binom{n}{2} + 2^{1-n}\sum\binom{n}{k}A_k$ является $A_n = 2\binom{n}{2}$ в соответствии с упр. 5.2.2–36. Следовательно, сумма цен данного случайного луча, представляющая ожидаемое время *полного* разложения $g(x)$, составляет $O(r^2(\log p)^3)$ при этих правдоподобных предположениях. Правдоподобность предположений становится абсолютно справедливой при выборе

случайного $t(x)$ степени $< rd$ вместо того, чтобы ограничиться выбором полинома степени $< 2d$.

30. Пусть $T(x) = x + x^p + \cdots + x^{p^{d-1}}$ — *след* x и пусть $v(x) = T(t(x)) \bmod q(x)$. Поскольку $t(x)^{p^d} = t(x)$ в поле полиномиальных остатков по модулю $q(x)$, имеем в этом поле $v(x)^p = v(x)$; другими словами, $v(x)$ является одним из p корней уравнения $y^p - y = 0$. Значит, $v(x)$ — целое число.

Отсюда следует, что $\prod_{s=0}^{p-1} \gcd(g_d(x), T(t(x)) - s) = g_d(x)$. В частности, когда $p = 2$, можно, как в упр. 29, утверждать, что $\gcd(g_d(x), T(t(x)))$ будет собственным делителем $g_d(x)$ с вероятностью $\geq \frac{1}{2}$, если $g_d(x)$ имеет хотя бы два неприводимых множителя и $t(x)$ — случайный бинарный полином степени $< 2d$.

[Заметьте, что $T(t(x)) \bmod g(x)$ может быть вычислено, начиная с $u(x) \leftarrow t(x)$, и после установки $d - 1$ раз $u(x) \leftarrow (t(x) + u(x)^p) \bmod g(x)$. Метод этого упражнения основан на разложении полиномов $x^{p^d} - x = \prod_{s=0}^{p-1}(T(x) - s)$, которое справедливо для любого p, в то время как формула (21) основана на разложении $x^{p^d} - x = x(x^{(p^d-1)/2} + 1)(x^{(p^d-1)/2} - 1)$ для нечетных p.]

"След" был введен Ричардом Дедекиндом (Richard Dedekind), *Abhandlungen der Königl. Gesellschaft der Wissenschaften zu Göttingen* **29** (1882), 1–56. Использование метода вычисления $\gcd(f(x), T(x) - s)$ для поиска множителей $f(x)$ было прослежено до А. Эрвина (A. Arwin), *Arkiv för Mat., Astr. och Fys.* **14**, 7 (1918), 1–46; однако его метод был не полон, потому что он не рассматривал $T(t(x))$ для $t(x) \neq x$. Алгоритм полного разложения с использованием следов был разработан позже Р. Д. Мак-Элисом (R. J. McEliece), *Math. Comp.* **23** (1969), 861–867; см. также von zur Gathen and Shoup, *Computational Complexity* **2** (1992), 187–224, алгоритм 3.6 (дающий асимптотически быстрые результаты).

Генри Кохен (Henri Cohen) обнаружил, что при использовании этого метода для $p = 2$ достаточно проверить не более d специальных случаев $t(x) = x, x^3, \ldots, x^{2d-1}$. Один из этих выборов $t(x)$ гарантированно разбивает $g_d(x)$ на множители, если g_d приводим, потому что можно получить результаты всех полиномов $t(x)$ степени $< 2d$ из этих частных случаев, если использовать тот факт, что $T(t(x)^p) \equiv T(t(x))$ и $T(u(x) + t(x)) \equiv T(u(x)) + T(t(x))$ (по модулю $g_d(x)$). [*A Course in Computational Algebraic Number Theory* (Springer, 1993), Algorithm 3.4.8.]

31. Если α — элемент поля из p^d элементов, обозначим через $d(\alpha)$ *степень* α, а именно — наименьший показатель степени e, такой, что $\alpha^{p^e} = \alpha$. Затем рассмотрим полином

$$P_\alpha(x) = (x - \alpha)(x - \alpha^p) \ldots (x - \alpha^{p^{d-1}}) = q_\alpha(x)^{d/d(\alpha)},$$

где $q_\alpha(x)$ — неприводимый полином степени $d(\alpha)$. При α, пробегающем по всем элементам этого поля, соответствующий полином $q_\alpha(x)$ пробегает по всем неприводимым полиномам степени e, делящим d, где каждая такая "неприводимость" встречается в точности e раз. Имеем $(x + t)^{(p^d-1)/2} \bmod q_\alpha(x) = 1$ тогда и только тогда, когда в поле $(\alpha + t)^{(p^d-1)/2} = 1$. Если t — целое число, имеем $d(\alpha + t) = d(\alpha)$; следовательно, $n(p, d)$ в d^{-1} раз превышает число элементов α степени d, таких, что $\alpha^{(p^d-1)/2} = 1$. Аналогично, если $t_1 \neq t_2$, нужно выяснить количество элементов степени d, таких, что $(\alpha + t_1)^{(p^d-1)/2} = (\alpha + t_2)^{(p^d-1)/2}$ или, что то же самое, $((\alpha + t_1)/(\alpha + t_2))^{(p^d-1)/2} = 1$. При α, пробегающем по всем элементам степени d, справедливо равенство $(\alpha + t_1)/(\alpha + t_2) = 1 + (t_1 - t_2)/(\alpha + t_2)$.

[Имеем $n(p, d) = \frac{1}{4} d^{-1} \sum_{c \backslash d} (3 + (-1)^c) \mu(c) (p^{d/c} - 1)$, что составляет около половины общего числа "неприводимостей" (в точности половину при нечетном d). Это доказывает, что $\gcd(g_d(x), (x + t)^{(p^d-1)/2} - 1)$ дает хорошие шансы найти множители $g_d(x)$ при фиксированном t и случайным образом выбранном $g_d(x)$; однако рандомизированный алгоритм предлагается для работы с гарантированной вероятностью для *фиксированного* $g_d(x)$ и *случайного* t, как в упр. 29.]

32. (a) Ясно, что $x^n-1=\prod_{d\backslash n}\Psi_d(x)$, поскольку каждый комплексный n-й корень единицы является примитивным d-м корнем некоторого уникального $d\backslash n$. Второе тождество следует из первого; $\Psi_n(x)$ имеет целые коэффициенты, поскольку они выражаются через члены произведений и частных нормированных полиномов с целыми коэффициентами.

(b) Условия, приведенного в указании, достаточно для доказательства того, что $f(x)=\Psi_n(x)$. Когда p не делит n, имеем $x^n-1\perp nx^{n-1}$ по модулю p, следовательно, полином x^n-1 свободен от квадратов по модулю p. При данных $f(x)$ и ζ, таких, как описано в указании, обозначим через $g(x)$ неприводимый множитель $\Psi_n(x)$, такой, что $g(\zeta^p)=0$. Если $g(x)\neq f(x)$, то $f(x)$ и $g(x)$ — различные множители $\Psi_n(x)$. Значит, они являются различными множителями x^n-1 и, следовательно, не имеют неприводимых множителей по общему модулю p. Однако ζ является корнем $g(x^p)$, так что $\gcd(f(x),g(x^p))\neq 1$ над кольцом целых чисел, и поэтому $f(x)$ является делителем $g(x^p)$. Согласно (5) $f(x)$ — делитель $g(x)^p$ по модулю p, а это противоречит предположению, что $f(x)$ и $g(x)$ не имеют общих неприводимых множителей. Поэтому $f(x)=g(x)$. [Неприводимость $\Psi_n(x)$ впервые была доказана для простых чисел n Гауссом в *Disquisitiones Arithmeticæ* (Leipzig, 1801), Art. 341, и Кронекером для произвольных n в *J. de Math. Pures et Appliquées* **19** (1854), 177–192.]

(c) $\Psi_1(x)=x-1$, и при простом p $\Psi_p(x)=1+x+\cdots+x^{p-1}$. Если $n>1$ нечетно, нетрудно доказать, что $\Psi_{2n}(x)=\Psi_n(-x)$. Если p делит n, второе тождество в п. (a) показывает, что $\Psi_{pn}(x)=\Psi_n(x^p)$. Если p не делит n, то имеем $\Psi_{pn}(x)=\Psi_n(x^p)/\Psi_n(x)$. Для составного же $n\le 15$ имеем $\Psi_4(x)=x^2+1$, $\Psi_6(x)=x^2-x+1$, $\Psi_8(x)=x^4+1$, $\Psi_9(x)=x^6+x^3+1$, $\Psi_{10}(x)=x^4-x^3+x^2-x+1$, $\Psi_{12}(x)=x^4-x^2+1$, $\Psi_{14}(x)=x^6-x^5+x^4-x^3+x^2-x+1$, $\Psi_{15}(x)=x^8-x^7+x^5-x^4+x^3-x+1$. [Формулу $\Psi_{pq}(x)=(1+x^p+\cdots+x^{(q-1)p})(x-1)/(x^q-1)$ можно использовать для того, чтобы показать, что все коэффициенты $\Psi_{pq}(x)$ равны ±1 или 0 при простых p и q; однако коэффициенты $\Psi_{pqr}(x)$ могут быть произвольно велики.]

33. Ложно; мы теряем все p_j с e_j, делимыми на p. Истинно, если $p>\deg(u)$ [см. упр. 36.]

34. [D. Y. Y. Yun, *Proc. ACM Symp. Symbolic and Algebraic Comp.* (1976), 26–35.] Установить $(t(x),v_1(x),w_1(x))\leftarrow \mathrm{GCD}(u(x),u'(x))$. Если $t(x)=1$, установить $e\leftarrow 1$; в противном случае устанавливать $(u_i(x),v_{i+1}(x),w_{i+1}(x))\leftarrow \mathrm{GCD}(v_i(x),w_i(x)-v_i'(x))$ для $i=1,2,\ldots,e-1$ до тех пор, пока не будет найдено $w_e(x)-v_e'(x)=0$. И наконец установить $u_e(x)\leftarrow v_e(x)$.

Для доказательства корректности этого алгоритма заметим, что он вычисляет полиномы $t(x)=u_2(x)u_3(x)^2u_4(x)^3\ldots$, $v_i(x)=u_i(x)u_{i+1}(x)u_{i+2}(x)\ldots$ и

$$w_i(x)=u_i'(x)u_{i+1}(x)u_{i+2}(x)\ldots+2u_i(x)u_{i+1}'(x)u_{i+2}(x)\ldots+3u_i(x)u_{i+1}(x)u_{i+2}'(x)\ldots+\cdots.$$

Имеем $t(x)\perp w_1(x)$, поскольку неприводимый множитель $u_i(x)$ делит все, кроме i-го, члены $w_1(x)$ и является взаимно простым по отношению к этому члену. Ясно и то, что $u_i(x)\perp v_{i+1}(x)$.

[Хотя в упр. 2, (b) доказывается, что большинство полиномов свободно от квадратов, на практике часто встречаются полиномы "с квадратами"; следовательно, этот метод становится весьма важным. Предложения по его усовершенствованию приводятся в Paul S. Wang and Barry M. Trager, *SICOMP* **8** (1979), 300–305. Разложение по модулю p, свободное от квадратов, рассматривается также в Bach and Shallit, *Algorithmic Number Theory* **1** (MIT Press, 1996), ответ к упр. 7.27.]

35. Имеем $w_j(x)=\gcd(u_j(x),v_j^*(x))\cdot\gcd(u_{j+1}(x),v_j(x))$, где

$$u_j^*(x)=u_j(x)u_{j+1}(x)\ldots \qquad \text{и} \qquad v_j^*(x)=v_j(x)v_{j+1}(x)\ldots.$$

[Юнь отмечает, что время работы свободного от квадратов разложения по методу упр. 34 не более чем в два раза превышает время вычисления $\gcd(u(x),u'(x))$. Кроме того, если

дан произвольный метод свободного от квадратов разложения, метод из этого упражнения приводит к процедуре поиска наибольшего общего делителя. (В случае, когда $u(x)$ и $v(x)$ свободны от квадратов, их наибольший общий делитель просто равен $w_2(x)$, где $w(x) = u(x)v(x) = w_1(x)w_2(x)^2$; все полиномы $u_j(x)$, $v_j(x)$, $u_j^*(x)$ и $v_j^*(x)$ свободны от квадратов.) Следовательно, задача преобразования примитивного полинома степени n в его свободное от квадратов представление с точки зрения вычислений *эквивалентна* задаче поиска наибольшего общего делителя двух полиномов степени n в смысле асимптотического времени работы для наихудшего случая.]

36. Пусть $U_j(x)$ — значение, вычисленное для "$u_j(x)$" по процедуре из упр. 34. Если $\deg(U_1)+2\deg(U_2)+\cdots = \deg(u)$, то $u_j(x) = U_j(x)$ для всех j. Однако в общем случае у нас будет $e < p$ и $U_j(x) = \prod_{k\geq 0} u_{j+pk}(x)$ для $1 \leq j < p$. Для дальнейшего разделения этих множителей можно вычислить $t(x)/(U_2(x)U_3(x)^2 \dots U_{p-1}(x)^{p-2}) = \prod_{j\geq p} u_j(x)^{p\lfloor j/p\rfloor} = z(x^p)$. После рекурсивного поиска свободного от квадратов представления $z(x) = (z_1(x), z_2(x), \dots)$ получим $z_k(x) = \prod_{0\leq j<p} u_{j+pk}(x)$, так что можно вычислить отдельные $u_i(x)$ по формуле $\gcd(U_j(x), z_k(x)) = u_{j+pk}(x)$ для $1 \leq j < p$. Полином $u_{pk}(x)$ останется, в то время как другие множители $z_k(x)$ будут удалены.

Примечание. Эта процедура очень проста, но реализующая ее программа длинна. Если необходима короткая, а не предельно эффективная программа для полного разложения полиномов по модулю p, то, вероятно, проще будет модифицировать программу разложения с различными степенями так, чтобы она давала $\gcd(x^{p^d} - x, u(x))$ несколько раз для одного и того же значения d до тех пор, пока наибольший общий делитель не станет равным 1. В этом случае нет необходимости начинать с вычисления $\gcd(u(x), u'(x))$ и удаления кратных множителей, как было предложено в тексте раздела, поскольку полином $x^{p^d} - x$ свободен от квадратов.

37. Точное значение вероятности составляет $\prod_{j\geq 1}(a_{jp}/p^j)^{k_j}/k_j!$, где k_j — количество d_i, равных j. Поскольку согласно упр. 4 $a_{jp}/p^j \approx 1/j$, получим формулу из упр. 1.3.3–21.

Примечания. Из этого упражнения следует, что если зафиксировать простое число p и случайным образом выбрать полином $u(x)$, то возникнет определенная вероятность его разложения указанным путем по модулю p. Более сложная задача состоит в определении вероятности при фиксированном полиноме $u(x)$ и "случайном" простом числе p, которое приводит к одному и тому же асимптотическому результату для почти всех $u(x)$. Г. Фробениус (G. Frobenius) доказал в 1880 году (хотя результат не был опубликован до 1896 года), что целый полином $u(x)$ разбивается по модулю p на множители степени $d_1, \dots, d_r$ при случайным образом выбранном большом простом p с вероятностью, равной числу перестановок в группе Галуа G из полиномов $u(x)$, которые имеют длины циклов $\{d_1, \dots, d_r\}$, деленной на общее количество перестановок в G. [Если $u(x)$ имеет рациональные коэффициенты и различные корни $\xi_1, \dots, \xi_n$ над полем комплексных чисел, его группа Галуа представляет собой единственную группу G перестановок, такую, что полином $\prod_{p(1)\dots p(n)\in G}(z + \xi_{p(1)}y_1 + \cdots + \xi_{p(n)}y_n) = U(z, y_1, \dots, y_n)$ имеет рациональные коэффициенты и неприводим над полем рациональных чисел; см. G. Frobenius, *Sitzungsberichte Königl. preuß. Akad. Wiss.* (Berlin, 1896), 689–703. Линейное отображение $x \mapsto x^p$ традиционно называется автоморфизмом Фробениуса, как в его знаменитой статье.] Кроме того, Б. Л. ван дер Варден (B. L. van der Waerden) доказал в 1934 году, что почти все полиномы степени n имеют множество всех $n!$ перестановок в качестве их группы Галуа [*Math. Annalen* **109** (1934), 13–16]. Поэтому почти все фиксированные неприводимые полиномы $u(x)$ будут множителями, как можно ожидать, по отношению к случайному выбору большого простого числа p. [См. также работу N. Chebotarev, *Math. Annalen* **95** (1926), в которой представлено обобщение теоремы Фробениуса для сопряженных классов групп Галуа.]

38. Из условий вытекает, что, когда $|z| = 1$, мы имеем либо $|u_{n-2}z^{n-2} + \cdots + u_0| < |u_{n-1}|-1 \le |z^n + u_{n-1}z^{n-1}|$, либо $|u_{n-3}z^{n-3}+\cdots+u_0| < u_{n-2}-1 \le |z^n+u_{n-2}z^{n-2}|$. Поэтому согласно теореме Роше [*J. École Polytechnique* **21**, 37 (1858), 1–34] $u(z)$ имеет минимум $n-1$ или $n-2$ корня внутри окружности $|z| = 1$. Если $u(z)$ приводим, он может быть записан как $v(z)w(z)$, где v и w — нормированные целые полиномы. Произведения корней v и корней w представляют собой ненулевые целые числа, так что каждый множитель имеет корень с абсолютной величиной ≥ 1. Следовательно, единственная возможность заключается в том, что v и w оба имеют в точности по одному такому корню и что $u_{n-1} = 0$. Эти корни должны быть действительны, поскольку корнями являются комплексно-сопряженные числа. Значит, $u(z)$ имеет действительный корень z_0 с $|z_0| \ge 1$. Но этого не может быть, так как, если $r = 1/z_0$, имеем $0 = |1+u_{n-2}r^2+\cdots+u_0r^n| \ge 1+u_{n-2}r^2-|u_{n-3}|r^3-\cdots-|u_0|r^n > 1$. [O. Perron, *Crelle* **132** (1907), 288–307; обобщения даны в A. Brauer, *Amer. J. Math.* **70** (1948), 423–432, **73** (1951), 717–720.]

39. Сначала докажем утверждение из указания. Пусть полином $u(x) = a(x-\alpha_1)\ldots(x-\alpha_n)$ имеет целые коэффициенты. Результант полинома $u(x)$ с полиномом $y - t(x)$ представляет собой детерминант, так что он является полиномом $r_t(y) = a^{\deg(t)}(y - t(\alpha_1))\ldots(y - t(\alpha_n))$ с целыми коэффициентами (см. упр. 4.6.1–12). Если $u(x)$ делит $v(t(x))$, то $v(t(\alpha_1)) = 0$. Следовательно, $r_t(y)$ имеет общий множитель с $v(y)$. Так что, если v неприводим, мы имеем $\deg(u) = \deg(r_t) \ge \deg(v)$.

Для данного неприводимого полинома $u(x)$, для которого требуется короткое доказательство неприводимости, можно предположить его нормированность согласно упр. 18, а также считать, что $\deg(u) \ge 3$. Идея заключается в том, чтобы показать существование полинома $t(x)$, такого, что полином $v(y) = r_t(y)$ является неприводимым по критерию из упр. 38. Тогда все множители $u(x)$ делят полином $v(t(x))$, и это доказывает, что $u(x)$ неприводим. Доказательство будет "укорочено", если коэффициенты $t(x)$ достаточно малы.

Можно показать, что полином $v(y) = (y - \beta_1)\ldots(y - \beta_n)$ удовлетворяет критерию упр. 38, если $n \ge 3$ и $\beta_1\ldots\beta_n \ne 0$ и если выполняется следующее "условие малости": $|\beta_j| \le 1/(4n)$, за исключением случаев, когда $j = n$ или $\beta_j = \overline{\beta}_n$ и $|\Re\beta_j| \le 1/(4n)$. Вычисления производятся непосредственно, с учетом того факта, что $|v_0| + \cdots + |v_n| \le (1 + |\beta_1|)\ldots(1 + |\beta_n|)$.

Пусть $\alpha_1, \ldots, \alpha_r$ — действительные числа, а $\alpha_{r+1}, \ldots, \alpha_{r+s}$ — комплексные, где $n = r + 2s$ и $\alpha_{r+s+j} = \overline{\alpha}_{r+j}$ для $1 \le j \le s$. Рассмотрим линейные выражения $S_j(a_0, \ldots, a_{n-1})$, определенные как $\Re(\sum_{i=0}^{n-1} a_i\alpha_j^i)$ для $1 \le j \le r+s$ и $\Im(\sum_{i=0}^{n-1} a_i\alpha_j^i)$ для $r+s < j \le n$. Если $0 \le a_i < b$ и $B = \lceil \max_{j=1}^{n-1} \sum_{i=0}^{n-1} |\alpha_i|^j \rceil$, имеем $|S_j(a_1, \ldots, a_{n-1})| < bB$. Таким образом, если выбрать $b > (16nB)^{n-1}$, должны существовать различные векторы $(a_0, \ldots, a_{n-1})$ и $(a'_0, \ldots, a'_{n-1})$, такие, что $\lfloor 8nS_j(a_0, \ldots, a_{n-1}) \rfloor = \lfloor 8nS_j(a'_0, \ldots, a'_{n-1}) \rfloor$ для $1 \le j < n$, поскольку имеется b^n векторов, но не более $(16nbB)^{n-1} < b^n$ возможных $(n-1)$-элементных наборов значений. Пусть $t(x) = (a_0 - a'_0) + \cdots + (a_{n-1} - a'_{n-1})x^{n-1}$ и $\beta_j = t(\alpha_j)$. Тогда "условие малости" удовлетворено. Кроме того, $\beta_j \ne 0$; в противном случае $t(x)$ должно делить $u(x)$. [*J. Algorithms* **2** (1981), 385–392.]

40. В данном множителе-кандидате $v(x) = x^d + a_{d-1}x^{d-1} + \cdots + a_0$ заменим каждый a_j рациональной дробью (по модулю p^e) с числителем и знаменателем $\le B$. Затем выполним умножение на наименьший общий знаменатель и посмотрим, осуществляет ли полученный полином деление $u(x)$ над кольцом целых чисел. Если нет, то не существует множителя $u(x)$ с коэффициентами, ограниченными B, которые сравнимы по модулю p^e с кратным $v(x)$.

41. Дэвид Бойд (David Boyd) заметил, что $4x^8+4x^6+x^4+4x^2+4 = (2x^4+4x^3+5x^2+4x+2) \times (2x^4 - 4x^3 + 5x^2 - 4x + 2)$, и нашел пример более высокой степени для доказательства того, что $c > 2$, если оно существует.

РАЗДЕЛ 4.6.3

1. x^m, где $m = 2^{\lfloor \lg n \rfloor}$ — наивысшая степень 2, меньшая или равная n.

2. Положим, что входное значение x находится в регистре A, а n — в памяти по адресу NN; выходное значение помещается в регистр X.

```
01  A1 ENTX 1       1        A1. Инициализация.
02     STX  Y       1        Y ← 1.
03     STA  Z       1        Z ← x.
04     LDA  NN      1        N ← n.
05     JAP  2F      1        Переход к шагу A2.
06     JMP  DONE    0        В противном случае ответ равен 1.
07  5H SRB  1     L+1-K
09     STA  N     L+1-K      N ← ⌊N/2⌋.
09  A5 LDA  Z       L        A5. Возведение Z в квадрат.
10     MUL  Z       L
11     STX  Z       L        Z ← Z × Z mod w.
12  A2 LDA  N       L        A2. Деление N пополам.
13  2H JAE  5B    L+1        Переход к шагу A5 при четном N.
14     SRB  1       K
15  A4 JAZ  4F      K        Переход, если N = 1.
16     STA  N     K-1        N ← ⌊N/2⌋.
17  A3 LDA  Z     K-1        A3. Умножение Y на Z.
18     MUL  Y     K-1
19     STX  Y     K-1        Y ← Z × Y mod w.
20     JMP  A5    K-1        Переход к шагу A5.
21  4H LDA  Z       1
22     MUL  Y       1        Окончательное умножение. ▮
```

Время работы составляет $21L + 16K + 8$, где $L = \lambda(n)$ на единицу меньше количества битов в двоичном представлении n, а $K = \nu(n)$ — количество единичных битов в таком представлении.

Для последовательной программы можно положить, что n достаточно мало́ для размещения в индексном регистре; в противном случае последовательное возведение в степень выходит за рамки данной задачи. Приведенная ниже программа помещает выходное значение в регистр A.

```
01  S1 LD1  NN     1     rI1 ← n.
02     STA  X      1     X ← x.
03     JMP  2F     1
04  1H MUL  X    N-1     rA × X mod w
05     SLAX 5    N-1          → rA.
06  2H DEC1 1      N     rI1 ← rI1 - 1.
07     J1P  1B     N     Опять умножить при rI1 > 0. ▮
```

Время ее работы составляет $14N - 7$; эта программа работает быстрее предыдущей при $n \leq 7$ и медленнее — при $n \geq 8$.

3. Последовательности показателей степени для этих методов следующие: (a) 1, 2, 3, 6, 7, 14, 15, 30, 60, 120, 121, 242, 243, 486, 487, 974, 975 [16 умножений]; (b) 1, 2, 3, 4, 8, 12, 24, 36, 72, 108, 216, 324, 325, 650, 975 [14 умножений]; (c) 1, 2, 3, 6, 12, 15, 30, 60, 120, 240, 243, 486, 972, 975 [13 умножений]; (d) 1, 2, 3, 6, 12, 15, 30, 60, 75, 150, 300, 600, 900, 975 [13 умножений]. [Наименьшее возможное число умножений равно 12; оно может

быть получено путем комбинирования метода множителя с бинарным методом, поскольку $975 = 15 \cdot (2^6 + 1)$.]

4. $(777777)_8 = 2^{18} - 1$.

5. T1. [Инициализация.] Установить LINKU$[j] \leftarrow 0$ для $0 \le j \le 2^r$ и установить $k \leftarrow 0$, LINKR$[0] \leftarrow 1$, LINKR$[1] \leftarrow 0$.

T2. [Изменение уровня.] (Теперь уровень k дерева связан слева направо, начиная с LINKR[0].) Если $k = r$, алгоритм завершается. В противном случае установить $n \leftarrow$ LINKR$[0]$, $m \leftarrow 0$.

T3. [Подготовка n.] (Теперь n — узел уровня k и m указывает на крайний справа в данный момент узел уровня $k+1$.) Установить $q \leftarrow 0$, $s \leftarrow n$.

T4. [Уже в дереве?] (Теперь s — узел на пути от корня дерева до n.) Если LINKU$[n+s] \neq 0$, перейти к шагу T6 (значение $n+s$ уже имеется в дереве).

T5. [Вставка под n.] Если $q = 0$, установить $m' \leftarrow n + s$. Затем установить LINKR$[n+s] \leftarrow q$, LINKU$[n+s] \leftarrow n$, $q \leftarrow n + s$.

T6. [Перемещение вверх.] Установить $s \leftarrow$ LINKU$[s]$. Если $s \neq 0$, вернуться к шагу T4.

T7. [Присоединение группы.] Если $q \neq 0$, установить LINKR$[m] \leftarrow q$, $m \leftarrow m'$.

T8. [Перемещение n.] Установить $n \leftarrow$ LINKR$[n]$. Если $n \neq 0$, вернуться к шагу T3.

T9. [Конец уровня.] Установить LINKR$[m] \leftarrow 0$, $k \leftarrow k + 1$ и вернуться к шагу T2. ▮

6. Докажем по индукции, что путем к числу $2^{e_0} + 2^{e_1} + \cdots + 2^{e_t}$ при $e_0 > e_1 > \cdots > e_t \ge 0$ является последовательность $1, 2, 2^2, \ldots, 2^{e_0}, 2^{e_0} + 2^{e_1}, \ldots, 2^{e_0} + 2^{e_1} + \cdots + 2^{e_t}$. Кроме того, последовательность показателей степеней на каждом уровне рассортирована в убывающем лексикографическом порядке.

7. Бинарный метод и метод множителя требуют при вычислении x^{2n} на один шаг больше, чем при вычислении x^n; метод дерева степеней требует не более одного дополнительного шага. Следовательно, (a) $15 \cdot 2^k$; (b) $33 \cdot 2^k$; (c) $23 \cdot 2^k$; $k = 0, 1, 2, 3, \ldots$.

8. Дерево степеней всегда включает узел $2m$ на уровень ниже m, кроме случая, когда он уже встречался на том же или на одном из предыдущих уровней; кроме того, дерево всегда включает узел $2m+1$ на один уровень ниже узла $2m$, если только его нет на том же уровне или на одном из предыдущих. [Неверно, что $2m$ является дочерним узлом m в дереве степеней для всех m; наименьший пример, когда это не так, — $m = 2138$, который появляется на уровне 15, в то время как узел 4276 появляется на уровне 16 в другом месте. В действительности $2m$ иногда встречается даже на том же уровне, что и m; наименьший такой пример — $m = 6029$.]

9. Начнем с установок $N \leftarrow n$, $Z \leftarrow x$ и $Y_q \leftarrow 1$ для $1 \le q < m$, где q — нечетно; в общем случае получим $x^n = Y_1 Y_3^3 Y_5^5 \ldots Y_{m-1}^{m-1} Z^N$ после выполнении алгоритма. Полагая, что $N > 0$, установим $k \leftarrow N \bmod m$, $N \leftarrow \lfloor N/m \rfloor$. Затем, если $k = 0$, установим $Z \leftarrow Z^m$ и повторим шаг; в противном случае, если $k = 2^p q$, где q — нечетно, установим $Z \leftarrow Z^{2^p}$, $Y_q \leftarrow Y_q \cdot Z$ и, если $N > 0$, установим $Z \leftarrow Z^{2^{e-p}}$ и повторим шаг. И наконец установим $Y_k \leftarrow Y_k \cdot Y_{k+2}$ для $k = m-3, m-5, \ldots, 1$. Искомый ответ — $Y_1 (Y_3 Y_5 \ldots Y_{m-1})^2$. (Около $m/2$ умножений представляют собой умножения на 1.)

10. Примените представление PARENT, обсуждавшееся в разделе 2.3.3: используйте таблицу $p[j]$, $1 \le j \le 100$, такую, что $p[1] = 0$ и $p[j]$ является номером узла, расположенного непосредственно над j для $j \ge 2$. (Тот факт, что каждый узел этого дерева имеет степень, не превышающую двух, не влияет на эффективность представления. Это только улучшает внешний вид дерева, используемого в качестве иллюстрации.)

11. 1, 2, 3, 5, 10, 20, (23 или 40), 43; 1, 2, 4, 8, 9, 17, (26 или 34), 43; 1, 2, 4, 8, 9, 17, 34, (43 или 68), 77; 1, 2, 4, 5, 9, 18, 36, (41 или 72), 77. Если бы любой из двух последних путей имелся в дереве, $n = 43$ было бы невозможно, поскольку дерево должно содержать либо 1, 2, 3, 5, либо 1, 2, 4, 8, 9.

12. Такое дерево невозможно, поскольку $l(n) \neq l^*(n)$ для некоторых n.

13. Для случая 1 используйте цепочку типа 1, за которой следует $2^{A+C} + 2^{B+C} + 2^A + 2^B$, либо воспользуйтесь методом множителя. Для случая 2 используйте цепочку типа 2, за которой следует $2^{A+C+1} + 2^{B+C} + 2^A + 2^B$. Для случая 3 используйте либо цепочку типа 5, за которой следует $2^A + 2^{A-1}$, либо метод множителя. Для случая 4 $n = 135 \cdot 2^D$, так что можно использовать метод множителя.

14. (a) Легко убедиться, что шаги $r-1$ и $r-2$ не являются малыми. Это позволяет считать, что шаг $r-1$ является малым, а шаг $r-2$ — нет. Если $c = 1$, то $\lambda(a_{r-1}) = \lambda(a_{r-k})$, так что $k = 2$; а поскольку $4 \le \nu(a_r) = \nu(a_{r-1}) + \nu(a_{r-k}) - 1 \le \nu(a_{r-1}) + 1$, имеем $\nu(a_{r-1}) \ge 3$, делая шаг $r-1$ звездным (чтобы цепочка $a_0, a_1, \ldots, a_{r-3}, a_{r-1}$ не включала только один малый шаг). Тогда $a_{r-1} = a_{r-2} + a_{r-q}$ для некоторого q, и, если заменить a_{r-2}, a_{r-1}, a_r на $a_{r-2}, 2a_{r-2}, 2a_{r-2} + a_{r-q} = a_r$, получится другая цепочка-контрпример, в которой шаг r является малым. Однако это невозможно. С другой стороны, если $c \ge 2$, то $4 \le \nu(a_r) \le \nu(a_{r-1}) + \nu(a_{r-k}) - 2 \le \nu(a_{r-1})$; следовательно, $\nu(a_{r-1}) = 4$, $\nu(a_{r-k}) = 2$ и $c = 2$. Это легко приводит к невозможной ситуации, возникающей в результате рассмотрения шести типов из доказательства теоремы B.

(b) Если $\lambda(a_{r-k}) < m-1$, имеем $c \ge 3$, так что $\nu(a_{r-k}) + \nu(a_{r-1}) \ge 7$ согласно (22); поэтому как $\nu(a_{r-k})$, так и $\nu(a_{r-1})$ оба ≥ 3. Все малые шаги должны быть $\le r-k$, а $\lambda(a_{r-k}) = m-k+1$. Если $k \ge 4$, необходимо иметь $c = 4$, $k = 4$, $\nu(a_{r-1}) = \nu(a_{r-4}) = 4$. Таким образом, $a_{r-1} \ge 2^m + 2^{m-1} + 2^{m-2}$ и a_{r-1} должно быть равно $2^m + 2^{m-1} + 2^{m-2} + 2^{m-3}$, но из $a_{r-4} \ge \frac{1}{8} a_{r-1}$ вытекает, что $a_{r-1} = 8a_{r-4}$. Значит, $k = 3$ и $a_{r-1} > 2^m + 2^{m-1}$. Поскольку $a_{r-2} < 2^m$ и $a_{r-3} < 2^{m-1}$, шаг $r-1$ должен быть удвоением, однако шаг $r-2$ не является удвоением, так как $a_{r-1} \neq 4a_{r-3}$. Кроме того, поскольку $\nu(a_{r-3}) \ge 3$, $r-3$ является звездным шагом и из $a_{r-2} = a_{r-3} + a_{r-5}$ должно следовать, что $a_{r-5} = 2^{m-2}$. Поэтому необходимо получить $a_{r-2} = a_{r-3} + a_{r-4}$. Как и в подобном случае, рассмотренном в тексте раздела, единственная возможность заключается в том, чтобы выполнялось $a_{r-4} = 2^{m-2} + 2^{m-3}$, $a_{r-3} = 2^{m-2} + 2^{m-3} + 2^{d+1} + 2^d$, $a_{r-1} = 2^m + 2^{m-1} + 2^{d+2} + 2^{d+1}$, но и это невозможно.

15. Ахим Фламменкамп (Achim Flammenkamp) [Diplomarbeit in Mathematics (Bielefeld University, 1991), Part 1] показал, что все числа n с $\lambda(n) + 3 = l(n) < l^*(n)$ имеют вид $2^A + 2^B + 2^C + 2^D + 2^E$, где $A > B > C > D > E$ и $B + E = C + D$; более того, они точно описываются как не соответствующие ни одному из восьми шаблонов (здесь $|\epsilon| \le 1$):

$$2^A + 2^{A-3} + 2^C + 2^{C-1} + 2^{2C+2-A}, \qquad 2^A + 2^{A-1} + 2^C + 2^D + 2^{C+D+1-A},$$
$$2^A + 2^B + 2^{2B-A+3} + 2^{2B+2-A} + 2^{3B+5-2A}, \qquad 2^A + 2^B + 2^{2B-A+\epsilon} + 2^D + 2^{B+D+\epsilon-A},$$
$$2^A + 2^B + 2^{B-1} + 2^D + 2^{D-1}, \qquad 2^A + 2^B + 2^{B-2} + 2^D + 2^{D-2} \ (A > B+1),$$
$$2^A + 2^B + 2^C + 2^{2B+\epsilon-A} + 2^{B+C+\epsilon-A}, \qquad 2^A + 2^B + 2^C + 2^{B+C+\epsilon-A} + 2^{2C+\epsilon-A}.$$

16. $l^B(n) = \lambda(n) + \nu(n) - 1$, так что, если $n = 2^k$, $l^B(n)/\lambda(n) = 1$, но если $n = 2^{k+1} - 1$, $l^B(n)/\lambda(n) = 2$.

17. Пусть $i_1 < \cdots < i_t$. Удалим все интервалы I_k, которые можно удалить без изменения объединения $I_1 \cup \cdots \cup I_t$. (Интервал $(j_k \,..\, i_k]$ может быть удален, если $j_{k+1} \le j_k$ либо $j_1 < j_2 < \cdots$ и $j_{k+1} \le i_{k-1}$.) Теперь объединим перекрывающиеся интервалы $(j_1 \,..\, i_1], \ldots, (j_d \,..\, i_d]$ в интервал $(j' \,..\, i'] = (j_1 \,..\, i_d]$ и заметим, что

$$a_{i'} < a_{j'}(1+\delta)^{i_1 - j_1 + \cdots + i_d - j_d} \le a_{j'}(1+\delta)^{2(i'-j')},$$

поскольку каждая точка $(j'\,..\,i']$ покрывается объединением $(j_i\,..\,i_1]\cup\cdots\cup(j_d\,..\,i_d]$ не более двух раз.

18. Назовем $f(m)$ "хорошей" функцией, если $(\log f(m))/m \to 0$ при $m\to\infty$. Произвольный полином от m является "хорошим". Произведение хороших функций — хорошая функция. Если $g(m)\to 0$ и c — положительная константа, то $c^{mg(m)}$ — хорошая функция; хороша также и $\binom{2m}{mg(m)}$, так как с учетом аппроксимации Стирлинга это эквивалентно утверждению $g(m)\log(1/g(m))\to 0$.

Теперь заменим каждый член суммы максимальным по s, t, v членом. Общее число членов — хорошее, так же как и $\binom{m+s}{t+v}$, $\binom{t+v}{v}\le 2^{t+v}$ и β^{2v}, потому что $(t+v)/m\to 0$. И наконец $\binom{(m+s)^2}{t}\le(2m)^{2t}/t! < (4em^2/t)^t$, где $(4e)^t$ — хорошая функция. Замещая t его верхней границей $(1-\epsilon/2)m/\lambda(m)$, показываем, что $(m^2/t)^t\le 2^{m(1-\epsilon/2)}f(m)$, где $f(m)$ — хорошая функция. Следовательно, вся сумма меньше, чем α^m для больших m, если $\alpha = 2^{1-\eta}$, где $0<\eta<\frac12\epsilon$.

19. (a) $M\cap N$, $M\cup N$, $M\uplus N$ соответственно; см. 4.5.2–(6), 4.5.2–(7).

(b) $f(z)g(z)$, $\operatorname{lcm}(f(z),g(z))$, $\gcd(f(z),g(z))$. (По тем же причинам, что и в п. (a), потому что нормированные неприводимые полиномы над полем комплексных чисел в точности представляют собой полиномы $z-\zeta$.)

(c) Законы коммутативности: $A\uplus B = B\uplus A$, $A\cup B = B\cup A$, $A\cap B = B\cap A$. Законы ассоциативности: $A\uplus(B\uplus C) = (A\uplus B)\uplus C$, $A\cup(B\cup C) = (A\cup B)\cup C$, $A\cap(B\cap C) = (A\cap B)\cap C$. Законы дистрибутивности: $A\cup(B\cap C) = (A\cup B)\cap(A\cup C)$, $A\cap(B\cup C) = (A\cap B)\cup(A\cap C)$, $A\uplus(B\cup C) = (A\uplus B)\cup(A\uplus C)$, $A\uplus(B\cap C) = (A\uplus B)\cap(A\uplus C)$. Законы идемпотентности: $A\cup A = A$, $A\cap A = A$. Законы поглощения: $A\cup(A\cap B) = A$, $A\cap(A\cup B) = A$, $A\cap(A\uplus B) = A$, $A\cup(A\uplus B) = A\uplus B$. Законы единицы и нуля: $\emptyset\uplus A = A$, $\emptyset\cup A = A$, $\emptyset\cap A = \emptyset$, где $\emptyset$ обозначает пустое мультимножество. Закон перечисления: $A\uplus B = (A\cup B)\uplus(A\cap B)$. Прочие свойства, аналогичные соответствующим свойствам множеств, вытекают из частичной упорядоченности, определенной правилом $A\subseteq B$ тогда и только тогда, когда $A\cap B = A$ (тогда и только тогда, когда $A\cup B = B$).

Примечания. Другие общие применения мультимножеств — нули и полюсы мероморфных функций, инварианты матриц в канонической форме, инварианты конечных Абелевых групп и т. д. Мультимножества могут быть полезны в комбинаторике и в теории мер. Терминальные строки нециркулярной контекстно-свободной грамматики образуют мультимножество, которое является множеством тогда и только тогда, когда грамматика свободна от неопределенностей. В статье автора в *Theoretical Studies in Computer Science*, edited by J. D. Ullman (Academic Press, 1992), 1–13, обсуждаются дальнейшие применения к контекстно-свободным грамматикам и вводится операция $A\dot\cap B$, где каждый элемент, который встречается a раз в A и b раз в B, встречается в $A\dot\cap B$ ab раз.

Хотя мультимножества появляются в математике достаточно часто, во многих случаях они трактуются неуклюже, поскольку в настоящее время не существует стандартного пути рассмотрения множеств с повторяющимися элементами. Некоторые математики выразили свое мнение о том, что отсутствие адекватной терминологии и обозначений для этой глобальной концепции определенно затрудняет развитие математики. (Мультимножество, конечно, формально эквивалентно отображению множества на неотрицательные целые числа, но эта эквивалентность имеет слишком малое значение для творческого математического мышления.) Автор обсуждал этот вопрос со многими специалистами в 60-х годах, пытаясь найти приемлемое решение вопроса. Вот некоторые из предлагавшихся для этой концепции названий: список (list), связка (bunch), мешок (bag), куча (heap), проба

(sample), взвешенное множество (weighted set), коллекция (collection*), комплект (suite). Однако все эти они конфликтуют с существующей терминологией, имеют неуместное дополнительное значение или слишком неудобны для произношения и написания. Наконец, стало ясно, что для такой важной концепции необходимо собственное имя, и оно — "мультимножество" (multiset) — было предложено Н. Г. де Брейном (N. G. de Bruijn). Его предложение широко распространилось в 70-х годах и теперь является стандартным термином.

Запись "$A \uplus B$" была выбрана автором, чтобы избежать конфликтов с существующими обозначениями и усилить аналогию с объединением множеств. Для этого нежелательно использовать "$A + B$", поскольку специалисты в области алгебры приняли $A + B$ как запись для мультимножества $\{\alpha + \beta \mid \alpha \in A \text{ и } \beta \in B\}$. Пусть A является мультимножеством неотрицательных целых чисел, а $G(z) = \sum_{n \in A} z^n$ — производящая функция, соответствующая A. (Очевидно, что производящие функции с неотрицательными целыми коэффициентами находятся во взаимно однозначном соответствии с мультимножествами неотрицательных чисел.) Если $G(z)$ соответствует A, а $H(z)$ соответствует B, то $G(z) + H(z)$ соответствует $A \uplus B$, а $G(z)H(z)$ соответствует $A + B$. Если построить производящие функции Дирихле $g(z) = \sum_{n \in A} 1/n^z$, $h(z) = \sum_{n \in B} 1/n^z$, то произведение $g(z)h(z)$ будет соответствовать произведению мультимножеств AB.

20. Тип 3: $(S_0, \ldots, S_r) = (M_{00}, \ldots, M_{r0}) = (\{0\}, \ldots, \{A\}, \{A-1, A\}, \{A-1, A, A\}, \{A-1, A-1, A, A, A\}, \ldots, \{A+C-3, A+C-3, A+C-2, A+C-2, A+C-2\})$.

Тип 5: $(M_{00}, \ldots, M_{r0}) = (\{0\}, \ldots, \{A\}, \{A-1, A\}, \ldots, \{A+C-1, A+C\}, \{A+C-1, A+C-1, A+C\}, \ldots, \{A+C+D-1, A+C+D-1, A+C+D\})$; $(M_{01}, \ldots, M_{r1}) = (\emptyset, \ldots, \emptyset, \emptyset, \ldots, \emptyset, \{A+C-2\}, \ldots, \{A+C+D-2\})$, $S_i = M_{i0} \uplus M_{i1}$.

21. Пусть, например, $u = 2^{8q+5}$, $x = (2^{(q+1)u} - 1)/(2^u - 1) = 2^{qu} + \cdots + 2^u + 1$, $y = 2^{(q+1)u} + 1$. Тогда $xy = (2^{2(q+1)u} - 1)/(2^u - 1)$. Если $n = 2^{4(q+1)u} + xy$, имеем $l(n) \le 4(q+1)u + q + 2$ по теореме F, но $l^*(n) = 4(q+1)u + 2q + 2$ согласно теореме H.

22. Подчеркните все, кроме $u - 1$ вставок, использованных при вычислении x.

23. Теорема G (подчеркнуто все).

24. Используйте числа $(B^{a_i} - 1)/(B - 1)$, $0 \le i \le r$, подчеркнутые, когда подчеркнуто a_i, и $c_k B^{i-1}(B^{b_j} - 1)/(B-1)$ для $0 \le j < t$, $0 < i \le b_{j+1} - b_j$, $1 \le k \le l^0(B)$, подчеркнутые, когда подчеркнуто c_k, где $c_0, c_1, \ldots$ — l^0-цепочка минимальной длины для B. Для доказательства второго неравенства положите $B = 2^m$ и используйте (3). (Второе неравенство редко приводит к улучшению теоремы G, если вообще приводит.)

25. Будем считать, что $d_k = 1$. Используем правило R $A_{k-1} \ldots A_1$, где A_j = "XR", если $d_j = 1$, и A_j = "R" в противном случае, и где "R" обозначает взятие квадратного корня, а "X" — умножение на x. Например, если $y = (.1101101)_2$, правило выглядит следующим образом: R R XR XR R XR XR. (Существуют бинарные алгоритмы для извлечения квадратного корня, пригодные для использования на аппаратном уровне, время работы которых сравнимо со временем выполнения деления; компьютеры с таким аппаратным обеспечением могли бы таким образом вычислять более общие дробные степени с помощью технологии, описанной в этом упражнении.)

26. Если известна пара (F_k, F_{k-1}), то имеем $(F_{k+1}, F_k) = (F_k + F_{k-1}, F_k)$ и $(F_{2k}, F_{2k-1}) = (F_k^2 + 2F_kF_{k-1}, F_k^2 + F_{k-1}^2)$, поэтому для вычисления (F_n, F_{n-1}) может быть применен бинарный метод с $O(\log n)$ арифметическими операциями. Возможно, лучше применить пары значений (F_k, L_k), где $L_k = F_{k-1} + F_{k+1}$ (см. упр. 4.5.4–15); тогда имеем $(F_{k+1}, L_{k+1}) = (\frac{1}{2}(F_k + L_k), \frac{1}{2}(5F_k + L_k))$, $(F_{2k}, L_{2k}) = (F_kL_k, L_k^2 - 2(-1)^k)$.

---

* Заметим, что этот термин, хотя и не прижился в математике, широко распространен в программировании, особенно — в объектно-ориентированном. — *Прим. перев.*

В случае обобщенной линейной рекуррентности $x_n = a_1x_{n-1} + \cdots + a_dx_{n-d}$ x_n можно найти за $O(d^3 \log n)$ арифметических операций, вычислив n-ю степень соответствующей матрицы размера $d \times d$. [Это наблюдение было сделано в работе J. C. P. Miller and D. J. Spencer Brown, *Comp. J.* **9** (1966), 188–190.] На самом деле, как заметил Ричард Брент (Richard Brent), количество операций может быть сведено до $O(d^2 \log n)$ или даже до $O(d \log d \log n)$ с помощью упр. 4.7–6, если сначала вычислить $x^n \bmod (x^d - a_1x^{d-1} - \cdots - a_d)$, а затем заменить x^j на x_j.

27. Наименьшее n, требующее s малых шагов, должно быть равным $c(r)$ для некоторого r. Если $c(r) < n < c(r+1)$, то $l(n) - \lambda(n) \le r - \lambda(c(r)) = l(c(r) - \lambda(c(r)))$. Поэтому ответ для $1 \le s \le 6$ — 3, 7, 29, 127, 1903, 65131; вероятно, $c(28)$ потребует 7.

28. (a) $x \nabla y = x \vee y \vee (x+y)$, где "$\vee$" означает побитовое "или" (см. упр. 4.6.2–26). Ясно, что $\nu(x \nabla y) \le \nu(x \vee y) + \nu(x \wedge y) = \nu(x) + \nu(y)$. (b) Заметим, во-первых, что $A_{i-1}/2^{d_{i-1}} \subseteq A_i/2^{d_i}$ для $1 \le i \le r$, и, во-вторых, что в неудвоении $d_j = d_{i-1}$; в противном случае $a_{i-1} \ge 2a_j \ge a_j + a_k = a_i$. Следовательно, $A_j \subseteq A_{i-1}$ и $A_k \subseteq A_{i-1}/2^{d_j - d_k}$. (c) Простая индукция по i, но близкие шаги требуют более пристального внимания. Будем говорить, что m обладает свойством $P(\alpha)$, если все единицы в его бинарном представлении располагаются в последовательных блоках $\ge \alpha$ в строке. Если m и m' обладают свойством $P(\alpha)$, то им обладает и $m \nabla m'$; если m обладает свойством $P(\alpha)$, то $\rho(m)$ обладает свойством $P(\alpha + \delta)$. Следовательно, B_i обладает свойством $P(1 + \delta c_i)$. И наконец, если m обладает свойством $P(\alpha)$, то $\nu(\rho(m)) \le (\alpha+\delta)\nu(m)/\alpha$, для $\nu(m) = \nu_1 + \cdots + \nu_q$, где каждый размер блока $\nu_j \ge \alpha$. Следовательно, $\nu(\rho(m)) \le (\nu_1 + \delta) + \cdots + (\nu_q + \delta) \le (1 + \delta/\alpha)\nu_1 + \cdots + (1 + \delta/\alpha)\nu_q$. (d) Пусть $f = b_r + c_r$ — число неудвоений, а s — число малых шагов. Если $f \ge 3.271 \lg \nu(n)$, то $s \ge \lg \nu(n)$, как и требовалось, в соответствии с (16). В противном случае $a_i \le (1 + 2^{-\delta})^{b_i} 2^{c_i + d_i}$ для $0 \le i \le r$. Значит, $n \le ((1 + 2^{-\delta})/2)^{b_r} 2^r$ и $r \ge \lg n + b_r - b_r \lg(1 + 2^{-\delta}) \ge \lg n + \lg \nu(n) - \lg(1 + \delta c_r) - b_r \lg(1 + 2^{-\delta})$. Пусть $\delta = \lceil \lg(f+1) \rceil$; тогда $\ln(1 + 2^{-\delta}) \le \ln(1 + 1/(f+1)) \le 1/(f+1) \le \delta/(1+\delta f)$. Отсюда следует, что $\lg(1+\delta x) + (f-x)\lg(1+2^{-\delta}) \le \lg(1+\delta f)$ для $0 \le x \le f$. Поэтому окончательно $l(n) \ge \lg n + \lg \nu(n) - \lg(1 + (3.271 \lg \nu(n)) \lceil \lg(1 + 3.271 \lg \nu(n)) \rceil)$. [*Theoretical Comp. Sci.* **1** (1975), 1–12.]

29. В только что процитированной работе Шёнхаге усовершенствовал метод из упр. 28 для доказательства того, что $l(n) \ge \lg n + \lg \nu(n) - 2.13$ для всех n. Может ли оставшийся пробел быть закрытым?

30. $n = 31$ представляет собой наименьший пример; $l(31) = 7$, но цепочка со сложением и вычитанием 1, 2, 4, 8, 16, 32, 31 имеет длину 6. [После доказательства теоремы E Эрдеш (Erdős) указал, что тот же результат справедлив и для аддитивно-вычитательных цепочек. Шёнхаге распространил нижнюю границу из упр. 28 на цепочки со сложением и вычитанием с $\nu(n)$, замененные $\overline{\nu}(n)$, как определено в упр. 4.1–34. Обобщенный бинарный метод возведения в степень справа налево, в котором используется $\lambda(n) + \overline{\nu}(n) - 1$ умножений, когда заданы и x, и x^{-1}, может основываться на представлении α_n из этого упражнения.]

32. См. *Discrete Math.* **23** (1978), 115–119. [Эта модель цен соответствует умножению больших чисел классическим методом, подобным алгоритму 4.3.1M. Эмпирические результаты с более общей моделью, в которой цена равна $(a_j a_k)^{\beta/2}$, были получены в работе D. P. McCarthy, *Math. Comp.* **46** (1986), 603–608; эта модель более близка к методам "быстрого умножения" из раздела 4.3.3, когда два n-битовых числа умножаются за $O(n^\beta)$ шагов, но функция цены $a_j a_k^{\beta-1}$ в действительности подходит в большей степени (см. упр. 4.3.3–13). X. Зантема (H. Zantema) проанализировал аналогичную задачу при стоимости i-го шага, равной $a_j + a_k$, а не $a_j a_k$ (см. *J. Algorithms* **12** (1991), 281–307). В этом случае оптимальные цепочки имеют общую цену $\frac{5}{2}n + O(n^{1/2})$. Кроме того, оптимальная

аддитивная цена при нечетном n не ниже $\frac{5}{2}(n-1)$; она равна этой величине тогда и только тогда, когда n может быть записано как произведение чисел вида 2^k+1.]

33. Восемь; четыре пути вычисления $39 = 12+12+12+3$ и два пути вычисления $79 = 39+39+1$.

34. Утверждение истинно. Метками в приведенном графе бинарной цепочки являются $\lfloor n/2^k \rfloor$ для $k = e_0, \ldots, 0$; в дуальном графе они равны $1, 2, \ldots, 2^{e_0}, n$. [Аналогично m-арный метод "справа налево" из упр. 9 дуален по отношению к методу "слева направо".]

35. Бинарной цепочке эквивалентны 2^t цепочек; это число должно составлять 2^{t-1}, если $e_0 = e_1 + 1$. Количество цепочек, эквивалентных схеме алгоритма A, равно количеству путей для вычисления суммы $t+2$ чисел, два из которых одинаковы. Эта величина равна $\frac{1}{2}f_{t+1} + \frac{1}{2}f_t$, где f_m — количество способов вычисления суммы $m+1$ различных чисел. Учитывая коммутативность, мы видим, что f_m равно 2^{-m}, умноженному на $(m+1)!$, умноженному на количество бинарных деревьев из m узлов, так что $f_m = (2m-1)(2m-3)\ldots 1$.

36. Построим $2^m - m - 1$ произведений $x_1^{e_1}\ldots x_m^{e_m}$ для каждой последовательности степеней, таких, что $0 \le e_k \le 1$ и $e_1 + \cdots + e_m \ge 2$. Пусть $n_k = (d_{k\lambda}\ldots d_{k1}d_{k0})_2$. Чтобы завершить вычисления, найдем $x_1^{d_{1\lambda}}\ldots x_m^{d_{m\lambda}}$, затем возведем в квадрат и умножим на $x_1^{d_{1i}}\ldots x_m^{d_{mi}}$ для $i = \lambda-1, \ldots, 1, 0$. [Страус показал в *AMM* **71** (1964), 807–808, что $2\lambda(n)$ может быть заменено $(1+\epsilon)\lambda(n)$ для любого $\epsilon > 0$ посредством обобщения этого бинарного метода на 2^k-арный, как в теореме D.]

37. Сначала вычислим 2^q для $1 \le q \le \lambda(n_m)$, а затем — каждый $n = n_j$ при помощи следующего варианта 2^k-арного метода: для всех нечетных $q < 2^k$ вычислим $f_q = \sum\{2^{kt+e} \mid d_t = 2^e q\}$, где $n = (\ldots d_1 d_0)_{2^k}$, за не более чем $\lfloor \frac{1}{k}\lg n \rfloor$ шагов и вычислим $n = \sum q f_q$ за не более чем $\sum l(q) + 2^{k-1}$ последующих шагов. Количество шагов на одно $n_j \le \lfloor \frac{1}{k}\lg n \rfloor + O(k2^k)$ и равно $\lambda(n)/\lambda\lambda(n) + O(\lambda(n)\lambda\lambda\lambda(n)/\lambda\lambda(n)^2)$ при $k = \lfloor \lg\lg n - 3\lg\lg\lg n \rfloor$.

[Обобщение теоремы E дает соответствующую нижнюю границу (*SICOMP* **5** (1976), 100–103).]

38. Следующее построение Д. Дж. Ньюмена (D. J. Newman) доказывает наилучшую известную верхнюю границу: пусть $k = p_1\ldots p_r$ — произведение первых r простых чисел. Вычислите k и все квадратичные остатки по модулю k за $O(2^{-r}k\log k)$ шагов (поскольку имеется примерно $2^{-r}k$ квадратичных остатков). Вычислите также все множители k, которые $\le m^2$, примерно за m^2/k последующих шагов. Теперь m сложений хватит для вычисления $1^2, 2^2, \ldots, m^2$. Имеем $k = \exp(p_r + O(p_r/(\log p_r)^{1000}))$, где p_r получается из ответа к упр. 4.5.4–36 [см., например, Greene and Knuth, *Math. for the Analysis of Algorithms* (Boston: Birkhäuser, 1981), §4.1.6]. Так, из выбора

$$r = \lfloor (1 + \tfrac{1}{2}\ln 2/\lg\lg m)\ln m/\ln\ln m \rfloor$$

следует, что $l(1^2, \ldots, m^2) = m + O(m\cdot\exp(-(\frac{1}{2}\ln 2 - \epsilon)\ln m/\ln\ln m))$.

С другой стороны, Д. Добкин (D. Dobkin) и Р. Липтон (R. Lipton) показали, что для любого $\epsilon > 0$, $l(1^2, \ldots, m^2) > m + m^{2/3-\epsilon}$ при достаточно большом m [*SICOMP* **9** (1980), 121–125].

39. Величина $l([n_1, n_2, \ldots, n_m])$ равна минимуму величины

$$\text{дуги} - \text{вершины} + m,$$

взятой по всем ориентированным графам, имеющим m вершин s_j, входные степени которых равны нулю, и одну вершину t с нулевой выходной степенью, где имеется ровно n_j ориентированных путей от s_j до t для $1 \le j \le m$. $l(n_1, n_2, \ldots, n_m)$ представляет собой

минимум величины

$$\text{дуги} - \text{вершины} + 1,$$

взятой по всем ориентированным графам, имеющим одну вершину s, входная степень которой равна нулю, и m вершин t_j, выходные степени которых равны нулю, где имеется ровно n_j ориентированных путей от s до t_j для $1 \le j \le m$. Эти задачи дуальны одна по отношению к другой, если изменить направление всех дуг. [См. *J. Algorithms* **2** (1981), 13–21.]

Примечание. Х. Х. Пападимитру (C. H. Papadimitriou) заметил, что рассмотренная задача является частным случаем более общей теоремы. Пусть $N = (n_{ij})$ — матрица неотрицательных целых чисел размера $m \times p$, не имеющая полностью нулевых строк или столбцов. Можно определить $l(N)$ как минимальное число умножений, требующихся для вычисления множества монономов $\{x_1^{n_{1j}} \dots x_m^{n_{mj}} \mid 1 \le j \le p\}$. Теперь $l(N)$ является также минимумом величины

$$\text{дуги} - \text{вершины} + m,$$

взятой по всем ориентированным графам, которые имеют m вершин s_i с нулевыми входящими степенями и p вершин t_j с нулевыми выходными степенями, где имеется в точности n_{ij} ориентированных путей от s_i к t_j для каждых i и j. В соответствии с дуальностью имеем $l(N) = l(N^T) + m - p$. [*Bulletin of the Europ. Assoc. Theor. Comp. Sci.* **13** (February, 1981), 2–3.]

Н. Пиппенгер (N. Pippenger) доказал глубокое обобщение результатов упр. 36 и 37. Пусть $L(m,p,n)$ — максимум $l(N)$, который получен по всем матрицам N размера $m \times p$, состоящих из неотрицательных целых чисел $n_{ij} \le n$. Тогда $L(m,p,n) = \min(m,p) \lg n + H/\lg H + O(m+p+H(\log\log H)^{1/2}(\log H)^{-3/2})$, где $H = mp\lg(n+1)$. [*SICOMP* **9** (1980), 230–250.]

40. Согласно упр. 39 достаточно показать, что $l(m_1 n_1 + \cdots + m_t n_t) \le l(m_1, \dots, m_t) + l([n_1, \dots, n_t])$. Но это очевидно, поскольку сначала можно построить $\{x^{m_1}, \dots, x^{m_t}\}$, а затем вычислить мононом $(x^{m_1})^{n_1} \dots (x^{m_t})^{n_t}$.

Примечание. Ниже приведен один строгий способ формулировки теоремы Оливоса: если $a_0, \dots, a_r$ и $b_0, \dots, b_s$ являются произвольными аддитивными цепочками, то

$$l(\textstyle\sum c_{ij} a_i b_j) \le r + s + \sum c_{ij} - 1$$

для любой матрицы размера $(r+1)\times(s+1)$, состоящей из неотрицательных целых чисел c_{ij}.

41. [*SICOMP* **10** (1981), 638–646.] Указанная формула может быть доказана, только если $A \ge 9m^2$. Поскольку это полином от m и поскольку задача поиска минимального покрытия вершин NP-сложна (см. раздел 7.9), задача вычисления $l(n_1, \dots, n_m)$ является NP-полной. [Неизвестно, является ли задача вычисления $l(n)$ NP-полной. Однако весьма правдоподобно, что оптимальная цепочка для, скажем, $\sum_{k=0}^{m-1} n_{k+1} 2^{Ak^2}$ приведет к появлению оптимальной цепочки для $\{n_1, \dots, n_m\}$ при достаточно больших A.]

РАЗДЕЛ 4.6.4

1. Присвоить $y \leftarrow x^2$, затем вычислить $((\dots(u_{2n+1}y + u_{2n-1})y + \cdots)y + u_1)x$.

2. Замена x в (2) полиномом $x + x_0$ приводит к следующей процедуре.

G1. Шаг G2 для $k = n, n-1, \dots, 0$ (в таком порядке) и остановиться.

G2. Присвоить $v_k \leftarrow u_k$, затем присвоить $v_j \leftarrow v_j + x_0 v_{j+1}$ для $j = k, k+1, \dots, n-1$. (Когда $k = n$, просто присвойте $v_n \leftarrow u_n$.) ▮

Вычисления оказываются идентичными вычислениям на шагах H1 и H2, но выполняются в другом порядке. (Фактически это воплощение первоначальных идей Ньютона, связанные с использованием схемы (2).)

3. Коэффициент при x^k равен полиному от y, который можно вычислить по правилу Горнера: $(\dots(u_{n,0}x+(u_{n-1,1}y+u_{n-1,0}))x+\cdots)x+((\dots(u_{0,n}y+u_{0,n-1})y+\cdots)y+u_{0,0})$. [Для "однородного" полинома, такого как $u_nx^n+u_{n-1}x^{n-1}y+\cdots+u_1xy^{n-1}+u_0y^n$, более эффективна другая схема: если $0<|x|\le|y|$, сначала разделить x на y, вычислить полином от x/y, а затем умножить на y^n.]

4. Правило (2) включает $4n$ или $3n$ вещественных умножений и $4n$ или $7n$ вещественных сложений; схема (3) хуже: она требует $4n+2$ или $4n+1$ умножений, $4n+2$ или $4n+5$ сложений.

5. Одно умножение для вычисления x^2; $\lfloor n/2\rfloor$ умножений и $\lfloor n/2\rfloor$ сложений для вычисления первой строки; $\lceil n/2\rceil$ умножений и $\lceil n/2\rceil-1$ сложений для вычисления второй строки и одно сложение для суммирования обеих строк. Всего $n+1$ умножений и n сложений.

6. **J1.** Вычислить и запомнить значения x_0^2, x_0^3, $\dots$, $x_0^{\lceil n/2\rceil}$.

J2. Присвоить $v_j\leftarrow u_jx_0^{j-\lfloor n/2\rfloor}$ для $0\le j\le n$.

J3. При $k=0,1,\dots,n-1$ присвоить $v_j\leftarrow v_j+v_{j+1}$ при $j=n-1,\dots,k+1,k$.

J4. Присвоить $v_j\leftarrow v_jx_0^{\lfloor n/2\rfloor-j}$ для $0\le j\le n$. ▮

Здесь $(n^2+n)/2$ сложений, $n+\lceil n/2\rceil-1$ умножений и n делений. Еще одно умножение и деление можно сэкономить, если трактовать v_n и v_0 как частные случаи [см. *SIGACT News* **7**, 3 (Summer, 1975), 32–34].

7. Пусть $x_j=x_0+jh$. Рассмотрим (42) и (44). Присвоить $y_j\leftarrow u(x_j)$ для $0\le j\le n$. Для $k=1,2,\dots,n$ (в таком порядке) присвоить $y_j\leftarrow y_j-y_{j-1}$ при $j=n,n-1,\dots,k$ (в таком порядке). Присвоить $\beta_j\leftarrow y_j$ для всех j.

Тем не менее, как объяснялось в разделе, ошибка округления будет накапливаться, даже если операции (5) выполняются с великолепной точностью. Лучшим способом осуществления инициализации, когда (5) выполняется с арифметикой с фиксированной точкой, является выбор $\beta_0,\dots,\beta_n$ таким образом, что

$$\begin{pmatrix}\binom{0}{0}&\binom{0}{1}&\cdots&\binom{0}{n}\\ \binom{d}{0}&\binom{d}{1}&\cdots&\binom{d}{n}\\ \vdots&\vdots&&\vdots\\ \binom{nd}{0}&\binom{nd}{1}&\cdots&\binom{nd}{n}\end{pmatrix}\begin{pmatrix}\beta_0\\ \beta_1\\ \vdots\\ \beta_n\end{pmatrix}=\begin{pmatrix}u(x_0)\\ u(x_d)\\ \vdots\\ u(x_{nd})\end{pmatrix}+\begin{pmatrix}\epsilon_0\\ \epsilon_1\\ \vdots\\ \epsilon_n\end{pmatrix},$$

где $|\epsilon_0|$, $|\epsilon_1|$, $\dots$, $|\epsilon_n|$ настолько малы, насколько это возможно. [H. Hassler, *Proc. 12th Spring Conf. Computer Graphics* (Bratislava: Comenius University, 1996), 55–66.]

8. См. (43).

9. [*Combinatorial Mathematics* (Buffalo: Math. Assoc. of America, 1963), 26–28.] Эту формулу можно рассматривать как применение принципа включения и исключения (раздел 1.3.3), поскольку сумма членов для $n-\epsilon_1-\cdots-\epsilon_n=k$ равна сумме всех $x_{1j_1}x_{2j_2}\dots x_{nj_n}$, для которых k значений j_i не появятся. Прямое доказательство можно получить, если заметить, что коэффициент при $x_{1j_1}\dots x_{nj_n}$ равен

$$\sum(-1)^{n-\epsilon_1-\cdots-\epsilon_n}\epsilon_{j_1}\dots\epsilon_{j_n};$$

если j_k различны, это выражение равно единице, однако, если $j_1,\dots,j_n\ne k$, равно нулю, поскольку члены с $\epsilon_k=0$ погашают члены с $\epsilon_k=1$.

Для эффективного вычисления суммы можно начать с $\epsilon_1 = 1, \epsilon_2 = \cdots = \epsilon_n = 0$, а затем пройти все комбинации ϵ_k так, чтобы заменить только одно ϵ при переходе от одного члена к другому одним ближайшим членом (см. "серый код" в главе 7). Вычисление первого члена сводится к $n-1$ умножению; из последующих 2^n-2 членов каждый требует n сложений, $n-1$ умножений и еще одно сложение. Общая сумма операций такова: $(2^n-1)(n-1)$ умножений и $(2^n-2)(n+1)$ сложений. Необходимо только $n+1$ ячеек для временного хранения промежуточных результатов, одна — для основной частичной суммы и одна — для каждого множителя текущего произведения.

10. $\sum_{1\le k<n}(k+1)\binom{n}{k+1} = n(2^{n-1}-1)$ умножений и $\sum_{1\le k<n} k\binom{n}{k+1} = n2^{n-1}-2^n+1$ сложений. Это приблизительно половина арифметических операций, необходимых в методе из упр. 9, хотя этот метод требует более сложной программы проверки последовательности. Приблизительно $\binom{n}{\lceil n/2\rceil}+\binom{n}{\lceil n/2\rceil-1}$ ячеек памяти должно быть использовано для временного хранения результатов, и это число растет экспоненциально (порядка $2^n/\sqrt{n}$).

Метод в данном упражнении эквивалентен необычному матричному разложению перманента функции, приведенному в работе Юрката и Райзера (Jurkat and Ryser, *J. Algebra* **5** (1967), 342–357). В некотором смысле его также можно рассматривать как применение (39) и (40).

11. Если матрица достаточно плотная, то эффективные методы вычисления приближенного значения известны; см. A. Sinclair, *Algorithms for Random Generation and Counting* (Boston: Birkhäuser, 1993). Но поставленная задача требует вычисления точных значений. Возможно, существует метод вычисления перманента с $O(c^n)$ операциями для некоторого $c < 2$.

12. Здесь кратко изложены результаты развития этой замечательной научно-исследовательской проблемы: Дж. Гопкрофт (J. Hopcroft) и Л. Р. Кер (L. R. Kerr) между прочим доказали, что необходимо семь умножений для умножения матриц размера 2×2 по модулю 2 [*SIAM J. Appl. Math.* **20** (1971), 30–36]. Р. Л. Проберт (R. L. Probert) показал, что все схемы с семью умножениями, в которых каждое умножение — это умножение линейной комбинации элементов одной матрицы на линейную комбинацию элементов другой матрицы, должно иметь по крайней мере 15 сложений [*SICOMP* **5** (1976), 187–203]. Ранг тензора умножений матриц размера 2×2 больше 7 для любого поля [В. Я. Пан, *J. Algorithms* **2** (1981), 301–310], если ранг тензора $T(2,3,2)$ произведения матриц размера 2×3 и 3×2 равен 11 [В. Б. Алексеев, *J. Algorithms* **6** (1985), 71–85]. Для умножения матриц размера $n\times n$ лучшая верхняя грань известна при $n=3$. Ее нашел Д. Д. Ладерман (J. D. Laderman) [*Bull. Amer. Math. Soc.* **82** (1976), 126–128], который показал, что достаточно 23 некоммутативных умножений. Его конструкцию обобщил Андрей Сикора. Он предложил метод, требующий $n^3-(n-1)^2$ некоммутативных умножений и $n^3-n^2+11(n-1)^2$ сложений; этот результат также сводится к (36), когда $n=2$ [*Lecture Notes in Comp. Sci.* **53** (1977), 504–512]. Для $n=5$ рекорд в настоящее время составляет 100 некоммутативных умножений [О. М. Макаров, *СССР, Вычисл. мат. и матем. физика* **27**, 1 (1987), 205–207]. Лучшую нижнюю грань, насколько известно, предложили Ж.-К. Лафон (J.-C. Lafon) и Ш. Виноград (S. Winograd), которые показали, что необходимо $2n^2-1$ нескалярных умножений и $mn+ns+m-n-1$ для размерности $m\times n\times s$ ["A lower bound for the multiplicative complexity of the product of two matrices", Centre de Calcul, Univ. Louis Pasteur (Strasbourg, 1979)]. Если все вычисления производятся без деления, то для числа операций немного лучшие нижние грани получены Н. Х. Бшути (N. H. Bshouty) [*SICOMP* **18** (1989), 759–765]. Он показал, что для умножения по модулю 2 матрицы размера $m\times n$ на матрицу размера $n\times s$ требуется по крайней мере

$$\textstyle\sum_{k=0}^{j-1}\lfloor ms/2^k\rfloor + \frac{1}{2}(n+(n \bmod j))(n-(n \bmod j)-j) + n \bmod j$$

умножений, когда $n \geq s \geq j \geq 1$. Полагая $m = n = s$ и $j \approx \lg n$, получим, что это число равно $2.5n^2 - \frac{1}{2}n \lg n + O(n)$.

Лучшая верхняя грань, известная для больших n, обсуждалась в разделе после формулы (36).

13. Суммируя геометрические ряды, найдем, что $F(t_1, \ldots, t_n)$ равно

$$\sum_{0 \leq s_1 < m_1, \ldots, 0 \leq s_n < m_n} \exp(-2\pi i(s_1 t_1/m_1 + \cdots + s_n t_n/m_n) f(s_1, \ldots, s_n))/m_1 \ldots m_n.$$

Обратное преобразование от $m_1 \ldots m_n$ можно найти, выполнив обычное преобразование и заменив t_j значениями $m_j - t_j$, когда $t_j \neq 0$ (см. упр. 4.3.3–9).

[Если рассматривать $F(t_1, \ldots, t_n)$ как коэффициент $x_1^{t_1} \ldots x_n^{t_n}$ в полиноме от нескольких переменных, то дискретное преобразование Фурье приравнивается к вычислению этого полинома в корнях из единицы и обратное преобразование равнозначно нахождению интерполяционного полинома.]

14. Пусть $m_1 = \cdots = m_n = 2$, $F(t_1, t_2, \ldots, t_n) = F(2^{n-1}t_n + \cdots + 2t_2 + t_1)$ и $f(s_1, s_2, \ldots, s_n) = f(2^{n-1}s_1 + 2^{n-2}s_2 + \cdots + s_n)$. Заметим, что между t_k и s_i существует взаимно обратное соответствие. Пусть также $g_k(s_k, \ldots, s_n, t_k)$ — это ω в степени $2^{k-1}t_k(s_n + 2s_{n-1} + \cdots + 2^{n-k}s_k)$.

На каждой итерации мы, по существу, берем 2^{n-1} пар комплексных чисел (α, β) и заменяем их числами $(\alpha + \zeta\beta, \alpha - \zeta\beta)$, где ζ — подходящая степень ω. Следовательно, $\zeta = \cos\theta + i \sin\theta$ для некоторого θ. Если предпочесть упрощения, когда $\zeta = \pm 1$ или $\pm i$, вся работа сведется к $((n-3) \cdot 2^{n-1} + 2)$ комплексным умножениям и $n \cdot 2^n$ комплексным сложениям. Техника упр. 41 может быть использована для уменьшения числа умножений и сложений действительных чисел с помощью операций для комплексных чисел.

Количество умножений комплексных чисел можно уменьшить приблизительно на 25% без изменения числа сложений, объединяя шаги k и $k+1$ для $k = 1, 3, \ldots$. Это означает, что 2^{n-2} четверок $(\alpha, \beta, \gamma, \delta)$ заменятся выражением

$$(\alpha + \zeta\beta + \zeta^2\gamma + \zeta^3\delta,\ \alpha + i\zeta\beta - \zeta^2\gamma - i\zeta^3\delta,\ \alpha - \zeta\beta + \zeta^2\gamma - \zeta^3\delta,\ \alpha - i\zeta\beta - \zeta^2\gamma + i\zeta^3\delta).$$

Общее число умножений комплексных чисел, когда n — четное, в результате уменьшается до $(3n-2)2^{n-3} - 5\lfloor 2^{n-1}/3 \rfloor$.

В этих вычислениях предполагается, что заданные числа $F(t)$ — комплексные. Если $F(t)$ — действительные числа, то $f(s)$ — комплексно-сопряженные к $f(2^n - s)$. Таким образом, можно избежать операций, вычисляя только 2^n независимых действительных чисел $f(0)$, $\Re f(1)$, $\ldots$, $\Re f(2^{n-1}-1)$, $f(2^{n-1})$, $\Im f(1)$, $\ldots$, $\Im f(2^{n-1}-1)$. Все вычисления в этом случае могут быть сведены к операциям с 2^n действительными числами, если учитывается тот факт, что $f_k(s_{n-k+1}, \ldots, s_n, t_1, \ldots, t_{n-k})$ будут комплексно-сопряженными к $f_k(s'_{n-k+1}, \ldots, s'_n, t_1, \ldots, t_{n-k})$, когда $(s_1 \ldots s_n)_2 + (s'_1 \ldots s'_n)_2 \equiv 0$ по модулю 2^n. Около половины умножений и сложений так же необходимы, как и для комплексных чисел.

[Алгоритм быстрого преобразования Фурье открыт К. Ф. Гауссом в 1805 году и независимо открывался в дальнейшем. Наиболее интересно это сделали Дж. В. Кули (J. W. Cooley) и Дж. У. Тьюки (J. W. Tukey), *Math. Comp.* **19** (1965), 297–301. Его интересная история прослежена Дж. В. Кули, П. А. У. Левисом и П. Д. Уэлчем (J. W. Cooley, P. A. W. Lewis and P. D. Welch, *Proc. IEEE* **55** (1967), 1675–1677); M. T. Heideman, D. H. Johnson, and C. S. Burrus, *IEEE ASSP Magazine* **1**, 4 (October, 1984), 14–21. Детали, связанные с его применением, обсуждались сотнями авторов; это обсуждение очень кратко изложил Чарлз Ван Лоан (Charles Van Loan, *Computational Frameworks for the Fast Fourier Transform* (Philadelphia: SIAM, 1992)). Обзор быстрого преобразования Фурье на конечных группах приводится в работе M. Clausen and U. Baum, *Fast Fourier Transforms* (Mannheim: Bibliographisches Institut Wissenschaftsverlag, 1993).]

15. (a) Указание вытекает из интегрирования и индукции. Пусть $f^{(n)}(\theta)$ берется по всем значениям, лежащим между A и B включительно, когда θ изменяется от $\min(x_0,\ldots,x_n)$ до $\max(x_0,\ldots,x_n)$. Заменяя $f^{(n)}$ каждой из этих граней в вышеприведенном интеграле, получим $A/n! \le f(x_0,\ldots,x_n) \le B/n!$. (b) Достаточно доказать это для $j = n$. Пусть f — интерполяционный полином Ньютона, тогда $f^{(n)}$ равна постоянной $n!\alpha_n$. [См. *The Mathematical Papers of Isaac Newton*, edited by D. T. Whiteside, **4** (1971), 36–51, 70–73.]

16. Выполнить умножения и сложения (43), как операции над полиномами. (Частный случай, когда $x_0 = x_1 = \cdots = x_n$, рассмотрен в упр. 2. Мы воспользовались этим методом на шаге C8 алгоритма 4.3.3T.)

17. Например, когда $n = 5$, имеем

$$u_{[5]}(x) = \frac{\dfrac{y_0}{x-x_0} - \dfrac{5y_1}{x-x_1} + \dfrac{10y_2}{x-x_2} - \dfrac{10y_3}{x-x_3} + \dfrac{5y_4}{x-x_4} - \dfrac{y_5}{x-x_5}}{\dfrac{1}{x-x_0} - \dfrac{5}{x-x_1} + \dfrac{10}{x-x_2} - \dfrac{10}{x-x_3} + \dfrac{5}{x-x_4} - \dfrac{1}{x-x_5}}$$

независимо от значения h.

18. $\alpha_0 = \frac{1}{2}(u_3/u_4 + 1)$, $\beta = u_2/u_4 - \alpha_0(\alpha_0 - 1)$, $\alpha_1 = \alpha_0\beta - u_1/u_4$, $\alpha_2 = \beta - 2\alpha_1$, $\alpha_3 = u_0/u_4 - \alpha_1(\alpha_1 + \alpha_2)$, $\alpha_4 = u_4$.

19. Поскольку α_5 — основной коэффициент, можно предположить без потери общности, что $u(x)$ — нормированный полином (т. е. что $u_5 = 1$). Тогда α_0 — корень уравнения $40z^3 - 24u_4z^2 + (4u_4^2 + 2u_3)z + (u_2 - u_3u_4) = 0$. Это уравнение всегда имеет по крайней мере один действительный корень, а может иметь три. Поскольку α_0 определено, получим $\alpha_3 = u_4 - 4\alpha_0$, $\alpha_1 = u_3 - 4\alpha_0\alpha_3 - 6\alpha_0^2$, $\alpha_2 = u_1 - \alpha_0(\alpha_0\alpha_1 + 4\alpha_0^2\alpha_3 + 2\alpha_1\alpha_3 + \alpha_0^3)$, $\alpha_4 = u_0 - \alpha_3(\alpha_0^4 + \alpha_1\alpha_0^2 + \alpha_2)$.

Для заданного полинома решим кубическое уравнение $40z^3 - 120z^2 + 80z = 0$; это приведет к трем решениям: $(\alpha_0,\alpha_1,\alpha_2,\alpha_3,\alpha_4,\alpha_5) = (0,-10,13,5,-5,1)$, $(1,-20,68,1,11,1)$, $(2,-10,13,-3,27,1)$.

20.
```
LDA    X          STA   TEMP2      FADD  =α1=       FMUL  TEMP1
FADD   =α3=       FMUL  TEMP2      FMUL  TEMP2      FADD  =α4=
STA    TEMP1      STA   TEMP2      FADD  =α2=       FMUL  =α5=
FADD   =α0-α3=
```

21. $z = (x+1)x - 2$, $w = (x+5)z + 9$, $u(x) = (w + z - 8)w - 8$; or $z = (x+9)x + 26$, $w = (x-3)z + 73$, $u(x) = (w + z - 24)w - 12$.

22. $\alpha_6 = 1$, $\alpha_0 = -1$, $\alpha_1 = 1$, $\beta_1 = -2$, $\beta_2 = -2$, $\beta_3 = -2$, $\beta_4 = 1$, $\alpha_3 = -4$, $\alpha_2 = 0$, $\alpha_4 = 4$, $\alpha_5 = -2$. Образуем $z = (x-1)x + 1$, $w = z + x$ и $u(x) = ((z - x - 4)w + 4)z - 2$. Одно из семи суммирований можно сэкономить, если вычислить $w = x^2 + 1$, $z = w - x$.

23. (a) Можно применить индукцию по n; результат тривиальный, если $n < 2$. Если $f(0) = 0$, то результат справедлив для полинома $f(z)/z$; значит, он выполняется и для $f(z)$. Если $f(iy) = 0$ для некоторого действительного $y \neq 0$, то $g(\pm iy) = h(\pm iy) = 0$. Так как результат справедлив для $f(z)/(z^2+y^2)$, он выполняется и для $f(z)$. Следовательно, можно предположить, что $f(z)$ не имеет корней, действительная часть которых равна нулю. Сейчас точное количество обходов от начала координат траекторией равно числу корней $f(z)$ внутри области, которых не больше одного. Когда R большое, траектория $f(Re^{it})$ для $\pi/2 \le t \le 3\pi/2$ будет обходить начало координат по часовой стрелке приблизительно $n/2$ раз, так что траектории $f(it)$ для $-R \le t \le R$ должны обходить начало координат

против часовой стрелки по крайней мере $n/2-1$ раз. Для четного n это означает, что $f(it)$ пересекает мнимую ось по крайней мере $n-2$ раз и действительную ось — $n-3$ раз. Для n нечетных $f(it)$ пересекает действительную ось по крайней мере $n-2$ раз и мнимую ось — $n-3$ раз. Это и будут корни $g(it)=0$ и $h(it)=0$ соответственно.

(b) Если нет, то g или h должны иметь корни вида $a+bi$ с $a\neq 0$ и $b\neq 0$. Но должно подразумеваться существование хотя бы трех других корней, а именно $a-bi$ и $-a\pm bi$, тогда как $g(z)$ и $h(z)$ имеют максимум n корней.

24. Корнями u являются -7, $-3\pm i$, $-2\pm i$ и -1; допустимые значения c равны 2 и 4 (но *не* 3, так как при $c=3$ сумма корней равна нулю). Случай 1, $c=2$: $p(x)=(x+5)(x^2+2x+2)(x^2+1)(x-1)=x^6+6x^5+6x^4+4x^3-5x^2-2x-10$; $q(x)=6x^2+4x-2=6(x+1)(x-\frac{1}{3})$. Пусть $\alpha_2=-1$, $\alpha_1=\frac{1}{3}$; $p_1(x)=x^4+6x^3+5x^2-2x-10=(x^2+6x+\frac{16}{3})(x^2-\frac{1}{3})-\frac{74}{9}$; $\alpha_0=6$, $\beta_0=\frac{16}{3}$, $\beta_1=-\frac{74}{9}$. Случай 2, $c=4$: аналогично получаем $\alpha_2=9$, $\alpha_1=-3$, $\alpha_0=-6$, $\beta_0=12$, $\beta_1=-26$.

25. $\beta_1=\alpha_2$, $\beta_2=2\alpha_1$, $\beta_3=\alpha_7$, $\beta_4=\alpha_6$, $\beta_5=\beta_6=0$, $\beta_7=\alpha_1$, $\beta_8=0$, $\beta_9=2\dot{\alpha}_1-\alpha_8$.

26. (a) $\lambda_1=\alpha_1\times\lambda_0$, $\lambda_2=\alpha_2+\lambda_1$, $\lambda_3=\lambda_2\times\lambda_0$, $\lambda_4=\alpha_3+\lambda_3$, $\lambda_5=\lambda_4\times\lambda_0$, $\lambda_6=\alpha_4+\lambda_5$. (b) $\kappa_1=1+\beta_1x$, $\kappa_2=1+\beta_2\kappa_1x$, $\kappa_3=1+\beta_3\kappa_2x$, $u(x)=\beta_4\kappa_3=\beta_1\beta_2\beta_3\beta_4x^3+\beta_2\beta_3\beta_4x^2+\beta_3\beta_4x+\beta_4$. (c) Если любой коэффициент равен нулю, то коэффициент при x^3 также должен быть нулем в (b), в то время как (a) дает произвольный полином $\alpha_1x^3+\alpha_2x^2+\alpha_3x+\alpha_4$ степени ≤ 3.

27. Иначе должен существовать ненулевой полином $f(q_n,\ldots,q_1,q_0)$ с целыми коэффициентами, такой, что $q_n\cdot f(q_n,\ldots,q_1,q_0)=0$ для всех множеств действительных чисел $(q_n,\ldots,q_0)$. Это невозможно, так как легко доказать индукцией по n, что ненулевой полином всегда принимает ненулевые значения. (См. упр. 4.6.1–16. Однако этот результат несправедлив, если рассматривать *конечные* поля вместо полей действительных чисел.)

28. Неопределенные величины $\alpha_1,\ldots,\alpha_s$ образуют алгебраический базис для области полиномов $Q[\alpha_1,\ldots,\alpha_s]$, где Q — поле рациональных чисел. Так как $s+1$ больше числа элементов базиса, то полиномы $f_j(\alpha_1,\ldots,\alpha_s)$ алгебраически зависимы. Это означает, что существует ненулевой полином g с рациональными коэффициентами, такой, что $g(f_0(\alpha_1,\ldots,\alpha_s),\ldots,f_s(\alpha_1,\ldots,\alpha_s))$ тождественно равен нулю.

29. Пусть заданы $j_0,\ldots,j_t\in\{0,1,\ldots,n\}$. Существуют ненулевые полиномы с целыми коэффициентами, такие, что $g_j(q_{j_0},\ldots,q_{j_t})=0$ для всех $(q_n,\ldots,q_0)$ в R_j, $1\leq j\leq m$. Поэтому произведение $g_1g_2\ldots g_m$ равно нулю для всех $(q_n,\ldots,q_0)$ в $R_1\cup\cdots\cup R_m$.

30. Начиная с конструкции в теореме M, докажем, что $m_p+(1-\delta_{0m_c})$ из β_k можно эффективно исключить. Если μ_i соответствует параметру умножения, то $\mu_i=\beta_{2i-1}\times(T_{2i}+\beta_{2i})$. Нужно прибавить $c\beta_{2i-1}\beta_{2i}$ к каждому β_j, для которого $c\mu_i$ появляется в T_j, и заменить β_{2i} нулем. Это приведет к удалению одного параметра для каждого умножения параметров. Если μ_i — первое умножение в цепочке, то $\mu_i=(\gamma_1x+\theta_1+\beta_{2i-1})\times(\gamma_2x+\theta_2+\beta_{2i})$, где γ_1, γ_2, θ_1, θ_2 — полиномы от $\beta_1,\ldots,\beta_{2i-2}$ с целыми коэффициентами. Здесь θ_1 и θ_2 может быть "поглощено" β_{2i-1} и β_{2i} соответственно. Таким образом, можно предположить, что $\theta_1=\theta_2=0$. Сейчас добавим $c\beta_{2i-1}\beta_{2i}$ к каждому β_j, для которого $c\mu_i$ появляется в T_j; добавим $\beta_{2i-1}\gamma_2/\gamma_1$ к β_{2i} и положим β_{2i-1} равным нулю. Множество результатов не изменится после этого удаления β_{2i-1}, кроме величин $\alpha_1,\ldots,\alpha_s$, таких, что γ_1 равно нулю. [Это доказательство, по существу, предложено В. Я. Паном, *Успехи мат. наук* **21**, 1 (Январь-февраль, 1966), 103–134.] Последний случай можно рассмотреть, как в доказательстве теоремы A, поскольку полиномы с $\gamma_1=0$ можно вычислить, устраняя β_{2i} (как и в первой конструкции, где μ_i соответствует умножению параметра).

31. В противном случае можно добавить одно умножение параметра в качестве последнего шага и прийти к противоречию теоремы C. (Данное упражнение является улучшенным

вариантом теоремы A в этом частном случае, поскольку существует только n степеней свободы для коэффициентов нормированного полинома степени n.)

32. $\lambda_1 = \lambda_0 \times \lambda_0$, $\lambda_2 = \alpha_1 \times \lambda_1$, $\lambda_3 = \alpha_2 + \lambda_2$, $\lambda_4 = \lambda_3 \times \lambda_1$, $\lambda_5 = \alpha_3 + \lambda_4$. Необходимо по крайней мере три умножения, чтобы вычислить u_4x^4 (см. раздел 4.6.3), и по крайней мере два сложения по теореме A.

33. Необходимо иметь $n+1 \leq 2m_c + m_p + \delta_{0m_c}$ и $m_c + m_p = (n+1)/2$. Таким образом, не существует умножений параметров. Сейчас первые λ_i, главный коэффициент которых (как полиномов от x) не является целым числом, должны быть получены посредством сложения в цепочке и должен существовать по крайней мере $n+1$ параметр. Таким образом, должно быть хотя бы $n+1$ сложений параметров.

34. Преобразовать данную цепочку шаг за шагом и также определить "содержание" c_i в λ_i следующим образом (интуитивно понятно, что c_i — старший коэффициент в λ_i). Определим $c_0 = 1$. (a) Если шаг имеет вид $\lambda_i = \alpha_j + \lambda_k$, заменить его шагом вида $\lambda_i = \beta_j + \lambda_k$, где $\beta_j = \alpha_j/c_k$, и определить $c_i = c_k$. (b) Если шаг имеет вид $\lambda_i = \alpha_j - \lambda_k$, заменить его шагом $\lambda_i = \beta_j + \lambda_k$, где $\beta_j = -\alpha_j/c_k$, и определить $c_i = -c_k$. (c) Если шаг вида $\lambda_i = \alpha_j \times \lambda_k$, заменить его шагом $\lambda_i = \lambda_k$ (шаг будет позже удален) и определить $c_i = \alpha_j c_k$. (d) Если шаг имеет вид $\lambda_i = \lambda_j \times \lambda_k$, оставить его без изменения и определить $c_i = c_j c_k$.

После того как этот процесс будет завершен, удалить все шаги вида $\lambda_i = \lambda_k$, заменяя λ_i на λ_k в каждом последующем шаге, который использует λ_j. Затем добавить последний шаг $\lambda_{r+1} = \beta \times \lambda_r$, где $\beta = c_r$. Это и есть требуемая схема, так как легко проверить, что новые λ_i — это только старые λ_i, деленные на множитель c_i. β_k — заданные функции от α_j; деление на нуль — не проблема, так как, если какое-то $c_k = 0$, должно быть $c_r = 0$ (следовательно, коэффициент при x^n равен нулю); иначе λ_k никогда не приведут к окончательному результату.

35. Так как существует по крайней мере пять шагов параметров, результат тривиален, если не существует хотя бы одного умножения параметров. Рассмотрим метод, в котором три умножения могут образовать u_4x^4. Мы видим, что должно быть три умножения параметров и два умножения в цепочке, поэтому четыре сложения-вычитания должны быть шагами параметров и упр. 34 применимо. Сейчас можно предположить, что используются только операции сложения и что имеется цепочка вычисления общего вида *нормированного* полинома четвертой степени с *двумя* умножениями в цепочке и четырьмя сложениями параметров. Единственно возможная схема такого вида, которая вычисляет полином четвертой степени, имеет вид

$$\begin{aligned}
\lambda_1 &= \alpha_1 + \lambda_0 \\
\lambda_2 &= \alpha_2 + \lambda_0 \\
\lambda_3 &= \lambda_1 \times \lambda_2 \\
\lambda_4 &= \alpha_3 + \lambda_3 \\
\lambda_5 &= \alpha_4 + \lambda_3 \\
\lambda_6 &= \lambda_4 \times \lambda_5 \\
\lambda_7 &= \alpha_5 + \lambda_6
\end{aligned} .$$

Фактически эта цепочка имеет на одну операцию сложения больше, чем нужно, но любая корректная схема может быть задана в таком виде, если одни из α_k являются функциями от других α_j. λ_7 имеет вид $(x^2+Ax+B)(x^2+Ax+C)+D = x^4+2Ax^3+(E+A^2)x^2+EAx+F$, где $A = \alpha_1 + \alpha_2$, $B = \alpha_1\alpha_2 + \alpha_3$, $C = \alpha_1\alpha_2 + \alpha_4$, $D = \alpha_6$, $E = B + C$, $F = BC + D$. Поскольку они содержат только три независимых параметра, то λ_7 не может представлять нормированный полином четвертой степени общего вида.

36. Как и в решении к упр. 35, можно предположить, что цепочка вычисляет нормированный полином шестой степени общего вида, используя только три умножения в цепочке

и шесть сложений параметров. Вычисления должны проводиться по одной из двух общих схем:

$$\begin{array}{l}\lambda_1 = \alpha_1 + \lambda_0\\ \lambda_2 = \alpha_2 + \lambda_0\\ \lambda_3 = \lambda_1 \times \lambda_2\\ \lambda_4 = \alpha_3 + \lambda_0\\ \lambda_5 = \alpha_4 + \lambda_3\\ \lambda_6 = \lambda_4 \times \lambda_5\\ \lambda_7 = \alpha_5 + \lambda_6\\ \lambda_8 = \alpha_6 + \lambda_6\\ \lambda_9 = \lambda_7 \times \lambda_8\\ \lambda_{10} = \alpha_7 + \lambda_9\end{array}, \qquad \begin{array}{l}\lambda_1 = \alpha_1 + \lambda_0\\ \lambda_2 = \alpha_2 + \lambda_0\\ \lambda_3 = \lambda_1 \times \lambda_2\\ \lambda_4 = \alpha_3 + \lambda_3\\ \lambda_5 = \alpha_4 + \lambda_3\\ \lambda_6 = \lambda_4 \times \lambda_5\\ \lambda_7 = \alpha_5 + \lambda_3\\ \lambda_8 = \alpha_6 + \lambda_6\\ \lambda_9 = \lambda_7 \times \lambda_8\\ \lambda_{10} = \alpha_7 + \lambda_9\end{array},$$

где, как и в упр. 35, дополнительное сложение введено для включения более общего случая. Ни одна из этих схем не может вычислить нормированный полином шестой степени общего вида, поскольку в первом случае это полином вида

$$(x^3 + Ax^2 + Bx + C)(x^3 + Ax^2 + Bx + D) + E,$$

а во втором — вида

$$(x^4 + 2Ax^3 + (E + A^2)x^2 + EAx + F)(x^2 + Ax + G) + H;$$

оба полинома содержат только пять независимых параметров.

37. Пусть $p_0(x) = u_n x^n + u_{n-1}x^{n-1} + \cdots + u_0$ и $q_0(x) = x^n + v_{n-1}x^{n-1} + \cdots + v_0$. Для $1 \le j \le n$ разделим $p_{j-1}(x)$ на нормированный полином $q_{j-1}(x)$ и получим $p_{j-1}(x) = \alpha_j q_{j-1}(x) + \beta_j q_j(x)$. Предположим, что нормированный полином $q_j(x)$ степени $n - j$ существует и удовлетворяет этому соотношению. Оно справедливо для почти всех рациональных функций. Пусть $p_j(x) = q_{j-1}(x) - xvq_j(x)$. Это определение означает, что $\deg(p_n) < 1$, поэтому можно положить $\alpha_{n+1} = p_n(x)$.

Для данной рациональной функции имеем

j	α_j	β_j	$q_j(x)$	$p_j(x)$
0			$x^2 + 8x + 19$	$x^2 + 10x + 29$
1	1	2	$x + 5$	$3x + 19$
2	3	4	1	5

,

значит, $u(x)/v(x) = p_0(x)/q_0(x) = 1 + 2/(x + 3 + 4/(x + 5))$.

Замечание. Обычная рациональная функция установленного вида имеет $2n + 1$ "степень свободы" в том смысле, что она, по существу, имеет $2n + 1$ независимых параметров. Если расширить понятие цепочки полиномов на понятие цепочки отношений полиномов, которые допускают операции деления так же, как и операции сложения, вычитания и умножения (см. упр. 71), то можно получить следующий результат с незначительными изменениями в доказательствах теорем A и M: *цепочка отношений полиномов с q шагами сложений-вычитаний имеет максимум $q + 1$ степеней свободы. А цепочка отношений полиномов с m шагами умножений-делений имеет максимум $2m + 1$ степеней свободы.* Следовательно, цепочка отношений полиномов, которая вычисляет почти все рациональные функции заданного вида, должна иметь по крайней мере $2n$ сложений-вычитаний и n умножений-делений. Метод этого упражнения оптимален.

38. Если $n = 0$, то теорема, несомненно, справедлива. Предположим, что n положительное и что задана цепочка полиномов, которая вычисляет $P(x; u_0, \ldots, u_n)$, где каждый из параметров α_j заменен действительным числом. Пусть $\lambda_i = \lambda_j \times \lambda_k$ — первое умножение в цепочке, которое включает один из $u_0, \ldots, u_n$; такой шаг должен существовать, если учесть значения ранга матрицы A. Без потери общности можно предположить, что λ_j

включают в себя u_n; таким образом, λ_j имеют вид $h_0u_0+\cdots+h_nu_n+f(x)$, где $h_0,\ldots,h_n$ — действительные, $h_n\neq 0$, и $f(x)$ — полином с действительными коэффициентами. (h_k и коэффициенты $f(x)$ получены из значений, определяемых α_j.)

Сейчас заменим шаг i шагом $\lambda_i=\alpha\times\lambda_k$, где α — произвольное действительное число. (Можно взять $\alpha=0$. Вообще говоря, α используется здесь только для того, чтобы показать достаточную гибкость доказательства.) Выполним дополнительно следующие шаги:

$$\lambda=(\alpha-f(x)-h_0u_0-\cdots-h_{n-1}u_{n-1})/h_n.$$

Новые шаги включают только операции сложения и умножения (на подходящие новые параметры). Наконец всюду в цепочке заменим $\lambda_{-n-1}=u_n$ этим новым элементом λ. В результате получим цепочку, которая вычисляет

$$Q(x;u_0,\ldots,u_{n-1})=P(x;u_0,\ldots,u_{n-1},(\alpha-f(x)-h_0u_0-\cdots-h_{n-1}u_{n-1})/h_n)$$

и имеет на одно умножение в цепочке меньше. Доказательство будет окончено, если можно показать, что Q удовлетворяет предположениям. Значение $(\alpha-f(x))/h_n$ приводит к, возможно, уменьшенному значению m и новому вектору B'. Если A_0, A_1, ..., A_n — столбцы матрицы A (эти векторы линейно независимы относительно вещественных чисел), соответствующая матрице Q новая матрица A' имеет вектор-столбцы

$$A_0-(h_0/h_n)A_n,\qquad\ldots,\qquad A_{n-1}-(h_{n-1}/h_n)A_n$$

плюс, возможно, несколько строк нулей, отвечающих за уменьшение значения m. Эти столбцы, естественно, также линейно независимы. По индукции цепочка, которая вычисляет Q, имеет по крайней мере $n-1$ умножений в цепочке, тогда как начальная цепочка имеет по крайней мере n.

[Пан также показал, что деление не приводит к улучшению правила (см. *Проблемы кибернетики* **7** (1962), 21–30). Обобщения вычисления нескольких полиномов с несколькими переменными с различного рода предпосылками и без них рассмотрены Ш. Виноградом (S. Winograd, *Comm. Pure and Applied Math.* **23** (1970), 165–179).]

39. Индукцией по m. Пусть $w_m(x)=x^{2m}+u_{2m-1}x^{2m-1}+\cdots+u_0$, $w_{m-1}(x)=x^{2m-2}+v_{2m-3}x^{2m-3}+\cdots+v_0$, $a=\alpha_1+\gamma_m$, $b=\alpha_m$ и пусть

$$f(r)=\textstyle\sum_{i,j\geq 0}(-1)^{i+j}\binom{i+j}{j}u_{r+i+2j}a^ib^j.$$

Следовательно, $v_r=f(r+2)$ для $r\geq 0$ и $\delta_m=f(1)$. Если $\delta_m=0$ и a задано, то получим полином степени $m-1$ от b с главным коэффициентом $\pm(u_{2m-1}-ma)=\pm(\gamma_2+\cdots+\gamma_m-m\gamma_m)$.

В неопубликованных заметках Моцкин почти всегда полагал $\delta_k=0$, выбирая γ_k таким образом, что основной коэффициент не равен нулю, когда m четное, и равен нулю, когда m нечетное. Тогда можно почти всегда предположить, что b — действительный корень полинома нечетной степени.

40. Нет; Ш. Виноград (S. Winograd) нашел метод вычисления всех полиномов степени 13 только с семью (возможно, комплексными) умножениями [*Comm. Pure and Applied Math.* **25** (1972), 455–457]. Л. Ревах (L. Revah) нашла схемы, которые вычисляют почти все полиномы степени $n\geq 9$ с $\lfloor n/2\rfloor+1$ (возможно, комплексными) умножениями [*SICOMP* **4** (1975), 381–392]. Она также показала, что, когда $n=9$, можно достичь $\lfloor n/2\rfloor+1$ умножений, но по крайней мере с $n+3$ сложениями. К слову, достаточно много операций сложения (см. упр. 39), оговорок "почти все" и "возможно, комплексными" исчезли. В. Я. Пан [*STOC* **10** (1978), 162–172; IBM Research Report RC7754 (1979)] нашел схемы с $\lfloor n/2\rfloor+1$ комплексными умножениями и минимальным числом $n+2+\delta_{n9}$ комплексных

сложений для всех нечетных $n \geq 9$. Его метод для $n = 9$ имеет вид

$$v(x) = ((x+\alpha)^2 + \beta)(x+\gamma), \qquad w(x) = v(x) + x,$$
$$t_1(x) = (v(x)+\delta_1)(w(x)+\epsilon_1), \qquad t_2(x) = (v(x)+\delta_2)(w(x)+\epsilon_2),$$
$$u(x) = (t_1(x)+\zeta)(t_2(x) - t_1(x) + \eta) + \kappa.$$

Минимальное число *действительных* сложений, необходимое, когда достигнуто минимальное число действительных умножений, для $n \geq 9$ остается неизвестным.

41. $a(c+d) - (a+b)d + i(a(c+d) + (b-a)c)$. [Остерегайтесь численной неустойчивости. Три умножения необходимы, поскольку в частном случае (71) с $p(u) = u^2 + 1$ умножение комплексное. Без ограничения на сложение существуют другие возможности. Например, в 1963 году Питером Унгером (Peter Ungar) предложена симметричная формула $ac - bd + i((a+b)(c+d) - ac - bd)$. Равенство 4.3.3–(2) похоже на это с 2^n в роли i. См. также работы I. Munro, *STOC* **3** (1971), 40–44; S. Winograd, *Linear Algebra and its Applications* **4** (1971), 381–388.]

Наоборот, если $a^2 + b^2 = 1$ и $t = (1-a)/b = b/(1+a)$, то для вычисления произведения $(a+bi)(c+di) = u + iv$ Оскар Банеман (Oscar Buneman) предложил алгоритм "$w = c - td$, $v = d + bw$, $u = w - tv$" [*J. Comp. Phys.* **12** (1973), 127–128]. Этим методом, если $a = \cos\theta$ и $b = \sin\theta$, получаем $t = \tan(\theta/2)$.

Гельмут Альт (Helmut Alt) и Ян Ван Ливен (Jan van Leeuwen) [*Computing* **27** (1981), 205–215] показали, что необходимо четыре действительных умножения или деления для вычисления $1/(a+bi)$ и достаточно четырех операций для вычисления

$$\frac{a}{b+ci} = \frac{a}{b+c(c/b)} - i\frac{(c/b)a}{b+c(c/b)}.$$

Шесть операций умножения-деления и три сложения-вычитания необходимо и достаточно для вычисления $(a+bi)/(c+di)$ [T. Lickteig, *SICOMP* **16** (1987), 278–311].

Несмотря на эти нижние грани следует помнить, что нет необходимости выполнять комплексную арифметику в терминах действительной арифметики. Например, при использовании быстрого преобразования Фурье время, необходимое для умножения двух n-значных комплексных чисел, асимптотически равно приблизительно только удвоенному времени умножения двух n-значных действительных чисел.

42. (a) Пусть $\pi_1, \ldots, \pi_m$ — это λ_i, соответствующие умножениям в цепочке, тогда $\pi_i = P_{2i-1} \times P_{2i}$ и $u(x) = P_{2m+1}$, где каждое P_j имеет вид $\beta_j + \beta_{j0}x + \beta_{j1}\pi_1 + \cdots + \beta_{jr(j)}\pi_{r(j)}$, где $r(j) \leq \lceil j/2 \rceil - 1$, и каждое β_j и β_{jk} — полином от α_i с целыми коэффициентами. Можно постоянно модифицировать цепочку таким образом (см. упр. 30), чтобы $\beta_j = 0$ и $\beta_{jr(j)} = 1$ для $1 \leq j \leq 2m$. Более того, можно предположить, что $\beta_{30} = 0$. Множество результатов сейчас имеет максимум $m + 1 + \sum_{j=1}^{2m}(\lceil j/2 \rceil - 1) = m^2 + 1$ степеней свободы.

(b) Любая такая цепочка полиномов с максимум m умножениями может походить на одну из рассмотренных в п. (a) форм, но полагаем, что $r(j) = \lceil j/2 \rceil - 1$ для $1 \leq j \leq 2m+1$, и не предполагаем, что $\beta_{30} = 0$ или $\beta_{jr(j)} = 1$ для $j \geq 3$. Эта простая каноническая форма включает $m^2 + 2m$ параметров. Так, как α_j пробегают все целые значения, и так, как мы проходим все цепочки, так и β_k пробегает максимально 2^{m^2+2m} множеств значений по модулю 2. Следовательно, и множество результатов такое же. Для того чтобы получить все 2^n полиномов степени n с 0–1 коэффициентами, должно выполняться $m^2 + 2m \geq n$.

(c) Присвоим $m \leftarrow \lfloor \sqrt{n} \rfloor$ и вычислим $x^2, x^3, \ldots, x^m$. Пусть $u(x) = u_{m+1}(x)x^{(m+1)m} + \cdots + u_1(x)x^m + u_0(x)$, где каждое $u_j(x)$ — полином степени $\leq m$ с целыми коэффициентами (значит, его можно вычислить, не увеличивая число умножений). Вычислим $u(x)$ по правилу (2), как полином от x^m, с известными коэффициентами. (Используемое число сложений приближенно равно сумме абсолютных значений коэффициентов. Таким образом, данный

алгоритм эффективен на 0–1 полиномах. Петерсон (Paterson) и Стокмейер (Stockmeyer) предложили также другой алгоритм, который использует около $\sqrt{2n}$ умножений.)

См. *SICOMP* **2** (1973), 60–66, а также J. E. Savage, *SICOMP* **3** (1974), 150–158; J. Ganz, *SICOMP* **24** (1995), 473–483. Аналогичные результаты относительно сложений приводятся в работах Borodin and Cook, *SICOMP* **5** (1976), 146–157; Rivest and Van de Wiele, *Inf. Proc. Letters* **8** (1979), 178–180.

43. Если $a_i = a_j + a_k$ — шаг в некоторой оптимальной цепочке сложений для $n+1$, вычислим $x^i = x^j x^k$ и $p_i = p_k x^j + p_j$, где $p_i = x^{i-1} + \cdots + x + 1$. Опустим окончательное вычисление x^{n+1}. Одно умножение экономится, как только $a_k = 1$, в частности когда $i = 1$ (см. упр. 4.6.3–31 с $\epsilon = \frac{1}{2}$).

44. Пусть $l = \lfloor \lg n \rfloor$, и предположим, что x, x^2, x^4, ..., x^{2^l} предварительно вычислены. Если $u(x)$ — нормированный полином степени $n = 2m+1$, то его можно записать как $u(x) = (x^{m+1} + \alpha)v(x) + w(x)$, где $v(x)$ и $w(x)$ — нормированные многочлены степени m. Здесь приведен метод для $n = 2^{l+1} - 1 \geq 3$, который требует дополнительно $2^l - 1$ операций умножения и $2^{l+1} + 2^{l-1} - 2$ операций сложения. Если $n = 2^l$, то можно применить правило Горнера для уменьшения n на 1. И если $m = 2^l < n < 2^{l+1} - 1$, можно записать $u(x) = x^m v(x) + w(x)$, где v и w — нормированные полиномы степени $n-m$ и m соответственно. Индукцией по l можно показать, что потребуется максимум $\frac{1}{2}n + l - 1$ умножений и $\frac{5}{4}n$ сложений после предварительных вычислений. [См. S. Winograd, *IBM Tech. Disclosure Bull.* **13** (1970), 1133–1135.]

Замечание. Можно также вычислить $u(x)$, выполнив $\frac{1}{2}n + O(\sqrt{n})$ операций умножения и $n + O(\sqrt{n})$ операций сложения, по таким же обоснованным правилам, если необходимо минимизировать число умножений и сложений. Характерный для данного класса полином

$$p_{jkm}(x) = \Big(\Big(\ldots(((x^m + \alpha_0)(x^{j+1} + \beta_1) + \alpha_1)(x^{j+2} + \beta_2) + \alpha_2)\cdots\Big)(x^k + \beta_{k-j}) + \alpha_{k-j}\Big)(x^j + \beta_0)$$

"охватывает" коэффициенты при степенях $\{j, j+k, j+k+(k-1), \ldots, j+k+(k-1)+\cdots+(j+1), m'-k, m'-k+1, \ldots, m'-j\}$, где

$$m' = m + j + (j+1) + \cdots + k = m + \binom{k+1}{2} - \binom{j}{2}.$$

Сложив такие полиномы $p_{1km_1}(x)$, $p_{2km_2}(x)$, ..., $p_{kkm_k}(x)$ для $m_j = \binom{j+1}{2} + \binom{k-j+2}{2}$, получим произвольный нормированный полином степени $k^2 + k + 1$. [В статье Rabin and Winograd, *Comm. on Pure and Applied Math.* **25** (1972), 433–458, §2, доказано, что конструкция с $\frac{1}{2}n + O(\log n)$ умножениями и $\leq (1+\epsilon)n$ сложениями возможна для всех $\epsilon > 0$, если n достаточно большое.]

45. Достаточно показать, что ранг (T_{ijk}) не *больше*, чем ранг (t_{ijk}), так как можно получить (t_{ijk}) из (T_{ijk}), преобразовав его таким же способом с F^{-1}, G^{-1}, H^{-1}. Если $t_{ijk} = \sum_{l=1}^{r} a_{il} b_{jl} c_{kl}$, то немедленно следует, что

$$T_{ijk} = \sum_{1 \leq l \leq r} \Big(\sum_{i'=1}^{m} F_{ii'} a_{i'l}\Big)\Big(\sum_{j'=1}^{n} G_{jj'} b_{j'l}\Big)\Big(\sum_{k'=1}^{s} H_{kk'} c_{k'l}\Big).$$

[Г. Ф. де Гроот (H. F. de Groote) доказал, что все нормальные схемы, которые дают произведение матриц размера 2×2 в результате семи умножений в цепочке, эквивалентны в том смысле, что их можно получить одну из другой, выполнив умножение на невырожденную матрицу, как в данном упражнении. В этом смысле алгоритм Штрассена (Strassen) единственный. См. *Theor. Comp. Sci.* **7** (1978), 127–148.]

46. Согласно упр. 45 можно добавить любое кратное строке, столбцу или матрице к другой строке, столбцу или матрице без изменения ранга; также можно умножить строку, столбец или матрицу на не равную нулю константу или транспонировать тензор. Всегда можно найти последовательность операций, которая приводит данный тензор размера $2 \times 2 \times 2$ к одной из таких форм: $\binom{0\,0}{0\,0}\binom{0\,0}{0\,0}$, $\binom{1\,0}{0\,0}\binom{0\,0}{0\,0}$, $\binom{1\,0}{0\,1}\binom{0\,0}{0\,0}$, $\binom{1\,0}{0\,0}\binom{0\,0}{0\,1}$, $\binom{1\,0}{0\,1}\binom{0\,1}{q\,r}$. По теореме W последний тензор имеет ранг 3 или 2 в зависимости от того, сколько неприводимых множителей имеет полином $u^2 - ru - q$ (один или два) в интересующем нас поле (см. (74)).

47. Тензор размера $m \times n \times s$ имеет mns степеней свободы. Согласно упр. 28 все тензоры размера $m \times n \times s$ можно выразить в терминах $(m + n + s)r$ элементов реализации (A, B, C), кроме случаев, когда $(m + n + s)r \geq mns$. С другой стороны, предположим, что $m \geq n \geq s$. Максимальный ранг матрицы размера $m \times n$ равен n. Таким образом, можно получить любой тензор за ns умножений в цепочке, получая отдельно каждую матрицу. [В упр. 46 показано, что такая нижняя грань для максимального ранга тензора не является наилучшей возможной гранью и верхней гранью. Томас Д. Хоуэл (Thomas D. Howell, Ph. D. thesis, Cornell Univ., 1976) показал, что существуют тензоры, имеющие ранг $\geq \lceil mns/(m + n + s - 2) \rceil$ над комплексными числами.]

48. Если (A, B, C) и (A', B', C') — реализации (t_{ijk}) и (t'_{ijk}) длиной r и r' соответственно, то $A'' = A \oplus A'$, $B'' = B \oplus B'$, $C'' = C \oplus C'$ и $A''' = A \otimes A'$, $B''' = B \otimes B'$, $C''' = C \otimes C'$ — реализации (t''_{ijk}) и (t'''_{ijk}) длиной $r + r'$ и $r \cdot r'$ соответственно.

Замечание. Многие, естественно, предполагают, что $\operatorname{rank}((t_{ijk}) \oplus (t'_{ijk})) = \operatorname{rank}(t_{ijk}) + \operatorname{rank}(t'_{ijk})$, однако в упр. 60, (b) и 65 показано, что это выглядит менее правдоподобным, чем кажется.

49. Согласно лемме T $\operatorname{rank}(t_{ijk}) \geq \operatorname{rank}(t_{i(jk)})$. Обратно, если M — матрица ранга r, то ее можно преобразовать с помощью операций со строками и столбцами, найдя такие невырожденные матрицы F и G, что матрица FMG будет содержать все 0, за исключением r диагональных элементов, равных 1 (см. алгоритм 4.6.2N). Следовательно, ранг тензора $FMG \leq r$ и он такой же, как ранг тензора M согласно упр. 45.

50. Пусть $i = \langle i', i'' \rangle$, где $1 \leq i' \leq m$ и $1 \leq i'' \leq n$, тогда $t_{\langle i', i'' \rangle jk} = \delta_{i''j}\delta_{i'k}$. Очевидно, что $\operatorname{rank}(t_{i(jk)}) = mn$, так как $(t_{i(jk)})$ — матрица перестановок. По лемме L $\operatorname{rank}(t_{ijk}) \geq mn$. Обратно, так как (t_{ijk}) имеет всего mn ненулевых элементов, то, очевидно, ее ранг $\leq mn$. (Существует, следовательно, ненормальная схема, требующая менее mn явных умножений. Подобной анормальной схемы не существует [*Comm. Pure and Appl. Math.* **3** (1970), 165–179]. Но может быть достигнута некоторая экономия, если такая же матрица используется с $s > 1$ различными вектор-столбцами, поскольку это эквивалент умножения матриц размера $(n \times s)$ и $(m \times n)$).

51. (a) $s_1 = y_0 + y_1$, $s_2 = y_0 - y_1$; $m_1 = \frac{1}{2}(x_0 + x_1)s_1$, $m_2 = \frac{1}{2}(x_0 - x_1)s_2$; $w_0 = m_1 + m_2$, $w_1 = m_1 - m_2$. (b) Здесь несколько промежуточных шагов, использующих методику из раздела: $((x_0 - x_2) + (x_1 - x_2)u)((y_0 - y_2) + (y_1 - y_2)u) \bmod (u^2 + u + 1) = ((x_0 - x_2)(y_0 - y_2) - (x_1 - x_2)(y_1 - y_2)) + ((x_0 - x_2)(y_0 - y_2) - (x_1 - x_0)(y_1 - y_0))u$. Первая реализация —

$$\begin{pmatrix} 1 & 1 & \bar{1} & 0 \\ 1 & 0 & 1 & 1 \\ 1 & \bar{1} & 0 & \bar{1} \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & \bar{1} & 0 \\ 1 & 0 & 1 & 1 \\ 1 & \bar{1} & 0 & \bar{1} \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & 1 & \bar{2} \\ 1 & 1 & \bar{2} & 1 \\ 1 & \bar{2} & 1 & 1 \end{pmatrix} \times \frac{1}{3}.$$

Вторая реализация —

$$\begin{pmatrix} 1 & 1 & 1 & \bar{2} \\ 1 & 1 & \bar{2} & 1 \\ 1 & \bar{2} & 1 & 1 \end{pmatrix} \times \frac{1}{3}, \quad \begin{pmatrix} 1 & 1 & \bar{1} & 0 \\ 1 & \bar{1} & 0 & \bar{1} \\ 1 & 0 & 1 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 1 & \bar{1} & 0 \\ 1 & 0 & 1 & 1 \\ 1 & \bar{1} & 0 & \bar{1} \end{pmatrix}.$$

Окончательно алгоритм вычисляет $s_1 = y_0 + y_1$, $s_2 = y_0 - y_1$, $s_3 = y_2 - y_0$, $s_4 = y_2 - y_1$, $s_5 = s_1 + y_2$; $m_1 = \frac{1}{3}(x_0 + x_1 + x_2)s_5$, $m_2 = \frac{1}{3}(x_0 + x_1 - 2x_2)s_2$, $m_3 = \frac{1}{3}(x_0 - 2x_1 + x_2)s_3$,

$m_4 = \frac{1}{3}(-2x_0 + x_1 + x_2)s_4$; $t_1 = m_1 + m_2$, $t_2 = m_1 - m_2$, $t_3 = m_1 + m_3$, $w_0 = t_1 - m_3$, $w_1 = t_3 + m_4$, $w_2 = t_2 - m_4$.

52. Пусть $k = \langle k', k'' \rangle$, когда $k \bmod n' = k'$ и $k \bmod n'' = k''$. Нужно вычислить $w_{\langle k',k'' \rangle} = \sum x_{\langle i',i'' \rangle} y_{\langle j',j'' \rangle}$, где суммирование производится так, что $i' + j' \equiv k'$ (по модулю n') и $i'' + j'' \equiv k''$ (по модулю n''). Это можно сделать, применив алгоритм n' к $2n'$ векторам $X_{i'}$ и $Y_{j'}$ длиной n'' и получив n' векторов $W_{k'}$. Каждое векторное сложение становится n'' сложениями, каждое умножение параметров становится n'' умножением параметров и каждое умножение в цепочке векторов заменяется циклической сверткой степени n''. [Если в подалгоритмах используется минимальное число умножений в цепочке над рациональными числами, то в этом алгоритме используется на $2(n' - d(n'))(n'' - d(n''))$ операций больше минимального числа, $d(n)$ — число делителей n, что следует из упр. 4.6.2–32 и теоремы W.]

53. (a) Пусть $n(k) = (p-1)p^{e-k-1} = \varphi(p^{e-k})$ для $0 \le k < e$ и $n(k) = 1$ для $k \ge e$. Представьте числа $\{1, \ldots, m\}$ в виде $a^i p^k$ (по модулю m), где $0 \le k \le e$ и $0 \le i < n(k)$, и a — фиксированный первообразный элемент по модулю p^e. Например, когда $m = 9$, можно положить $a = 2$; значениями являются $\{2^0 3^0, 2^1 3^0, 2^0 3^1, 2^2 3^0, 2^5 3^0, 2^1 3^1, 2^4 3^0, 2^3 3^0, 2^0 3^2\}$. Тогда $f(a^i p^k) = \sum_{0 \le l \le e} \sum_{0 \le j < n(l)} \omega^{g(i,j,k,l)} F(a^j p^l)$, где $g(i,j,k,l) = a^{i+j} p^{k+l}$.

Вычислим $f_{ikl} = \sum_{0 \le j < n(l)} \omega^{g(i,j,k,l)} F(a^j p^l)$ для $0 \le i < n(k)$ и для каждых k и l. Это циклическая свертка степени $n(k+l)$ значений

$$x_i = \omega^{a^i p^{k+l}} \quad \text{и} \quad y_s = \sum_{0 \le j < n(l)} [s + j \equiv 0 \ (\text{по модулю } n(k+l))]\, F(a^j p^l),$$

так как f_{ikl} равно $\sum x_r y_s$ (суммирование по $r + s \equiv i (\text{по модулю } n(k+l))$). Преобразование Фурье получается посредством суммирования соответствующих f_{ikl}. [*Замечание.* Когда образованы линейные комбинации x_i, как, например, в (69), результат будет чисто действительным или чисто мнимым, если циклическая свертка сконструирована по правилу (59) с $u^{n(k)} - 1 = (u^{n(k)/2} - 1)(u^{n(k)/2} + 1)$. Это связано с тем, что операция по модулю $(u^{n(k)/2} - 1)$ дает полином с действительными коэффициентами $\omega^j + \omega^{-j}$, тогда как операция по модулю $(u^{n(k)/2} + 1)$ дает полином с мнимыми коэффициентами $\omega^j - \omega^{-j}$.]

Когда $p = 2$, применяют подобную конструкцию, используя представление $(-1)^i a^j 2^k$ (по модулю m), где $0 \le k \le e$, $0 \le i \le \min(e-k, 1)$ и $0 \le j < 2^{e-k-2}$. В таком случае используется конструкция упр. 52 с $n' = 2$ и $n'' = 2^{e-k-2}$; среди этих чисел не существует взаимно простых чисел; конструкция дает требуемое прямое произведение циклической свертки.

(b) Пусть $a'm' + a''m'' = 1$ и пусть $\omega' = \omega^{a''m''}$, $\omega'' = \omega^{a'm'}$. Определим $s' = s \bmod m'$, $s'' = s \bmod m''$, $t' = t \bmod m'$, $t'' = t \bmod m''$ таким образом, что $\omega^{st} = (\omega')^{s't'} (\omega'')^{s''t''}$. Отсюда следует, что $f(s', s'') = \sum_{t'=0}^{m'-1} \sum_{t''=0}^{m''-1} (\omega')^{s't'} (\omega'')^{s''t''} F(t', t'')$. Другими словами, одномерное преобразование Фурье m элементов — это фактически несколько измененное двумерное преобразование Фурье $m' \times m''$ элементов.

Мы будем рассматривать "нормальные" алгоритмы, состоящие из: (i) сумм s_i от $F(k)$ и $s(j)$, (ii) произведений m_j, каждое из которых получается путем умножения одного из $F(k)$ или $S(j)$ на действительное или мнимое число α_j, (iii) дополнительных сумм t_k, каждая из которых образуется из m_k или t_j (а не $F(k)$ или $s(j)$). Окончательными значениями должны быть m_k или t_j. Например, "нормальная" схема преобразования Фурье для $m = 5$, построенная по (69) и по методу из п. (a), имеет следующий вид: $s_1 = F(1) + F(4)$, $s_2 = F(3) + F(2)$, $s_3 = s_1 + s_2$, $s_4 = s_1 - s_2$, $s_5 = F(1) - F(4)$, $s_6 = F(2) - F(3)$, $s_7 = s_5 - s_6$; $m_1 = \frac{1}{4}(\omega + \omega^2 + \omega^4 + \omega^3)s_3$, $m_2 = \frac{1}{4}(\omega - \omega^2 + \omega^4 - \omega^3)s_4$, $m_3 = \frac{1}{2}(\omega + \omega^2 - \omega^4 - \omega^3)s_5$, $m_4 = \frac{1}{2}(-\omega + \omega^2 + \omega^4 - \omega^3)s_6$, $m_5 = \frac{1}{2}(\omega^3 - \omega^2)s_7$, $m_6 = 1 \cdot F(5)$, $m_7 = 1 \cdot s_3$; $t_0 = m_1 + m_6$, $t_1 = t_0 + m_2$, $t_2 = m_3 + m_5$, $t_3 = t_0 - m_2$, $t_4 = m_4 - m_5$, $t_5 = t_1 + t_2$, $t_6 = t_3 + t_4$, $t_7 = t_1 - t_2$, $t_8 = t_3 - t_4$, $t_9 = m_6 + m_7$. Отметим умножения на единицу в m_6 и m_7. Ясно, что реально

выполнять умножение здесь не нужно. Однако в силу принятых обозначений этот случай отмечается. Кроме того, он широко используется в рекурсивных конструкциях. Здесь $m_6 = f_{001}$, $m_7 = f_{010}$, $t_5 = f_{000} + f_{001} = f(2^0)$, $t_6 = f_{100} + f_{101} = f(2^1)$ и т. д. Схему можно улучшить, включив $s_8 = s_3 + F(5)$, заменив m_1 выражением $(\frac{1}{4}(\omega + \omega^2 + \omega^4 + \omega^3) - 1)s_3$ [что равно $-\frac{5}{4}s_3$], а m_6 — выражением $1 \cdot s_8$ и удалив m_7 и t_9. Это сокращает одно из тривиальных умножений на 1 и позволяет использовать данную схему для построения большей схемы. В улучшенной схеме $f(5) = m_6$, $f(1) = t_5$, $f(2) = t_6$, $f(3) = t_8$, $f(4) = t_7$.

Предположим, что есть нормальные одномерные схемы для m' и m'', использующие соответственно (a', a'') комплексных сложений, (t', t'') тривиальных умножений на ± 1 или $\pm i$ и вообще (c', c'') комплексных умножений, включая тривиальные умножения. (Все нетривиальные комплексные умножения "простые", так как они включают только два действительных умножения и недействительных сложения.) Можно построить нормальную схему для двумерного $(m' \times m'')$-случая, применив схему m' к векторам $F(t', *)$ длиной m''. Каждый шаг s_i превращается в m'' сложений; каждое m_j становится преобразованием Фурье m'' элементов, но в этом алгоритме со всеми α_k, умноженными на α_j; каждое t_k превращается в m'' сложений. Таким образом, новый алгоритм имеет $(a'm'' + c'a'')$ комплексных сложений, $t't''$ обычных умножений и $c'c''$ комплексных умножений.

Используя эту технику, Виноград нашел нормальные одномерные схемы для малых значений m со следующими затратами (a, t, c).

$m = 2$	$(2, 2, 2)$	$m = 7$	$(36, 1, 9)$
$m = 3$	$(6, 1, 3)$	$m = 8$	$(26, 6, 8)$
$m = 4$	$(8, 4, 4)$	$m = 9$	$(46, 1, 12)$
$m = 5$	$(17, 1, 6)$	$m = 16$	$(74, 8, 18)$

Комбинируя эти схемы, как описано выше, получим методы, в которых используется меньше арифметических операций, чем в быстром преобразовании Фурье (БПФ), рассматриваемом в упр. 14. Например, когда $m = 1008 = 7 \cdot 9 \cdot 16$, затраты равны $(17946, 8, 1944)$. Таким образом можно осуществить преобразование Фурье 1 008 комплексных чисел с 3 872 действительными умножениями и 35 892 действительными сложениями. Улучшить метод Винограда можно, комбинируя взаимно простые модули и используя многомерные свертки, как показано в работе Nussbaumer and Quandalle, *IBM J. Res. and Devel.* **22** (1978), 134–144. Подход авторов позволяет сократить количество необходимых вычислений для комплексного преобразования Фурье 1 008 чисел до 3 084 действительных умножений и 34 668 действительных сложений. Для сравнения БПФ 1 024 комплексных чисел включает 14 344 действительных умножения и 27 652 действительных сложения. Однако, если используется улучшение подхода из ответа к упр. 14, для БПФ 1 024 комплексных чисел необходимо только 10 936 действительных умножений и 25 948 сложений, что совсем несложно выполнить. Поэтому тонкие методы быстрее работают только на машинах, которые значительно дольше умножают, чем складывают.

[См. *Proc. Nat. Acad. Sci. USA* **73** (1976), 1005–1006; *Math. Comp.* **32** (1978), 175–199; *Advances in Math.* **32** (1979), 83–117; *IEEE Trans.* **ASSP-27** (1979), 169–181.]

54. $\max(2e_1\deg(p_1) - 1, \ldots, 2e_q\deg(p_q) - 1, q + 1)$.

55. $2n' - q'$, где n' — степень минимального полинома P (нормированного полинома μ по крайней мере такой степени, что $\mu(P)$ — нулевая матрица) и q' — число различных неразложимых множителей этого полинома. (Привести P, выполнив подобные преобразования.)

56. Пусть $t_{ijk} + t_{jik} = \tau_{ijk} + \tau_{jik}$ для всех i, j, k. Если (A, B, C) — реализация (t_{ijk}) ранга r, то $\sum_{l=1}^{r} c_{kl}(\sum_i a_{il}x_i)(\sum_j b_{jl}x_j) = \sum_{i,j} t_{ijk}x_ix_j = \sum_{i,j} \tau_{ijk}x_ix_j$ для всех k. Обратно, пусть l-е умножение в цепочке полиномов для $1 \le l \le r$ является произведением $(\alpha_l + \sum_i \alpha_{il}x_i)(\beta_l + \sum_j \beta_{jl}x_j)$, где α_l и β_l определяют возможные постоянные члены и/или нелинейные члены. Все члены степени 2, появляющиеся на любом шаге цепочки, могут

быть выражены в виде линейной комбинации $\sum_{l=1}^{r} c_l (\sum_i a_{il} x_i)(\sum_j b_{jl} x_j)$, поэтому цепочка определяет тензор (t_{ijk}) ранга $\le r$, такой, что $t_{ijk} + t_{jik} = \tau_{ijk} + \tau_{jik}$. Это доказывает утверждение указания. Тогда $\operatorname{rank}(\tau_{ijk} + \tau_{jik}) = \operatorname{rank}(t_{ijk} + t_{jik}) \le \operatorname{rank}(t_{ijk}) + \operatorname{rank}(t_{jik}) = 2\operatorname{rank}(t_{ijk})$.

Билинейная форма от $x_1, \ldots, x_m, y_1, \ldots, y_n$ является квадратичной формой от $m+n$ переменных, где $\tau_{ijk} = t_{i,j-m,k}$ для $i \le m$ и $j > m$; в остальных случаях $\tau_{ijk} = 0$. Тогда $\operatorname{rank}(\tau_{ijk}) + \operatorname{rank}(\tau_{jik}) \ge \operatorname{rank}(t_{ijk})$, поскольку мы получим реализацию (t_{ijk}), удалив последние n строк матрицы A и первые m строк матрицы B в реализации (A, B, C) $(\tau_{ijk} + \tau_{jik})$.

57. Пусть N — наименьшая степень 2, превосходящая $2n$, и пусть $u_{n+1} = \cdots = u_{N-1} = v_{n+1} = \cdots = v_{N-1} = 0$. Если $U_s = \sum_{t=0}^{N-1} \omega^{st} u_t$ и $V_s = \sum_{t=0}^{N-1} \omega^{st} v_t$ для $0 \le s < N$, где $\omega = e^{2\pi i/N}$, то $\sum_{s=0}^{N-1} \omega^{-st} U_s V_s = N \sum u_{t_1} v_{t_2}$, где последняя сумма берется по всем t_1 и t_2 с $0 \le t_1, t_2 < N$, $t_1 + t_2 \equiv t$ по модулю N. Если неравенства $t_1 \le n$ и $t_2 \le n$, не выполняются, то члены сумм становятся нулями. Следовательно, $t_1 + t_2 < N$ и сумма равна коэффициенту при z^t в произведении $u(z)v(z)$. Если воспользоваться методом вычисления прямого и обратного преобразований Фурье из упр. 14, то количество комплексных операций будет равно $O(N \log N) + O(N \log N) + O(N) + O(N \log N)$ и $N \le 4n$. [См. раздел 4.3.3C и статью Дж. М. Полларда (J. M. Pollard, *Math. Comp.* **25** (1971), 365–374).]

При умножении полиномов с целыми коэффициентами можно использовать *целое* число ω, которое является порядком 2^t по модулю простого числа p, и определять результаты по модулю достаточного количества простых чисел. Подходящие в этом отношении простые числа вместе с их наименьшими первообразными корнями r (для которых выберем $\omega = r^{(p-1)/2^t} \bmod p$, где $p \bmod 2^t = 1$) можно найти, как описано в разделе 4.5.4. Для $t = 9$ десятью наибольшими числами $< 2^{35}$ являются $p = 2^{35} - 512a + 1$, где $(a, r) = (28, 7)$, $(31, 10)$, $(34, 13)$, $(56, 3)$, $(58, 10)$, $(76, 5)$, $(80, 3)$, $(85, 11)$, $(91, 5)$, $(101, 3)$; десятью наибольшими числами $< 2^{31}$ являются $p = 2^{31} - 512a + 1$, где $(a, r) = (1, 10)$, $(11, 3)$, $(19, 11)$, $(20, 3)$, $(29, 3)$, $(35, 3)$, $(55, 19)$, $(65, 6)$, $(95, 3)$, $(121, 10)$. Для больших t все простые числа p вида $2^t q + 1$, где $q < 32$, являются нечетными, и $2^{24} < p < 2^{36}$ заданы следующим образом: $(p-1, r) = (11 \cdot 2^{21}, 3)$, $(25 \cdot 2^{20}, 3)$, $(27 \cdot 2^{20}, 5)$, $(25 \cdot 2^{22}, 3)$, $(27 \cdot 2^{22}, 7)$, $(5 \cdot 2^{25}, 3)$, $(7 \cdot 2^{26}, 3)$, $(27 \cdot 2^{26}, 13)$, $(15 \cdot 2^{27}, 31)$, $(17 \cdot 2^{27}, 3)$, $(3 \cdot 2^{30}, 5)$, $(13 \cdot 2^{28}, 3)$, $(29 \cdot 2^{27}, 3)$, $(23 \cdot 2^{29}, 5)$. Несколько последних простых чисел можно использовать с $\omega = 2^e$ для достаточно малых e. Такие простые числа обсуждаются в работах R. M. Robinson, *Proc. Amer. Math. Soc.* **9** (1958), 673-681, и S. W. Golomb, *Math. Comp.* **30** (1976), 657–663. Дополнительные ссылки на методы для всех целых чисел можно найти в ответе к упр. 4.6–5.

Тем не менее метод из упр. 59 почти всегда предпочтительнее на практике.

58. (a) Вообще, если (A, B, C) реализует (t_{ijk}), то $((x_1, \ldots, x_m)A, B, C)$ является реализацией матрицы размера $1 \times n \times s$, на пересечении строки j и столбца k которой находится $\sum x_i t_{ijk}$. В таком случае должно быть по крайней мере такое же множество ненулевых элементов в $(x_1, \ldots, x_m)A$, каков ранг этой матрицы. Случаю, когда тензор имеет размер $m \times n \times (m+n-1)$, соответствует умножение полинома степени $m-1$ на полином степени $n-1$, соответствующая матрица имеет ранг n всякий раз, когда $(x_1, \ldots, x_m) \ne (0, \ldots, 0)$. Подобное утверждение имеет место при $A \leftrightarrow B$ и $m \leftrightarrow n$.

Замечание. В частности, работая в поле из 2 элементов, говорят, что строки A по модулю 2 образуют "линейный код" m векторов, находящихся на расстоянии по крайней мере n, всякий раз, когда (A, B, C) — реализация, полностью состоящая из целых чисел. Эти наблюдения, приведенные в работах R. W. Brockett and D. Dobkin, *Linear Algebra and its Applications* **19** (1978), 207–235, теорема 14; Lempel and Winograd, *IEEE Trans.* **IT-23** (1977), 503–508; Lempel, Seroussi, and Winograd, *Theoretical Comp. Sci.* **22** (1983), 285–296, можно использовать для получения нетривиальных нижних граней рангов целочисленных матриц. Например, М. Р. Браун (M. R. Brown) и Д. Добкин (D. Dobkin) [*IEEE Trans.*

C-29 (1980), 337–340] использовали это, чтобы показать, что реализации $n \times n$ умножений полиномов в целых числах должны иметь ранг $\geq \alpha n$ для всех достаточно больших n, когда α — *любое действительное число, меньшее, чем число*

$$\alpha_{\min} = 3.52762\,68026\,32407\,48061\,54754\,08128\,07512\,70182+;$$

здесь $\alpha_{\min} = 1/H(\sin^2\theta, \cos^2\theta)$, где $H(p,q) = p\lg(1/p) + q\lg(1/q)$ — бинарная функция энтропии и $\theta \approx 1.34686$ — корень уравнения $\sin^2(\theta - \pi/4) = H(\sin^2\theta, \cos^2\theta)$. Реализация ранга $O(n \log n)$ для всех целых чисел, основанная на круговых полиномах, построена М. Камински (M. Kaminski) [*J. Algorithms* **9** (1988), 137–147].

(b) $\begin{pmatrix} 1&0&0&0&0&1&1&1\\ 0&1&0&0&1&1&0&1\\ 0&0&1&1&0&0&1&1 \end{pmatrix}$, $\begin{pmatrix} 1&0&0&0&0&1&1&1\\ 0&1&0&0&0&1&0&1\\ 0&0&1&0&0&0&1&1\\ 0&0&0&1&1&0&0&1 \end{pmatrix}$, $\begin{pmatrix} 1&0&0&0&0&0&0&0\\ \bar1&\bar1&0&0&\bar0&1&0&0\\ \bar1&1&\bar1&0&0&0&1&0\\ 1&0&0&\bar1&\bar1&\bar1&\bar1&1\\ 0&0&1&0&1&0&0&0\\ 0&0&0&1&0&0&0&0 \end{pmatrix}$.

Следующий экономный способ реализации умножения обычных полиномов степеней 2, 3 и 4 предложен Г. Кохеном и А. К. Ленстра (H. Cohen and A. K. Lenstra, *Math. Comp.* **48** (1987), S1–S2):

$$\begin{pmatrix} 1&0&0&1&1&0\\ 0&1&0&1&0&1\\ 0&0&1&0&1&1 \end{pmatrix} \quad \text{и} \quad \begin{pmatrix} 1&0&0&0&0&0\\ \bar1&\bar1&0&1&0&0\\ \bar1&1&\bar1&0&1&0\\ 0&\bar1&\bar1&0&0&1\\ 0&0&1&0&0&0 \end{pmatrix};$$

$$\begin{pmatrix} 1&0&0&0&1&1&0&0&1\\ 0&1&0&0&1&0&0&1&1\\ 0&0&1&0&0&1&1&0&1\\ 0&0&0&1&0&0&1&1&1 \end{pmatrix} \quad \text{и} \quad \begin{pmatrix} 1&0&0&0&0&0&0&0&0\\ \bar1&\bar1&0&0&1&0&0&0&0\\ \bar1&1&\bar1&0&0&1&0&0&0\\ 1&1&1&1&\bar1&\bar1&\bar1&\bar1&1\\ 0&\bar1&1&\bar1&0&0&0&1&0\\ 0&0&\bar1&\bar1&0&0&1&0&0\\ 0&0&0&1&0&0&0&0&0 \end{pmatrix};$$

$$\begin{pmatrix} 1&0&0&1&1&0&1&0&1&1&0&0&0&0\\ 0&1&0&1&0&1&0&1&1&0&1&0&0&0\\ 0&0&1&0&1&1&0&0&0&1&1&0&0&0\\ 0&0&0&0&0&0&1&0&1&1&0&1&0&1\\ 0&0&0&0&0&0&0&1&1&0&1&0&1&1 \end{pmatrix} \quad \text{и}$$

$$\begin{pmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ \bar1&\bar1&0&1&0&0&0&0&0&0&0&0&0&0&0\\ \bar1&1&\bar1&0&1&0&0&0&0&0&0&0&0&0&0\\ \bar1&\bar1&\bar1&0&0&1&1&0&0&0&0&\bar1&0&0\\ 1&1&1&\bar1&0&0&\bar1&\bar1&1&0&0&1&1&\bar1\\ 1&\bar1&0&0&\bar1&0&\bar1&1&0&1&0&0&\bar1&0\\ 0&1&0&0&0&\bar1&0&\bar1&0&0&1&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&\bar1&\bar1&1\\ 0&0&0&0&0&0&0&0&0&0&0&1&0&\bar0 \end{pmatrix}$$

В каждом случае матрицы A и B идентичны.

59. [*IEEE Trans.* **ASSP-28** (1980), 205–215.] Заметим, что циклическая свертка является умножением полинома по модулю $u^n - 1$, а отрицательная циклическая свертка — умножением полинома по модулю $u^n + 1$. Изменим запись, заменив n числом 2^n, и рассмотрим рекурсивные алгоритмы для циклической и отрицательной циклической сверток $(z_0, \ldots, z_{2^n-1})$ последовательности $(x_0, \ldots, x_{2^n-1})$ с $(y_0, \ldots, y_{2^n-1})$. Алгоритмы

представлены в неоптимальном виде для краткости и простоты изложения; те читатели, которые реализуют данные алгоритмы, заметят, что многое можно ускорить. Например, окончательное значение $Z_{2m-1}(w)$ на шаге N5 всегда равно нулю.

C1. [Критерий для простого случая.] Если $n = 1$, то присвоить

$$z_0 \leftarrow x_0y_0 + x_1y_1, \qquad z_1 \leftarrow (x_0 + x_1)(y_0 + y_1) - z_0$$

и завершить выполнение, иначе — присвоить $m \leftarrow 2^{n-1}$.

C2. [Переход к разностям.] Для $0 \le k < m$ присвоить $(x_k, x_{m+k}) \leftarrow (x_k + x_{m+k}, x_k - x_{m+k})$ и $(y_k, y_{m+k}) \leftarrow (y_k + y_{m+k}, y_k - y_{m+k})$. (Получим $x(u) \bmod (u^m - 1) = x_0 + \cdots + x_{m-1}u^{m-1}$ и $x(u) \bmod (u^m + 1) = x_m + \cdots + x_{2m-1}u^{m-1}$; вычислим $x(u)y(u) \bmod (u^m - 1)$ и $x(u)y(u) \bmod (u^m + 1)$, а затем сгруппируем результаты согласно (59).)

C3. [Рекурсия.] Присвоить $(z_0, \ldots, z_{m-1})$ циклической свертке $(x_0, \ldots, x_{m-1})$ с $(y_0, \ldots, y_{m-1})$. Присвоить также $(z_m, \ldots, z_{2m-1})$ отрицательной циклической свертке $(x_m, \ldots, x_{2m-1})$ с $(y_m, \ldots, y_{2m-1})$.

C4. [Переход от разностей.] Для $0 \le k < m$ присвоить $(z_k, z_{m+k}) \leftarrow \frac{1}{2}(z_k + z_{m+k}, z_k - z_{m+k})$, тогда требуемый ответ — $(z_0, \ldots, z_{2m-1})$. ▮

N1. [Критерий для простого случая.] Если $n = 1$, присвоить $t \leftarrow x_0(y_0 + y_1)$, $z_0 \leftarrow t - (x_0 + x_1)y_1$, $z_1 \leftarrow t + (x_1 - x_0)y_0$ и закончить выполнение, иначе — присвоить $m \leftarrow 2^{\lfloor n/2 \rfloor}$ и $r \leftarrow 2^{\lceil n/2 \rceil}$. (На следующих шагах используется 2^{n+1} вспомогательных переменных X_{ij} для $0 \le i < 2m$ и $0 \le j < r$, представляющих $2m$ полиномов $X_i(w) = X_{i0} + X_{i1}w + \cdots + X_{i(r-1)}w^{r-1}$. Аналогично используется 2^{n+1} вспомогательных переменных Y_{ij}.)

N2. [Инициализация вспомогательных полиномов.] Присвоить $X_{ij} \leftarrow X_{(i+m)j} \leftarrow x_{mj+i}$, $Y_{ij} \leftarrow Y_{(i+m)j} \leftarrow y_{mj+i}$ для $0 \le i < m$ и $0 \le j < r$. (Здесь получим $x(u) = X_0(u^m) + uX_1(u^m) + \cdots + u^{m-1}X_{m-1}(u^m)$; аналогичная формула имеет место для $y(u)$. Наша стратегия — умножить эти полиномы по модулю $(u^{mr}+1) = (u^{2^n} + 1)$, выполнив операции по модулю $(w^r + 1)$ с полиномами $X(w)$ и $Y(w)$ и найдя их циклическую свертку длиной $2m$, и тем самым получить с ее помощью $x(u)y(u) \equiv Z_0(u^m) + uZ_1(u^m) + \cdots + u^{2m-1}Z_{2m-1}(u^m)$.)

N3. [Преобразовать.] (Сейчас, по существу, выполняется быстрое преобразование Фурье полиномов $(X_0, \ldots, X_{m-1}, 0, \ldots, 0)$ и $(Y_0, \ldots, Y_{m-1}, 0, \ldots, 0)$ с помощью $w^{r/m}$ в качестве $(2m)$-го корня единицы. Это эффективно, поскольку умножение на степень w не является реальным умножением.) Для $j = \lfloor n/2 \rfloor - 1, \ldots, 1, 0$ (в таком порядке) вычислить для всех m двоичных чисел $s + t = (s_{\lfloor n/2 \rfloor} \ldots s_{j+1}0 \ldots 0)_2 + (0 \ldots 0t_{j-1} \ldots t_0)_2$. Заменить $(X_{s+t}(w), X_{s+t+2^j}(w))$ парой полиномов $(X_{s+t}(w) + w^{(r/m)s'}X_{s+t+2^j}(w), X_{s+t}(w) - w^{(r/m)s'}X_{s+t+2^j}(w))$, где $s' = 2^j(s_{j+1} \ldots s_{\lfloor n/2 \rfloor})_2$. (Мы вычисляем 4.3.3–(39) с $K = 2m$ и $\omega = w^{r/m}$; отметим замену двоичного разряда в s'.) Точнее говоря, операция $X_i(w) \leftarrow X_i(w) + w^kX_l(w)$ означает присвоение $X_{ij} \leftarrow X_{ij} + X_{l(j-k)}$ для $k \le j < r$ и $X_{ij} \leftarrow X_{ij} - X_{l(j-k+r)}$ для $0 \le j < k$. Копирование $X_l(w)$ можно выполнить без больших затрат памяти. Произвести такие же преобразования с Y_k.

N4. [Рекурсия.] Для $0 \le i < 2m$ присвоить $(Z_{i0}, \ldots, Z_{i(r-1)})$ отрицательную свертку $(X_{i0}, \ldots, X_{i(r-1)})$ и $(Y_{i0}, \ldots, Y_{i(r-1)})$.

N5. [Обратное преобразование.] Для $j=0, 1, \ldots, \lfloor n/2\rfloor$ (в таком порядке) присвоить

$$(Z_{s+t}(w), Z_{s+t+2^j}(w)) \leftarrow \frac{1}{2}(Z_{s+t}(w) + Z_{s+t+2^j}(w),\ w^{-(r/m)s'}(Z_{s+t}(w) - Z_{s+t+2^j}(w)))$$

всем m, выбирая s и t, как на шаге N3.

N6. [Переупорядочение.] (Сейчас наша цель — завершить то, что сформулировано в конце шага N2, так как легко видеть, что преобразование Z_k — это произведение преобразований X_k и Y_i.) Присвоить $z_i \leftarrow Z_{i0} - Z_{(m+i)(r-1)}$ и $z_{mj+i} \leftarrow Z_{ij} + Z_{(m+i)(j-1)}$ для $0 < j < r$ и $0 \le i < m$. ▮

Легко проверить, что в этих вычислениях для вспомогательных переменных необходимо максимум n дополнительных двоичных разрядов точности. Например, если $|x_i| \le M$ для $0 \le i < 2^n$ в начале алгоритма, то все переменные x и X всегда будут ограничены 2^nM. Все переменные z и Z будут ограничены числом $(2^nM)^2$, где на n больше двоичных разрядов, чем требуется для окончательной свертки.

Алгоритм N выполняет A_n сложений-вычитаний, D_n делений пополам и M_n умножений, где $A_1 = 5$, $D_1 = 0$, $M_1 = 3$; для $n > 1$ $A_n = \lfloor n/2\rfloor 2^{n+2} + 2^{\lfloor n/2\rfloor+1}A_{\lceil n/2\rceil} + (\lfloor n/2\rfloor+1)2^{n+1}+2^n$, $D_n = 2^{\lfloor n/2\rfloor+1}D_{\lceil n/2\rceil} + (\lfloor n/2\rfloor+1)2^{n+1}$ и $M_n = 2^{\lfloor n/2\rfloor+1}M_{\lceil n/2\rceil}$. Решение имеет вид $A_n = 11\cdot 2^{n-1+\lceil \lg n\rceil} - 3\cdot 2^n + 6\cdot 2^n S_n$, $D_n = 4\cdot 2^{n-1+\lceil \lg n\rceil} - 2\cdot 2^n + 2\cdot 2^n S_n$, $M_n = 3\cdot 2^{n-1+\lceil \lg n\rceil}$. Здесь S_n удовлетворяет рекуррентной формуле $S_1 = 0$, $S_n = 2S_{\lceil n/2\rceil} + \lfloor n/2\rfloor$, и несложно доказать неравенство $\frac{1}{2}n\lceil \lg n\rceil \le S_n \le S_{n+1} \le \frac{1}{2}n\lg n + n$ для всех $n \ge 1$. Алгоритм C выполняет примерно ту же работу, что и алгоритм N.

60. (a) В сумме Σ_1, например, можно сгруппировать все члены, имеющие общие значения j и k, в один трилинейный член; это даст ν^2 трилинейных членов, когда $(j,k) \in E\times E$, плюс ν^2, когда $(j,k) \in E\times O$, и ν^2, когда $(j,k) \in O\times E$. Когда $\tilde{j} = k$, число $-x_{j\tilde{j}}y_{j\tilde{j}}z_{\tilde{j}j}$ также без труда можно включить в Σ_1. [Для случая, когда $n = 10$, метод умножения матриц размера 10×10 дает 710 некоммутативных умножений; это почти так же хорошо, как семь умножений матриц размера 5×5 методом Макарова (метод упоминается в ответе к упр. 12), хотя в схеме Винограда (35) используется только 600, когда допускается коммутативность. В подобной схеме Пан показал, что для начала $M(n) < n^{2.8}$ для всех больших n. Это пробудило большой интерес к проблеме (см. *SICOMP* **9** (1980), 321–342).]

(b) Пусть здесь S — это все индексы (i,j,k) одной задачи, а $\tilde{S}$ — другой. [Когда $m = n = s = 10$, результат совершенно неожиданный: можно умножить две пары матриц размера 10×10, выполнив 1 300 некоммутативных умножений, в то время как схема для умножения каждой пары с 650 операциями неизвестна.]

61. (a) Заменить $a_{il}(u)$ величиной $ua_{il}(u)$. (b) Пусть $a_{il}(u) = \sum_\mu a_{il\mu}u^\mu$ и т. д. в реализации полинома длиной $r = \operatorname{rank}_d(t_{ijk})$. Тогда $t_{ijk} = \sum_{\mu+\nu+\sigma=d}\sum_{l=1}^r a_{il\mu}b_{jl\nu}c_{kl\sigma}$. [Этот результат можно улучшить так: $\operatorname{rank}(t_{ijk}) \le (2d+1)\operatorname{rank}_d(t_{ijk})$ в бесконечном поле, потому что трилинейная форма $\sum_{\mu+\nu+\sigma=d}a_\mu b_\nu c_\sigma$ соответствует умножению полиномов по модулю u^{d+1}, как показали Бини и Пан (Bini and Pan, *Calcolo* **17** (1980), 87–97).] (c, d) Это очевидно из реализаций упр. 48.

(e) Предположим, имеются реализации t и rt', такие, что $\sum_{l=1}^r a_{il}b_{jl}c_{kl} = t_{ijk}u^d + O(u^{d+1})$ и $\sum_{L=1}^R A_{\langle ii'\rangle L}B_{\langle jj'\rangle L}C_{\langle kk'\rangle L} = [i=j=k]\,t'_{i'j'k'}u^{d'} + O(u^{d'+1})$. Тогда

$$\sum_{L=1}^R\sum_{l=1}^r a_{il}A_{\langle li'\rangle L}\sum_{m=1}^r b_{jm}B_{\langle mj'\rangle L}\sum_{n=1}^r c_{kn}C_{\langle nk'\rangle L} = t_{ijk}t'_{i'j'k'}u^{d+d'} + O(u^{d+d'+1}).$$

62. Ранг равен 3 согласно методу доказательства в теореме W с $P = \left(\begin{smallmatrix}0&1\\0&0\end{smallmatrix}\right)$. Грань ранга не может быть равна 1, так как нельзя получить $a_1(u)b_1(u)c_1(u) \equiv a_1(u)b_2(u)c_2(u) \equiv u^d$

и $a_1(u)b_2(u)c_1(u) \equiv a_1(u)b_1(u)c_2(u) \equiv 0$ по модулю u^{d+1}. Грань ранга равна 2, потому что реализации равны $\left(\begin{smallmatrix}1&1\\u&0\end{smallmatrix}\right)$, $\left(\begin{smallmatrix}u&0\\1&1\end{smallmatrix}\right)$, $\left(\begin{smallmatrix}1&-1\\0&u\end{smallmatrix}\right)$.

Отметим, что грань ранга введена в работе Bini, Capovani, Lotti, and Romani, *Information Processing Letters* **8** (1979), 234–235.

63. (a) Пусть элементы $T(m,n,s)$ и $T(M,N,S)$ обозначены через $t_{\langle i,j'\rangle\langle j,k'\rangle\langle k,i'\rangle}$ и $T_{\langle I,J'\rangle\langle J,K'\rangle\langle K,I'\rangle}$ соответственно. Каждый элемент $\mathcal{T}_{\langle\mathcal{I},\mathcal{J}'\rangle\langle\mathcal{J},\mathcal{K}'\rangle\langle\mathcal{K},\mathcal{I}'\rangle}$ прямой суммы, где $\mathcal{I} = \langle i, I\rangle$, $\mathcal{J} = \langle j, J\rangle$ и $\mathcal{K} = \langle k, K\rangle$ равен $t_{\langle i,j'\rangle\langle j,k'\rangle\langle k,i'\rangle}T_{\langle I,J'\rangle\langle J,K'\rangle\langle K,I'\rangle}$ по определению; значит, он равен $[\mathcal{I}' = \mathcal{I}$ и $\mathcal{J}' = \mathcal{J}$ и $\mathcal{K}' = \mathcal{K}]$.

(b) Примените упр. 61, (e) с $M(N) = \underline{\text{rank}}_0(T(N,N,N))$.

(c) Имеем $M(mns) \le r^3$, так как $T(mns,mns,mns) = T(m,n,s)\otimes T(n,s,m)\otimes T(s,m,n)$. Если $M(n) \le R$, то $M(n^h) \le R^h$ для всех h, и, следовательно, $M(N) \le M(n^{\lceil\log_n N\rceil}) \le R^{\lceil\log_n N\rceil} \le RN^{\log R/\log n}$. [Этот результат появился в 1972 году в статье Пана.]

(d) Имеем $M_d(mns) \le r^3$ для некоторого d, где $M_d(n) = \text{rank}_d(T(n,n,n))$. Если $M_d(n) \le R$, то $M_{hd}(n^h) \le R^h$ для всех h и требуемая формула справедлива, так как $M(n^h) \le \binom{hd+2}{2}R^h$ согласно упр. 61, (b). В бесконечном поле остается множитель $\log N$. [Этот результат получен в 1979 году Бини (Bini) и Шёнхаге (Schönhage).]

64. Имеем $\sum_k(f_k(u) + \sum_{j\ne k} g_{j,k}(u)) = u^2\sum_{1\le i,j,k\le 3} x_{ij}y_{jk}z_{ki} + O(u^3)$, где $f_k(u) = (x_{k1} + u^2x_{k2})(y_{2k} + u^2y_{1k})z_{kk} + (x_{k1}+u^2x_{k3})y_{3k}((1+u)z_{kk} - u(z_{k1}+z_{k2}+z_{k3})) - x_{k1}(y_{2k}+y_{3k})(z_{k1}+z_{k2}+z_{k3})$ и $g_{jk}(u) = (x_{k1} + u^2x_{j2})(y_{2k} - uy_{1j})(z_{kj} - uz_{jk}) + (x_{k1} + u^2x_{j3})(y_{2k} + uy_{1j})z_{kj}$. [Лучшая верхняя грань, известная для $\text{rank}(T(3,3,3))$, равна 23 (см. ответ к упр. 12). Грань ранга $T(2,2,2)$ остается неизвестной.]

65. Полином в указании имеет вид $u^2\sum_{i=1}^m\sum_{j=1}^n(x_iy_jz_{ij} + X_{ij}Y_{ij}Z) + O(u^3)$. Пусть X_{ij} и Y_{ij} не определены для $1 \le i < m$ и $1 \le j < n$. Присвоим также $X_{in} = Y_{mj} = 0$, $X_{mj} = -\sum_{i=1}^{m-1} X_{ij}$, $Y_{in} = -\sum_{j=1}^{n-1} Y_{ij}$. Таким образом, с помощью $mn+1$ умножений полиномов в области неопределенности можно вычислить x_iy_j для каждого i и j, а также $\sum_{i=1}^m\sum_{j=1}^n X_{ij}Y_{ij} = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1} X_{ij}Y_{ij}$. [*SICOMP* **10** (1981), 434–455. В этой классической статье Шёнхаге среди прочих результатов получил результаты упр. 64, 66 и 67, (i).]

66. (a) Пусть $\omega = \liminf_{n\to\infty}\log M(n)/\log n$ по лемме T $\omega \ge 2$. Для всех $\epsilon > 0$ существует N с $M(N) < N^{\omega+\epsilon}$. Из доводов упр. 63, (c) следует, что $\log M(n)/\log n < \omega + 2\epsilon$ для всех достаточно больших N.

(b) Это непосредственно следует из упр. 63, (d).

(c) Пусть $r = \underline{\text{rank}}(t)$, $q = (mns)^{\omega/3}$, $Q = (MNS)^{\omega/3}$. Пусть задано $\epsilon > 0$, тогда существует целая постоянная c_ϵ, такая, что $M(p) \le c_\epsilon p^{\omega+\epsilon}$ для всех положительных целых p. Для каждого целого $h > 0$ получим $t^h = \bigoplus_k \binom{h}{k}T(m^kM^{h-k}, n^kN^{h-k}, s^kS^{h-k})$ и $\underline{\text{rank}}(t^h) \le r^h$. Пусть заданы h и k и пусть $p = \lfloor\binom{h}{k}^{1/(\omega+\epsilon)}\rfloor$. Тогда согласно упр. 63, (b)

$$\begin{aligned}\underline{\text{rank}}(T(pm^kM^{h-k}, pn^kN^{h-k}, ps^kS^{h-k})) &\le \underline{\text{rank}}(M(p)T(m^kM^{h-k}, n^kN^{h-k}, s^kS^{h-k}))\\ &\le \underline{\text{rank}}(c_\epsilon\tbinom{h}{k}T(m^kM^{h-k}, n^kN^{h-k}, s^kS^{h-k}))\\ &\le c_\epsilon r^h\end{aligned}$$

и из п. (b) следует, что

$$p^\omega q^k Q^{h-k} = (pm^kM^{h-k}pn^kN^{h-k}ps^kS^{h-k})^{\omega/3} \le c_\epsilon r^h.$$

Так как $p \ge \binom{h}{k}^{1/(\omega+\epsilon)}/2$, получим

$$\binom{h}{k}q^kQ^{h-k} \le \binom{h}{k}^{\epsilon/(\omega+\epsilon)}(2p)^\omega q^kQ^{h-k} \le 2^{\epsilon h/(\omega+\epsilon)}2^\omega c_\epsilon r^h.$$

Поэтому $(q+Q)^h \le (h+1)2^{\epsilon h/(\omega+\epsilon)}2^\omega c_\epsilon r^h$ для всех h. Следовательно, мы должны получить $q+Q \le 2^{\epsilon/(\omega+\epsilon)}r$ для всех $\epsilon > 0$.

(d) Пусть в упр. 65 $m = n = 4$, и заметим, что $16^{0.85} + 9^{0.85} > 17$.

67. (a) Матрица $(t_{\langle ij'\rangle\langle\langle jk'\rangle\langle ki'\rangle\rangle})$ размера $mn \times mns^2$ имеет ранг mn, потому что она является матрицей перестановок, когда ограничены mn строк, для которых $k = k' = 1$.

(b) $((t \oplus t')_{i(jk)})$, по существу, является $(t_{i(jk)}) \oplus (t'_{i(jk)})$ плюс $n's + sn'$ дополнительных нулевых столбцов. [Аналогично получаем $((t \otimes t')_{i(jk)}) = (t_{i(jk)}) \otimes (t'_{i(jk)})$ для прямых произведений.]

(c) Пусть D — диагональная матрица $\operatorname{diag}(d_1, \ldots, d_r)$, такая, что $ADB^T = O$. Из леммы T известно, что $\operatorname{rank}(A) = m$ и $\operatorname{rank}(B) = n$, отсюда $\operatorname{rank}(AD) = m$ и $\operatorname{rank}(DB^T) = n$. Без потери общности можно предположить, что первые m столбцов матрицы A линейно независимы. Так как столбцы B^T принадлежат пустому пространству AD, можно также предположить, что последние n столбцов матрицы B линейно независимы. Запишем матрицу A в виде разбиения $(A_1\, A_2\, A_3)$, где A_1 — матрица размера $m \times m$ (и невырожденная), A_2 — размера $m \times q$ и A_3 — размера $m \times n$. Разобьем D так, что $AD = (A_1D_1\, A_2D_2\, A_3D_3)$. Тогда существует матрица $W = (W_1\, I\, O)$ размера $q \times r$, такая, что $ADW^T = O$, т. е. $W_1 = -D_2A_2^TA_1^{-T}D_1^{-1}$. Аналогично можно записать $B = (B_1\, B_2\, B_3)$ и найти $VDB^T = O$, когда $V = (O\, I\, V_3)$ — матрица размера $q \times r$ с $V_3 = -D_2B_2^TB_3^{-T}D_3^{-1}$. Заметим, что $UDV^T = D_2$. Таким образом, утверждение указания более или менее установлено (в конце концов, это было всего лишь указание).

Сейчас положим, что $A_{il}(u) = a_{il}$ для $1 \le i \le m$, $A_{(m+i)l}(u) = uv_{il}/d_{m+i}$; $B_{jl}(u) = b_{jl}$ для $1 \le j \le n$, $B_{(n+j)l}(u) = w_{jl}u$; $C_{kl}(u) = u^2c_{kl}$ для $1 \le k \le s$, $C_{(s+1)l}(u) = d_l$. Следовательно, $\sum_{l=1}^r A_{il}(u)B_{jl}(u)C_{kl}(u) = u^2t_{ijk} + O(u^3)$, если $k \le s$, и $u^2[i > m][j > n]$, если $k = s+1$. [При этом доказательстве не будет необходимости в предположении, что t невырожденная относительно C.]

(d) Рассмотрим следующую реализацию $T(m, 1, n)$ с $r = mn + 1$: $a_{il} = [\lfloor l/n \rfloor = i-1]$, $b_{jl} = [l \bmod n = j]$, $b_{\langle ij\rangle l} = [l = (i-1)n + j]$, если $l \le mn$; $a_{ir} = 1$, $b_{jr} = -1$, $c_{\langle ij\rangle r} = 0$. Она допускает улучшение с $d_l = 1$ для $1 \le l \le r$.

(e) Идея состоит в нахождении допускающей улучшение реализации $T(m, n, s)$. Предположим, что (A, B, C) — реализация длиной r. Пусть заданы произвольные целые $\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_s$. Расширим A, B и C, определяя

$$A_{\langle ij'\rangle(r+p)} = \alpha_i[j' = p],\ B_{\langle jk'\rangle(r+p)} = \beta_{k'}[j = p],\ C_{\langle ki'\rangle(r+p)} = 0 \text{ для } 1 \le p \le n.$$

Пусть $d_l = \sum_{i'=1}^m \sum_{k=1}^s \alpha_{i'}\beta_k c_{\langle ki'\rangle l}$ для $l \le r$ и $d_l = -1$, в других случаях получим

$$\begin{aligned}\sum_{l=1}^{r+n} A_{\langle ij'\rangle l}B_{\langle jk'\rangle l}d_l &= \sum_{i'=1}^m \sum_{k=1}^s \alpha_{i'}\beta_k \sum_{l=1}^r A_{\langle ij'\rangle l}B_{\langle jk'\rangle l}C_{\langle ki'\rangle l} - \sum_{p=1}^n \alpha_i[j' = p]\,\beta_{k'}[j = p] \\ &= [j = j']\,\alpha_i\beta_{k'} - [j = j']\,\alpha_i\beta_{k'} = 0.\end{aligned}$$

Значит, такое расширение допускает улучшение, если $d_1 \ldots d_r \ne 0$. Но $d_1 \ldots d_r$ — полиномы от $(\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_s)$, не равные тождественно нулю, так как без потери общности можно предположить, что не все столбцы матрицы C равны нулю. Следовательно, будет работать несколько наборов α_k и β_i.

(f) Если $M(n) = n^\omega$, то получим $M(n^h) = n^{h\omega}$, отсюда

$$\underline{\operatorname{rank}}(T(n^h, n^h, n^h) \oplus T(1, n^{h\omega} - n^h(2n^h - 1), 1)) \le n^{h\omega} + n^h.$$

Из упр. 66, (c) следует неравенство $n^{h\omega} + (n^{h\omega} - 2n^{2h} + n^h)^{\omega/3} \le n^{h\omega} + n^h$ для всех h. Значит, $\omega = 2$, однако это противоречит существованию нижней грани $2n^2 - 1$ (см. ответ к упр. 12).

(g) Пусть $f(u)$ и $g(u)$ — полиномы, такие, что элементы $Vf(u)$ и $Wg(u)$ являются полиномами. Затем снова определим

$$A_{(i+m)l} = u^{d+1}v_{il}f(u)/d_{i+m},\ B_{(j+n)l} = u^{d+1}w_{jl}g(u)/p,\ C_{kl} = u^{d+e+2}c_{kl},$$

где $f(u)g(u) = pu^e + O(u^{e+1})$. Значит, $\sum_{l=1}^{r} A_{il}(u)B_{jl}(u)C_{kl}(u)$ равна $u^{d+e+2}t_{ijk} + O(u^{d+e+3})$, если $k \le s$, $u^{d+e+2}[i > m][j > n]$, если $k = s+1$. [*Замечание.* Следовательно, результат (e) имеет место для любого поля, если rank_2 заменить rank, так как можно выбрать α_k и β_i полиномами вида $1 + O(u)$.]

(h) Пусть строка p матрицы C приписывается компоненте $T(1, 16, 1)$. Основным является то, что $\sum_{l=1}^{r} a_{il}(u)b_{jl}(u)c_{pl}(u)$ равна нулю (не просто $O(u^{d+1})$) для всех i и j, оставшихся после удаления. Кроме того, $c_{pl}(u) \neq 0$ для всех l. Данные свойства справедливы для конструкций пп. (c) и (g), и они также остаются справедливыми для прямых произведений.

(i) Доказательство просто обобщается для полиномов с двумя и со многими переменными.

(j) Из п. (h) следует неравенство $81^{\omega/3} + 2(36^{\omega/3}) + 34^{\omega/3} \le 100$, поэтому $\omega < 2.52$. Возведение в квадрат дает $\operatorname{rank}(T(81,1,81) \oplus 4T(27,4,27) \oplus 2T(9,34,9) \oplus 4T(9,16,9) \oplus 4T(3,136,3) \oplus T(1,3334,1)) \le 10000$, что дает $\omega < 2.4999$. Успех! Продолжаем возводить в квадрат и получаем все лучшую и лучшую грань, которая быстро сходится к значению **2.497**723729083.... Если начать с $T(4,1,4) \oplus T(1,9,1)$, а не с $T(3,1,3) \oplus T(1,4,1)$, то предельной гранью будет значение 2.51096309....

[С помощью подобного ловкого приема получим $\omega < 2.496$ (см. *SICOMP* **11** (1982), 472–492).]

68. Т. М. Вэри (T. M. Vari) показал, что необходимо $n-1$ умножений, доказывая, что n умножений необходимы для вычисления $x_1^2 + \cdots + x_n^2$ [Cornell Computer Science Report 120 (1972)]. Ч. Панду Ранган (C. Pandu Rangan) показал, что если вычислять такие полиномы, как $L_1R_1 + \cdots + L_{n-1}R_{n-1}$, где L_i и R_i — линейные комбинации x_k, то необходимо хотя бы $n-2$ сложений для образования L_i и R_i [*J. Algorithms* **4** (1983), 282–285]. Но его нижняя грань, очевидно, не относится ко всем цепочкам полиномов.

69. Пусть $y_{ij} = x_{ij} - [i=j]$. Примените рекурсивную конструкцию (31) к матрице $I + Y$, используя арифметику степенных рядов от n^2 переменных y_{ij} и игнорируя все члены общей степени $> n$. Каждый элемент h матрицы представлен как сумма $h_0 + h_1 + \cdots + h_n$, где h_k — значение однородного полинома степени k. Затем каждый шаг сложения приводит к $n+1$ сложению и каждый шаг умножения приводит к $\approx \frac{1}{2}n^2$ умножениям и $\approx \frac{1}{2}n^2$ сложениям. Кроме того, каждое деление — это величина вида $1 + h_1 + \cdots + h_n$, так как все деления в рекуррентной конструкции — это деления на 1, когда y_{ij} — полностью равны нулю; поэтому деление является незначительно более простым, чем умножение (см. 4.7–(3), где $V_0 = 1$). Так как мы останавливаемся, когда размер определителя становится равным 2×2, нет необходимости вычитать 1 из y_{jj}, когда $j > n-2$. Оказывается, если избавиться от избыточных вычислений, этот метод потребует $20\binom{n}{5} + 8\binom{n}{4} + 12\binom{n}{3} - 4\binom{n}{2} + 5n - 4$ умножений и $20\binom{n}{5} + 8\binom{n}{4} + 4\binom{n}{3} + 24\binom{n}{2} - n$ сложений, т. е. $\frac{1}{6}n^5 - O(n^4)$ операций. Подобный метод можно использовать для исключения деления во многих других случаях (см. *Crelle* **264** (1973), 184–202). (Однако в следующем упражнении строится конструкция, которая даже быстрее схемы без делений для определителей.)

70. Положим, что $A = \lambda - x$, $B = -u$, $C = -v$ и $D = \lambda I - Y$ в указанном равенстве, затем вычислим определитель обеих частей, используя то, что $I/\lambda + Y/\lambda^2 + Y^2/\lambda^3 + \cdots$ — обратная к D как формальный степенной ряд от $1/\lambda$. Следует вычислить uY^kv только для $0 \le k \le n-2$, так как известно, что $f_X(\lambda)$ — полином степени n. Таким образом, необходимо всего $n^3 + O(n^2)$ умножений и $n^3 + O(n^2)$ сложений для перехода от степени

$n-1$ к степени n. Используя рекурсию, получим коэффициенты f_X из элементов X после $6\binom{n}{4}+7\binom{n}{3}+2\binom{n}{2}$ умножений и $6\binom{n}{4}+5\binom{n}{3}+2\binom{n}{2}$ сложений-вычитаний.

Вычислив только $\det X=(-1)^n f_X(0)$, можно сэкономить $3\binom{n}{2}-n+1$ умножений и $\binom{n}{2}$ сложений. Этот свободный от деления метод для вычисления определителя на самом деле совершенно экономный, когда n — умеренная величина; это вариант схемы разложения по алгебраическим дополнениям, когда $n>4$.

Если ω — показатель умножения матриц в упр. 66, то такое же приближение приводит к свободному от деления вычислению за $O(n^{\omega+1+\epsilon})$ шагов, поскольку векторы uY^k для $0\le k<n$ можно вычислить за $O(M(n)\log n)$ шагов. Возьмем матрицу, первые 2^l строк которой равны uY^k для $0\le k<2^l$, и умножим ее на Y^{2^l}. Тогда первые 2^l строк произведения будут равны uY^k для $2^l\le k<2^{l+1}$. [См. S. J. Berkowitz, *Inf. Processing Letters* **18** (1984), 147–150.] Конечно, такое "быстрое" асимптотическое умножение матриц представляет определенный теоретический интерес. Э. Калтофен (E. Kaltofen) показал, как вычислять определители только с помощью $O(n^{2+\epsilon}\sqrt{M(n)})$ операций сложения, вычитания и умножения [*Proc. Int. Symp. Symb. Alg. Comp.* **17** (1992), 342–349]; его метод представляет интерес даже при $M(n)=n^3$.

71. Предположим, что $g_1=u_1\circ v_1$, ..., $g_r=u_r\circ v_r$ и $f=\alpha_1 g_1+\cdots+\alpha_r g_r+p_0$, где $u_k=\beta_{k1}g_1+\cdots+\beta_{k(k-1)}g_{k-1}+p_k$, $v_k=\gamma_{k1}g_1+\cdots+\gamma_{k(k-1)}g_{k-1}+q_k$, каждый знак "$\circ$" является "$\times$" или "$/$" и каждое p_j или q_j — полиномом степени ≤ 1 от x_1, ..., x_n. Вычислим произвольные величины w_k, y_k, z_k для $k=r, r-1, \ldots, 1$ следующим образом: $w_k=\alpha_k+\beta_{(k+1)k}y_{k+1}+\gamma_{(k+1)k}z_{k+1}+\cdots+\beta_{rk}y_r+\gamma_{rk}z_r$ и

$$\begin{aligned} y_k&=w_k\times v_k, \quad & z_k&=w_k\times u_k, \quad & &\text{если } g_k=u_k\times v_k;\\ y_k&=w_k/v_k, \quad & z_k&=-y_k\times g_k, \quad & &\text{если } g_k=u_k/v_k. \end{aligned}$$

Затем $f'=p_0'+p_1'y_r+q_1'z_1+\cdots+p_r'y_r+q_r'z_r$, где $'$ означает производную по любому из $x_1,\ldots,x_n$. [W. Baur and V. Strassen, *Theoretical Comp. Sci.* **22** (1983), 317–330. Родственный метод опубликовал С. Линнаинмаа (S. Linnainmaa, *BIT* **16** (1976), 146–160), который применил его к анализу округления ошибок.] Мы сэкономим два умножения в цепочке, если $g_r=u_r\times v_r$, так как $w_r=\alpha_r$. Повторив конструкцию, можно задать все вторые частные производные с максимум $9m+3d$ умножениями в цепочке и $4d$ делениями.

72. Существует алгоритм вычисления ранга тензора над алгебраически замкнутыми полями, подобными комплексным числам, так как это частный случай результатов Альфреда Тарски (Alfred Tarski, *A Decision Method for Elementary Algebra and Geometry*, 2nd edition (Berkeley, California: Univ. of California Press, 1951)). Однако известные методы не делают эти вычисления действительно осуществимыми, за исключением очень малых тензоров. Неизвестно, будет ли решена эта проблема над полем рациональных чисел за конечное время.

73. В таких цепочках полиномов от N переменных определитель любой матрицы размера $N\times N$ для N линейных форм, полученных после l шагов сложений-вычитаний, равен максимум 2^l. И в дискретном преобразовании Фурье матрица последних $N=m_1\ldots m_n$ линейных форм имеет определитель $N^{N/2}$, поскольку ее квадрат равен N перестановкам матрицы из упр. 13 [*JACM* **20** (1973), 305–306].

74. (a) Если $k=(k_1,\ldots,k_s)^T$ — вектор взаимно простых целых чисел, значит, существует Uk, так как любой общий делитель элементов матрицы Uk делит все элементы $k=U^{-1}Uk$. Следовательно, матрица VUk не может иметь все целые компоненты.

(b) Допустим, существует цепочка полиномов для Vx с t умножениями. Если $t=0$, все элементы матрицы V должны быть целыми числами, значит, $s=0$. Иначе пусть $\lambda_i=\alpha\times\lambda_k$ или $\lambda_i=\lambda_j\times\lambda_k$ — первый шаг умножения. Можно предположить, что $\lambda_k=n_1x_1+\cdots+n_sx_s+\beta$, где n_1, ..., n_s — целые числа, не все равные нулю, и β — постоянные. Найдем

унимодулярную матрицу U, такую, что $(n_1,\ldots,n_s)U=(0,\ldots,0,d)$, где $d=\gcd(n_1,\ldots,n_s)$. (Приведенный перед равенством 4.5.2–(14) алгоритм безоговорочно определил такую матрицу U.) Построим новую цепочку полиномов с входными значениями y_1, ..., y_{s-1} следующим образом: сначала вычислим $x=(x_1,\ldots,x_s)^T=U(y_1,\ldots,y_{s-1},-\beta/d)^T$, затем продолжим вычисления с предполагаемой цепочкой полиномов для Vx. Когда достигнем шага i цепочки, получим $\lambda_k=(n_1,\ldots,n_s)x+\beta=0$, значит, можно просто положить $\lambda_i=0$, а не выполнять умножение. Затем вычисляем Vx и добавляем постоянный вектор $w\beta/d$ к результату, где w — наибольший правый столбец VU. Пусть W — $s-1$ остальных столбцов VU. Новая цепочка полиномов вычисляет $Vx+w\beta/d=VU(y_1,\ldots,y_{s-1},-\beta/d)^T+w\beta/d=W(y_1,\ldots,y_{s-1})^T$ с $t-1$ умножениями. Однако столбцы W являются Z-независимыми согласно п. (a); следовательно, $t-1\ge s-1$ по индукции по s и получаем $t\ge s$.

(c) Пусть $x_j=0$ для $t-s$ значений j, которых нет в Z-независимых столбцах. С помощью любой цепочки для Vx вычисляем $V'x'$ для матрицы V', применив результаты п. (b).

(d) $\lambda_1=x-y$, $\lambda_2=\lambda_1+\lambda_1$, $\lambda_3=\lambda_2+x$, $\lambda_4=(1/6)\times\lambda_3$, $\lambda_5=\lambda_4+\lambda_4$, $\lambda_6=\lambda_5+y$ $(=x+y/3)$, $\lambda_7=\lambda_6-\lambda_1$, $\lambda_8=\lambda_7+\lambda_4$ $(=x/2+y)$. Но для $\{x/2+y, x+y/2\}$ необходимы два умножения, поскольку столбцы $\left(\begin{smallmatrix}1/2 & 1\\ 1 & 1/2\end{smallmatrix}\right)$ Z-независимы. [*Journal of Information Processing* **1** (1978), 125–129.]

РАЗДЕЛ 4.7

1. Найдите первый не равный нулю коэффициент V_m, как в (4), и разделите $U(z)$ и $V(z)$ на z^m (сдвигая коэффициенты на m мест влево). Отношение будет степенным рядом тогда и только тогда, когда $U_0=\cdots=U_{m-1}=0$.

2. Имеем $V_0^{n+1}W_n=V_0^nU_n-(V_0^1W_0)(V_0^{n-1}V_n)-(V_0^2W_1)(V_0^{n-2}V_{n-1})-\cdots-(V_0^nW_{n-1})(V_0^0V_1)$. Тогда можно начать с замены (U_j,V_j) на $(V_0^jU_j,V_0^{j-1}V_j)$ для $j\ge1$, затем присвоить $W_n\leftarrow U_n-\sum_{k=0}^{n-1}W_kV_{n-k}$ для $n\ge0$ и наконец заменить W_j на W_j/V_0^{j+1} для $j\ge0$. Подобная техника возможна в сочетании с другими алгоритмами данного раздела.

3. Да. Если $\alpha=0$, то легко доказать индукцией, что $W_1=W_2=\cdots=0$. Когда $\alpha=1$, найдем, что $W_n=V_n$ согласно тождеству

$$\sum_{k=1}^{n}\left(\frac{k-(n-k)}{n}\right)V_kV_{n-k}=V_nV_0.$$

4. Если $W(z)=e^{V(z)}$, то $W'(z)=V'(z)W(z)$; находим $W_0=e^{V_0}$ и

$$W_n=\sum_{k=1}^{n}\frac{k}{n}V_kW_{n-k}\qquad\text{для } n\ge1.$$

Если $W(z)=\ln V(z)$, то V и W меняются ролями. Значит, когда $V_0=1$, устанавливается правило: $W_0=0$ и $W_n=V_n+\sum_{k=1}^{n-1}(k/n-1)V_kW_{n-k}$ для $n\ge1$.

[Используя упр. 6, коэффициент W_n для логарифма можно вычислить за $O(n\log n)$ операций. Р. П. Брент (R. P. Brent) заметил, что $\exp(V(z))$ также можно вычислить асимптотически с такой же скоростью, применяя метод Ньютона к $f(x)=\ln x-V(z)$. Следовательно, обычное возведение в степень $(1+V(z))^\alpha=\exp(\alpha\ln(1+V(z)))$ также можно выполнить за $O(n\log n)$ операций. См. *Analytic Computational Complexity*, edited by J. F. Traub (New York: Academic Press, 1975), 172–176.]

5. Получится начальный степенной ряд. Это можно использовать в качестве критерия алгоритма обращения.

6. $\phi(x)=x+x(1-xV(z))$ (см. алгоритм 4.3.3R). Затем, когда W_0, ..., W_{N-1} известны, основная идея заключается во вводе V_N, ..., V_{2N-1}, вычислении $(W_0+\cdots+W_{N-1}z^{N-1})\times(V_0+\cdots+V_{2N-1}z^{2N-1})=1+R_0z^N+\cdots+R_{N-1}z^{2N-1}+O(z^{2N})$ и использовании равенства

$W_N + \cdots + W_{2N-1}z^{N-1} = -(W_0 + \cdots + W_{N-1}z^{N-1})(R_0 + \cdots + R_{N-1}z^{N-1}) + O(z^N)$. [*Numer. Math.* **22** (1974), 341–348; данный алгоритм, по существу, впервые опубликован в работе M. Sieveking, *Computing* **10** (1972), 153–156.] Заметим, что общие затраты на вычисление N коэффициентов равняются $O(N \log N)$ арифметическим операциям, если использовать "быстрое" умножение полиномов (см. упр. 4.6.4–57).

7. $W_n = \binom{mk}{k}/n$, когда $n = (m-1)k + 1$; иначе — 0 (см. упр. 2.3.4.4–11).

8. **G1.** Ввести G_1 и V_1; присвоить $n \leftarrow 1$, $U_0 \leftarrow 1/V_1$; вывести $W_1 = G_1U_0$.

G2. Увеличить n на 1. Если $n > N$, работа алгоритма окончена; иначе — ввести V_n и G_n.

G3. Присвоить $U_k \leftarrow (U_k - \sum_{j=1}^{k} U_{k-j}V_{j+1})/V_1$ для $k = 0, 1, \ldots, n-2$ (в таком порядке), затем присвоить $U_{n-1} \leftarrow -\sum_{k=2}^{n} kU_{n-k}V_k/V_1$.

G4. Вывести $W_n = \sum_{k=1}^{n} kU_{n-k}G_k/n$ и возвратиться к шагу G2. ▮

(Время работы алгоритма — порядка N^3; таким образом, оно увеличилось только относительно порядка N^2.)

Замечание. Алгоритмы T и N определяют $V^{[-1]}(U(z))$, алгоритм данного упражнения определяет $G(V^{[-1]}(z))$, что не одно и то же. Безусловно, все результаты можно получить, выполнив операцию обращения, а затем — композиции (упр. 11), но полезно иметь для каждого случая более прямой алгоритм.

9.

	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$
T_{1n}	1	1	2	5	14
T_{2n}		1	2	5	14
T_{3n}			1	3	9
T_{4n}				1	4
T_{5n}					1

10. С помощью (9) получаем $y^{1/\alpha} = x(1 + a_1x + a_2x^2 + \cdots)^{1/\alpha} = x(1 + c_1x + c_2x^2 + \cdots)$ и затем обращаем последние ряды. (См. замечание, следующее после уравнения 1.2.11.3–(11).)

11. Присвоить сначала $W_0 \leftarrow U_0$, а затем — $(T_k, W_k) \leftarrow (V_k, 0)$ для $1 \le k \le N$. После этого для $n = 1, 2, \ldots, N$ выполнить следующие операции: присвоить сначала $W_j \leftarrow W_j + U_nT_j$ для $n \le j \le N$, а затем — $T_j \leftarrow T_{j-1}V_1 + \cdots + T_nV_{j-n}$ для $j = N, N-1, \ldots, n+1$.

Здесь вместо $T(z)$ стоит $V(z)^N$. Можно построить *интерактивный* алгоритм степенных рядов для этой задачи, аналогичный алгоритму T, но потребуется около $N^2/2$ ячеек для хранения данных. Для выполнения данного упражнения также существует интерактивный алгоритм, которому необходимо только $O(N)$ ячеек: можно предположить, что $V_1 = 1$, если U_k заменить на $U_kV_1^k$ и V_k заменить на V_k/V_1 для всех k. Тогда с помощью алгоритма L можно обратить $V(z)$ и использовать его выходные данные в качестве входных данных в алгоритм упр. 8 с $G_1 = U_1$, $G_2 = U_2$ и т. д. Тогда вычислим $U(V^{[-1][-1]}(z)) - U_0$. (См. также упр. 20.)

Брент и Кунг построили несколько асимптотически быстрых алгоритмов. Например, можно вычислить $U(x)$ для $x = V(z)$ с помощью варианта алгоритма, который предложен в упр. 4.6.4–42(c), выполнив около $2\sqrt{N}$ умножений в цепочке с $M(N)$ или около N умножений параметров с количеством операций, равным N, где $M(N)$ — число операций, необходимых для умножения степенных рядов порядка N. Следовательно, общее число операций равно $O(\sqrt{N}M(N) + N^2) = O(N^2)$. Еще более быстрый метод может быть основан на тождестве $U(V_0(z) + z^mV_1(z)) = U(V_0(z)) + z^mU'(V_0(z))V_1(z) + z^{2m}U''(V_0(z))V_1(z)^2/2! + \cdots$, включающем около N/m членов, из которых выбираем $m \approx \sqrt{N/\log N}$. Для вычисления первых членов $U(V_0(z))$ потребуется $O(mN(\log N)^2)$ операций, если использовать метод.

отчасти подобный рассмотренному в упр. 4.6.4–43. Так как перейти от $U^{(k)}(V_0(z))$ к $U^{(k+1)}(V_0(z))$ можно за $O(N \log N)$ операций, выполнив дифференцирование и деление на $V_0'(z)$, то полностью процедура будет содержать $O(mN(\log N)^2 + (N/m)\, N \log N) = O(N \log N)^{3/2}$ операций. [*JACM* **25** (1978), 581–595.]

Когда полином имеет целые коэффициенты, состоящие из m двоичных разрядов, данный алгоритм включает приблизительно $N^{3/2+\epsilon}$ умножений чисел, содержащих $(N \lg m)$ двоичных разрядов. Таким образом, общее время работы алгоритма больше $N^{5/2}$. Альтернативный подход с асимптотическим временем счета $O(N^{2+\epsilon})$ развит в работе P. Ritzmann, *Theoretical Comp. Sci.* **44** (1986), 1–16. Композиция может выполняться быстрее по модулю малого простого числа p (см. упр. 26).

12. Деление полиномов тривиально, кроме случая, когда $m \geq n \geq 1$. В последнем случае уравнение $u(x) = q(x)v(x) + r(x)$ эквивалентно уравнению $U(z) = Q(z)V(z) + z^{m-n+1}R(z)$, где $U(x) = x^m u(x^{-1})$, $V(x) = x^n v(x^{-1})$, $Q(x) = x^{m-n}q(x^{-1})$ и $R(x) = x^{n-1}r(x^{-1})$ — "обратные" полиномы от u, v, q и r.

Чтобы найти $q(x)$ и $r(x)$, вычислим первые $m-n+1$ коэффициентов степенного ряда $U(z)/V(z) = W(z) + O(z^{m-n+1})$, затем вычислим степенной ряд $U(z) - V(z)W(z)$, который имеет вид $z^{m-n+1}T(z)$, где $T(z) = T_0 + T_1 z + \cdots$. Заметим, что $T_j = 0$ для всех $j \geq n$, следовательно, $Q(z) = W(z)$ и $R(z) = T(z)$ удовлетворяют требованиям.

13. Используем упр. 4.6.1–3 с $u(z) = z^N$ и $v(z) = W_0 + \cdots + W_{N-1}z^{N-1}$. Ожидаемые аппроксимации — это значения $v_3(z)/v_2(z)$, полученные, конечно, с помощью алгоритма. Из упр. 4.6.1–26 следует, что не существует дополнительных возможностей со взаимно простыми числителем и знаменателем. Если каждое W_i — целое, то алгоритм 4.6.1C, примененный ко всем целым числам, будет обладать требуемыми свойствами.

Замечание. Те, кого интересует более полная информация, могут обратиться к книге Claude Brezinski, *History of Continued Fractions and Padé Approximants* (Berlin: Springer, 1991). Случай, когда $N = 2n+1$ и $\deg(w_1) = \deg(w_2) = n$, в частности, представляет интерес, поскольку он эквивалентен так называемой системе Теплица. Асимптотически быстрые методы для систем Теплица рассмотрены в работе Bini and Pan, *Polynomial and Matrix Computations* **1** (Boston: Birkhäuser, 1994), §2.5. Метод, предложенный в этом упражнении, можно обобщить на произвольную рациональную интерполяцию вида $W(z) \equiv p(z)/q(z)$ (по модулю $(z-z_1)\ldots(z-z_N)$), где z_i необязательно должны быть различны. Таким образом, можно точно определить значения $W(z)$ и несколько их производных в нескольких точках (см. работу Richard P. Brent, Fred G. Gustavson, and David Y. Y. Yun, *J. Algorithms* **1** (1980), 259–295).

14. Если $U(z) = z + U_k z^k + \cdots$ и $V(z) = z^k + V_{k+1}z^{k+1} + \cdots$, находим, что разность $V(U(z)) - U'(z)V(z)$ равна $\sum_{j\geq 1} z^{2k+j-1} j(U_k V_{k+j} - U_{k+j} +$ (полином включает только $U_k, \ldots, U_{k+j-1}, V_{k+1}, \ldots, V_{k+j-1}$)). Следовательно, $V(z)$ определен единственным образом, если задан $U(z)$, как и $U(z)$, для заданных $V(z)$ и U_k.

Решение зависит от двух вспомогательных алгоритмов; первый из них позволяет решить уравнение $V(z + z^k U(z)) = (1 + z^{k-1}W(z))V(z) + z^{k-1}S(z) + O(z^{k-1+n})$ для $V(z) = V_0 + V_1 z + \cdots + V_{n-1}z^{n-1}$, где $U(z)$, $W(z)$, $S(z)$ и n заданы. Если $n = 1$, положим $V_0 = -S(0)/W(0)$ или выберем V_0 произвольно, когда $S(0) = W(0) = 0$. Перейдем от n к $2n$. Пусть

$$\begin{aligned} V(z + z^k U(z)) &= (1 + z^{k-1}W(z))V(z) + z^{k-1}S(z) - z^{k-1+n}R(z) + O(z^{k-1+2n}), \\ 1 + z^{k-1}\hat{W}(z) &= (z/(z + z^k U(z)))^n (1 + z^{k-1}W(z)) + O(z^{k-1+n}), \\ \hat{S}(z) &= (z/(z + z^k U(z)))^n R(z) + O(z^n) \end{aligned}$$

и пусть $\hat V(z) = V_n + V_{n+1}z + \cdots + V_{2n-1}z^{n-1}$ удовлетворяет уравнению

$$\hat V(z + z^k U(z)) = (1 + z^{k-1}\hat W(z))\hat V(z) + z^{k-1}\hat S(z) + O(z^{k-1+n}).$$

С помощью второго алгоритма решаем $W(z)U(z) + zU'(z) = V(z) + O(z^n)$ для $U(z) = U_0 + U_1 z + \cdots + U_{n-1}z^{n-1}$, где $V(z)$, $W(z)$ и n заданы. Если $n = 1$, то положим $U_0 = V(0)/W(0)$ или выберем U_0 произвольно для случая, когда $V(0) = W(0) = 0$. Перейдем от n к $2n$. Пусть $W(z)U(z) + zU'(z) = V(z) - z^n R(z) + O(z^{2n})$ и пусть $\hat U(z) = U_n + \cdots + U_{2n-1}z^{n-1}$ — решение уравнения $(n + W(z))\hat U(z) + z\hat U'(z) = R(z) + O(z^n)$.

Применив обозначения (27), первый алгоритм можно использовать для решения уравнения $\hat V(U(z)) = U'(z)(z/U(z))^k \hat V(z)$ с любой требуемой точностью. Положим $V(z) = z^k \hat V(z)$. Чтобы найти $P(z)$, предположим, что $V(P(z)) = P'(z)V(z) + O(z^{2k-1+n})$. Данное уравнение справедливо для $n = 1$, когда $P(z) = z + \alpha z^k$ и α — произвольное. Можно перейти от n к $2n$, полагая, что $V(P(z)) = P'(z)V(z) + z^{2k-1+n}R(z) + O(z^{2k-1+2n})$, и заменяя $P(z)$ на $P(z) + z^{k+n}\hat P(z)$, где второй алгоритм используется для нахождения полинома $\hat P(z)$, такого, что $(k + n - zV'(P(z))/V(z))\hat P(z) + z\hat P'(z) = (z^k/V(z))R(z) + O(z^n)$.

15. Из дифференциального уравнения $U'(z)/U(z)^k = 1/z^k$ следует, что $U(z)^{1-k} = z^{1-k} + c$ для некоторой постоянной c. Таким образом, находим, что $U^{[n]}(z) = z/(1 + cnz^{1-k})^{1/(k-1)}$.

Аналогично решаем (27) для произвольного $V(z)$: если $W'(z) = 1/V(z)$, то $W(U^{[n]}(z)) = W(z) + nc$ для некоторого c.

16. Нужно показать, что $[t^n]\, t^{n+1}((n+1)R'_{k+1}(t)/V(t)^n - nR'_k(t)/V(t)^{n+1}) = 0$. Это следует из равенства $(n+1)R'_{k+1}(t)/V(t)^n - nR'_k(t)/V(t)^{n+1} = \frac{d}{dt}(R_k(t)/V(t)^{n+1})$. Поэтому получаем $n^{-1}[t^{n-1}]\, R'_1(t)\, t^n/V(t)^n = (n-1)^{-1}[t^{n-2}]\, R'_2(t)\, t^{n-1}/V(t)^{n-1} = \cdots = 1^{-1}[t^0]\, R'_n(t)\, t/V(t) = [t]\, R_n(t)/V_1 = W_n$.

17. Соберем коэффициенты при $x^l y^m$. Из формулы свертки следует, что $\binom{l+m}{m} v_{n(l+m)} = \sum_k \binom{n}{k} v_{kl} v_{(n-k)m}$, а это эквивалентно равенству $[z^n]\, V(z)^{l+m} = \sum_k ([z^k]V(z)^l)([z^{n-k}]\, V(z)^m)$, являющемуся частным случаем (2).

Замечание. Название "степеноид" введено Й. Ф. Стеффенсеном (J. F. Steffensen), который первым из многих авторов изучил удивительные свойства этих полиномов [*Acta Mathematica* **73** (1941), 333–366]. Обзор литературы и дальнейшее обсуждение предмета можно найти в следующих нескольких упражнениях, а также в статье D. E. Knuth, *The Mathematica Journal* **2** (1992), 67–78. Одним из результатов, полученных в этой статье, является асимптотическая формула $V_n(x) = e^{xV(s)}(\frac{n}{es})^n(1 - V_2 y + O(y^2) + O(x^{-1}))$, если $V_1 = 1$, и $sV'(s) = y$ и $y = n/x$ ограничены, когда $x \to \infty$ и $n \to \infty$.

18. Имеем $V_n(x) = \sum_k x^k n!\, [z^n]\, V(z)^k/k! = n!\, [z^n]\, e^{xV(z)}$. Поэтому

$$V_n(x)/x = (n-1)!\, [z^{n-1}]\, V'(z)\, e^{xV(z)},$$

когда $n > 0$. Равенство можно получить, если приравнять коэффициенты при z^{n-1} в равенстве $V'(z)\, e^{(x+y)V(z)} = V'(z)\, e^{xV(z)} e^{yV(z)}$.

19. Согласно полиномиальной теореме 1.2.6–(42) имеем

$$\begin{aligned} v_{nm} &= \frac{n!}{m!}\,[z^n]\left(\frac{v_1}{1!}z + \frac{v_2}{2!}z^2 + \frac{v_3}{3!}z^3 + \cdots\right)^m \\ &= \sum_{\substack{k_1+k_2+\cdots+k_n=m \\ k_1+2k_2+\cdots+nk_n=n \\ k_1,k_2,\ldots,k_n\ge 0}} \frac{n!}{k_1!\,k_2!\ldots k_n!}\left(\frac{v_1}{1!}\right)^{k_1}\left(\frac{v_2}{2!}\right)^{k_2}\cdots\left(\frac{v_n}{n!}\right)^{k_n} \end{aligned}$$

Эти коэффициенты появляются также в формуле Арбогаста (упр. 1.2.5–21). Эти члены уравнения можно привести в соответствие с множеством разбиений, как объясняется в

ответе к данному упражнению. Рекуррентная формула

$$v_{nk} = \sum_j \binom{n-1}{j-1} v_j v_{(n-j)(k-1)}$$

позволяет вычислить столбец k по столбцам 1 и $k-1$; она легко интерпретируется в терминах разбиений $\{1,\ldots,n\}$, поскольку существует $\binom{n-1}{j-1}$ способов включения элемента n в подмножество размера j. Несколько первых строк матрицы имеют следующий вид.

$$\begin{array}{lllll} v_1 & & & & \\ v_2 & v_1^2 & & & \\ v_3 & 3v_1v_2 & v_1^3 & & \\ v_4 & 4v_1v_3+3v_2^2 & 6v_1^2v_2 & v_1^4 & \\ v_5 & 5v_1v_4+10v_2v_3 & 15v_1v_2^2+10v_1^2v_3 & 10v_1^3v_2 & v_1^5 \end{array}$$

20. $[z^n]\,W(z)^k = \sum_j ([z^j]\,U(z)^k)([z^n]\,V(z)^j)$, следовательно, $w_{nk} = (n!/k!)\sum_j ((k!/j!)u_{jk}) \times ((j!/n!)v_{nj})$. [E. Jabotinsky, *Comptes Rendus Acad. Sci. Paris* **224** (1947), 323–324.]

21. (a) Если $U(z) = \alpha W(\beta z)$, получим $u_{nk} = \frac{n!}{k!}\,[z^n]\,(\alpha W(\beta(z))^k = \alpha^k\beta^n w_{nk}$; в частности, если $U(z) = V^{[-1]}(z) = -W(-z)$, то $u_{nk} = (-1)^{n-k}w_{nk}$. Значит, согласно упр. 20 $\sum_k u_{nk}v_{km}$ и $\sum_k v_{nk}u_{km}$ соответствуют функции z.

(b) [Решение И. Гессель (Ira Gessel).] Данное тождество фактически эквивалентно формуле обращения Лагранжа: имеем $w_{nk} = (-1)^{n-k}u_{nk} = (-1)^{n-k}\frac{n!}{k!}\,[z^n]\,V^{[-1]}(z)^k$, и согласно упр. 16 коэффициент при z^n в $V^{[-1]}(z)^k$ равен $n^{-1}\,[t^{n-1}]\,kt^{n+k-1}/V(t)^n$. С другой стороны, определим $v_{(-k)(-n)}$. Это $(-k)^{\underline{n-k}}\,[z^{n-k}]\,(V(z)/z))^{-n}$, в свою очередь, равно $(-1)^{n-k}(n-1)\ldots(k+1)k\,[z^{n-1}]\,z^{n+k-1}/V(z)^n$.

22. (a) Если $V(z) = U^{\{\alpha\}}(z)$ и $W(z) = V^{\{\beta\}}(z)$, то

$$W(z) = V(zW(z)^\beta) = U(zW(z)^\beta\,V(zW(z)^\beta)^\alpha) = U(zW(z)^{\alpha+\beta}).$$

(Отметим разницу между данным законом и подобными формулами $U^{[1]}(z) = U(z)$, $U^{[\alpha][\beta]}(z) = U^{[\alpha\beta]}(z)$, которые применяются в итерации.)

(b) $B^{\{2\}}(z)$ — производящая функция для бинарных деревьев, 2.3.4.4–(12), равная $W(z)/z$, в примере $z = t - t^2$, который следует за алгоритмом L. Кроме того, $B^{\{t\}}(z)$ — производящая функция для t-ичных деревьев (упр. 2.3.4.4–11).

(c) Указание эквивалентно равенству $zU^{\{\alpha\}}(z)^\alpha = W^{[-1]}(z)$, которое, в свою очередь, эквивалентно формуле $zU^{\{\alpha\}}(z)^\alpha/U(zU^{\{\alpha\}}(z)^\alpha)^\alpha = z$. Из теоремы обращения Лагранжа (упр. 8) следует, что $[z^n]\,W^{[-1]}(z)^x = \frac{x}{n}[z^{-x}]\,W(z)^{-n}$, где x — положительное целое число. (Здесь $W(z)^{-n}$, ряд Лорана, степенной ряд, деленный на степень z; можно воспользоваться обозначениями $[z^m]\,V(z)$ для ряда Лорана, как и для степенного ряда.) Следовательно, $[z^n]\,U^{\{\alpha\}}(z)^x = [z^n]\,(W^{[-1]}(z)/z)^{x/\alpha} = [z^{n+x/\alpha}]\,W^{[-1]}(z)^{x/\alpha}$ равняется

$$\tfrac{x/\alpha}{n+x/\alpha}\,[z^{-x/\alpha}]\,W(z)^{-n-x/\alpha} = \tfrac{x}{x+n\alpha}\,[z^{-x/\alpha}]\,z^{-n-x/\alpha}U(z)^{x+n\alpha},$$

где x/α — положительное целое число. Мы получили результат для бесконечного множества α. Этого достаточно, так как коэффициенты $U^{\{\alpha\}}(z)^x$ — полиномы от α.

Выше уже рассматривались частные случаи данного результата (в упр. 1.2.6–25 и 2.3.4.4–29). Одно запоминающееся следствие указания — это случай, когда $\alpha = -1$:

$$W(z) = zU(z) \qquad \text{тогда и только тогда, когда} \qquad W^{[-1]}(z) = z/U^{\{-1\}}(z)\,.$$

(d) Если $U_0 = 1$ и $V_n(x)$ — степеноид для $V(z) = \ln U(z)$, то, как уже было доказано, $xV_n(x+n\alpha)/(x+n\alpha)$ — степеноид для $\ln U^{\{\alpha\}}(z)$. Значит, можно вставить данный степеноид в первое тождество, заменяя y на $y - \alpha n$ во второй формуле.

23. (a) Имеем $U = I + T$, где T^n равно нулю в строках $\le n$. Значит, $\ln U = T - \frac{1}{2}T^2 + \frac{1}{3}T^3 - \cdots$ обладает таким свойством $\exp(\alpha \ln U) = I + \binom{\alpha}{1}T + \binom{\alpha}{2}T^2 + \cdots = U^\alpha$. Каждое вводимое U^α — это полином от α, и соотношения упр. 19 выполняются всякий раз, когда α — положительное целое число; кроме того, U^α — степенная матрица для всех α и ее первые столбцы определяют $U^{[\alpha]}(z)$. (В частности, U^{-1} — степенная матрица; это другой метод обращения $U(z)$.)

(b) Так как $U^\epsilon = I + \epsilon \ln U + O(\epsilon^2)$, получим

$$l_{nk} = [\epsilon]u_{nk}^{[\epsilon]} = \frac{n!}{k!}[z^n][\epsilon]\,(z + \epsilon L(z) + O(\epsilon^2))^k = \frac{n!}{k!}[z^n]\,kz^{k-1}L(z).$$

(c) $\frac{\partial}{\partial\alpha}U^{[\alpha]}(z) = [\epsilon]\,U^{[\alpha+\epsilon]}(z)$, и получаем

$$U^{[\alpha+\epsilon]}(z) = U^{[\alpha]}(U^{[\epsilon]}(z)) = U^{[\alpha]}(z + \epsilon L(z) + O(\epsilon^2)).$$

Также $U^{[\alpha+\epsilon]}(z) = U^{[\epsilon]}(U^{[\alpha]}(z)) = U^{[\alpha]}(z) + \epsilon L(U^{[\alpha]}(z)) + O(\epsilon^2)$.

(d) Равенство следует из того факта, что U меняются местами с $\ln U$. Оно определяет l_{n-1}, когда $n \ge 4$, потому что коэффициент при l_{n-1} в левой части равенства равен nu_2, а коэффициент в правой части равенства есть $u_{n(n-1)} = \binom{n}{2}u_2$. Аналогично, если $u_2 = \cdots = u_{k-1} = 0$ и $u_k \ne 0$, получаем $l_k = u_k$ и по рекуррентной формуле для $n \ge 2k$ определяем l_{k+1}, l_{k+2}, ... Левая часть равенства имеет вид $l_n + \binom{n}{k-1}l_{n+1-k}u_k + \cdots$, а правая — $l_n + \binom{n}{k}l_{n+1-k}u_k + \cdots$. Вообще, $l_2 = u_2$, $l_3 = u_3 - \frac{3}{2}u_2^2$, $l_4 = u_4 - 5u_2u_3 + \frac{9}{2}u_2^3$, $l_5 = u_5 - \frac{15}{2}u_2u_4 - 5u_3^2 + \frac{185}{6}u_2^2u_3 - 20u_2^4$.

(e) Справедливо $U = \sum_m (\ln U)^m/m!$, и для фиксированного m вклад в $u_n = u_{n1}$ от m-го члена равен $\sum l_{n_m n_{m-1}} \ldots l_{n_2 n_1} l_{n_1 n_0}$, где суммирование по $n = n_m > \cdots > n_1 > n_0 = 1$. Применим полученный результат к п. (b). [См. *Trans. Amer. Math. Soc.* **108** (1963), 457–477.]

24. (a) Из (21) и упр. 20 получаем $U = VDV^{-1}$, где V — степенная матрица функции Шрёдера и D — диагональная матрица $\operatorname{diag}(u, u^2, u^3, \ldots)$. Тогда можно взять

$$\ln U = V \operatorname{diag}(\ln u, 2\ln u, 3\ln u, \ldots)V^{-1}.$$

(b) Равенство $WVDV^{-1} = VDV^{-1}W$ влечет равенство $(V^{-1}WV)D = D(V^{-1}WV)$. Диагональные элементы матрицы D различны, значит, $V^{-1}WV$ должна быть диагональной матрицей D'. Таким образом, $W = VD'V^{-1}$ и W имеет такую же функцию Шрёдера, как и U. Следовательно, $W_1 \ne 0$ и $W = VD^\alpha V^{-1}$, где $\alpha = (\ln W_1)/(\ln U_1)$.

25. Должно быть $k = l$, потому что $[z^{k+l-1}]U(V(z)) = U_{k+l-1} + V_{k+l-1} + kU_kV_l$. Для полного доказательства достаточно показать, что $U_k = V_k$ и $U(V(z)) = V(U(z))$ влечет $U(z) = V(z)$. Предположим, что l минимально с $U_l \ne V_l$, и пусть $n = k + l - 1$. Тогда получаем $u_{nk} - v_{nk} = \binom{n}{l}(u_l - v_l)$, $u_{nj} = v_{nj}$ для $j > k$, $u_{nl} = \binom{n}{k}u_k$ и $u_{nj} = 0$ для $l < j < n$. Сейчас сумма $\sum_j u_{nj}v_j = u_n + u_{nk}v_k + \cdots + u_{nl}v_l + v_n$ должна равняться $\sum_j v_{nj}u_j$. Значит, $\binom{n}{l}(u_l - v_l)v_k = \binom{n}{k}v_k(u_l - v_l)$, но $\binom{k+l-1}{k} = \binom{k+l-1}{l}$ тогда и только тогда, когда $k = l$.

[Из данного и предыдущего упражнений можно предположить, что $U(V(z)) = V(U(z))$ только тогда, когда одно из U и V является итерацией другого. Но это необязательно, если U_1 и V_1 — корни из единицы. Например, если $V_1 = -1$ и $U(z) = V^{[2]}(z)$, V не является итерацией $U^{[1/2]}$ и $U^{[1/2]}$ — итерацией V.]

26. Записав $U(z) = U_{[0]}(z^2) + zU_{[1]}(z^2)$, получаем $U(V(z)) \equiv U_{[0]}(V_1z^2 + V_2z^4 + \cdots) + V(z)U_{[1]}(V_1z^2 + V_2z^4 + \cdots)$ (по модулю 2). Время счета удовлетворяет равенству $T(N) = 2T(N/2) + C(N)$, где $C(N)$ — это, по существу, время, расходуемое на умножение полиномов по модулю z^N. Можно сделать $C(N) = O(N^{1+\epsilon})$ методом, упр. 4.6.4–59 (см. также ответ к упр. 4.6–5).

Аналогичный метод работает по модулю p за время $O(pN^{1+\epsilon})$. [D. J. Bernstein, в печати.]

27. Из $(W(qz) - W(z))V(z) = W(z)(V(q^m z) - V(z))$ получаем рекуррентную формулу $W_n = \sum_{k=1}^{n} V_k W_{n-k}(q^{km} - q^{n-k})/(q^n - 1)$. [*J. Difference Eqs. and Applics.* **1** (1995), 57–60.]

28. Вначале заметим, что $\delta(U(z)V(z)) = (\delta U(z))V(z) + U(z)(\delta V(z))$, так как $t(mn) = t(m) + t(n)$. Кроме того, индукцией по n получаем $\delta(V(z)^n) = nV(z)^{n-1}\delta V(z)$ для всех $n \geq 0$. Это равенство необходимо, чтобы показать, что $\delta e^{V(z)} = \sum_{n\geq 0} \delta(V(z)^n/n!) = e^{V(z)}\delta V(z)$. Заменив в данном равенстве $V(z)$ на $\ln V(z)$, получаем $V(z)\,\delta \ln V(z) = \delta V(z)$, поэтому $\delta(V(z)^\alpha) = \delta e^{\alpha \ln V(z)} = e^{\alpha \ln V(z)}\delta(\alpha \ln V(z)) = \alpha V(z)^{\alpha-1}$ для всех комплексных чисел α.

Следовательно, требуемое рекуррентное соотношение имеет вид

(a) $W_1 = 1$, $W_n = \sum_{d\backslash n,\, d>1}((\alpha+1)t(d)/t(n) - 1)V_d W_{n/d}$;
(b) $W_1 = 1$, $W_n = \sum_{d\backslash n,\, d>1}(t(d)/t(n))V_d W_{n/d}$;
(c) $W_1 = 0$, $W_n = V_n + \sum_{d\backslash n,\, d>1}(t(d)/t(n) - 1)V_d W_{n/d}$.

[См. H. W. Gould, *AMM* **81** (1974), 3–14. Данная формула справедлива, когда t — любая функция, такая, что $t(m) + t(n) = t(mn)$ и $t(n) = 0$ тогда и только тогда, когда $n = 1$, но в предположении, что t не раскладывается на простые множители. Этот метод можно применять и для степенных рядов с произвольным множеством переменных. Тогда t — общая степень члена ряда.]

ТАБЛИЦЫ ЗНАЧЕНИЙ НЕКОТОРЫХ КОНСТАНТ

Таблица 1

ВЕЛИЧИНЫ, ЧАСТО ИСПОЛЬЗУЕМЫЕ В СТАНДАРТНЫХ ПОДПРОГРАММАХ И ПРИ АНАЛИЗЕ КОМПЬЮТЕРНЫХ ПРОГРАММ (40 ДЕСЯТИЧНЫХ ЗНАКОВ)

$\sqrt{2} = 1.41421\ 35623\ 73095\ 04880\ 16887\ 24209\ 69807\ 85697-$
$\sqrt{3} = 1.73205\ 08075\ 68877\ 29352\ 74463\ 41505\ 87236\ 69428+$
$\sqrt{5} = 2.23606\ 79774\ 99789\ 69640\ 91736\ 68731\ 27623\ 54406+$
$\sqrt{10} = 3.16227\ 76601\ 68379\ 33199\ 88935\ 44432\ 71853\ 37196-$
$\sqrt[3]{2} = 1.25992\ 10498\ 94873\ 16476\ 72106\ 07278\ 22835\ 05703-$
$\sqrt[3]{3} = 1.44224\ 95703\ 07408\ 38232\ 16383\ 10780\ 10958\ 83919-$
$\sqrt[4]{2} = 1.18920\ 71150\ 02721\ 06671\ 74999\ 70560\ 47591\ 52930-$
$\ln 2 = 0.69314\ 71805\ 59945\ 30941\ 72321\ 21458\ 17656\ 80755+$
$\ln 3 = 1.09861\ 22886\ 68109\ 69139\ 52452\ 36922\ 52570\ 46475-$
$\ln 10 = 2.30258\ 50929\ 94045\ 68401\ 79914\ 54684\ 36420\ 76011+$
$1/\ln 2 = 1.44269\ 50408\ 88963\ 40735\ 99246\ 81001\ 89213\ 74266+$
$1/\ln 10 = 0.43429\ 44819\ 03251\ 82765\ 11289\ 18916\ 60508\ 22944-$
$\pi = 3.14159\ 26535\ 89793\ 23846\ 26433\ 83279\ 50288\ 41972-$
$1^\circ = \pi/180 = 0.01745\ 32925\ 19943\ 29576\ 92369\ 07684\ 88612\ 71344+$
$1/\pi = 0.31830\ 98861\ 83790\ 67153\ 77675\ 26745\ 02872\ 40689+$
$\pi^2 = 9.86960\ 44010\ 89358\ 61883\ 44909\ 99876\ 15113\ 53137-$
$\sqrt{\pi} = \Gamma(1/2) = 1.77245\ 38509\ 05516\ 02729\ 81674\ 83341\ 14518\ 27975+$
$\Gamma(1/3) = 2.67893\ 85347\ 07747\ 63365\ 56929\ 40974\ 67764\ 41287-$
$\Gamma(2/3) = 1.35411\ 79394\ 26400\ 41694\ 52880\ 28154\ 51378\ 55193+$
$e = 2.71828\ 18284\ 59045\ 23536\ 02874\ 71352\ 66249\ 77572+$
$1/e = 0.36787\ 94411\ 71442\ 32159\ 55237\ 70161\ 46086\ 74458+$
$e^2 = 7.38905\ 60989\ 30650\ 22723\ 04274\ 60575\ 00781\ 31803+$
$\gamma = 0.57721\ 56649\ 01532\ 86060\ 65120\ 90082\ 40243\ 10422-$
$\ln \pi = 1.14472\ 98858\ 49400\ 17414\ 34273\ 51353\ 05871\ 16473-$
$\phi = 1.61803\ 39887\ 49894\ 84820\ 45868\ 34365\ 63811\ 77203+$
$e^\gamma = 1.78107\ 24179\ 90197\ 98523\ 65041\ 03107\ 17954\ 91696+$
$e^{\pi/4} = 2.19328\ 00507\ 38015\ 45655\ 97696\ 59278\ 73822\ 34616+$
$\sin 1 = 0.84147\ 09848\ 07896\ 50665\ 25023\ 21630\ 29899\ 96226-$
$\cos 1 = 0.54030\ 23058\ 68139\ 71740\ 09366\ 07442\ 97660\ 37323+$
$-\zeta'(2) = 0.93754\ 82543\ 15843\ 75370\ 25740\ 94567\ 86497\ 78979-$
$\zeta(3) = 1.20205\ 69031\ 59594\ 28539\ 97381\ 61511\ 44999\ 07650-$
$\ln \phi = 0.48121\ 18250\ 59603\ 44749\ 77589\ 13424\ 36842\ 31352-$
$1/\ln \phi = 2.07808\ 69212\ 35027\ 53760\ 13226\ 06117\ 79576\ 77422-$
$-\ln \ln 2 = 0.36651\ 29205\ 81664\ 32701\ 24391\ 58232\ 66946\ 94543-$

Таблица 2

ВЕЛИЧИНЫ, ЧАСТО ИСПОЛЬЗУЕМЫЕ В СТАНДАРТНЫХ ПОДПРОГРАММАХ И ПРИ АНАЛИЗЕ КОМПЬЮТЕРНЫХ ПРОГРАММ (45 ВОСЬМЕРИЧНЫХ ЗНАКОВ)

Величины, расположенные слева от знака "=", заданы в десятичной системе счисления

$0.1 =$	*0.06314 63146 31463 14631 46314 63146 31463 14631 46315−*
$0.01 =$	*0.00507 53412 17270 24365 60507 53412 17270 24365 60510−*
$0.001 =$	*0.00040 61115 64570 65176 76355 44264 16254 02030 44672+*
$0.0001 =$	*0.00003 21556 13530 70414 54512 75170 33021 15002 35223−*
$0.00001 =$	*0.00000 24761 32610 70664 36041 06077 17401 56063 34417−*
$0.000001 =$	*0.00000 02061 57364 05536 66151 55323 07746 44470 26033+*
$0.0000001 =$	*0.00000 00153 27745 15274 53644 12741 72312 20354 02151+*
$0.00000001 =$	*0.00000 00012 57143 56106 04303 47374 77341 01512 63327+*
$0.000000001 =$	*0.00000 00001 04560 27640 46655 12262 71426 40124 21742+*
$0.0000000001 =$	*0.00000 00000 06676 33766 35367 55653 37265 34642 01627−*
$\sqrt{2} =$	*1.32404 74631 77167 46220 42627 66115 46725 12575 17435+*
$\sqrt{3} =$	*1.56663 65641 30231 25163 54453 50265 60361 34073 42223−*
$\sqrt{5} =$	*2.17067 36334 57722 47602 57471 63003 00563 55620 32021−*
$\sqrt{10} =$	*3.12305 40726 64555 22444 02242 57101 41466 33775 22532+*
$\sqrt[3]{2} =$	*1.20505 05746 15345 05342 10756 65334 25574 22415 03024+*
$\sqrt[3]{3} =$	*1.34233 50444 22175 73134 67363 76133 05334 31147 60121−*
$\sqrt[4]{2} =$	*1.14067 74050 61556 12455 72152 64430 60271 02755 73136+*
$\ln 2 =$	*0.54271 02775 75071 73632 57117 07316 30007 71366 53640+*
$\ln 3 =$	*1.06237 24752 55006 05227 32440 63065 25012 35574 55337+*
$\ln 10 =$	*2.23273 06735 52524 25405 56512 66542 56026 46050 50705+*
$1/\ln 2 =$	*1.34252 16624 53405 77027 35750 37766 40644 35175 04353+*
$1/\ln 10 =$	*0.33626 75425 11562 41614 52325 33525 27655 14756 06220−*
$\pi =$	*3.11037 55242 10264 30215 14230 63050 56006 70163 21122+*
$1^\circ = \pi/180 =$	*0.01073 72152 11224 72344 25603 54276 63351 22056 11544+*
$1/\pi =$	*0.24276 30155 62344 20251 23760 47257 50765 15156 70067−*
$\pi^2 =$	*11.67517 14467 62135 71322 25561 15466 30021 40654 34103−*
$\sqrt{\pi} = \Gamma(1/2) =$	*1.61337 61106 64736 65247 47035 40510 15273 34470 17762−*
$\Gamma(1/3) =$	*2.53347 35234 51013 61316 73106 47644 54653 00106 66046−*
$\Gamma(2/3) =$	*1.26523 57112 14154 74312 54572 37655 60126 23231 02452+*
$e =$	*2.55760 52130 50535 51246 52773 42542 00471 72363 61661+*
$1/e =$	*0.27426 53066 13167 46761 52726 75436 02440 52371 03355+*
$e^2 =$	*7.30714 45615 23355 33460 63507 35040 32664 25356 50217+*
$\gamma =$	*0.44742 14770 67666 06172 23215 74376 01002 51313 25521−*
$\ln \pi =$	*1.11206 40443 47503 36413 65374 52661 52410 37511 46057+*
$\phi =$	*1.47433 57156 27751 23701 27634 71401 40271 66710 15010+*
$e^\gamma =$	*1.61772 13452 61152 65761 22477 36553 53327 17554 21260+*
$e^{\pi/4} =$	*2.14275 31512 16162 52370 35530 11342 53525 44307 02171−*
$\sin 1 =$	*0.65665 24436 04414 73402 03067 23644 11612 07474 14505−*
$\cos 1 =$	*0.42450 50037 32406 42711 07022 14666 27320 70675 12321+*
$-\zeta'(2) =$	*0.74001 45144 53253 42362 42107 23350 50074 46100 27706+*
$\zeta(3) =$	*1.14735 00023 60014 20470 15613 42561 31715 10177 06614+*
$\ln \phi =$	*0.36630 26256 61213 01145 13700 41004 52264 30700 40646+*
$1/\ln \phi =$	*2.04776 60111 17144 41512 11436 16575 00355 43630 40651+*
$-\ln \ln 2 =$	*0.27351 71233 67265 63650 17401 56637 26334 31455 57005−*

Значения некоторых констант с 40 знаками в табл. 1 вычислены на настольном калькуляторе Джоном У. Ренчем (мл.) (John W. Wrench, Jr.) для первого издания этой книги. Когда в 70-х годах новое программное обеспечение позволило вычислить их на компьютере, оказалось, что значения, полученные Д. Ренчем, правильны. Значения других фундаментальных констант с 40 знаками приведены в формулах 4.5.2–(60), 4.5.3–(26), 4.5.3–(41), 4.5.4–(9) и ответах к упр. 4.5.4–8, 4.5.4–25 и 4.6.4–58.

Таблица 3

ЗНАЧЕНИЯ ГАРМОНИЧЕСКИХ ЧИСЕЛ, ЧИСЕЛ БЕРНУЛЛИ И ЧИСЕЛ ФИБОНАЧЧИ ДЛЯ МАЛЫХ ЗНАЧЕНИЙ n

n	H_n	B_n	F_n	n
0	0	1	0	0
1	1	$-1/2$	1	1
2	3/2	1/6	1	2
3	11/6	0	2	3
4	25/12	$-1/30$	3	4
5	137/60	0	5	5
6	49/20	1/42	8	6
7	363/140	0	13	7
8	761/280	$-1/30$	21	8
9	7129/2520	0	34	9
10	7381/2520	5/66	55	10
11	83711/27720	0	89	11
12	86021/27720	$-691/2730$	144	12
13	1145993/360360	0	233	13
14	1171733/360360	7/6	377	14
15	1195757/360360	0	610	15
16	2436559/720720	$-3617/510$	987	16
17	42142223/12252240	0	1597	17
18	14274301/4084080	43867/798	2584	18
19	275295799/77597520	0	4181	19
20	55835135/15519504	$-174611/330$	6765	20
21	18858053/5173168	0	10946	21
22	19093197/5173168	854513/138	17711	22
23	444316699/118982864	0	28657	23
24	1347822955/356948592	$-236364091/2730$	46368	24
25	34052522467/8923714800	0	75025	25
26	34395742267/8923714800	8553103/6	121393	26
27	312536252003/80313433200	0	196418	27
28	315404588903/80313433200	$-23749461029/870$	317811	28
29	9227046511387/2329089562800	0	514229	29
30	9304682830147/2329089562800	8615841276005/14322	832040	30

Пусть для любого x $H_x = \sum_{n\ge1}\left(\frac{1}{n} - \frac{1}{n+x}\right)$. Тогда

$$
\begin{aligned}
H_{1/2} &= 2 - 2\ln 2,\\
H_{1/3} &= 3 - \tfrac{1}{2}\pi/\sqrt{3} - \tfrac{3}{2}\ln 3,\\
H_{2/3} &= \tfrac{3}{2} + \tfrac{1}{2}\pi/\sqrt{3} - \tfrac{3}{2}\ln 3,\\
H_{1/4} &= 4 - \tfrac{1}{2}\pi - 3\ln 2,\\
H_{3/4} &= \tfrac{4}{3} + \tfrac{1}{2}\pi - 3\ln 2,\\
H_{1/5} &= 5 - \tfrac{1}{2}\pi\phi^{3/2}5^{-1/4} - \tfrac{5}{4}\ln 5 - \tfrac{1}{2}\sqrt{5}\ln\phi,\\
H_{2/5} &= \tfrac{5}{2} - \tfrac{1}{2}\pi\phi^{-3/2}5^{-1/4} - \tfrac{5}{4}\ln 5 + \tfrac{1}{2}\sqrt{5}\ln\phi,\\
H_{3/5} &= \tfrac{5}{3} + \tfrac{1}{2}\pi\phi^{-3/2}5^{-1/4} - \tfrac{5}{4}\ln 5 + \tfrac{1}{2}\sqrt{5}\ln\phi,\\
H_{4/5} &= \tfrac{5}{4} + \tfrac{1}{2}\pi\phi^{3/2}5^{-1/4} - \tfrac{5}{4}\ln 5 - \tfrac{1}{2}\sqrt{5}\ln\phi,\\
H_{1/6} &= 6 - \tfrac{1}{2}\pi\sqrt{3} - 2\ln 2 - \tfrac{3}{2}\ln 3,\\
H_{5/6} &= \tfrac{6}{5} + \tfrac{1}{2}\pi\sqrt{3} - 2\ln 2 - \tfrac{3}{2}\ln 3
\end{aligned}
$$

и в общем случае, когда $0 < p < q$ (см. упр. 1.2.9–19),

$$H_{p/q} = \frac{q}{p} - \frac{\pi}{2}\cot\frac{p}{q}\pi - \ln 2q + 2\sum_{1\le n<q/2}\cos\frac{2pn}{q}\pi\cdot\ln\sin\frac{n}{q}\pi.$$

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

Буквы в формулах, если не оговорено дополнительно, имеют следующий смысл.

j, k Арифметическое выражение, значением которого является целое число

m, n Арифметическое выражение, значением которого, является неотрицательное целое число

x, y Арифметическое выражение, принимающее действительное значение

z Арифметическое выражение, принимающее комплексное значение

f Функция, принимающая действительное или комплексное значение

S, T Множество или мультимножество

Обозначение	Значение	Раздел
$\blacksquare$	Конец алгоритма, программы или доказательства	1.1
A_n или $A[n]$	n-й элемент линейного множества A	1.1
A_{mn} или $A[m, n]$	Элемент, стоящий в строке m и столбце n прямоугольной таблицы (матрицы) A	1.1
$V \leftarrow E$	Присвоить переменной V значение выражения E	1.1
$U \leftrightarrow V$	Значения переменных U и V поменять местами	1.1
$(B \Longrightarrow E;\ E')$	Условное выражение: означает E, если B истинно, и E', если B ложно	
$[B]$	Характеристическая функция условия B: $(B \Longrightarrow 1;\ 0)$	1.2.3
δ_{kj}	Символ Кронекера: $[j = k]$	1.2.3
$[z^n]\, g(z)$	Коэффициент при z^n в степенном ряду $g(z)$	1.2.9
$\sum_{R(k)} f(k)$	Сумма всех $f(k)$, таких, что значение k — целое и выполняется соотношение $R(k)$	1.2.3
$\prod_{R(k)} f(k)$	Произведение всех $f(k)$, таких, что значение k — целое и выполняется соотношение $R(k)$	1.2.3
$\min_{R(k)} f(k)$	Минимальное значение из всех $f(k)$, таких, что значение k — целое и выполняется соотношение $R(k)$	1.2.3
$\max_{R(k)} f(k)$	Максимальное значение из всех $f(k)$, таких, что значение k — целое и выполняется соотношение $R(k)$	1.2.3

Обозначение	Значение	Раздел
$\Re z$	Действительная часть z	1.2.2
$\Im z$	Мнимая часть z	1.2.2
$\overline{z}$	Комплексное число, сопряженное к z: $\Re z - i\,\Im z$	1.2.2
A^T	Транспонированная прямоугольная таблица (матрица) A: $A^T[j,k] = A[k,j]$	1.2.3
x^y	x в степени y (когда x — положительное число)	1.2.2
x^k	x в степени k: $\left(k \geq 0 \Rightarrow \prod_{0 \leq j < k} x;\quad 1/x^{-k}\right)$	1.2.2
$x^{\bar{k}}$	$\Gamma(x+k)/\Gamma(x) = \left(k \geq 0 \Rightarrow \prod_{0 \leq j < k} (x+j);\quad 1/(x+k)^{\overline{-k}}\right)$	1.2.5
$x^{\underline{k}}$	$x!/(x-k)! = \left(k \geq 0 \Rightarrow \prod_{0 \leq j < k} (x-j);\quad 1/(x-k)^{\underline{-k}}\right)$	1.2.5
$n!$	n факториал: $\Gamma(n+1) = n^{\underline{n}}$	1.2.5
$f'(x)$	Производная от f по x	1.2.9
$f''(x)$	Вторая производная от f по x	1.2.10
$f^{(n)}(x)$	n-я производная от f по x: $\big(n = 0 \Rightarrow f(x);\ g'(x)\big)$, где $g(x) = f^{(n-1)}(x)$	1.2.11.2
$f^{[n]}(x)$	n-я итерация: $\big(n = 0 \Rightarrow x; f(f^{[n-1]}(x))\big)$	4.7
$f^{\{n\}}(x)$	n-я индуцированная функция: $f^{\{n\}}(x) = f\big(x f^{\{n\}}(x)^n\big)$	4.7
$H_n^{\dot{(x)}}$	Гармоническое число порядка x: $\sum_{1 \leq k \leq n} 1/k^x$	1.2.7
H_n	Гармоническое число: $H_n^{(1)}$	1.2.7
F_n	Число Фибоначчи: $(n \leq 1 \Rightarrow n;\ F_{n-1} + F_{n-2})$	1.2.8
B_n	Число Бернулли: $n!\,[z^n]\,z/(e^z - 1)$	1.2.11.2
$X \cdot Y$	Скалярное произведение векторов $X = (x_1, \ldots, x_n)$ и $Y = (y_1, \ldots, y_n)$: $x_1y_1 + \cdots + x_ny_n$	3.3.4
$j \backslash k$	j делит k: $k \bmod j = 0$ и $j > 0$	1.2.4
$S \setminus T$	Разность множеств: $\{a \mid a$ принадлежит S и a не принадлежит $T\}$	
$\oplus\ \ominus\ \otimes\ \oslash$	Округление или специальные операции	4.2.1
$(\ldots a_1 a_0 . a_{-1} \ldots)_b$	Представление числа в позиционной системе счисления с основанием b: $\sum_k a_k b^k$	4.1

Обозначение	Значение	Раздел
$//x_1, x_2, \ldots, x_n//$	Цепная дробь: $1/\bigl(x_1 + 1/(x_2 + 1/(\cdots + 1/(x_n)\ldots))\bigr)$	4.5.3
$\binom{x}{k}$	Биномиальный коэффициент: $(k < 0 \Rightarrow 0;\ x^{\underline{k}}/k!)$	1.2.6
$\binom{n}{n_1, n_2, \ldots, n_m}$	Полиномиальный коэффициент (определен только тогда, когда $n = n_1 + n_2 + \cdots + n_m$)	1.2.6
$\left[{n \atop m}\right]$	Число Стирлинга первого рода: $\sum_{0<k_1<k_2<\cdots<k_{n-m}<n} k_1 k_2 \ldots k_{n-m}$	1.2.6
$\left\{{n \atop m}\right\}$	Число Стирлинга второго рода: $\sum_{1\le k_1\le k_2\le\cdots\le k_{n-m}\le m} k_1 k_2 \ldots k_{n-m}$	1.2.6
$\{a \mid R(a)\}$	Множество всех a, таких, что выполняется соотношение $R(a)$	
$\{a_1, \ldots, a_n\}$	Множество или мультимножество $\{a_k \mid 1 \le k \le n\}$	
$\{x\}$	Дробная часть: $x - \lfloor x \rfloor$ (используется, когда x — действительное число, а не множество)	1.2.11.2
$[a\,..\,b]$	Замкнутый интервал: $\{x \mid a \le x \le b\}$	1.2.2
$(a\,..\,b)$	Открытый интервал: $\{x \mid a < x < b\}$	1.2.2
$[a\,..\,b)$	Полузамкнутый интервал: $\{x \mid a \le x < b\}$	1.2.2
$(a\,..\,b]$	Полуоткрытый интервал: $\{x \mid a < x \le b\}$	1.2.2
$\lvert S\rvert$	Число элементов множества S	
$\lvert x\rvert$	Абсолютная величина x: $(x \ge 0 \Rightarrow x;\ -x)$	
$\lvert z\rvert$	Абсолютная величина z: $\sqrt{z\bar{z}}$	1.2.2
$\lfloor x \rfloor$	Наибольшее целое число, не превосходящее x: $\max_{k\le x} k$	1.2.4
$\lceil x \rceil$	Наименьшее целое число $> x$: $\min_{k\ge x} k$	1.2.4
$((x))$	Пилообразная функция	3.3.3
$\langle X_n \rangle$	Бесконечная последовательность X_0, X_1, X_2, ... (здесь n — часть обозначения)	1.2.9
γ	Константа Эйлера: $\lim_{n\to\infty}(H_n - \ln n)$	1.2.7
$\gamma(x, y)$	Неполная гамма-функция: $\int_0^y e^{-t} t^{x-1}\,dt$	1.2.11.3
$\Gamma(x)$	Гамма-функция: $(x-1)! = \gamma(x, \infty)$	1.2.5
$\delta(x)$	Характеристическая функция целых чисел	3.3.3
e	Основание натурального логарифма: $\sum_{n\ge 0} 1/n!$	1.2.2
$\zeta(x)$	Дзета-функция: $\lim_{n\to\infty} H_n^{(x)}$ (где $x > 1$)	1.2.7

Обозначение	Значение	Раздел
$K_n(x_1, \ldots, x_n)$	Континуант	4.5.3
$\ell(u)$	Старший коэффициент полинома u	4.6
$l(n)$	Наименьшая длина аддитивной цепочки для n	4.6.3
$\Lambda(n)$	Функция фон Мангольдта	4.5.3
$\mu(n)$	Функция Мёбиуса	4.5.2
$\nu(n)$	Количество единиц в двоичном представлении числа	4.6.3
$O\big(f(n)\big)$	О большое от $f(n)$ при $n \to \infty$	1.2.11.1
$O\big(f(z)\big)$	О большое от $f(z)$ при $z \to 0$	1.2.11.1
$\Omega\big(f(n)\big)$	Омега большое от $f(n)$ при $n \to \infty$	1.2.11.1
$\Theta\big(f(n)\big)$	Тета большое от $f(n)$ при $n \to \infty$	1.2.11.1
$\pi(x)$	Количество простых чисел: $\sum_{n \le x}[n \text{ простое число}]$	4.5.4
π	Отношение длины окружности к ее диаметру: $4\sum_{n \ge 0}(-1)^n/(2n+1)$	4.3.1
ϕ	Золотое сечение: $\frac{1}{2}\big(1+\sqrt{5}\big)$	1.2.8
$\emptyset$	Пустое множество: $\{x \mid 0 = 1\}$	
$\varphi(n)$	Функция Эйлера: $\sum_{0 \le k < n}[k \perp n]$	1.2.4
∞	Бесконечность: больше любого числа	4.2.2
$\det(A)$	Определитель квадратной матрицы A	1.2.3
$\operatorname{sign}(x)$	Знак x: $\big(x = 0 \Longrightarrow 0;\ x/\lvert x\rvert\big)$	
$\deg(u)$	Степень полинома u	4.6
$\operatorname{cont}(u)$	Содержание полинома u	4.6.1
$\operatorname{pp}\big(u(x)\big)$	Примитивная часть полинома u	4.6.1
$\log_b x$	Логарифм x по основанию b: y такое, что $x = b^y$ (когда $x > 0$, $b > 0$ и $b \ne 1$)	1.2.2
$\ln x$	Натуральный логарифм: $\log_e x$	1.2.2
$\lg x$	Логарифм по основанию 2: $\log_2 x$	1.2.2
$\exp x$	Показательная функция от x: e^x	1.2.2
$j \perp k$	j взаимно простое с k: $\gcd(j,k) = 1$	1.2.4
$\gcd(j,k)$	Наибольший общий делитель j и k: $\Big(j = k = 0 \Longrightarrow 0;\ \max\limits_{d\backslash j,\, d\backslash k} d\Big)$	4.5.2
$\operatorname{lcm}(j,k)$	Наименьшее общее кратное j и k: $\Big(jk = 0 \Longrightarrow 0;\ \min\limits_{d>0,\, j\backslash d,\, k\backslash d} d\Big)$	4.5.2

Обозначение	Значение	Раздел
$x \bmod y$	x по модулю y: $(y = 0 \Rightarrow x;\ x - y\lfloor x/y \rfloor)$	1.2.4
$u(x) \bmod v(x)$	Остаток от деления полинома u на полином v	4.6.1
$x \equiv x' (\text{по модулю } y)$	Сравнимость (конгруэнтность) по модулю y: $x \bmod y = x' \bmod y$	1.2.4
$x \approx y$	x приближенно равно y	3.5, 4.2.2
$\Pr(S(n))$	Вероятность того, что утверждение $S(n)$ справедливо для случайных положительных целых чисел n	3.5
$\Pr(S(X))$	Вероятность того, что утверждение $S(X)$ справедливо для случайных величин X	1.2.10
$\mathrm{E}\,X$	Математическое ожидание (среднее значение) случайной величины X: $\sum_x x \Pr(X = x)$	1.2.10
$\mathrm{mean}(g)$	Среднее значение распределения вероятностей, которое задано производящей функцией g: $g'(1)$	1.2.10
$\mathrm{var}(g)$	Дисперсия распределения вероятностей, которое задано производящей функцией g: $g''(1) + g'(1) - g'(1)^2$	1.2.10
$(\min x_1, \mathrm{ave}\ x_2, \max x_3, \mathrm{dev}\ x_4)$	Случайная величина с минимальным значением x_1, средним значением (математическим ожиданием) x_2, максимальным значением x_3, среднеквадратичным отклонением x_4	1.2.10
␣	Один пробел	1.3.1
rA	Регистр A (сумматор) компьютера MIX	1.3.1
rX	Регистр X (расширение) компьютера MIX	1.3.1
rI1, ..., rI6	Индексные регистры I1, ..., I6 компьютера MIX	1.3.1
rJ	Регистр перехода J компьютера MIX	1.3.1
(L:R)	Частичное поле слова компьютера MIX, $0 \le \mathtt{L} \le \mathtt{R} \le 5$	1.3.1
OP ADDRESS,I(F)	Обозначение команды компьютера MIX	1.3.1, 1.3.2
u	Единица времени компьютера MIX	1.3.1
*	"Сам" ("self") в языке MIXAL	1.3.2
0F, 1F, 2F, ..., 9F	"Вперед" ("forward") — локальный символ в языке MIXAL	1.3.2
0B, 1B, 2B, ..., 9B	"Назад" ("backward") — локальный символ в языке MIXAL	1.3.2
0H, 1H, 2H, ..., 9H	"Здесь" ("here") — локальный символ в языке MIXAL	1.3.2