Беккель Л. С.
Кушнир Д. В.
Шемякин С. Н.

Математические основы информационной безопасности

Учебно-методическое пособие

СПб ГУТ)))

Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования «Санкт-Петербургский государственный университет
телекоммуникаций им. проф. М. А. Бонч-Бруевича»
(СПбГУТ)

Л. С. Беккель, Д. В. Кушнир, С. Н. Шемякин

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Учебно-методическое пособие

Электронное издание
локального распространения

Санкт-Петербург
Наукоемкие технологии
2026

© Беккель Л. С., Кушнир Д. В., Шемякин С. Н., 2026
© Санкт-Петербургский государственный университет
телекоммуникаций им. проф. М. А. Бонч-Бруевича, 2026
ISBN 978-5-00271-169-7

УДК 004.056:519.7(075.8)
ББК 16.8:22.18я73
Б42

Рецензент:
А. Ю. Матюхин, кандидат технических наук,
доцент кафедры сетей связи и передачи данных

Б42 **Беккель, Л. С.**
Математические основы информационной безопасности [Электронный ресурс]: учебно-методическое пособие / Л. С. Беккель, Д. В. Кушнир, С. Н. Шемякин; СПбГУТ. – Электрон, текстовые дан. (3,0 Мб). – СПб.: Наукоемкие технологии, 2026. – 102 с. – 1 электрон., опт. диск (CD-ROM).

ISBN 978-5-00271-169-7

Рассматриваются некоторые главы теории чисел, которые наиболее тесно связаны с дальнейшими курсами в области информационной безопасности. Вводятся основные понятия и инструменты, необходимые для понимания способов построения и свойств криптографических преобразований. При изложении материала учитывается специфика преподавания вопросов информационной безопасности в ВУЗе и сложившееся разделение изучения отдельных тем между рядом предметов, вводящих те или иные криптографические методы защиты информации.

Предназначается для студентов, обучающихся по направлениям подготовки 10.03.01 «Информационная безопасность», 10.05.02 «Информационная безопасность телекоммуникационных систем».

Утверждено редакционно-издательским советом СПбГУТ
в качестве учебно-методического пособия

Текстовое электронное издание
Минимальные системные требования:

- процессор: Intel x86, x64, AMD x86, x64 не менее 1 ГГц;
- оперативная память RAM ОЗУ: не менее 512 МБайт;
- свободное место на жестком диске (HDD): не менее 120 МБайт;
- операционная система: Windows XP и выше;
- Adobe Acrobat Reader;
- дисковод CD-ROM;
- мышь.

УДК 004.056:519.7(075.8)
ББК 16.8:22.18я73

© Беккель Л. С., Кушнир Д. В., Шемякин С. Н., 2026
© Санкт-Петербургский государственный университет телекоммуникаций им. проф. М. А. Бонч-Бруевича, 2026

ISBN 978-5-00271-169-7

Учебное издание

Беккель Людмила Сергеевна
Кушнир Дмитрий Викторович
Шемякин Сергей Николаевич

Математические основы информационной безопасности

Учебно-методическое пособие

Электронное издание
локального распространения

Издается в авторской редакции
Публикуется из оригинал-макета, подготовленного авторами

Издательство «Наукоемкие технологии»
ООО «Корпорация «Интел Групп»
https://publishing.intelgr.com
E-mail: publishing@intelgr.com
Тел.: +7 (812) 945-50-63
Интернет-магазин издательства
https://shop.intelgr.com/

Подписано к использованию 19.08.2026 г.
Объем издания – 3,0 Мб.
Комплектация издания – 1 CD.
Тираж 100 CD.

ISBN 978-5-00271-169-7



СОДЕРЖАНИЕ

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ 6
ПРЕДИСЛОВИЕ 7
1. СЛОЖНОСТЬ АЛГОРИТМОВ И ЗАДАЧ 7
1.1. Введение 7
1.2. Сведения из теории алгоритмов 10
1.3. О понятии алгоритма 10
1.4. Примеры задач и алгоритмов 11
1.5. Машина с произвольным доступом 13
1.6. Меры сложности алгоритмов 14
1.7. Полиномиальные алгоритмы 16
1.8. Полиномиальность и эффективность 17
1.9. Сложность и полиномиальность в среднем 17
1.10. Оценки функции сложности 19
1.11. Сложность арифметических операций с целыми числами 20
2. АРИФМЕТИКА ЦЕЛЫХ ЧИСЕЛ 22
2.1. Свойства чисел 22
2.2. Алгоритм Евклида 27
2.3. Расширенный алгоритм Евклида 30
2.4. Функция Эйлера 32
2.5. Непрерывные и подходящие дроби 33
2.6. Арифметика остатков 38
2.7. Модульная арифметика. Инверсии 44
3. КИТАЙСКАЯ ТЕОРЕМА ОБ ОСТАТКАХ 47
4. РЕШЕНИЕ СРАВНЕНИЙ 51
4.1. Линейные сравнения с одним неизвестным 51
4.2. Сравнения степени n 54
4.3. Символы Лежандра и Якоби 55
4.4. Извлечение квадратного корня 58
4.4.1. Извлечение квадратного корня по простому модулю 58
4.4.2. Извлечение квадратного корня по составному модулю 60
5. ПЕРВООБРАЗНЫЕ КОРНИ И ИНДЕКСЫ 62
5.1. Первообразный корень 62

5.2. Индексы и логарифмы 64
5.3. Применение индексов к решению сравнений 67
6. ЛИНЕЙНЫЕ ДИОФАНТОВЫ УРАВНЕНИЯ 68
6.1. Общие сведения о диофантовых уравнениях 68
6.2. Диофантовы уравнения первой степени с двумя переменными 70
7. РАЗЛОЖЕНИЕ НА МНОЖИТЕЛИ 73
7.1. Алгоритм разложения на множители методом пробных делений 74
7.2. Алгоритм разложения на множители Ферма 74
7.3. Другие алгоритмы разложения на множители 75
7.4. Контрольные вопросы 75
8. ПРОСТЫЕ ЧИСЛА 76
8.1. Простые и составные числа. 76
8.2. О распределении простых чисел 76
8.3. Критерии простоты 77
8.4. Алгоритмы проверки простоты 79
8.5. Тест Ферма. 80
8.6. Тест Соловея-Штрассена 83
8.7. Тест Миллера-Рабина 84
8.8. Сертификат простоты числа 84
8.9. Построение больших простых чисел 85
8.9.1. Общие подходы. 85
8.9.2. Критерий Лукаса 86
8.9.3. Теорема Поклингтона 87
8.9.4. Теорема Диемитко 88
9. АЛГЕБРАИЧЕСКАЯ СТРУКТУРА 89
9.1. Группа, Кольцо, Поле 89
9.2. Расширенные поля 93
9.3. Задающие полиномы 94
9.4. Расширенный алгоритм Евклида для полиномов. 97
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 100
УКАЗАТЕЛЬ 101

ОСНОВНЫЕ ОБОЗНАЧЕНИЯ

$\mathbb{N}$	множество натуральных чисел;
$\mathbb{Z}$	кольцо целых чисел;
$\mathbb{Q}$	поле рациональных чисел;
$\mathbb{Z}_2 = \{0,1\}$	множество из двух чисел;
$\mathbb{Z}_k$	кольцо классов вычетов по модулю *k;*
$\mathbb{P}$	множество простых чисел;
$x \in \mathbb{Z}_2,$	переменная скалярная;
$\boldsymbol{x} \in \mathbb{Z}_2^n$	переменная вектор;
$GF\ (2)$	простое поле Галуа, результат сложения и умножения приводится по модулю 2;
$GF\ (2^n)$	расширение поля Галуа; линейное пространство над двоичным полем, размерности *n*;
$H(X)$	энтропия случайной величины *X*;
$H(X\|Z)$	условная энтропия случайной величины *X* относительно случайной величины *Z*;
$I(X,Z)$	взаимная информация *X* и *Z*;
$\oplus$	сложение по модулю два;
$\in$	принадлежит;
$\exists$	означает «существует» или «для некоторого»;
(a, b)	наибольший общий делитель (a, b) = НОД(a, b) ;
$[a, b]$	наименьшее общее кратное $[a, b]$ = НОК(a, b) ;
$a \mid b$	*a* делит *b*;
$a \nmid b$	*a* не делит *b*;
$\varphi(n)$	функция Эйлера от натурального числа *n*;
$\pi\ (x)$	функция равная количеству простых чисел меньших или равных *x*;
$\left(\frac{a}{p}\right)$	символ Лежандра для простого *p*;
$\left(\frac{a}{n}\right)$	символ Якоби для составного *n*;
■	завершение доказательства;

ПРЕДИСЛОВИЕ

Теория чисел начала широко применяться в криптографии примерно 50 лет назад. Это было вызвано необходимостью обмена большими массивами конфиденциальной информации (не только государственной и военной, но и банковской, экономической, медицинской, юридической и т.п.), а также возможностью такого обмена в связи с появлением доступных и эффективных компьютерных средств обработки этой информации.

Криптографические потребности стимулировали исследования в некоторых областях теории чисел, стали источником постановки новых фундаментальных проблем. Стойкость криптографических алгоритмов напрямую зависит от невозможности найти быстрые алгоритмы для решения некоторых задач, другими словами, от того, что некоторые задачи теории чисел сложны в вычислительном отношении.

1. СЛОЖНОСТЬ АЛГОРИТМОВ И ЗАДАЧ

1.1. Введение

Криптография делится на два направления: собственно шифры и криптографический анализ. Количество различных отечественных и зарубежных шифров достаточно велико. Поэтому разработчик информационной системы при выборе криптографической системы должен уметь составить собственное мнение о безопасности рассматриваемых шифров. При этом необходимо иметь представление об уровне стойкости криптосистемы в настоящий момент, каким он ожидается в будущем, чем именно обеспечивается стойкость шифра и основные уязвимости к заданным методам криптографического анализа.

Стойкость криптографических систем основывается на трудности решения некоторых задач. Перечислим некоторые из них:

- разложение большого числа на простые множители;
- определение индекса вычета по заданному основанию;
- решение линейного уравнения в группе точек эллиптической кривой;
- определение состояния конечного автомата по его выходной последовательности.

Первая задача имеет название задачи факторизации, две последующих – задачи дискретного логарифмирования.

Теорию, необходимую для решения задач, возникающих при проведении криптографического анализа, даёт дискретная (прикладная) математика. Особенностью этой области математики является ограниченность или

по крайней мере счётность множества значений переменных. Поэтому значительная часть дискретных и комбинаторных задач может быть решена при полном переборе значений всех переменных. Такой способ решения задач называют переборным.

Достоинством переборного алгоритма является универсальность, то есть теоретическая возможность решения практически всех задач дискретной математики. Недостатком переборного алгоритма является рост вычислительных расходов с ростом размера задачи, причём рост по экспоненте. В настоящий момент границей применимости переборного алгоритма является количество переменных, равное трёмстам. В то же время для некоторых задач дискретной математики известны существенно менее трудоёмкие алгоритмы решения.

После уточнения понятия алгоритма обнаружились алгоритмически неразрешимые задачи. Известным примером можно считать десятую проблему Гильберта: «Существует ли алгоритм, который по данному полиному $p(\boldsymbol{x})$ над кольцом целых чисел распознаёт существование решения уравнения $p(\boldsymbol{x}) = 0$ в целых числах». Как показал Ю. В. Матиясевич, такого алгоритма в принципе не существует.

Ограничимся задачами с ограниченным множеством значений переменных. Такие задачи можно назвать переборными. Для них имеется конечное число возможных вариантов значений переменных, проверка всех этих вариантов обязательно даст решение. В конечной области аналогом алгоритмической неразрешимости является перебор вариантов как единственный способ решения задачи, аналогом алгоритмической разрешимости – существование алгоритма существенно более экономного, чем перебор.

Общепринято считать переборную задачу решаемой эффективно, если существует алгоритм, решающий её за время, ограниченное полиномом от «размера записи условия задачи».

Рассматриваемый здесь раздел дискретной математики фактически есть теория полиномиальной сводимости переборных задач. Говорят, что переборная задача $П_1$ сводится к переборной задаче $П_2$, если метод решения задачи $П_2$ можно преобразовать в метод решения задачи $П_1$. Сводимость называется полиномиальной, если подобное преобразование можно выполнить за полиномиальное время. Основными объектами теории являются: класс $\mathcal{NP}$ всех переборных задач и класс $\mathcal{P}$ переборных задач, разрешимых за полиномиальное время на машине Тьюринга.

В классе $\mathcal{NP}$ выявлены $\mathcal{N}$-полные задачи, к которым полиномиально сводится любая задача из $\mathcal{NP}$. В настоящее время установлена $\mathcal{NP}$-полнота многих задач, эквивалентных между собой, то есть полиномиально сводимых друг к другу. Имеются три возможности:

1. Найдётся полиномиальный алгоритм решения хотя бы одной $\mathcal{N}$-полной задачи. Тогда $\mathcal{P} = \mathcal{NP}$, и можно надеяться на отыскание эффективных алгоритмов для различных классов дискретных задач.
2. $\mathcal{P} \neq \mathcal{NP}$, тогда любая $\mathcal{N}$-полная задача может быть разбита на некоторое множество частных полиномиальных подзадач и задачу, для которой не существует полиномиального алгоритма решения.
3. Возможно, что гипотезу $\mathcal{P} \neq \mathcal{NP}$ нельзя ни доказать, ни опровергнуть на существующем уровне развития теории алгоритмов.

Будем говорить, что функция $f\,(n)$ ограничена полиномом $g\,(n)$, если существует константа c, такая, что $|f(n)| \leq c|g(n)|$ для всех значений n, начиная с некоторого n_1. Обозначение этого факта: $f\,(n) = O\,(g\,(n))$.

Определение 1.1. *Полиномиальный алгоритм* решения некоторого класса задач или *алгоритм полиномиальной временной сложности* - алгоритм, у которого время решения задачи равно $O\,(p\,(n))$, где $p\,(n)$ – некоторый полином, n - объём записи задачи на входе алгоритма.

Алгоритмы, временная сложность которых не поддаётся подобной оценке, называют *экспоненциальными*. Класс задач, для которых неизвестен или не существует полиномиальный алгоритм решения, называют *трудноразрешимым*. Данное различение алгоритмов имеет смысл только для задач большой размерности. Для дискретных задач с малым числом переменных перебор их вариантов значений зачастую оказывается достаточным для решения за допустимое время.

Эквивалентность труднорешаемых задач по сложности определяется по сводимости за полиномиальное время. Если одна задача сводится некоторым алгоритмом за полиномиальное время к некоторой другой, то любой полиномиальный алгоритм решения второй задачи может быть преобразован в полиномиальный алгоритм решения первой. Можно сказать, в таком случае, что сложность первой задачи не больше сложности второй задачи.

Существуют переборные задачи, класс $\mathcal{NP}$, которые могут быть решены за полиномиальное время на недетерминированном вычислительном устройстве. Примером являются задачи распознавания свойств. (Это задачи, решениями которой являются значения логической переменной – ДА или НЕТ.) Большинство практических не поддающихся решению задач после переоформления их в виде задачи распознавания попадают в этот класс.

Определился и продолжает пополняться список $\mathcal{NP}$ задач со свойством сводимости к ним любой переборной задачи. Этот класс задач называют классом *$\mathcal{N}$-полных задач*.

1.2. Сведения из теории алгоритмов

Решение задач, возникающих во всех возможных видах государственной деятельности, потребовало подготовки кадров, способных справиться с ними. Описание способов решения вычислительных задач дало начало понятию и термину алгоритма.

Необходимость формализации и развития данного объекта математической теории возникла в разведцентрах при решении сложных задач криптографического анализа. Вскрытие шифров потребовало целесообразной организации работы нескольких тысяч расчётчиков, доступны же, как всегда, оказались только кадры с низкой квалификацией, которые способны выполнять только простые вычислительные операции.

Возникновение теории алгоритмов можно связать с исследованиями А. Чёрча (1936 г.), связанные с попыткой понять, что же можно вычислить с помощью функций. Чёрч уточнил понятие вычислимой функции и привёл примеры функций, не являющихся вычислимыми. *Вычислимая функция* есть функция, для которой существует алгоритм её вычисления. С этого момента начались исследования, уточняющие понятие и сущность алгоритма.

1.3. О понятии алгоритма

До середины XIX века всё многообразие вычислений укладывалось 10-15 операций арифметики, тригонометрии и анализа. Поэтому понятие метода вычисления было ясным и не нуждалось в специальных исследованиях.

Определение 1.2. *Алгоритм* (*алгорифм*) – точное предписание, которое задаёт вычислительный процесс, начинающийся с произвольного исходного данного и направленный на получение полностью определённого этим исходным данным результата.

С точки зрения современной практики, алгоритм – это программа, критерием алгоритмичности процесса является возможность его запрограммировать. Разработка алгоритмов, накопление их стало областью прикладной информатики. Сравнение алгоритмов потребовало разработки точного языка для описания и оценки параметров алгоритмов.

Оказалось, что понятие алгоритма, подобно понятиям множества и натурального числа, относится к фундаментальным, оно не может быть выражено через другие понятия. Приведём ещё определения, в какой-то степени его определяющие.

Определение 1.3. Алгоритм есть точное предписание, которое задаёт вычислительный процесс нахождения значения вычислимой функции по заданному значению её аргумента.

Определение 1.4. Алгоритм есть предписание, однозначно определяющее ход некоторых процессов.

Требования к алгоритмам.

1. Алгоритм применяется к *исходным данным* и выдаёт результат. Всегда существует *конструктивный объект*, к которому применяется алгоритм. В качестве конструктивных объектов выступают данные, символы, структуры.
2. Данные для своего размещения требуют наличия *памяти*.
3. Алгоритм состоит из *элементарных шагов* (действий), в литературе также используется термин *дискретность алгоритма*. Число выполняемых шагов является конечным.
4. Последовательность шагов *детерминирована*, после каждого шага указывается следующий шаг, либо алгоритм останавливается.
5. Алгоритм должен быть *результативным*, то есть должен выдавать результат после конечного числа шагов.
6. Сущность понятия алгоритм включает в себя:

– описание алгоритма вычислений или управления (текст программы);
– средства реализации алгоритма – вычислительная техника, средства запуска, остановки, управления;
– процесс реализации алгоритма (алгоритмический процесс).

Данные требования понятие алгоритма уточняют, но не более, поэтому при исследовании данного объекта математики используют следующие формально определённые средства реализации алгоритмов:

– машина Тьюринга;
– частично-рекурсивные схемы;
– машина Колмогорова;
– нормальные алгоритмы Маркова;
– канонические системы Поста и др.

Оказалось, что указанные модели полиномиально сводимы и дают эквивалентные оценки сложности алгоритмов.

1.4. Примеры задач и алгоритмов

Задача 1. Даны три натуральных числа x, n, m. Вычислить $y = x^n \bmod m$.
Тривиальный алгоритм вычисления степени в кольце классов вычетов: в цикле от 1 до n выполнить следующее действие: $y \leftarrow y \cdot x \bmod m$. Сложность алгоритма, то есть число элементарных операций, есть величина $O(n)$.
Бинарный алгоритм вычисления степени в кольце классов вычетов имеет сложность $O(\log n)$. Рассмотрим двоичное разложение числа n:

$$n = \sum_{i=1}^{k} a_i 2^i, \qquad a_i \in \{0, 1\}.$$

Получаем, что

$$x^n = \prod_{\{i:\, a_i = 1\}} x^{2^i}.$$

Достаточно провести $k = \log_2 n$ возведений в квадрат, получить $x^{2^i} \bmod m$, $i = 1 \ldots k$ и выбрать необходимые сомножители для вычисления искомой степени. Разница в вычислительных расходах для тривиального и для бинарного способа вычисления степени является экспоненциальной.

Процедура расчёта модульной экспоненты применяется в сетевых протоколах, например, при установке защищённого интернет-соединения. Всякий раз, когда в браузере адрес начинается с https://, работают криптографические алгоритмы, вычисляется степень в кольце классов вычетов. При длине показателя степени 128 бит получим следующее сравнение количества умножений в обоих алгоритмах:

Длина показателя: 56 бит.
Количество умножений: $2^{56} \approx 7{,}2 \cdot 10^{16}$ в тривиальном алгоритме.
Количество умножений: ≈ 100 в бинарном алгоритме.
Длина показателя: 128 бит.
Количество умножений: $2^{128} \approx 3{,}4 \cdot 10^{38}$ в тривиальном алгоритме.
Количество умножений: ≈ 130 в бинарном алгоритме.

Задача 2. Дискретный логарифм. Даны натуральные числа a, b и простое число p. Найти значение неизвестного x, такое, что

$$a^x = b \bmod p.$$

Экспоненциальный алгоритм решения этой задачи существует, это перебор значений x в интервале от 1 до p. В настоящее время вычисление дискретного логарифма является очень сложной задачей, самые быстрые из известных алгоритмов требуют сверх полиномиального времени. На этом важном свойстве, то есть свойстве *односторонней* эффективной вычислимости модульной экспоненты, основано много протоколов криптографии.

Задача 3. Факторизация большого числа, то есть разложение на простые множители. Пусть p и q – два простых числа. Вычисление их произведения

$n = p \cdot q$ – задача несложная. Но если известно значение n и требуется найти p и q, то это уже труднорешаемая задача, для которой неизвестны эффективные алгоритмы решения. На трудности задачи факторизации основана криптографическая система RSA.

1.5. Машина с произвольным доступом

Количественное определение меры эффективности алгоритма предполагает использование формально определённого средства его реализации – машины Тьюринга, рекурсивных функций, частично-рекурсивных схем или иного аппарата.

В настоящее время считается, что наиболее удобным способом сравнения сложности разнородных алгоритмов является понятие *асимптотической сложности*. Выбор вычислительной модели для реализации алгоритма не имеет особо важного значения, а для оценки сложности алгоритма число выполненных операторов (например, на языке Python) оказывается приемлемым. Основная причина такого подхода к выбору модели заключается в том, что существуют эффективные способы моделирования (или *трансляции программ*) формальных или технически реализованных вычислительных моделей с помощью других. При этом сохраняется класс эффективных алгоритмов: как правило, алгоритмы более эффективные в одной модели оказываются более эффективными и в других.

Приведём краткое описание только одной вычислительной модели – одноадресной машины с произвольным доступом к памяти (*random access machines*). RAM-машина программируется в терминах инструкций процессора. Из известных моделей она наиболее напоминает современный компьютер.

Состав RAM-машины:

1. Конечная входная read-only лента, на которой записаны исходные данные задачи.
2. Полубесконечная выходная write-only лента для записи результата работы машины.
3. Бесконечное число регистров $r_0, r_1, r_2, \ldots$, каждый из которых содержит адрес и команду с оператором (начальное состояние - нули). Регистр r_0 является выделенным и называется «сумматором» - этот регистр используется при арифметических операциях как накопитель, то есть второй операнд и место хранения результата.
4. Программа, состоящая из конечного числа инструкций, каждая из которых содержит адрес и команду с оператором. В качестве операнда используются регистры, возможна косвенная адресация, то есть обращение по адресу, записанному в регистре.
5. Регистр-счётчик «PC» - указатель текущей команды.

Машина последовательно выполняет команды, на которые указывает регистр «PC», читает входную ленту, изменяет значения регистров и записывает результат на выходную ленту. Высокоуровневые конструкции языков программирования можно смоделировать на ассемблере RAM-машины. Переменные других типов (булевы, строки, структуры) можно представить с помощью целых чисел и косвенной адресации.

Список команд RAM-машины

LOAD OP	$OP \rightarrow r_0$	Загрузить операнд в сумматор
STORE OP	$r_0 \rightarrow r_{op}$	Сохранить сумматор в регистре
ADD OP	$r_0 \rightarrow r_0 + OP$	Прибавить операнд к сумматору
SUB OP	$r_0 \rightarrow r_0 - OP$	Вычесть операнд из сумматора
READ OP	$input \rightarrow r_{op}$	Загрузить ячейку из входа в r_{op}, сдвиг к следующей
WRITE OP	$r_{op} \rightarrow output$	Записать *OP* в текущую ячейку выхода, сдвиг
JUMP OP	$OP \rightarrow PC$	Установить счётчик команд в *OP*
JGTZ OP	$OP \rightarrow PC \ / r_0 > 0$	Установить счётчик команд в *OP,* если $r_0 > 0$
JZERO OP	$OP \rightarrow PC / r_0 = 0$	Установить счётчик команд в *OP,* если $r_0 = 0$
HALT		Останов

Операнд OP: целое число, регистр, указатель.

RAM-машина идеальна, то есть реализована быть не может, так как она работает с бесконечной памятью, с мгновенным доступом к памяти, неограниченным размером используемых целых чисел. Однако данная модель даёт возможность формально определить сложность алгоритма как число исполняемых инструкций до остановки машины.

Следует отметить, что частные случаи модели RAM-машины описывают неветвящиеся программы, булевы схемы, сама машина является основой модели, которая используется в теории параллельных вычислений.

1.6. Меры сложности алгоритмов

Однородная мера сложности алгоритма (*unit-cost RAM*): число выполненных инструкций при решении задачи.

Логарифмическая мера сложности алгоритма. В данном показателе сложности учитывается время выполнения каждой команды, это есть единица плюс величина, пропорциональная сумме логарифмов значений операндов.

Логарифмическая мера сложности всегда больше однородной, но с точки зрения оценки эффективности алгоритма они равносильны, выбор меры определяется спецификой рассматриваемой задачи.

С другой стороны, если к элементарным операциям добавить умножение и деление, равносильность данных мер сложности исчезает. Например, известный алгоритм быстрого переполнения памяти карманного калькулятора имеет однородную сложность порядка $t + 1$, и логарифмическую порядка 2^t.

Обычно при определении сложности алгоритма в качестве основных вычислительных ресурсов рассматривают *время*, затраченное на вычисление и использованную *память*. Затраты этих ресурсов могут зависеть от размера входных данных.

Размер данных можно определить как *размерность задачи*, то есть *число ячеек во входное ленте RAM-машины.*

С другой стороны, можно определить *размер входных данных* как суммарную *битовую длину записи всех входных параметров*, то есть число заполненных ячеек на входной ленте машины Тьюринга.

В большинстве алгоритмов решения известных задач время работы зависит только от размерности задач и не зависит от размерности входных данных. Кроме того, этим же свойством обладает любой алгоритм, все циклы которого имеют вид:

for i = 1 to $f(n)$ **do**,

где $f(n)$ – функция, зависящая только от размерности задачи. Алгоритмы с таким свойством называются *однородными*.

Сложность однородного алгоритма – целочисленная функция целого аргумента $t(n)$, равная времени его работы для входных данных размерности n.

Для неоднородных алгоритмов приходится использовать две другие меры сложности.

Сложность в худшем случае. В этом случае $t(n)$ – время работы, которое данный алгоритм может гарантировать заведомо, если известно, что суммарная битовая длина входных данных не превышает n. При этом время работы алгоритма для некоторых (или для большинства) наборов входных данных может быть существенно меньше.

Сложность в среднем. Задаётся вероятностное распределение (обычно равномерное) на наборах входных данных длины меньшей n, тогда сложностью $t(n)$ называется математическое ожидание времени работы рассматриваемого алгоритма на случайном наборе данных.

Оказывается, что существуют полиномиальные в среднем алгоритмы для некоторых $\mathcal{N}$-трудных задач.

1.7. Полиномиальные алгоритмы

Алгоритмы для одной и той же задачи сравнивают по асимптотическому поведению их сложности $t\,(n)$. Мультипликативные множители в оценке сложности игнорируются, так как они обычно имеют своим источником не особо важные внешние факторы. Имеет место приблизительное равенство: «реальное время работы = среднее время выполнения одного оператора × однородная сложность».

Теория сложности вычислений отвечает за второй множитель. Первый множитель зависит от грамотности разработки программы, качества трансляции на язык низшего уровня, быстродействия АЛУ, времени обращения к оперативной и долговременной памяти и т. д.

Результат оценки сложности алгоритма имеет вид $O\,(f\,(n))$. Понимать запись надо следующим образом: если результат имеет вид «данный алгоритм работает $O(n^2\log\,n)$ единиц времени», то это означает, что «существует положительная константа *c* такая, что время работы алгоритма в худшем случае не превышает $c \cdot n^2 \cdot \log n$ единиц времени начиная с некоторого значения *n*».

Для подавляющего большинства известных алгоритмов константа *c* находится в разумных пределах, поэтому, как правило, имеет место следующий тезис: *алгоритмы, более эффективные с точки зрения их асимптотического поведения, оказываются также более эффективными и при тех сравнительно небольших размерах входных данных, для которых они реально используются на практике.*

По определению считается, что алгоритм, работающий за время $O(n^2 \log n)$ лучше алгоритма с временем работы $O(n^3)$ и в подавляющем большинстве задач это утверждение отражает реально существующую на практике ситуацию.

Таким образом, можно определять классы алгоритмов, замкнутых относительно выбора вычислительной модели.

Определение 1.5. Алгоритм называется полиномиальным, если его сложность в наихудшем случае $t\,(n)$ ограничена сверху некоторым полиномом от n.

Это определение оказывается вполне независимым относительно выбора вычислительной модели: любые «разумные» модели (модели, в которых не может происходить экспоненциального нарастания памяти), оказываются равносильными в смысле этого определения.

1.8. Полиномиальность и эффективность

Может ли полиномиальный алгоритм быть неэффективным? Разумеется, может, если в полиномиальной оценке времени его работы $t(n) \le C \cdot n^k$ либо мультипликативная константа C, либо показатель k чрезмерно велики. Опыт показывает, что такое случается крайне редко, и подавляющее большинство полиномиальных алгоритмов для естественных задач удовлетворяет оценке $t(n) \le 10 \cdot n^3$.

Может ли неполиномиальный алгоритм быть эффективным? Ответ утвердительный, по меньшей мере, по трем причинам.

Во-первых, может случиться так, что примеры, на которых время работы алгоритма велико, настолько редки, что вероятность обнаружить хотя бы один из них на практике пренебрежимо мала. В математических терминах это означает, что алгоритм полиномиален в среднем относительно любого «разумного» вероятностного распределения.

Во-вторых, многие псевдополиномиальные алгоритмы являются эффективными, когда возникающие на практике числовые параметры не слишком велики.

Наконец, субэкспоненциальные алгоритмы, время работы которых, например, имеет вид $O(n^{c \log\log\log n})$, при всех разумных длинах входа n могут быть весьма практически эффективными.

Примеров задач, на которых нарушается основополагающее равенство

«полиномиальность» = «эффективность»

крайне мало по сравнению с числом примеров, на которых оно подтверждается. Единственную конкуренцию машинам RAM как основы для построения теории сложности вычислений на сегодняшний день составляют машины Тьюринга. Их принципиальное отличие состоит в отсутствии непрямой индексации: после проведения каких-либо действий с некоторой ячейкой процессор («головка») может перейти лишь в одну из «соседних» ячеек. Машины Тьюринга удобны при рассмотрении комбинаторных задач с небольшим количеством числовых параметров, а также (в силу своей структурной простоты) для теоретических исследований.

1.9. Сложность и полиномиальность в среднем

Среди подходов к решению $\mathcal{NP}$-трудных задач можно выделить два основных. Первый заключается в построении приближенных алгоритмов с гарантированными оценками точности получаемого решения, а второй — в

отказе от анализа сложности алгоритмов по наихудшему случаю и переходе к анализу сложности в среднем.

Известно, что большинство задач дискретной оптимизации является $\mathcal{N}$-трудными, и для них существование полиномиальных алгоритмов маловероятно. Несмотря на это, для многих исходных (входов) такие задачи бывают легко разрешимы на практике. Дело в том, что вся трудность задачи может заключаться в небольшом подмножестве входов. Проблема нахождения таких входов важна для экспериментальной оценки эффективности алгоритмов и в то же время играет заметную роль в математической криптографии. Концепция сложности в среднем для таких задач представляется более адекватной, чем концепция сложности в худшем случае.

Имеется еще одно соображение в пользу перехода от анализа по худшему случаю к анализу сложности в среднем. Оно состоит в том, что некоторые практически эффективные алгоритмы не являются эффективными при анализе по худшему случаю. Наиболее известным примером такого алгоритма является симплекс-метод решения задач линейного программирования, который, не являясь полиномиальным алгоритмом, поразительно хорошо зарекомендовал себя при решении практических задач.

Таким образом, попытки ослабить требования в определении эффективности привели к понятию сложности в среднем (average case complexity), под которым, можно понимать математическое ожидание времени работы алгоритма при заданном вероятностном распределении на исходных данных. С середины 80-х годов начала развиваться теория сложности в среднем, в некотором смысле аналогичная теории полиномиальной сводимости для $\mathcal{N}$-трудных задач. Кроме того, для ряда $\mathcal{N}$-полных задач удалось построить полиномиальные в среднем алгоритмы. Однако достигнутые здесь успехи оказались гораздо скромнее результатов в теории сложности при анализе по худшему случаю.

Пусть для данного алгоритма A время его работы на входе I обозначается через $T_A(I)$. Пусть также для каждого натурального n задано распределение вероятностей на входах I длины n, обозначаемое $P_n(I)$. Поскольку T является случайной величиной, определим среднее время работы алгоритма A как математическое ожидание времени работы алгоритма $\mathbf{E}_n\ T_A$ на входах длины n.

Определение 1.6. «Полиномиальный в среднем (точно)» Алгоритм A называется полиномиальным в среднем, если среднее время его работы ограничено полиномом от длины входа, т. е.

$$\exists\ C > 0 : \mathbf{E}_n\ T_A = O\ (n^{\ C}\ ).$$

Вместе с тем в теоретических работах, касающихся теории сводимости и сложности в среднем, полиномиальность в среднем принято определять по-другому. Причина состоит в том, что приведенное выше определение слишком чувствительно: изменение T на T^2 (типичная оценка при моделировании одной вычислительной модели на другой) может привести к тому, что алгоритм, полиномиальный в среднем, скажем, на модели RAM, не является полиномиальным в среднем на машине Тьюринга.

Поэтому в теоретических работах, касающихся теории сводимости и сложности в среднем, используют обычно другое определение.

Определение 1.7. «Полиномиальный в среднем». Алгоритм называется полиномиальным в среднем, если для времени работы алгоритма T выполняется:

$$\exists\, \varepsilon > 0 : \mathbf{E}_n\, T^{\varepsilon} = O(n).$$

1.10. Оценки функции сложности

Будем предполагать, что функции сложности $f(n)$ обладают двумя свойствами:

(a) $f(n) \geq n$;

(b) $k\, f(n) \leq f(kn)$, $k > 1$.

Для получения оценок функций сложности важны следующие леммы.

Лемма. Если функции $f(n)$ и $g\{n)$ удовлетворяют свойствам (a) и (b) и при некоторых константах $a > 1$, $b > 0$ и $d > 0$ выполняется неравенство

$$f(n) \leq f\left(\frac{n}{a}\right) + b \cdot g(n) + d \cdot n,$$

то $f(n) = O(g\{n))$.

Доказательство. Так как в силу свойства (b)

$$a \cdot f\left(\frac{n}{a}\right) \leq f\left(a \cdot \frac{n}{a}\right) = f(n),$$

то

$$f(n) \leq \frac{1}{a} f(n) + b \cdot g(n) + d \cdot n.$$

Отсюда получаем, что

$$f(n) \leq \frac{b+d}{1-\frac{1}{a}} \cdot g(n) = O\big(g(n)\big). \qquad \blacksquare$$

Лемма. Если при некоторых константах $a > 0$, $c > 0$ и $d > 0$ функция $f(n)$ удовлетворяет условию $f(1) = d$ и $f(n) = c \cdot f\left(\frac{n}{a}\right) + d \cdot n$ при $n > 1$, то она имеет вид:

$$f(n) = \begin{cases} O(n), & a > c, \\ O(n \log n), & a = c, \\ O\left(n^{\log_a c}\right), & a < c. \end{cases}$$

Доказательство проведем только для случая $n = a^t$. Имеем

$$f(n) = d \cdot n \cdot \sum_{i=0}^{t-1} \left(\frac{c}{a}\right)^i + d \cdot c^t = d \cdot n \cdot \sum_{i=0}^{t} \left(\frac{c}{a}\right)^i,$$

так как $n = a^t$. Отсюда получаем:

$$f(n) \le \frac{d \cdot n}{1 - c/a} = O(n), \quad \text{если } a > c,$$

$$f(n) = d \cdot n \cdot t = d \cdot n \cdot \log n, \quad \text{если } a = c,$$

$$f(n) = d \cdot n \cdot \frac{(c/a)^{t+1}}{c/a - 1} \le \frac{c \cdot d \cdot n}{a} \cdot \frac{c^t}{a^t} \cdot \frac{1}{c/a - 1} = O(c^t) =$$
$$= O\left(n^{\log_a c}\right), \quad \text{если } a < c. \ \blacksquare$$

1.11. Сложность арифметических операций с целыми числами

Сложение и вычитание. Стандартные школьные алгоритмы сложения и вычитания чисел столбиком имеют оценку сложности $O(\log N)$, где N — большее из двух
чисел. Это минимальная возможная оценка сложности и поэтому нет смысла заниматься оптимизацией выполнения этих операций.

Умножение и деление. Для умножения и деления двух чисел N и M алгоритмы умножения столбиком и деления углом дают оценку $O(\log N \log M)$. Поэтому для функций оценки сложности умножения и деления двух n-разрядных чисел выполняется неравенство

$$n \le f(n) \le O(n^2).$$

Можно доказать, что сложности операций **умножения**, **деления с остатком**, **возведения в квадрат** и **нахождения обратного к данному числу** совпадают с точностью до константного множителя. Под обратным к данному числу N, записанному в виде последовательности из n двоичных знаков, здесь понимается правильная дробь, имеющая n двоичных знаков после запятой и полученная из числа N отбрасыванием более младших разрядов.

Возведение в степень. Для вычисления степени N^a натуральное число a представляется в виде

$$a = \sum_{i=0}^{k-1} a_i 2^i = (\dots(a_{k-1}2 + a_{k-2})2 + \dots + a_1)2 + a_0,$$

где k – количество двоичных разрядов числа a.

Теперь возведение в данную степень можно свести к последовательному выполнению операций двух типов: умножению на N и возведению в квадрат. Отсюда получается следующая оценка сложности

$$f(n) = O(M(n)k) = O(kn^2).$$

Здесь n есть число разрядов в двоичной записи числа N, $M(n)$ – сложность операции умножения двух n-разрядных целых чисел.

Контрольные вопросы

Дайте определение понятия алгоритма.
Какие алгоритмы называют полиномиальными?
Какие алгоритмы называют экспоненциальными?
Какой класс задач называют классом $\mathcal{N}$-полных задач?
Какая функция является вычислимой?
Что понимается под размерностью задачи?
Какие алгоритмы являются однородными?
Какие алгоритмы называются полиномиальными в среднем?
Поясните запись оценки сложности алгоритма $O(kn^2)$.
Приведите оценки сложности таких операций как сложение, умножение, возведение в степень.

2. АРИФМЕТИКА ЦЕЛЫХ ЧИСЕЛ

2.1. Свойства чисел

Начнем рассмотрение с понятия *натуральных чисел* (от латинского слова naturalis — «естественный»). Множество натуральных чисел обозначается символом $\mathbb{N}$ и используется для счета или обозначения количества чего-либо.

$$\mathbb{N} = \{1, 2, 3, 4, 5, 6, \ldots\}$$

Отдельно стоит упомянуть о включении или не включении нуля в множество натуральных чисел. В русскоязычной литературе в множество $\mathbb{N}$ ноль включать не принято, но для некоторых аналитических рассуждений ноль может быть включен, обычно это будет указано отдельно.

Основу для представления данных и выполнения преобразований в криптографических алгоритмах составляют операции с целыми чисел.

Множество *целых чисел* обозначается $\mathbb{Z}$ (от немецкого слова Zahlen — «числа») и содержит все числа (без дробей) от минус бесконечности до плюс бесконечности:

$$\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$$

Возможные операции над целыми числами (входы и выходы $\in \mathbb{Z}$):

- Унарные (один вход и один выход): $-a$
- Бинарные (два входа и один выход): $a + b$, $a - b$, $a \cdot b$, $a \bmod n$, $a \operatorname{div} n$
- Деления (два входа и два выхода): $a/n = q(r)$,
  $a = q \cdot n + r$ (q – неполное частное, r – остаток).

В операции деления обычно предполагается, что остаток $r \geq 0$ и $n > 0$.

Пример:

Разделить 255 на 11:

$a = 255$, $n = 11$: $a = 23 \cdot 11 + 2$, т.е. $q = 23$, $r = 2$.



Рисунок 2.1. Целое и остаток числа, $a > 0$

Если $a < 0$, тогда после деления заменяем: $q \Rightarrow q - 1$, $r \Rightarrow r + n$.

Пример:
Разделить – 255 на 11:
$a = -255$, $n = 11$: $a = (-23) \cdot 11 - 2 \Leftrightarrow (-24) \cdot 11 + 9$, т.е. $q = -24$, r = 9

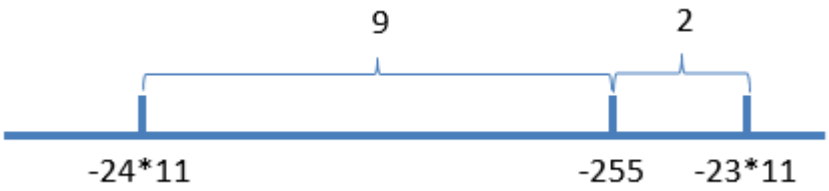


Рисунок 2.2. Целое и остаток числа, a <0

Можно отметить, что такое представление числа с помощью неполного частного, делителя и остатка всегда возможно.

Теорема 2.1. Теорема о делении с остатком (без доказательства).
Всякое целое число a представляется единственным способом через положительное целое число n в форме:

$$a = q \cdot n + r, \qquad 0 \le r < n.$$

В криптографии делимость чисел является важным свойством. Приведем основные обозначения.
В выражении $a = q \cdot n + r$, если $r = 0$, то $a = q \cdot n$, следовательно, n делит a нацело (или a делится без остатка на n), обозначается: $n \mid a$, и если $r \neq 0$, то n не делит a: $n \nmid a$.[1]

Пример:
4|8, 11|(–22), –6|120; 4 ∤ 9, 8 ∤ 4, 11 ∤(–32).

Можно отметить следующие свойства теории делимости:

Свойство 1: если a|1, то a = ±1.

Свойство 2: если a|b и b|a, то a = ±b

Свойство 3: если a|b и b|c, то a|c

[1] Запись $a|b$ может обозначать a делит b, но возможно, и что a делится на b (без остатка) – конкретный смысл зависит предпочтений авторов.

Свойство 4: если a|b и a|c, то a|(m · b + n · c), где m и n — произвольные целые числа.

Особую роль в теории чисел и в криптографии играют *простые числа*.

Определение 2.1. *Простое число* (иногда используется обозначение ℙ от английского prime number) – это натуральное число, которое имеет ровно два различных натуральных делителя: 1 и самого себя.

$$\mathbb{P} = \{2, 3, 5, 7, 11, 13, 17, 19, 23, \ldots\}$$

Все натуральные числа можно разделить на три множества, простые числа, составные и числа 0 и 1. ℕ = {простые} ∪ {0} ∪ {1} {составные}.

Сколько же всего простых чисел? Существует довольно много доказательств бесконечности количества простых чисел, одно из них было дано еще Евклидом (др.-греч. Εὐκλείδης, от «добрая слава» — древнегреческий математик III века до н.э.).

Теорема 2.2. Количество простых чисел – бесконечно.

Доказательство. Доказательство Евклида. Пусть известен конечный набор простых чисел. Перемножим эти числа и к результату добавим единицу. Полученное число не делится нацело ни на одно число из ранее известного набора (остаток при делении на любое из них будет единица).

Следовательно, полученное число само является простым, или делится на какое-то другое простое число вне исходного набора, что и доказывает, что всегда существует некоторое простое число вне исходного набора. ■

Другой вопрос, который может интересовать математиков и криптографов, это вопрос о том, как часто встречаются простые числа.

Теорема 2.3. Теорема о распределении простых чисел (без доказательства). Функция $\pi(n)$ (количество простых чисел, меньших n) приблизительно равна:

$$\pi(n) \approx \frac{n}{\ln n}.$$

Соотношение хорошо работает на больших n, но может давать значительные отклонения на малых значениях n, например:

$n = 10$;	$\pi(n) = 4$;	$\ln(n)/n \approx 4.34$,
$n = 100$;	$\pi(n) = 25$;	$\ln(n)/n \approx 21.71$,

$n = 1000$; $\pi(n) = 168$; $\ln(n)/n \approx 144.76$,
$n = 10^6$; $\pi(n) = 78498$; $\ln(n)/n \approx 72382$.

Из теоремы, в частности, следует, что для любого произвольного числа n, вероятность что оно простое примерно составляет $1/\ln(n)$.

Лемма Евклида. Если простое число p делит без остатка произведение двух целых чисел $x \cdot y$, то p делит без остатка либо x либо y.

Доказательство. Пусть $x \cdot y$ делится на p, но x не делится на p. Тогда x и p — взаимно простые (не имеют общих множителей), следовательно, найдутся такие целые числа u и v, что:

$$x \cdot u + p \cdot v = 1$$

Данное выражение называется соотношением Безу и будет подробно рассмотрено в материале о расширенном алгоритме Евклида.

В данном соотношении умножим обе части на y и получим:

$$(x \cdot y) \cdot u + p \cdot v \cdot y = y.$$

В результате оба слагаемых в левой части выражения делятся на p, значит, и правая часть выражения делится на p. (Мы доказали, что если x не делится на p, то y обязательно делится.) ■

Приведем некоторые свойства простых чисел.

Свойство 1. Наименьший отличный от единицы делитель целого числа n, $n > 1$, является простым числом.

Свойство 2. Любое простое число p, $p > 3$, может быть представлено в виде $p = 6 \cdot k + 1$ или $p = 6 \cdot k - 1$, где k — некоторое натуральное число.

Свойство 3. Если даны два числа p и a (p — простое, a — натуральное), то $a^p - a$ делится на p. Данное свойство формулируется в виде малой теоремы Ферма и будет рассмотрена далее.

Свойство 4. Если натуральное число $n > 1$, то всегда существует такое простое p, что $n < p < 2 \cdot n$ (постулат Бертрана).

Перейдем к рассмотрению теоремы, которая получила название основной теоремы арифметики.

Теорема 2.4. Основная теорема арифметики.

Каждое натуральное число n > 1 может быть представлено в виде:

$$n = p_1 \cdot p_2 \cdot p_3 \cdot \ldots \cdot p_k,$$

где p_1, p_2, p_3, …, p_k — простые числа, возможно совпадающие, и такое представление является единственным с точностью до порядка следования сомножителей.

Доказательство существования представления (индукция по n).

Предположим, что уже доказано существование такого представления для всех чисел меньших n. Так, например, для числа два мы уже знаем существование, так как число два является простым.

Если число n простое, то утверждение уже справедливо.

Если число n составное, то может быть представлено в виде $n = a \cdot b$, где $a < n$ и $b < n$. По предположению индукции у каждого чисел меньших n уже существует разложение;

Выполняя разложения для сомножителей a и b, получаем разложение и для n. ■

Доказательство единственности представления.

Пусть n — наименьшее натуральное число, которое может быть представлено в виде произведение простых чисел двумя разными способами:

$$n = r_1 \cdot r_2 \cdot r_3 \cdot \ldots \cdot r_k \text{ и } n = q_1 \cdot q_2 \cdot q_3 \cdot \ldots \cdot q_s,\ r_i \neq q_j \text{ для всех } i \text{ и } j.$$

Если оба разложения не содержат множителей (пустые), то они уже одинаковы.

Если оба выражения содержат множители, то пусть число p является любым из множителей в любом из двух разложений.

Если p входит и в другое разложение, то можно выполнить сокращение на этот множитель p каждого из разложений и получить два разных разложения числа «n/p», что невозможно (см. первоначальное предположение, что n – наименьшее число, которое может быть разложено двумя разными способами).

Если число p не входит в другое разложение, то в рассматриваемых выражениях одно произведение чисел делится на p, вместе с тем другое произведение на p делится, что противоречит их равенству. Пришли к противоречию с первоначальным предположением, утверждающим, что возможно два разложения. Следовательно, мы доказали, что число может быть разложено на произведение простых чисел единственным образом. ■

Следствие. Любое натуральное число n может быть единственным образом, с точностью до порядка следования сомножителей, представлено в виде:

$$n = p_1^{s_1} \cdot p_2^{s_2} \cdot p_3^{s_3} \cdot \ldots \cdot p_k^{s_k},$$

где p_1, p_2, p_3, …, p_k – различные простые числа, s_1, s_2, s_3, …, s_k некоторые натуральные числа.

Приведенное представление для числа n называется *каноническим разложением* на простые множители или *каноническим представлением*.

Рассмотрим некоторые свойства составных чисел.

Теорема 2.5. Наименьший отличный от единицы делитель составного числа a не превосходит $\sqrt{a}$.

Доказательство. Пусть b – наименьший делитель числа a отличный от единицы $a = b \cdot a_1$, тогда:

$$a_1 \geq b.$$

Умножаем обе части на a:

$$a_1 a \geq ba.$$

Подставляем в правую часть разложение для числа a на множители:

$$a_1 a \geq b^2 a_1;$$
$$a \geq b^2;$$
$$b \leq \sqrt{a}. \blacksquare$$

2.2. Алгоритм Евклида

Как уже было отмечено у одного числа может быть много делителей. Два целых положительных числа могут иметь много общих делителей. Любое целое число, которое делит целые числа a, b, … называется их *общим делителем*. Два числа могут иметь только один общий делитель, который является *наибольшим общим делителем*.

Функция, определяющая наибольший общий делитель обозначается как НОД(a, b), но в литературе часто используется краткое обозначение (a, b).

Наибольший общий делитель имеет очевидные свойства, исходящие из его определения: НОД(a,b)|a и НОД(a,b)|b. Важным свойством является соотношение НОД(a,0) = a.

Для составных чисел важной функцией является наименьшее общее кратное чисел НОК(a,b), для которого тоже может быть использовано краткое обозначение: [a,b]. Свойства наименьшего общего кратного: a | НОК(a,b) и b | НОК(a,b)

Пример.

105 = 3·5·7

84 = 2·2·3·7

Общие делители 105 и 84: 1, 3, 7.

НОД(105,84) = 21.
НОК(105,84) = 420.

Функция наибольший общий делитель и функция наименьшее общее кратное связаны между собой соотношениями. Пусть известны канонические разложения n и m:

$$n = p_1^{s_1} \cdot p_2^{s_2} \cdot p_3^{s_3} \cdot \ldots \cdot p_k^{s_k}$$
$$m = p_1^{e_1} \cdot p_2^{e_2} \cdot p_3^{e_3} \cdot \ldots \cdot p_k^{e_k},$$

где p_i – простые, несовпадающие числа, s_i и e_i – неотрицательные числа (могут быть нулями).

Тогда (m, n) и $[m, n]$ могут быть выражены как:

$$(n, m) = p_1^{min(s_1,e_1)} \cdot p_2^{min(s_2,e_2)} \cdot p_3^{min(s_3,e_3)} \cdot \ldots \cdot p_k^{min(s_k,e_k)},$$

$$[n, m] = p_1^{max(s_1,e_1)} \cdot p_2^{max(s_2,e_2)} \cdot p_3^{max(s_3,e_3)} \cdot \ldots \cdot p_k^{max(s_k,e_k)},$$

а связь между ними в виде:

$$\text{НОК}(n, m) = n \cdot m / \text{НОД}(n, m).$$

Наибольший общий делитель может быть вычислен не только между двумя числами, но для любого конечного количества чисел:

НОД (b_1, b_2, b_3, …, b_n) = d_n:
d_2 = НОД (b_1, b_2);
d_3 = НОД (d_2, b_3);
d_4 = НОД (d_3, b_4);
…
d_n = НОД (d_{n-1}, b_n).

С наибольшим общим делителем связано еще несколько понятий, в частности, понятие взаимной простоты чисел.

Определение 2.2. Попарная простота. Если $(a, b) = 1$, то числа a и b называются попарно простыми.

Часто для обозначения попарной простоты используют более общий термин – взаимная простота.

Определение 2.3. Взаимная простота. Если НОД (a_1, a_2, …, a_n) = 1, то числа a_1, a_2, …, a_n называются *взаимно простыми*.

Примеры.

Числа 23, 21, 41, 24 являются взаимно простыми, т.к. (23,21,41,24) = 1.

Числа 23, 33, 24 являются попарно простыми, т.к. (5, 13) = 1; (13, 22) = 1; (5, 22) = 1.

Для рассмотрения дальнейших свойств чисел, нам потребуется следующая теорема.

Теорема 2.6. (без доказательства). Если a кратно b ($b|a$), то совокупность общих делителей a и b совпадает с совокупностью делителей одного числа b, в частности, $(a, b) = b$.

Теорема 2.7. (без доказательства). Если $a = b \cdot q + c$, то совокупность общих делителей чисел a и b совпадает с совокупностью делителей чисел b и c, в частности $(a, b) = (b, c)$.

Результаты этой теоремы можно использовать для алгоритма пошагового вычисления наибольшего общего делителя, уменьшая аргументы функции на каждом шагу алгоритма.

Пример. Вычислить НОД (36,10):

НОД (36,10): [36 = 3·10+6] = НОД (10,6) =
= НОД (10,6): [10 = 1·6+4] = НОД (6,4) =
= НОД (6,4): [6 = 1·4+2] = НОД (4,2) =
= НОД (4,2): [4 = 2·2+0] = НОД (2,0) = 2.

Данный результат и возможность представить число a в виде $a = q \cdot n + r$, $0 \le r < n$ позволяет составить алгоритм вычисления наибольшего общего делителя.

Данный алгоритм получил название а*лгоритм Евклида* и выполняется приведенным далее набором шагов.

Пусть даны положительные целые числа a, b и пусть $a > b$ (если это неравенство не выполняется, то переобозначением переменным можно привести условие к заданному).

Согласно теореме о делении с остатком составляем последовательность равенств:

$a = b \cdot q_1 + r_1$ $0 \le r_1 < b$
$b = r_1 \cdot q_2 + r_2$ $0 \le r_2 < r_1$
$r_1 = r_2 \cdot q_3 + r_3$ $0 \le r_3 < r_2$
… …

$r_{n-3} = r_{n-2} \cdot q_{n-1} + \boldsymbol{r_{n-1}}$ $\quad 0 \le r_{n-1} < r_{n-2}$

$r_{n-2} = r_{n-1} \cdot q_n$ $\quad r_n = 0$

Алгоритм заканчивается, когда получается некоторое $r_n = 0$. Последнее неизбежно, т.к. ряд $b, r_1, r_2, \ldots$ не может содержать более чем b положительных чисел.

Таким образом, наибольший общий делитель равен последнему не нулевому остатку алгоритма Евклида. Поскольку, согласно теоремам (3.6) и (3.7): $(a, b) = (b, r_1) = (r_1, r_2) = \ldots = (r_{n-2}, r_{n-1}) = (r_{n-1}, 0) = r_{n-1}$.

Пример. Определить: НОД (25, 9).

$25 = 9 \cdot 2 + 7$

$9 = 7 \cdot 1 + 2$

$7 = 2 \cdot 3 + 1$

$2 = 1 \cdot 2 + 0$

Ответ. НОД (25, 9) = 1

2.3. Расширенный алгоритм Евклида

На основе рассмотренного алгоритма Евклида может быть получен несколько более общий алгоритм, позволяющий определить дополнительные соотношения между аргументами. Такой алгоритм получил название *расширенного алгоритма Евклида*. Рассмотрим теорему.

Теорема 2.8. (без доказательства)**.** Наибольший общий делитель d любых двух положительных целых чисел a и b единственным образом представляется их линейной комбинацией $d = u \cdot a + v \cdot b$, *где* $u, v \in \mathbb{Z}$.

Рассмотрим соотношения в каждой строке алгоритма Евклида и сформируем выражения для вычисления остатков на каждом шаге алгоритма, т.е. запишем формулы для r_i:

$r_1 = a + b \cdot (-q_1)$

$r_2 = b - r_1 \cdot q_2 = b - (a + b \cdot (-q_1)) \cdot q_2 = a \cdot (-q_2) + b \cdot (1 + q_2 \cdot q_1)$

……

НОД $(a, b) = r_{n-1} = a \cdot s + b \cdot t$,

где s и t целые.

Это представление наибольшего общего делителя называется *соотношением Безу*, а числа s и t — *коэффициентами Безу*.

Рассмотрим пример выполнения расширенного алгоритма Евклида и нахождение коэффициентов Безу. Пошаговое выполнение расширенного

алгоритма может выполняться как начиная с первых строк алгоритма Евклида, так и с последних. Рассмотрим выполнение расширенного алгоритма Евклида для чисел 25 и 9. Выполнять будем, начиная с последнего ненулевого остатка, т.е. «обратным ходом»:

$\mathbf{1} = 7 - \mathbf{2}\cdot 3 =$
$7 - (9 - 7\cdot 1)\cdot 3 = \mathbf{7}\cdot 4 - 9\cdot 3 =$
$= (25 - 9\cdot 2)\cdot 4 - 9\cdot 3 = \mathbf{4}\cdot 25 + (-\mathbf{11})\cdot 9.$

Возможно выполнение алгоритма и прямым ходом (последовательно каждый остаток выражаем через числа a и b из соотношений в алгоритме Евклида):

$\mathbf{7} = 25 - 9\cdot 2$
$\mathbf{2} = 9 - \mathbf{7} = 9 - (25 - 9\cdot 2) = -25+3\cdot 9$
$\mathbf{1} = \mathbf{7} - \mathbf{2}\cdot 3 = (25 - 9\cdot 2) - (-25+3\cdot 9)\cdot 3 = \mathbf{4}\cdot 25 + (-\mathbf{11})\cdot 9$

Рассмотрим алгоритм в общем виде.

$a = b\cdot q_1 + r_1$ и $r_1 = \alpha_1\cdot x_1 + \beta_1\cdot y_1$
$b = r_1\cdot q_2 + r_2$ и $r_2 = \alpha_2\cdot x_2 + \beta_2\cdot y_2$
……
$r_{n-3} = r_{n-2}\cdot q_{n-1} + r_{n-1}$ и $r_{n-1} = \boldsymbol{\alpha}\cdot x_{n-1} + \boldsymbol{\beta}\cdot y_{n-1}$
$r_{n-2} = r_{n-1}\cdot q_n$ $r_n = 0$

Ответ: НОД (a, b) = r_{n-1} (ответ – последний **ненулевой** остаток). Представим это в виде таблицы:

Таблица 3.1

остатки	Частные	x	y
a	*	$x_{(-1)}$	$y_{(-1)}$
b	*	x_0	y_0
r_1	q_1	x_1	y_1
r_2	q_2	x_2	y_2
…	…	…	…
r_{n-2}	q_{n-2}	x_{n-2}	y_{n-2}
r_{n-1}	q_{n-1}	x_{n-1}	y_{n-1}

Первые две строки в данной таблице добавлены для построения рекуррентного вычисления x_i и y_i на основе предыдущих значений по единообразным соотношениям. Они потребуются для вычисления x_1 и y_1.

Начальные значения x_i и y_i в этих строчках устанавливаются в $x_{(-1)} = 1$, $y_{(-1)} = 0$, $x_0 = 0$, $y_0 = 1$. Далее x_i и y_i вычисляются по соотношениям:

$$x_j = x_{j-2} - q_j \cdot x_{j-1};$$
$$y_j = y_{j-2} - q_j \cdot y_{j-1};$$

Рассмотрим пример выполнения расширенного алгоритма Евклида для чисел **1234** и **54**

Таблица 3.2

остатки	частные	X	y
1234	*	1	0
54	*	0	1
46	22	1 – 22·0 = 1	0 – 22·1 = –22
8	1	0 – 1·1 = –1	1 – 1· (–22) = 23
6	5	1 – 5· (– 1) = 6	– 22 – 5·23 = –137
2	1	– 1 – 1·6 = –7	23 – 1·(–137) = 160
0	3	*	*

α = –7; β = 160. Действительно: (–7)·1234 + 160·54 = 2.

2.4. Функция Эйлера

Важным элементом теории чисел является функция Эйлера.

Определение 2.4. *Функцией Эйлера* $\varphi(a)$ называется функция, которая для $\forall\ a \in \mathbb{N}$ равна количеству чисел в ряду 1, …, a – 1 попарно простых (взаимно простых) с числом a, где $a \geq 1$.

φ (1) = 1
φ (2) = 1
φ (3) = 2 (1, 2)
φ (4) = 2 (1, ~~2~~, 3) => (1, 3)
φ (5) = 4 (1, 2, 3, 4)
φ (6) = 2 (1, ~~2~~, ~~3~~, ~~4~~ ,5) => (1, 5)

Теорема 2.9. (без доказательства). Пусть число a представлено в виде канонического разложения.

$$a = p_1^{s_1} \cdot p_2^{s_2} \cdot \ldots \cdot p_n^{s_n},$$

Тогда:

$$\varphi(a) = \left(p_1^{s_1} - p_1^{s_1-1}\right) \cdot \left(p_2^{s_2} - p_2^{s_2-1}\right) \cdot \ldots \cdot \left(p_k^{s_k} - p_k^{s_k-1}\right)$$

Если p является простым числом, то справедливо:

$$\varphi(p) = (p^1 - p^0) = p - 1.$$

Кроме того, для взаимно простых чисел p и q справедливо соотношение (свойство мультипликативности):

$$\varphi(pq) = (p-1)(q-1).$$

2.5. Непрерывные (цепные) и подходящие дроби

Рассмотрим произвольное вещественное число a. Тогда его можно представить в виде:

$$a = q_1 + \frac{1}{a_2}$$

где q_1 – наибольшее целое, не превосходящее a; a_2 – вещественное число, a_2>1.

Точно так же при нецелых $a_2, \dots, a_{n-1}$ имеем:

$$a_2 = q_2 + \frac{1}{a_3}, a_3 > 1$$
$$\dots$$
$$a_{n-1} = q_{n-1} + \frac{1}{a_n}, a_n > 1$$

В результате формируем следующее разложение a в *непрерывную дробь* (иногда используется название *цепная дробь*):

$$a = q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \cfrac{1}{\dots + \cfrac{1}{q_{n-1} + \cfrac{1}{a_n}}}}}$$

Пусть число *a* рациональное и может быть представлено несократимой дробью $a=x/y$ с положительным знаменателем, то получение рассмотренного ранее разложения будет состоять из конечного числа шагов, каждый из которых может быть выполнен с помощью алгоритма Евклида.

Рассмотрим последовательное формирование цепной дроби:

$a = b \cdot q_1 + r_2$	$0 \le r_2 < b$	$\frac{a}{b} = q_1 + \frac{r_2}{b}$
$b = r_2 \cdot q_2 + r_3$	$0 \le r_3 < r_2$	$\frac{b}{r_2} = q_2 + \frac{r_3}{r_2}$
…		…
$r_{n-2} = r_{n-1} \cdot q_{n-1} + r_n$	$0 \le r_n < r_{n-1}$	$\frac{r_{n-2}}{r_{n-1}} = q_{n-1} + \frac{r_n}{r_{n-1}}$
$r_{n-1} = r_n \cdot q_n$		$\frac{r_{n-1}}{r_n} = q_n$

В итоге такая непрерывная дробь для рационального числа будет представлена в виде:

$$\frac{a}{b} = q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \cfrac{1}{\ldots + \cfrac{1}{q_{n-1} + \frac{1}{q_n}}}}}.$$

Числа q_1, q_2, … q_n, участвующие в разложении числа a в непрерывную дробь, являются неполными частными (в случае рационального a это будут неполные частные последовательных делений алгоритма Евклида).

При этом дроби, получаемые на шагах алгоритма, называют *подходящими дробями*:

$$\delta_1 = \frac{P_1}{Q_1} = q_1, \qquad \delta_2 = \frac{P_2}{Q_2} = q_1 + \frac{1}{q_2}, \qquad \delta_3 = \frac{P_3}{Q_3} = q_1 + \frac{1}{q_2 + \frac{1}{q_3}}$$

Пример. Пусть имеется рациональная дробь 9/13, необходимо построить непрерывную дробь и найти неполные частные и подходящие дроби.

$9 = 13 \cdot 0 + 9$	$q_1 = 0$	$\delta_1 = 0$
$13 = 9 \cdot 1 + 4$	$q_2 = 1$	$\delta_2 = 0 + \frac{1}{1} = 1$
$9 = 4 \cdot 2 + 1$	$q_3 = 2$	$\delta_3 = 0 + \frac{1}{1 + \frac{1}{2}}$
$4 = 1 \cdot 4 + 0$	$q_4 = 4$	$\delta_3 = 0 + \frac{1}{1 + \frac{1}{2 + \frac{1}{4}}}$

В итоге получаем:

$$\frac{9}{13} = 0 + \cfrac{1}{1 + \cfrac{1}{2 + \frac{1}{4}}}.$$

Возможно получение того же результата процедурами «переворачивания» дроби:

$$\frac{9}{13} = 0 + \frac{1}{\frac{13}{9}} = 0 + \frac{1}{1+\frac{4}{9}} = 0 + \frac{1}{1+\frac{1}{\frac{9}{4}}} = \frac{1}{1+\frac{1}{2+\frac{1}{4}}}$$

Теорема 2.10. Для подходящих дробей при s>1 справедливо соотношение:

$$\delta_s = \frac{P_s}{Q_s} = \frac{q_s P_{s-1} + P_{s-2}}{q_s Q_{s-1} + Q_{s-2}},$$

где

$$P_s = q_s P_{s-1} + P_{s-2};$$
$$Q_s = q_s Q_{s-1} + Q_{s-2}.$$

Доказательство. По индукции. Положим $P_0 = 1\ Q_0 = 0$.

Тогда, поскольку δ_s получается из δ_{s-1} заменой в выражении для δ_s числа q_{s-1} на $(q_{s-1} + \frac{1}{q_s})$, имеем:

$$\delta_1 = \frac{P_1}{Q_1} = \frac{q_1}{1}; \qquad P_1 = q_1;\ Q_1 = 1$$
$$\delta_2 = \frac{P_2}{Q_2} = q_1 + \frac{1}{q_2} = \frac{q_2 \cdot q_1 + 1}{q_2 \cdot 1 + 0} = \frac{q_2 \cdot P_1 + P_0}{q_2 \cdot Q_1 + Q_0}.$$

Проверим для $k > 2$ (при $k = 2$ уже убедились). Предположим, что справедливо для любого k:

$$\frac{P_k}{Q_k} = \frac{q_k \cdot P_{k-1} + P_{k-2}}{q_k \cdot Q_{k-1} + Q_{k-2}}$$

Тогда

$$\delta_{k+1} = \frac{P_{k+1}}{Q_{k+1}} = \frac{\left(q_k + \frac{1}{q_{k+1}}\right) \cdot P_{k-1} + P_{k-2}}{\left(q_k + \frac{1}{q_{k+1}}\right) \cdot Q_{k-1} + Q_{k-2}} =$$

$$= \frac{q_{k+1}(q_k \cdot P_{k-1} + P_{k-2}) + P_{k-1}}{q_{k+1}(q_k \cdot Q_{k-1} + Q_{k-2}) + Q_{k-1}} =$$

$$= \frac{q_{k+1} \cdot P_k + P_{k-1}}{q_{k+1} \cdot Q_k + Q_{k-1}}$$

т.е. получили для δ_{k+1} выражение такого же вида, что и для δ_k. ■

Алгоритм нахождения:

1. Определяем неполные частные (q_1, q_2, …, q_n) с помощью алгоритма Евклида.
2. Обозначаем $P_0 = 1$, $Q_0 = 0$, $P_1 = q_1$, $Q_1 = 1$.
3. Определяем (для $s = 2, 3, \ldots$):

$$P_s = q_s \cdot P_{s-1} + P_{s-2},$$
$$Q_s = q_s \cdot Q_{s-1} + Q_{s-2}.$$

4. Рассчитываем:

$$\delta_s = \frac{P_s}{Q_s}.$$

Вычисления P_i и Q_i можно свести в таблицу:

Таблица 3.3

q	–	q_1	q_2	…	q_{k-2}	q_{k-1}	q_k	…	q_n	q_{n+1}
P	1	q_1	P_2	…	P_{k-2}	P_{k-1}	P_k	…	P_n	a
Q	0	1	Q_2	…	Q_{k-2}	Q_{k-1}	Q_k	…	Q_n	b

Последний столбец пишем только в том случае, когда рассматриваем несократимую дробь с положительным знаменателем.

Рассмотрим еще один пример. Разложим в непрерывную дробь несократимую дробь 125/92. Получаем непрерывную дробь:

$$\frac{125}{92} = 1 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{1 + \cfrac{1}{2 + \frac{1}{2}}}}}}$$

Таблица 3.4

q	–	1	2	1	3	1	2	2
P	1	1	3	4	15	19	53	125
Q	0	1	2	3	11	14	39	92

Подходящие дроби для 125/92:

$$\delta_1 = 1, \delta_2 = \frac{3}{2}, \delta_3 = \frac{4}{3}, \delta_4 = \frac{15}{11}, \delta_5 = \frac{19}{14}, \delta_6 = \frac{53}{39}, \delta_7 = \frac{125}{92}.$$

Если обратить внимание на последовательные значения подходящих дробей и исходной величины, то можно заметить, что каждая следующая полученная подходящая дробь оказывается всё ближе к самому значению исходной величины:

$$\delta_1 = 1 < 125/92,$$
$$\delta_2 = \frac{3}{2} > 125/92,$$
$$\delta_3 = \frac{4}{3} < 125/92,$$
$$\delta_4 = \frac{15}{11} > 125/92,$$
$$\delta_4 = \frac{19}{14} < 125/92,$$
$$\delta_4 = \frac{53}{39} > 125/92,$$
$$\delta_4 = \frac{125}{92} = \frac{125}{9}.$$

Подходящие дроби и непрерывная дробь могут быть найдены не только для рациональных чисел. Представим в виде непрерывной дроби корень из числа 2:

$$\sqrt{2} = 1 + \sqrt{2} - 1 = 1 + \frac{\left(\sqrt{2}-1\right)\left(\sqrt{2}+1\right)}{\left(\sqrt{2}+1\right)} = 1 + \frac{2-1}{\left(\sqrt{2}+1\right)} = 1 + \frac{1}{1+\sqrt{2}}$$

Можно продолжить выполнять аналогичные действия и получим бесконечную непрерывную дробь:

$$\sqrt{2} = 1 + \frac{1}{2+\frac{1}{2+\frac{1}{\sqrt{2}}}} \text{ или } \sqrt{2} = 1 + \frac{1}{2+\frac{1}{2+\frac{1}{\dots}}}$$

На основе соотношений

$$P_s = q_s \cdot P_{s-1} + P_{s-2}$$
$$Q_s = q_s \cdot Q_{s-1} + Q_{s-2}$$

Рассчитываем подходящие дроби:

$$\delta_s = \frac{P_s}{Q_s}$$

Составим подходящие дроби: δ_1=1/1; δ_2=1+1/2=3/2.

$$\delta_3 = \frac{2*3+1}{2*2+1} = \frac{7}{5},\ \delta_4 = \frac{2*7+3}{2*5+2} = \frac{17}{12},\ \delta_5 = \frac{2*17+7}{2*12+5} = \frac{41}{29},\ \delta_6 = \frac{2*41+17}{2*29+12} = \frac{99}{70}$$

Полученная подходящая дробь дала приближение для значения корень из двух в виде отношения 99/70 ≈ 1,4142857…. , аналогичное вычисление на обычном калькуляторе выдает значение 1,41421356237…, т.е. полученное приближение совпало с самим значением с точностью до четвертого знака.

Подходящие дроби применяются для приближения действительных чисел рациональными с наилучшим приближением, сокращения обыкновенных дробей, решения диофантовых уравнений с двумя неизвестными и решения сравнений первой степени по модулю (будут рассмотрены позднее).

2.6. Арифметика остатков

Определение 2.5. Пусть m – некоторое целое число, $m > 1$. Пусть числа a и b при делении на m имеют один и тот же остаток:

$$\begin{matrix} a = m \cdot t_1 + r \\ b = m \cdot t_2 + r\,' \end{matrix} \qquad 0 \le r < m$$

Тогда числа a и b называют *равноостаточными*.

Если вычесть из одного соотношения в определении второе, то получим:

$$a - b = m \cdot t_1 + r - m \cdot t_2 - r = m(t_1 - t_2) = mt;$$

$$a = b + mt,$$

где t – целое число.

Числа a и b отличаются друг от друга только целым количеством добавленного числа m. Если теперь мы будем искать остатки от левой и правой частей полученного выражения при делении на число m, то получим равенство остатков.

Определение 2.6. Если a и b два целых числа и их разность $a - b$ делится на целое положительное число m, то говорят, что a сравнимо с b *по модулю* m:

$$a = b \bmod m.$$

В записи сравнимости может использоваться символ «≡», подчеркивающий эквивалентность чисел a и b, но мы ограничимся использованием символа «=».

Бинарный оператор mod используется для обозначения оператора нахождения остатка от деления, т.е. запись $a \bmod b$ представляет остаток от деления a на b. Из выражения $a - b = m \cdot k \rightarrow a = b + m \cdot k$, где $k \in \mathbb{Z}$

Если представить b в виде:

$$b = m \cdot t_1 + r, \text{ то}$$
$$a = m \cdot t_1 + r + m \cdot k = m \cdot (t_1 + k) + r,$$

таким образом, при делении чисел a и b на число m (модуль) будет получен один и тот же остаток.

Примеры сравнимых чисел:

$16 = 7 \bmod 3$	$16 = 7 + (3) \cdot 3$
$5 = 9 \bmod 4$	$5 = 9 + (-1) \cdot 4$
$12 = 27 \bmod 5$	$12 = 27 + (-3) \cdot 5$

Приведем свойства сравнимых чисел.

Свойство 1. Рефлективность.
$a = a \bmod n$

Свойство 2. Симметричность.
$a = b \bmod n$
$b = a \bmod n$

Свойство 3. Транзитивность.
если $a = b \bmod n$ и $b = c \bmod n$ то $a = c \bmod n$.
Любые целые a и b сравнимы по модулю 1

Свойство 4. Сравнимые числа можно почленно складывать.
$a_1 = b_1 \bmod n$
$a_2 = b_2 \bmod n$
$a_1 + a_2 = b_1 + b_2 \bmod n$

Свойство 5. Слагаемые можно переносить из одной части в другую.

$a = b + c \bmod n$

$a - c = b \bmod n$

Свойство 6. Сравнения можно почленно перемножать.

$a_1 = b_1 \bmod n$

$a_2 = b_2 \bmod n$

$a_1 \cdot a_2 = b_1 \cdot b_2 \bmod n$

Свойство 7.

Если $a = b \bmod n$ и d – делитель чисел a и b, причем так, что $a = a_1 \cdot d$, $b = b_1 \cdot d$ и НОД(d, m) = 1, то левую и правую часть сравнения можно сократить на d, т.е., если $a_1 \cdot d = b_1 \cdot d \bmod n$, то $a_1 = b_1 \bmod n$.

Свойство 8.

Если $a = b \bmod n$, a, b, n имеют общий делитель d – то на него можно сократить:

$a = a_1 \cdot d,\ b = b_1 \cdot d,\ n = n_1 \cdot d \quad => \quad a_1 = b_1 \bmod n_1$

Свойство 9.

Если $a = b \bmod n$ и $k = k \bmod n$, то все части исходного сравнения можно умножить на одно и тоже целое число k и получим:

$a \cdot k = b \cdot k \bmod n$.

Замечание. Если модулю сравнений не совпадают, то нельзя выполнять с ними совместные операции.

Пример 1.

$64 = 28 \bmod 9$

$16 \cdot 4 = 7 \cdot 4 \bmod 9$ (9, 4) = 1

$16 = 7 \bmod 9$

Пример 2.

$66 = 30 \bmod 18$

$11 \cdot 6 = 5 \cdot 6 \bmod 18$ $11 \neq 5 \bmod 18$, $(18, 6) \neq 1$

$11 \cdot 6 = 5 \cdot 6 \bmod 3 \cdot 6$

$11 = 5 \bmod 3$

Пример 3.

$11 = 17 \bmod 6$

$11 \cdot 4 = 17 \cdot 4 \bmod 6$
$44 = 68 \bmod 6$

Пример 4.
$11 = 17 \bmod 6$
$11 \cdot 4 = 17 \cdot 4 \bmod 6 \cdot 4$
$44 = 68 \bmod 24$

Введем понятие вычета.

Определение 2.7. Множество равноостаточных чисел по модулю n является классом чисел (понятие класса будет рассмотрено позднее), которые можно представить в виде: $a = b + n \cdot t$. Любое число класса называется *вычетом* по модулю n.

Другими словами, множество всех чисел, сравнимых с a по модулю m называется *классом вычетов* a по модулю n, *и* может быть обозначено $[a]_n$.

Таким образом, сравнение $a = b \bmod n$ равносильно равенству классов вычетов $[a]_n = [b]_n$.

Запись $a \bmod n$ представляет целое число в $\mathbb{Z}_n = \{0, \ldots, n-1\}$, где $\mathbb{Z}_n$ – система наименьших неотрицательных вычетов по модулю n.

Примеры. $\mathbb{Z}_2 = \{0,1\}$; $\mathbb{Z}_9 = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$; $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$.

Рассмотрение классов вычетов тесно связано с понятием эквивалентности.

Определение 2.8. Отношение эквивалентности на множестве $\mathbb{Z}_n$ — это бинарное отношение, для которого при любых элементов a_1, a_2, a_3 из $\mathbb{Z}_n$ выполнены следующие условия:

1. рефлексивность: a эквивалентно самому себе.
2. симметричность: если a_1 эквивалентно a_2, то a_2 эквивалентно a_1
3. транзитивность: если a_1 эквивалентно a_2 и a_2 эквивалентно a_3, то a_1 эквивалентно a_3.

Сравнение по модулю n является отношением эквивалентности на множестве целых чисел $\mathbb{Z}$, и, следовательно, классы вычетов по модулю n представляют собой классы эквивалентности и количество таких классов эквивалентности равно n.

Для обозначения множества всех классов вычетов по некоторому модулю n используют обозначения $\mathbb{Z}_n$ или $\mathbb{Z}/n\mathbb{Z}$ или $\mathbb{Z}/(n)$.

Пример. $[a]_n$ – набор всех чисел x, таких, что $x = a \bmod n$.
При $n = 7$ имеется 7 таких наборов:
$[0]_7 = \{\ldots, -21, -14, -7, 0, 7, 14, 21, \ldots\}$,
$[1]_7 = \{\ldots, -20, -13, -6, 1, 8, 15, 22, \ldots\}$,
$[2]_7 = \{\ldots, -19, -12, -5, 2, 9, 16, 23, \ldots\}$,
$[3]_7 = \{..., -18, -11, -4, 3, 10, 17, 24, \ldots\}$,
$[4]_7 = \{..., -17, -10, -3, 4, 11, 18, 25, \ldots\}$,
$[5]_7 = \{..., -16, -9, -2, 5, 12, 19, 26, \ldots\}$,
$[6]_7 = \{..., -15, -8, -1, 6, 13, 20, 27, \ldots\}$.

В каждом наборе есть один элемент, называемый наименьшим неотрицательным вычетом. Множество, которые показывает все наименьшие неотрицательные вычеты $\mathbb{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$.

Операции сложения и умножения на $\mathbb{Z}$ индуцируют соответствующие операции на множестве $\mathbb{Z}_m$:
$[a]_m + [b]_m = [a + b]_m$,
$[a]_m - [b]_m = [a - b]_m$,
$[a]_m \cdot [b]_m = [a \cdot b]_m$.

Свойства операций в модульной арифметике:
$(a + b) \bmod m = [(a \bmod m) + (b \bmod m)] \bmod m$,
$(a - b) \bmod m = [(a \bmod m) - (b \bmod m)] \bmod m$,
$(-b) \bmod m = (m - b) \bmod m$,
$(a \cdot b) \bmod m = [(a \bmod m) \cdot (b \bmod m)] \bmod m$,
$(a^k) \bmod m = [(a \bmod m)^k] \bmod m$.

Примеры использования свойств:
$(23876576528 + 3276749) \bmod 10 = (8+9) \bmod 10 = 7$,
$(29867349878 - 4356759) \bmod 10 = (8-9) \bmod 10 = 9$,
$(17233488 \cdot 2124949) \bmod 10 = (8 \cdot 9) \bmod 10 = 2$,
$(11^{20}) \bmod 10 = (11 \bmod 10)^{20} = 1^{20} = 1$.

Всего по модулю n существует n различных классов равноостаточных чисел (с остатками 0, 1, …, n–1).

Совокупность вычетов, произвольно выбранных по одному из каждого класса вычетов, называется *полной системой вычетов по модулю n*. В полной системе вычетов всего насчитывается n штук чисел.

Приведенная система вычетов по модулю n содержит элементы из каждого класса вычетов взаимно простых с модулем n. Обычно в приведенной системе вычетов указывают наименьшие неотрицательные вычеты $\{0, 1, \ldots, n - 1\}$. В приведенной системе существует $\varphi(n)$ различных вычетов и $\varphi(n)$

различных классов вычетов, так как именно функция Эйлера определяет количество чисел взаимно простых с числом n.

Пример. Определить систему вычетов для $n = 21$. Найдем $\varphi(21) = 12$.
Полная система наименьших неотрицательных вычетов:
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,19, 20}
Приведенная система наименьших неотрицательных вычетов:
{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20}

Для доказательства следующих теорем нам потребуется одно из свойств системы вычетов.

Свойство *: Если $(a, n) = 1$ и x пробегает приведенную систему вычетов (последовательно принимает значения из этой системы) по модулю n, то произведение чисел $a \cdot x$ тоже пробегает приведенную систему вычетов по модулю n.

Доказательство. Количество чисел $a \cdot x$ будет равным количеству чисел x, и будет равно числу $\varphi(n)$.

Предположим, что для некоторого x_1 произведение $a \cdot x_1$ не принадлежит приведенной системе вычетов, т.е. $(a \cdot x_1, n) \neq 1$. Следовательно существует некоторое целое число d, которое делит и $(a \cdot x_1)$ и n, но так как по условию $(a, n) = 1$, значит число d должно делить и x_1 и n, но по условию, так как x_1 принадлежит приведённой системе вычетов, то $(x_1, n) = 1$.

Получаем противоречие, следовательно, не существует такого числа x_1 в приведенной системе вычетов, для которого число $a \cdot x_1$ не принадлежит приведенной системе вычетов. ■

Теорема 2.10. *Теорема Эйлера*. Если числа a и n взаимно простые, т.е. $(a, n) = 1$, тогда справедливо:

$$a^{\varphi(n)} = 1 \bmod n,$$

где $\varphi(n)$ — функция Эйлера.

Доказательство. Пусть x пробегает приведенную систему вычетов по mod n: $x = r_1, r_2, \ldots, r_l$, где $l = \varphi(n)$ их количество.

Следовательно, наименьшие неотрицательные вычеты $\rho_1, \rho_2, \ldots, \rho_l$, соответствующие числам $a \cdot x$:

$$\rho_1 = (a \cdot r_1) \bmod n; \quad \rho_2 = (a \cdot r_2) \bmod n; \quad \ldots; \quad \rho_l = (a \cdot r_l) \bmod n;$$

согласно свойству (*) тоже пробегают эту же приведенную систему вычетов, но в каком-то другом порядке. Выполнив почленное умножение сравнений, получаем:

$$\rho_1 \cdot \rho_2 \cdot \rho_3 \cdot \ldots \cdot \rho_l = (a \cdot r_1) \cdot (a \cdot r_2) \cdot (a \cdot r_3) \cdot \ldots \cdot (a \cdot r_l) \bmod n,$$

$$\rho_1 \cdot \rho_2 \cdot \rho_3 \cdot \ldots \cdot \rho_l = a^l \cdot r_1 \cdot r_2 \cdot r_3 \cdot \ldots \cdot r_l \bmod n.$$

Разделив обе части на: $\rho_1 \cdot \rho_2 \cdot \rho_3 \cdot \ldots \cdot \rho_l$, которое равно (см. выше) произведению $r_1 \cdot r_2 \cdot r_3 \cdot \ldots \cdot r_l$, получаем:

$1 = a^l \bmod n \quad \rightarrow \quad a^l = 1 \bmod n \quad \rightarrow \quad a^{\varphi(n)} = 1 \bmod n$.■

Пример. Вычислить $3^6 \bmod 14$.

Вариант решения 1. $3^6 = 729$; $729 \bmod 3 = 1$.

Вариант решения 2. Так как φ (14)=6 и НОД(3,14)=1, то по теореме: $3^6 \bmod 14 = 1$.

Теорема 2.11. *Малая теорема Ферма*. Если число *p* простое и для чисел *a* и *p* справедливо (*a*, *p*) = 1 (числа взаимно простые), тогда справедливо:

$$a^{p-1} = 1 \bmod p.$$

Данная теорема является следствием доказанной ранее теоремы Эйлера при *n* = *p*, где *p* простое число. Малая теорема Ферма может быть доказана и отдельно.

Пример. Вычислить $9^{111} \bmod 5$.

Так как НОД(9,5) = 1 и φ(5) = 4, то можем провести ряд преобразований, выделяя структуры вида $9^4 \bmod 5$, которые согласно теореме равны единице:

$9^{111} \bmod 5 = (9^4)^{27} \cdot 9^3 \bmod 5 = (1)^{27} \cdot 9^3 \bmod 5$, так как: $9 \bmod 5 = 4$, то $9^3 \bmod 5 = 4^3 \bmod 5 = 16 \cdot 4 \bmod 5 = 4 \bmod 5$.

2.7. Модульная арифметика. Инверсии

Введем понятие инверсий.

Определение 2.9. *Аддитивная инверсия* (*противоположное*) – нахождение элемента, вычета (представителя класса эквивалентности) обратного по сложению. В модульной арифметике два числа *a* и *b* называют аддитивно инверсными друг другу, если выполняется равенство $b = n - a \bmod n$ или

$$a + b = 0 \bmod n.$$

Например, аддитивная инверсия числа 3 в $\mathbb{Z}_{11}$ равна 11 – 3 = 8.

Каждое целое число имеет аддитивную инверсию, причем только одну. Как следует из определения сумма целого числа с его аддитивной инверсией сравнима с нулем по модулю *n*. Аддитивная инверсия числа может совпадать с самым числом.

Пример. Все пары инверсные по сложению (аддитивные инверсии) в $\mathbb{Z}_{11}$: (0, 0), (1, 10), (2, 9), (3, 8), (4, 7), (5, 6). В этом списке 0 — инверсия самому себе.

Определение 2.10. *Мультипликативная инверсия* (*обратное*) – нахождение элемента, вычета (представителя класса эквивалентности) обратного по умножению. В $\mathbb{Z}_n$ два числа a и b являются мультипликативной инверсны друг другу, если

$$a \cdot b = 1 \bmod n.$$

В литературе для мультипликативной инверсии числа a часто использую обозначение a^{-1}, т.е. $a \cdot a^{-1} = 1 \bmod n$.

Замечание. В модульной арифметике число a имеет мультипликативную инверсию в $\mathbb{Z}_n$, только если числа a и n взаимно простые, $(n, a) = 1$.

Пример. Найти мультипликативную инверсию числа 10 в $\mathbb{Z}_{12}$: Мультипликативная инверсия не существует, потому что НОД(12,10) ≠ 1. Другими словами, невозможно найти число в промежутке от 0 до 11, такое, что при умножении этого числа на 8 результат будет сравним с 1 по модулю 12.

Пример. Найти все мультипликативные инверсии в $\mathbb{Z}_{12}$: (1, 1), (5, 5), (7, 7), (11,11). Числа a = 0, 2, 3, 4, 6, 8, 9, 10 не имеют мультипликативной инверсии, т.к. для них НОД(a,12) ≠ 1.

Пример. Найти все мультипликативные инверсии в $\mathbb{Z}_{11}$: Имеем семь пар: (1, 1), (2, 6), (3, 4), (5, 9), (7, 8), (9, 9) и (10, 10). При переходе от Z_{12} к Z_{11} число пар увеличивается. При $\mathbb{Z}_{11}$ НОД (11, a) = 1 (взаимно просто) для всех значений a, кроме 0. Таким образом все целые числа от 1 до 10 по модулю 11 имеют мультипликативные инверсии.

Введем обозначение $\mathbb{Z}_n^*$, как множество элементов $\mathbb{Z}_n$, имеющих мультипликативную инверсию (мультипликативная группа $\mathbb{Z}_n$):

Z_8={0,1,2,3,4,5,6,7}	Z_8*={1,3,5,7}
Z_{11}={0,1,2,3,4,5,6,7,8,9,10}	Z_{11}*={1,2,3,4,5,6,7,8,9,10}
Z_{12}={0,1,2,3,4,5,6,7,8,9,10,11}	Z_{12}*={1,5,7,11}

Если число p является простым, то $\mathbb{Z}_p = \mathbb{Z}_p^* \cup \{0\}$ (добавление нулевого элемента).

Контрольные вопросы

Приведите свойства делимости.

На какие три множества можно разделить все натуральные числа?

Какова примерная вероятность того, что случайно выбранное число m является простым?

Какое минимальное количество простых числе может быть на промежутке от m до $4 \cdot m$?

Приведите формулировку основной теоремы арифметики.

Приведите примеры нескольких трехзначных чисел и их канонические разложения.

Докажите, что у любого составного числа имеется делитель, который не превосходит корня из этого числа.

Приведите пример некратных чисел, наибольший общий делитель которых равен трём.

Приведите пример чисел наибольший общий делитель которых равен единице.

Дайте определение взаимно простых чисел и приведите примеры.

Какие существуют алгоритмы нахождения для двух чисел a и b значения (a, b).

Приведите общий вид соотношения Безу.

Дайте определение функции Эйлера.

Приведите пример непрерывной дроби.

Поясните связь между построением непрерывной дроби и алгоритмом Евклида.

Приведите пример чисел сравнимых по модулю и дайте определение таких чисел.

Поясните особенность почленного сокращения сравнений в модульной арифметике.

Поясните понятие вычета и класса вычетов. Приведите пример.

Сколько существует классов вычетов по заданному модулю m?

Приведите формулировку теоремы Эйлера.

Приведите малую теорему Ферма и докажите ее при наличии доказанной теоремы Эйлера.

Приведите определение аддитивной инверсии (противоположного элемента по модулю).

Приведите определение мультипликативной инверсии (обратного элемента по модулю).

Приведите пример числа, для которого по заданному модулю не существует.

3. КИТАЙСКАЯ ТЕОРЕМА ОБ ОСТАТКАХ

В теории чисел особое место занимает так называемая *китайская теорема об остатках*, для которой даже используют общепринятое сокращение (КТО). На основе этой теоремы можно восстановить целое число по заданному множеству остатков от деления исходного целого числа на числа из некоторого набора попарно взаимно простых чисел.

Теорема впервые была приведена в математическом трактате китайского ученого Сунь Цзы «Сунь Цзы Суань Цзин» (~ 3 век н.э.). Затем, через тысячу лет, была обобщена Цинь Цзюшао в книге «Математические рассуждения в 9 главах» (~ 13 век н.э.).

Теорема 3.1. Если заданные натуральные числа $m_1, m_2, \ldots, m_m$ попарно взаимно просты, то для любых целых $a_1, a_2, \ldots, a_m$ таких, что $0 < a_i < m_i$ при всех $i \in \{1, 2, \ldots, m\}$, найдётся число x, которое при делении на m_i даёт остаток a_i при всех $i \in \{1, 2, \ldots, m\}$.

Более того, если найдутся два таких числа x_1 и x_2, то:

$$x_1 = x_2 \bmod (M).$$

где $M = m_1 \cdot m_2 \cdot \ldots \cdot m_m$.

Другими словами, пусть НОД(m_i, m_j) = 1 для всех i≠j, тогда система:

$$\begin{cases} x = a_1 \bmod m_1 \\ x = a_2 \bmod m_2 \\ \cdots \\ x = a_n \bmod m_n \end{cases}$$

имеет решение.

Если два числа x_1 и x_2 являются решением данной системы, то они удовлетворяют сравнению:

$$x_1 = x_2 \bmod M,$$

т.е. решение единственно на промежутке от 0 до M.

Доказательство. Единственность решения, т.е. докажем, что $x_1 = x_2 \bmod M$.

Предположим, что есть два решения системы x_1 и x_2. Обозначим $y = x_1 - x_2$, тогда:

$$\begin{cases} y = 0 \bmod m_1 \\ y = 0 \bmod m_2 \\ \cdots \\ y = 0 \bmod m_n \end{cases}$$

следовательно, $y = 0 \bmod M$, так как m_i – взаимно простые и $M = m_1 \cdot m_2 \cdot \ldots \cdot m_m$. Отсюда и следует, что $x_1 = x_2 \bmod M$. ■

Доказательство. Существование решения. Для доказательства построим явно решение системы.

Вычислим произведение модулей M. $M = m_1 \cdot m_2 \cdot \ldots \cdot m_m$.

Для каждого i вычислим произведение всех модулей, кроме mi:

$$M_i = M/m_i .$$

Так как все m_i попарно взаимно простые, то числа M_i и m_i взаимно простые.

Вычислим для каждого i обратный элемент $M_i = M_i^{(-1)}$ для M_i по модулю m_i, такой что:

$$M_i \cdot M_i = 1 \bmod m_i.$$

Обратный элемент всегда существует, так как M_i и m_i взаимно простые.

Итоговое решение системы задается соотношением:

$$x = a_1 M_1 N_1 + a_2 M_2 N_2 + \cdots + a_n M_n N_n = \sum_{i=1}^{k} a_i M_i N_i,$$

Проверим, что данное выражение для x является решением всех исходных уравнений системы. Рассмотрим j-ое уравнение системы:

$$x = a_j \bmod m_j$$

Подставим в него выражение для x и получим:

$$x = \sum_{i=1}^{k} a_i M_i N_i = a_j \bmod m_j.$$

Отметим, что все слагаемые $a_i M_i N_i$ при $i \neq j$ равны нулю по модулю m_j, так как M_i делится на m_j при $i \neq j$, отсюда получаем:

$$x = a_j M_j N_j = a_j \bmod m_j,$$

но, так как $M_i N_i = 1 \bmod m_j$:

$$x = a_j \cdot 1 = a_j \bmod m_j$$

и сравнение справедливо. В итоге мы доказали, что данное представление x является решением каждого из уравнений системы и получили способ решения таких систем. ■

Пример. Дана система уравнений:

$$\begin{cases} x = 1 \bmod 2 \\ x = 2 \bmod 3 \\ x = 6 \bmod 7 \end{cases}$$

Находим произведение модулей уравнений: 2·3·7=42. Следовательно система имеет единственное решение на промежутке [0... 41].

Найдем это решение (перебором):

решения первого уравнения от 1 до 41: 1, 3, 5, 7, 9, 11, ... 37, 39 ,41

решения второго уравнения от 1 до 41: 2, 5, 8, 11, ... 38, 41

решения третьего уравнения от 1 до 41: 6, 13, 20, 27, 34, 41

Итоговое решение: $x = 41$ (единственное значение, удовлетворяющее всем уравнениям на промежутке [0…41]). Однако переборный алгоритм не является эффективным.

Рассмотрим аналитический метод решения системы. Пусть дана система:

$$\begin{cases} x = 1 \bmod 13 \\ x = 4 \bmod 15 \\ x = 8 \bmod 19 \end{cases}$$

Будем решать методом подстановки, т.е. находить x из первого уравнения, результат подставлять во второе и т.д.

Рассмотри первое уравнение: $x = 1 \bmod 13$. Раскроем оператор mod, тогда уравнение можно записать в виде: $x = 1 + 13 \cdot t$, где t – целое.

Подставляем полученный результат во второе уравнение: $1 + 13 \cdot t = 4 \bmod 15$. Раскроем оператор модуля во втором уравнении $13 \cdot t = 3 \bmod 15$. Найдем значение t: $t = 3 \cdot (13)^{(-1)} \bmod 15 = 3 \cdot 7 \bmod 15$, так как $(13)^{(-1)} = 7 \bmod 15$. В итоге $t = 6 \bmod 15$. Раскроем оператор модуля и получим: $t = 6 + 15 \cdot u$. Подставим это значение в выражение для x и получим: $x = 1 + 13 \cdot t = 1 + 13 \cdot (6 + 15 \cdot u) = 79 + 195 \cdot u$.

Подставляем полученный результат во третье уравнение: $79 + 195 \cdot u = 8 \bmod 19$. Выполняем преобразования, получаем: $u = 1 \bmod 19$, раскрываем оператор mod и получаем: $u = 1 + 19 \cdot v$.

Следовательно, итоговое решение: $x = 79 + 195 \cdot u = 79 + 195\,(1 + 19 \cdot v) = 274 + 3705 \cdot v$, где $v \in \mathbb{Z}$, т.е. x может принимать значения: …, –3431, 274, 3979, 7684, … (при v = …, –2, –1, 0, 1, 2, …). M=13·15·19=3705.

Теперь рассмотрим пример решения данной системы на основе самой китайской теоремы об остатках (см. доказательство существования решения):

$$\begin{cases} x = 2 \bmod 3 \\ x = 3 \bmod 5 \\ x = 2 \bmod 7 \end{cases}$$

Найдем произведение модулей: M=3·5·7=105.

Для каждого уравнения определим произведение модулей без текущего: M_1 = 105/3=35, M_2 = 105/5 = 21, M_3 = 105/7 = 15.

Найдем для каждого Mi обратное число Mi (в общем случае необходимо использовать расширенный алгоритм Евклида):

M_1 =2 (так как $M_1 \cdot M_1 = 35 \cdot 2 = 1 \bmod 3$), M_2 =1, M_3 =1.

Найдём решение с точностью до mod 105:

$$x = a_1 M_1 N_1 + a_2 M_2 N_2 + a_3 M_3 N_3 =$$
$$= 2 \cdot 35 \cdot 2 \; + \; 3 \cdot 21 \cdot 1 \; + \; 2 \cdot 15 \cdot 1 \; = \; 23 \bmod 105.$$

Китайская теорема об остатках позволяет производить вычисления не с самими числами, которые могут весьма велики, а с их остатками при делении на некоторый набор взаимно простых чисел. После выполнения вычислений итоговое значение может быть определено по остаткам. Работа с остатками может быть более эффективной.

Рассмотрим пример, необходимо вычислить: $x = 39^{43} \bmod 77$. Так как модуль 77 разлагается на множители 7 и 11, то запишем систему уравнений:

$$\begin{cases} x = 39^{43} \bmod 7 \\ x = 39^{43} \bmod 11 \end{cases}$$

Упростим:

$$\begin{cases} x = 4^{43} \bmod 7 \\ x = 6^{43} \bmod 11 \end{cases}$$

Перепишем:

$$\begin{cases} x = (4^6)^7 \cdot 4 \bmod 7 \\ x = (6^{10})^4 \cdot 6^3 \bmod 11 \end{cases}$$

Вычислим, с учетом малой теоремы Ферма:

$$\begin{cases} x = 4 \bmod 7 \\ x = 7 \bmod 11 \end{cases}$$

Находим *x* по китайской теореме об остатках:
$M = 7 \cdot 11 = 77$; $M_1 = 77/7 = 11$; $M_2 = 77/11 = 7$
$N_1 = M_1^{-1} = 11^{-1} = 4^{-1} = 2 \bmod 7$
$N_2 = M_2^{-1} = 7^{-1} = 8 \bmod 11$

$x = 4 \cdot 11 \cdot 2 + 7 \cdot 7 \cdot 8 = 11 + 7 = 18 \bmod 77$

Ответ: $39^{43} =$ **18** mod 77.

Контрольные вопросы

Какую задачу позволяет решить китайская теорема об остатках?

Сколько решений существует для системы сравнений в китайской теореме об остатках.

Что утверждается об единственности решения в китайской теореме об остатках.

Каким образом восстанавливать число по остаткам, приведите пример.

Всегда ли возможно восстановить число по остаткам?

Для решения каких задач может использоваться китайская теорема об остатках?

Приведите пример решения задачи с помощью китайской теоремы об остатках.

4. РЕШЕНИЕ СРАВНЕНИЙ

4.1. Линейные сравнения с одним неизвестным

Определение 4.1. Пусть

$$f(x) = a_0 \cdot x^n + a_1 \cdot x^{n-1} + a_2 \cdot x^{n-2} + \cdots + a_0 \cdot x^n,$$

где: $a_0, a_1, \ldots, a_n \in \mathbb{Z}$, $n \in \mathbb{Z}$, x – некоторая переменная величина. Функция *f(x)* называется степенным многочленом. Если $a_0 \neq 0$, *f(x)=Q_m(x) – многочлен степени m.*

Будем рассматривать этот многочлен на множестве целых чисел.

Рассмотрим сравнение вида:

$$f(x) = 0 \bmod n \qquad (4.1)$$

Решить сравнение – значит найти все целые значения x, ему удовлетворяющие, т.е. такие значения x, при подстановке которых в левую часть сравнения, мы получим верное числовое сравнение.

Если сравнению (4.1) удовлетворяет какое-либо $x = x_1$, то тому же сравнению будут удовлетворять и все числа сравнимые с x_1, по модулю m: $x = x_1 \bmod m$.

Рассмотрим сравнение: $a \cdot x + b \equiv 0 \bmod m$, его можно переписать в виде:

$$a \cdot x \equiv b \bmod m$$

С точки зрения количества решений такого уравнения (число решений в полной системе вычетов) возможны следующие варианты:

Вариант 1. НОД $(a, m) = 1$, то сравнение имеет одно решение.

Вариант 2. НОД $(a, m) = d$, то:

2.1. d не делит b (b не кратно d): решений нет.

Пример: $4x+3=0 \bmod 6$, решения нет среди всех возможных значений x по mod 6: 0, 1, 2, 3, 4, 5.

2.2. d делит b нацело: d решений.

Рассмотрим последний случай подробнее. Пусть $a = a_1 \cdot d$, $b = b_1 \cdot d$, $m = m_1 \cdot d$, тогда $a_1 \cdot d \cdot x = b_1 \cdot d \bmod m_1 \cdot d$. В итоге получаем: $a_1 \cdot x = b_1 \bmod m_1$.

Если НОД$(a_1, m_1) = 1$ существует единственное решение сравнения по модулю m_1:

$$x = x_1 \bmod m_1, \text{ т.е. } x = x_1 + t\, m_1$$

По модулю m для исходного сравнения существует не одно решение, а столько решений, сколько чисел вида $x_1 + t \cdot m_1$ найдется в ряде $0, 1, \ldots, m - 1$.

Числа $x_1 + 0 \cdot m_1$, $x_1 + 1 \cdot m_1$, …, $x_1 + (d - 1) \cdot m_1$, являются решениями сравнения по модулю m.

Пример. Решить сравнение $9 \cdot x = 6 \bmod 21$.

Находим наибольший общий делитель d = НОД(21,9) = 3. Записываем сравнение в виде: $(9/3) \cdot x_1 = (6/3) \bmod (21/3)$; $\Rightarrow 3 \cdot x_1 = 2 \bmod 7$.

Решение данного сравнения: $x_1 = 3$ (способ решения будет показан далее). Для исходного сравнения решениями будут числа: $x = 3$, $3 + 7$, $3 + 2 \cdot 7 \Rightarrow 3, 10, 17$.

Проверим, например, $x = 17$: $9 \cdot 17 = 153 = 6 \bmod 21$.

Способ решения 1. Согласно теореме Эйлера, если НОД(a,m)=1, то : $a^{\varphi(m)} = 1 \bmod m$. Следовательно, справедливо и такое выражение: $a^{\varphi(m)}b = b \bmod m$.

Пусть необходимо решить уравнение: $a \cdot x = b \bmod m$.

Так как: $a^{\varphi(m)} = 1 \bmod m$, следовательно $a \cdot (a^{\varphi(m)-1}) = 1 \bmod m$. Тогда справедливо: $a \cdot (a^{\varphi(m)-1}) \cdot b = b \bmod m$, сравнивая данное выражение с исходным, получаем:

$$x = a^{\varphi(m)-1} \cdot b \bmod m.$$

Пример. $3 \cdot x = 4 \bmod 5$; Определим НОД(3, 5) = 1. Следовательно можно применить теорему Эйлера. Определим функцию Эйлера от числа 5:φ(5) = 4.

Находим значение x: $x = 3^{\varphi(5)-1} \bmod 5 = 3^{(4-1)} \bmod 5 =$

$= 27 \cdot 4 \bmod 5 = 2 \cdot 4 \bmod 5 = 3$;

Ответ: $x = 3$.

Проверка: $3 \cdot x \bmod 5 = 3 \cdot 3 \bmod 5 = 9 \bmod 5 = 4$.

Способ решения 2. На практике же используют другой метод решения сравнений первой степени, так как рассмотренный выше требует возведения в степень, что хоть и выполняется алгоритмом полиномиальной сложности, но может потребовать значительных вычислительных ресурсов.

Еще раз рассмотрим сравнение $a \cdot x = b \bmod m$. Для определения возможности его решения находим НОД$(a, m) = d$. Если b не кратно d, то у сравнения нет решений. Если b кратно d, то у сравнения существует единственное решение по модулю m/d, или, что то же самое, d решений по модулю m.

В этом случае в результате сокращения исходного сравнения на d получается сравнение: $a_1 \cdot x = b_1 \bmod m_1$, где $a_1 = a/d$, $b_1 = b/d$, $m_1 = m/d$ – целые числа, a_1 и m – взаимно простые.

Следовательно число a_1 можно обратить по модулю m_1, т.е. найти такое число c, что $c \cdot a_1 = 1 \bmod m_1$:

$$c = a_1^{-1} \bmod m_1.$$

Тогда решение для x: $x = c \cdot a_1 \cdot x = c \cdot b_1 = a_1^{-1} \cdot b_1 \bmod m_1$. Решениями же исходного уравнения будут все числа вида: $x + k \cdot m_1 < m$, где k – целое число.

Для нахождения решения сравнения $a \cdot x = b \bmod m$ используют представление НОД $(a, m) = 1$ в виде линейной комбинации чисел a и m (Соотношение Безу). Действительно, используя расширенный алгоритм Евклида, представляем наибольший общий делитель «1» в виде:

$$1 = \alpha \cdot a + \beta \cdot m.$$

Умножая последнее равенство на b, получаем:

$$b = \alpha \cdot a \cdot b + \beta \cdot m \cdot b$$

Если теперь привести выражение по модулю «m», получаем:

$$\alpha \cdot a \cdot b = b \bmod m$$
$$a \cdot \boldsymbol{\alpha} \cdot \boldsymbol{b} = b \bmod m$$

а так как исходное соотношение: $a \cdot x = b \bmod m$, то число $x = \boldsymbol{\alpha} \cdot \boldsymbol{b} = \boldsymbol{a}^{-1} \cdot \boldsymbol{b}$ будет решением этого исходного сравнения.

Пример. $6 \cdot x = 26 \bmod 22$. НОД(6, 22) = 2, т.е. существуют два решения по mod 22. Упростим:

$$6 \cdot x = 26 \bmod 22;$$
$$3 \cdot x = 13 \bmod 11;$$
$$3 \cdot x = 2 \bmod 11;$$
$$x = 3^{-1} \cdot 2 \bmod m_1.$$

Находим: $3^{-1} = 1 \bmod 11$. $3^{-1} = 4$ (так как $3 \cdot 4 = 1 \bmod 11$).

$x = 4 \cdot 2 \bmod 11 = 8 \bmod 11$.

Решениями будут числа $8 + k \cdot 11$, k = 0, 1 (два решения, не превосходящие модуль исходного уравнения, числа 22), т.е. два решения $x = \mathbf{8} \bmod 22$ и $x = \mathbf{19} \bmod 22$.

4.2. Сравнения степени m

Рассмотрим сравнения степени $n > 1$. Будем рассматривать простейший случай – двучленные сравнения:

$$x^n = a \bmod p, \quad \text{НОД}(a, p) = 1$$

Если данное сравнение имеет решения, то число a называется вычетом степени n по модулю p.

Если $n = 2$ вычеты и невычеты называются *квадратичными*.

Если $n = 3$ вычеты и невычеты называются кубическими.

Если $n = 4$ вычеты и невычеты называются биквадратными.

В дальнейшем изложении будем рассматривать сравнения второй степени:

$$x^2 = a \bmod p;\ \text{НОД}(a, p) = 1 \qquad (4.2)$$

Отметим, что любое выражение вида $x^2 + a \cdot x + b \bmod p$ может быть преобразовано к виду $x^2 = a \bmod p$ с помощью замены переменных.

Если число a является квадратичным вычетом по модулю p и число p – простое, то сравнение (4.2) имеет два решения.

Действительно, если a является квадратичным вычетом, то сравнение (4.2) имеет, по крайней мере, одно решение $x = x_1 \bmod p$, тогда ввиду $(-x_1)^2 = x_1^2$, это же сравнение имеет и второе решение $x = -x_1 \bmod p$, которое не совпадает с первым, так как из $x_1 = -x_1 (\bmod p) \rightarrow 2x_1 = 0 = p \bmod p$, что невозможно, так как в этом случае p имеет делитель x_1 и $a = x_1^2 \bmod p$ имеет делитель x_1, а по условию НОД$(a, p) = 1$, НОД$(2, p) =$ НОД$(x_1, p) = 1$.

Вывод: если p – простое, то у сравнения (4.2) не может быть более двух решений, так сравнение второй степени более двух решений иметь не может.

Приведенная система вычетов по модулю p состоит из $(p–1)/2$ квадратичных вычетов, сравнимых с числами:

$$1^2, 2^2, 3^2, \dots, \qquad \left(\frac{p-1}{2}\right)^2 \qquad (4.3)$$

и из $(p–1)/2$ квадратичных невычетов.

Действительно, среди вычетов из приведенной системы вычетов по модулю p квадратичными вычетами являются те и только те вычеты, которые сравнимы с квадратами чисел (приведенной системы вычетов), то есть следующими числами из (4.3) $-\frac{p-1}{2}, \dots, -2, -1, 1, 2, \dots, \frac{p-1}{2}$. При этом числа в (4.3) по модулю p не сравнимы.

4.3. Символы Лежандра и Якоби

Определение 4.1. *Символ Лежандра*, для целого a и простого p, записывается в виде

$$\left(\frac{a}{p}\right)$$

и определяется для всех a, не делящихся на p:

$\left(\frac{a}{p}\right) = 0$, если a делится p без остатка;

$\left(\frac{a}{p}\right) = 1$, если a является квадратичным вычетом по модулю p;

$\left(\frac{a}{p}\right) = -1$, если a является квадратичным невычетом по модулю p.

Вычислить символ Лежандра позволяет *критерий Эйлера* (при a, не делящемся на p). Одновременно это позволит определить, является ли число a квадратичным вычетом или квадратичным невычетом по модулю p:

$$a^{\frac{p-1}{2}} = \left(\frac{a}{p}\right) \bmod p.$$

Теорема 4.1. Критерий Эйлера. Пусть число p является простым и $p > 2$. Число a, для которого $(a, p) = 1$, является квадратичным вычетом тогда и только тогда, когда

$$a^{\frac{p-1}{2}} = 1 \mod p.$$

Доказательство. Пусть число a является квадратичным вычетом по простому модулю p. Тогда всегда должно выполняться сравнение $x^2 = a \bmod p$. Следовательно справедливо: $a^{(p-1)/2} = x^{p-1} = 1 \bmod p$. ■

Пример. Определить является ли число 7 квадратичным вычетом по простому модулю 31.

$$\left(\frac{a}{p}\right) = \left(\frac{7}{31}\right) = a^{\frac{p-1}{2}} = 7^{\frac{31-1}{2}} = 7^{15} = 1 \bmod 31$$

Следовательно, 7 является квадратичным вычетом по простому модулю 31 и сравнение $x^2 = 7 \pmod{31}$ имеет два решения.

Пример. Определить является ли число 11 квадратичным вычетом или невычетом по простому модулю 31

$$\left(\frac{a}{p}\right) = \left(\frac{11}{31}\right) = a^{\frac{p-1}{2}} = 11^{\frac{31-1}{2}} = 11^{15} = 30 = -1 \bmod 31$$

Следовательно, число 11 является квадратичным невычетом по модулю 31 и сравнение $x^2 = 11 \bmod 31$ решений не имеет.

Перечислим основные свойства символа Лежандра.

1. Если $a \equiv a_1 \bmod p$, то $\left(\frac{a}{p}\right) = \left(\frac{a_1}{p}\right)$

2. $\left(\frac{1}{p}\right) = 1$. Действительно, 1 = 12, следовательно, 1 – квадратичный вычет

3. $\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}} = \begin{cases} 1, p = 1 \bmod 4 \\ -1, p = 3 \bmod 4 \end{cases}$, следствие из теоремы при a = –1.

4. $\left(\frac{a \cdot b \cdot \cdots \cdot l}{p}\right) = \left(\frac{a}{p}\right) \cdot \left(\frac{b}{p}\right) \cdot \cdots \cdot \left(\frac{l}{p}\right)$. Следствие $\left(\frac{ab^2}{p}\right) = \left(\frac{a}{p}\right)$.

5. $\left(\frac{2}{p}\right) = (-1)^{\frac{p^2-1}{8}} = \begin{cases} 1, p = \pm 1 \bmod 8 \\ -1, p = \pm 3 \bmod 8 \end{cases}$

6. Если p и q простые и нечетные, то (закон взаимности квадратичных вычетов):

$$\left(\frac{q}{p}\right) = (-1)^{\frac{p-1}{2} \cdot \frac{q-1}{2}} \left(\frac{p}{q}\right)$$

Откуда следует, что если $p = 1 \bmod 4$ или $q = 1 \bmod 4$, то знак сохраняется, иначе – меняется.

Для вычисления символа Лежандра можно воспользоваться следующими шагами:

1. Если a – чётно: $\left(\frac{2a}{p}\right) = \left(\frac{2}{p}\right)\left(\frac{a}{p}\right)$; $\left(\frac{2}{p}\right) = (-1)^{\frac{p^2-1}{8}}$; $\left(\frac{4a}{p}\right) = \left(\frac{a}{p}\right)$.

2. Если $a > p$, $a = p \cdot q + r$, тогда: $\left(\frac{a}{p}\right) = \left(\frac{r}{p}\right)$ и $r < p$.

3. Если $a < p$, используем свойство взаимности: $\left(\frac{a}{p}\right) = (-1)^{\frac{p-1}{2} \cdot \frac{a-1}{2}} \left(\frac{p}{a}\right)$ и $a > p$.

4. При получении $\left(\frac{1}{p}\right)$, используем свойство: $\left(\frac{1}{p}\right) = 1$

5. При получении $\left(\frac{-1}{p}\right)$, используем свойство: $\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}}$.

Для рассмотрения более общих случаев используется символ Якоби. Символ Якоби является обобщением уже известного нам символа Лежандра на составное P.

Определение 4.2. Пусть P – нечетное, составное число большее единицы, и известно его разложение на простые сомножители $P = p_1 \cdot p_2 \cdot \ldots \cdot p_r$. Пусть (a, P) = 1, тогда *символ Якоби* может быть определен равенством:

$$\left(\frac{a}{P}\right) = \left(\frac{a}{p_1}\right) \cdot \left(\frac{a}{p_2}\right) \cdot \left(\frac{a}{p_3}\right) \cdot \ldots \cdot \left(\frac{a}{p_r}\right)$$

Перечислим основные свойства символа Якоби:

1. Если $a \equiv a_1 (\mathrm{mod}\ P)$, то $\left(\frac{a}{P}\right) = \left(\frac{a_1}{P}\right)$
2. $\left(\frac{1}{P}\right) = 1$
3. $\left(\frac{-1}{P}\right) = (-1)^{\frac{P-1}{2}}$
4. $\left(\frac{a \cdot b \cdot \ldots \cdot l}{P}\right) = \left(\frac{a}{P}\right) \cdot \left(\frac{b}{P}\right) \cdot \ldots \cdot \left(\frac{l}{P}\right)$, следствие: $\left(\frac{a \cdot b^2}{P}\right) = \left(\frac{a}{P}\right)$
5. $\left(\frac{2}{P}\right) = (-1)^{\frac{P^2-1}{8}}$
6. Если P и Q простые, нечетные и взаимно простые, то выполняется:

$$\left(\frac{Q}{P}\right) = (-1)^{\frac{P-1}{2} \cdot \frac{Q-1}{2}} \left(\frac{P}{Q}\right)$$

Символ Лежандра, как и символ Якоби, может быть вычислен на основе их свойств. Такой метод вычисления позволяет произвести вычисления символа Лежандра за меньшее количество операций, чем по критерию Эйлера (теореме):

$$\left(\frac{219}{383}\right) = (-1)^{\frac{219-1}{2} \cdot \frac{383-1}{2}} \left(\frac{383}{219}\right) = -\left(\frac{383}{219}\right) = -\left(\frac{164}{219}\right) = -\left(\frac{41}{219}\right) =$$

$$= \left(\frac{219}{41}\right) = -\left(\frac{14}{41}\right) = -\left(\frac{2}{41}\right)\left(\frac{7}{41}\right) = -\left(\frac{7}{41}\right) = -\left(\frac{41}{7}\right) = -\left(\frac{-1}{7}\right) = 1$$

Следовательно, чисто 219 является квадратичным вычетом по модулю 383 и сравнение $x^2 = 219 \bmod 383$ имеет два решения.

4.4. Вычисление квадратного корня

4.4.1. Вычисление квадратного корня по простому модулю

Пусть p нечетное простое число и a – целое число, такое, что $(a, p) = 1$.

Если квадратичное сравнение $x^2 = a \bmod p$ имеет решения, т.е. $a^{(p-1)/2} = 1 \bmod p$, то корни квадратичного сравнения могут быть найдены следующим образом:

1) если $p = 3 \bmod 4$, то $x = a^{\frac{p+1}{4}} \bmod p$;

2) если $p = 5 \bmod 8$, то:

если $a^{\frac{p-1}{4}} = 1 \bmod p$, то $x = a^{\frac{p+3}{8}} \bmod p$,

если $a^{\frac{p-1}{4}} = -1 \bmod p$, то $x = 2a\,(4a)^{\frac{p-5}{8}} \bmod p$ или $x = 2^{\frac{p-1}{4}} a^{\frac{p+3}{8}} \bmod p$;

3) если $p \equiv 1 \bmod 4$ (этот случай включает предыдущие), то нахождение квадратичного корня производится по несколько более сложному, но имеющему полиномиальную сложность, алгоритму Тоннели-Шенкса:

3.1) Представим $p - 1$ в виде: $p - 1 = 2^s \cdot q$, где q — нечётное целое число, и $s \geq 1$.

3.2) Производим поиск квадратичного невычета. Определяем число z, которое будет являться квадратичным невычетом по модулю p. Число z находится случайным перебором, проверяя условие: $z^{(p-1)/2} = -1 \bmod p$.

3.3) Определим вспомогательные величины:

$c = z^q \bmod p$, $t = a^q \bmod p$, $R = a^{(p-1)/2} \bmod p$, $M = s$.

4) Выполняем в цикле:

если t = 1, то выходим из цикла, в противном случае:

перебирая i (0 < i < M) ,… находим наименьшее i, такое что $t^{2^i} = 1 \bmod p$.

Далее вычисляем $b = c^{2^{M-i-1}} \bmod p$ и обновляем переменные:

$$c = b^2 \bmod p,\ t = t \cdot b^2 \bmod p,\ R = R \cdot b \bmod p,\ M = i.$$

5) После завершения цикла первое решение сравнения: R, другое решение: $p–R$.

Пример. Решим сравнение $x^2 = 2 \bmod 17$, т.е. $p = 17$, $a = 2$

$$p = 1 \bmod 4$$

Проверим, является ли число 2 квадратичным вычетом по модулю 17:

$2^{(17-1)/2} = 2^8 = 1 \bmod 17$. Значит 2 квадратичный вычет по критерию Эйлера и существует решение сравнения.

Представим $p - 1$ в виде: $q \cdot 2^s$

$p = 17$: $p - 1 = 16 = 1 \cdot 2^4$, $q = 1$, $s = 4$.

Найдем квадратичный невычет перебором чисел от 2 до $p - 1$. Число 2 мы уже проверили, что не является невычетом, проверим число $z = 3$:

$3^{(17-1)/2} = 3^8 = 169 = -1 \bmod 17$.

Значит 3 квадратичный невычет, $z = 3$.

Зададим начальные значения переменных:

$c = z^q = 3^1 = 3 \bmod 17$

$t = a^q = 2^1 = 2 \bmod 17$

$R = a^{(q+1)/2} = 2^{(1+1)/2} = 2 \bmod 17$

$M = s = 4$

$t \neq 1$, следовательно выполняем первую итерацию.

Находим наименьшее i, такое что $t^{2^i} = 2^{2^i} = 1 \bmod 17$
$t^{2^0} = 2^1 = 2 \bmod 17$,
$t^{2^1} = 2^2 = 4 \bmod 17$,
$t^{2^2} = 2^4 = 16 \bmod 17$,
$t^{2^3} = 2^8 = 1 \bmod 17$.

Таким образом, $i = 3$.

$$b = c^{2^{M-i-1}} = 3^{2^{4-3-1}} = 3^{2^0} = 3 \bmod 17$$

Обновляем переменные:
$c = b^2 = 3^2 = 9 \bmod 17$
$t = t \cdot b^2 = 2 \cdot 3^2 = 2 \cdot 9 = 18 = 1 \bmod 17$
$R = R \cdot b = 2 \cdot 3 = 6 \bmod 17$
$M = i = 3$

$t = 1 \bmod 17$, следовательно цикл завершен.

Результат.
Первое решение: $x = R = 6 \bmod 17$,
второе решение $x = p - R = -6 = 11 \bmod 17$.

Проверка: $6^2 = 36 = 2 \bmod 17$.

4.4.2. Извлечение квадратного корня по составному модулю

Задача решения квадратичного сравнения по составному модулю предполагает нахождение разложения составного модуля на множители и решения соответствующих сравнений по каждому множителю составного модуля с последующим нахождением решения исходной задачи на основе китайской теоремы об остатках. Необходимость искать разложения числа на множители делает данную задачу экспоненциально сложной, в отличии от задачи извлечения корня по модулю простого числа.

Рассмотрим задачу извлечения квадратного корня по составному модулю на примере квадратичного сравнения.

Предположим, что $x^2 = 36 \bmod 77$.

Алгоритм извлечения квадратного корня по составному модулю является экспоненциально сложным, так как требует выполнения разложения модуля на множители.

Так как 77 = 7·11, то по китайской теореме об остатках можно записать два сравнения:

$x^2 = 36 \bmod 7$
$x^2 = 36 \bmod 11$

Решая первое из этих сравнений, получаем:

$x = \pm 1 \bmod 7$

Решая второе сравнение, получаем:

$x = \pm 5 \bmod 11$

Теперь можно составить четыре системы уравнений:

$$\begin{cases} x = 1 \bmod 7 \\ x = 5 \bmod 11 \end{cases}, \begin{cases} x = 1 \bmod 7 \\ x = -5 \bmod 11 \end{cases}, \begin{cases} x = -1 \bmod 7 \\ x = 5 \bmod 11 \end{cases}, \begin{cases} x = -1 \bmod 7 \\ x = -5 \bmod 11 \end{cases}.$$

Решаем системы (оставляем в качестве самостоятельного задания само решение) и получаем ответы $x = \pm 6 \bmod 77$ и $x = \pm 27 \bmod 77$

Проверим решения: $27^2 = 729 = 36 \bmod 77$.

Контрольные вопросы

Что в теории чисел понимают под сравнением, приведите пример.

Сколько решений возможно у линейного сравнения с одним неизвестным?

Приведите два метода решения линейного сравнения и укажите, какой из них является более эффективным.

Сколько решений возможно у квадратичного сравнения по простому модулю?

Что называют квадратичным вычетом?

Что называют квадратичным невычетом?

Найдите для модуля m=7 все квадратичные вычеты и невычеты.

Сколько квадратичных вычетов и невычетов существует по простому модулю?

Дайте определения символа Лежандра.

Приведите критерий Эйлера для символа Лежандра.

Приведите свойства символа Лежандра.

Дайте определение символа Якоби.

Приведите свойства символа Якоби.

Приведите методы решения квадратичного сравнения для $p = 4k+3$.

В каких случаях квадратичное сравнение имеет решение?

Каким образом находить решение квадратичного сравнения по составному модулю.

Сколько может быть решений у квадратичного сравнения по составному модулю?

5. ПЕРВООБРАЗНЫЕ КОРНИ И ИНДЕКСЫ

5.1. Первообразный корень

Определение 6.1. *Первообразным (примитивным) корнем* по модулю m называется такое число a, что все его степени по модулю m пробегают по всем числам, взаимно простым с m.

Другими словами: если число a является первообразным корнем по модулю m, то для любого целого b такого, что НОД(b, m) = 1, найдётся такое целое k, что:

$$a^{\mathrm{k}} = b \bmod n$$

В частности, для случая простого «m» степени первообразного корня пробегают по всем числам от 1 до m – 1.

Теорема 5.1. (без доказательства). Первообразный корень по модулю «m» существует тогда и только тогда, когда m является либо степенью нечётного простого, либо удвоенной степенью простого, а также в случаях m = 1, 2, 4.

Для первообразного корня a можно показать, что наименьшее число k, для которого $a^k = 1 \bmod n$ (т.е. k — показатель a), равно φ(m). Справедливо и обратное утверждение.

Кроме того, если по модулю m есть хотя бы один первообразный корень, то всего их φ(m).

Нахождение первообразного корня. Исходя из определения возможно выполнить поиск первообразный корень возведением потенциального кандидата во все его степени и проверкой, что все получаемые значения будут различны, т.е. проверкой, что для текущего a: $a^1 \neq a^2 \neq \ldots \neq a^{m-1} \bmod m$.

Описанный подход не является эффективным, можно показать, что чтобы число a являлось первообразным корнем достаточно проверить, что все степени числа a, которые являются делителями числа $\varphi(m)$ не сравнимы с 1 по модулю m.

Другими словами, для проверки, что a является первообразным корнем по модулю m, надо проверить, что для любого числа k ($k < \varphi(m)$) делителя $\varphi(m)$ справедливо:

$$a^k \neq 1 \bmod m$$

при этом $a^{\varphi(m)} = 1 \bmod m$, при $(a, m) = 1$.

Замечание. Достаточно проверять только простые делители $\varphi(m)$.

Пример. Рассмотрим модуль $m = 54$. Находим $\varphi(m)$ и делители $\varphi(m)$.

$\varphi(54) = \varphi(2 \cdot 27) = \varphi(2 \cdot 3^3) = \varphi(2)\ \varphi(3^3) = \varphi(3^3) = 3^3 - 3^2 = 18 = 2 \cdot 3 \cdot 3.$

Делителями числа 18 являются числа 1, 2, 3, 6, 9, 18. Рассмотрим число 5. Проверим, будут ли числа 1, 2, 3, 6, 9 показателями для числа 5 по модулю 54:

$5^1 \bmod 54 \equiv 5 \bmod 54;$
$5^2 \bmod 54 \equiv 25 \bmod 54;$
$5^3 \bmod 54 \equiv 125 \bmod 54 \equiv 17 \bmod 54;$
$5^6 \bmod 54 \equiv (17 \cdot 17) \bmod 54 \equiv 289 \bmod 54 \equiv 19 \bmod 54;$
$5^9 \bmod 54 \equiv (5^6 \bmod 54)(5^3 \bmod 54) \equiv 17 \cdot 19 \equiv 323 \bmod 54 \equiv 53 \bmod 54;$
$5^{18} \bmod 54 \equiv (5^9 \bmod 54)^2 \equiv 53 \cdot 53 \equiv 2809 \bmod 54 \equiv 1 \bmod 54.$

Из приведенных вычислений ясно, что минимальный показатель k, для которого $a^k = 1 \bmod m$ (такое значение k еще называют показателем, которому принадлежит число a) для числа $a = 5$ по модулю $m = 54$, равен 18, т. е. значению $\varphi(54)$. Отсюда следует, что число a и целый класс $[5]_{54}$ является первообразным корнем по модулю 54.

Первообразных корней по модулю m может и не быть. Для модуля $m = 20$, рассмотрим все классы взаимно простые с модулем (показатели вычислены аналогично предыдущим примерам):

для класса [1] показатель равен 1;
для классов [9], [11], [19] показатель равен 2;
для классов [3], [7], [13], [17] показатель равен 4.

Не существует ни одного класса взаимно простого с модулем, у которого показатель составляет $\varphi(20) = 8$.

Отсюда следует вывод, что для модуля 20 первообразный корень отсутствует.

Пример. Найти показатель δ числа 2 по mod 7.
$\varphi(7) = 7 - 1 = 6$
$2^1 = 2 \bmod 7$
$2^2 = 4 \bmod 7 \rightarrow \delta = 3 < \varphi(7) \rightarrow 2$ не является первообразным корнем.
$2^3 = 1 \bmod 7$

Пример. Найти показатель δ числа 3 по mod 7.
$3^1 = 3 \bmod 7$,
$3^2 = 2 \bmod 7$,
$3^3 = 6 \bmod 7$,
$3^4 = 4 \bmod 7$,
$3^5 = 243 \bmod 7 = 5 \bmod 7$,
$3^6 = 729 \bmod 7 = 1 \bmod 7$.

Следовательно, $\delta = 6$ и число 3 является первообразным корнем по модулю 7.

Пример. Найти показатель δ числа 5 по mod 7.
$5^1 = 5 \bmod 7$,
$5^2 = 4 \bmod 7$,
$5^3 = 125 \bmod 7 = 6 \bmod 7$,
$5^4 = 625 \bmod 7 = 2 \bmod 7$,
$5^5 = 3125 \bmod 7 = 3 \bmod 7$,
$5^6 = 15625 \bmod 7 = 1 \bmod 7$.
Следовательно, $\delta = 6$ и число 5 является первообразным корнем по модулю 7.

Вывод: по одному и тому же модулю могут существовать первообразные корни и сравнимые по одному модулю числа, принадлежащие одному показателю.

5.2. Индексы и логарифмы

Определение 5.2. Пусть числа a и b взаимно просты с m, т.е. $(a, m) = 1$ и $(b, m) = 1$. Число s называется *индексом* b по модулю m и основанию a, если

$$a^s = b \bmod m.$$

При фиксированном модуле m для индекса s принята следующая форма записи $s = \mathrm{imd}_a\, b$. Для данной формы из определения индекса получается тождество $a^{ind_a b} = b \bmod n$.

Отметим, что если s является индексом числа b по основанию a, т.е. $s = \text{imd}_a b \bmod m$, то s является индексом для любого числа β из класса $[b]$ по любому основанию α из класса чисел $[a]$.

Понятие индекса представляет собой аналогию понятия логарифма. (В дальнейшем часто будет использоваться понятие *дискретный логарифм*.)

Пример. Пусть даны модуль $m = 13$ и основание $a = 2$:

- $2^1 \equiv 2 \bmod 13$, т. е. $\text{imd}_2\, 2 = 1$;
- $2^2 \equiv 4 \bmod 13$, т. е. $\text{imd}_2\, 4 = 2$;
- $2^3 \equiv 8 \bmod 13$, т. е. $\text{imd}_2\, 8 = 3$;
- $2^4 \equiv 16 \bmod 13 \equiv 3 \bmod 13$, т. е. $\text{imd}_2\, 3 = 4$;
- $2^5 \equiv 32 \bmod 13 \equiv 6 \bmod 13$, т. е. $\text{imd}_2\, 6 = 5$;
- $2^6 \equiv 64 \bmod 13 \equiv 12 \bmod 13$, т. е. $\text{imd}_2\, 12 = 6$;
- …
- $2^{13} \equiv (2^6 \cdot 2^6 \cdot 2) \bmod 13 \equiv 288 \bmod 13 \equiv 2 \bmod 13$, т. е. $\text{imd}_2\, 2 = 13$.

Обратите внимание на два соотношения: $\text{imd}_2\, 2 = 1$ и $\text{imd}_2\, 2 = 13$, т.е. одно число может иметь разные индексы.

Пример. Пусть даны модуль $m = 21$ и основание $a = 5$:

- $5^2 \equiv 25 \bmod 21 \equiv 4 \bmod 21$, т. е. $\text{imd}_5 4 = 2$;
- $5^4 \equiv 625 \bmod 21 \equiv 16 \bmod 21$, т. е. $\text{imd}_5 16 = 4$;
- $5^6 \equiv 5452 \bmod 21 \equiv 64 \bmod 21 \equiv 1 \bmod 21$, т. е. $\text{imd}_5 1 = 6$;

Обратим внимание, что для модуля 21 не существует индекса для числа 2 по основанию 5, т. е. не существует значения $\text{imd}_5\, 2$.

Действительно, не существует такого числа s, для которого сравнение $5^s \equiv 2 \bmod 21$ справедливо.

Если p – простое и g первообразный корень по модулю p, то любой элемент из множества чисел $1, 2, \ldots, p-1$, имеет однозначные представления в виде $a = g^{\gamma}$. Для некоторого целого числа $\gamma \in \{0, 1, \ldots, p-1\}$. Число γ и называют дискретным логарифмом или индексом числа a по основанию g:

$$\gamma = \text{imd}_g\, a \bmod p.$$

Перечислим основные свойства индексов:

- $g^{\gamma} = g^{\delta} \bmod p$, если $\gamma = \delta \bmod (p-1)$;
- $imd\ 1 = 0 \bmod (p-1)$;
- $imd\ (a \cdot b \cdot c) = imd(a) \cdot imd(b) \cdot imd(c) \bmod (p-1)$;
- Индексы данного числа по данному основанию образуют класс вычетов по модулю $(p-1)$.

Два числа, принадлежащие одному и тому же классу вычетов, имеют индексы, принадлежащие одному и тому же классу вычетов. Важно отметить, что не существует эффективного (полиномиального) метода вычисления индексов (дискретных логарифмов).

Пример. Определить y из $3=5^{y} \bmod 7$ или $y = \mathrm{imd}_5 3$.

Вычислим подбором:

$y = 1$ $\quad 5^1 \bmod 7 \neq 3$

$y = 2$ $\quad 5^2 \bmod 7 = 25 \bmod 7 = 4 \neq 3$

$y = 3$ $\quad 5^3 \bmod 7 = 125 \bmod 7 = 4 \cdot 5 \bmod 7 = 6 \neq 3$

$y = 4$ $\quad 5^4 \bmod 7 = 625 \bmod 7 = 6 \cdot 5 \bmod 7 = 2 \neq 3$

$y = 5$ $\quad 5^5 \bmod 7 = 3125 \bmod 7 = 2 \cdot 5 \bmod 7 = 3$

Ответ: $\mathrm{imd}_5 3 = 5$.

Для решения задач, связанных с отысканием значений индексов могут быть использованы заранее построенные *таблицы индексов*.

Составленные таблицы индексов для простых модулей p дают возможность по числу находить его индекс и, наоборот, по индексу – число. В качестве основания выбирается один из первообразных корней числа p.

Первые таблицы индексов для простых модулей до 200 составил в 1837 г. русский математик М.В. Остроградский, немецким математиком К. Якоби эти таблицы были доведены до 1000. В настоящее время используются специальные алгоритмы, например, Алгоритм Шенкса, алгоритм на основе «решета числового поля», однако, следует отметить, что все эти алгоритмы вычислительно сложные.

Таблицы обычно содержат наименьшие неотрицательные вычеты по модулю $\varphi(p) = p - 1$ (первая таблица) и наименьшие неотрицательные приведенные вычеты чисел (вторая таблица).

Пример. Построить таблицы для определения индексов по числам и чисел по индексам по модулю $p = 7$

В качестве основания a удобно взять наименьший первообразный корень по модулю 7, возьмем $a = 3$.

Запишем две приведенные системы вычетов по модулю 7:

- 1) 1, 2, 3, 4, 5, 6;
- 2) $3^0, 3^1, 3^2, 3^3, 3^4, 3^5$.

С помощью сравнения устанавливаем соотношение между элементами в пунктах 1) и 2). Определим элементы второй приведенные системы вычетов по модулю 7 и сведем данные в таблицу.

$3^0 = 1 \bmod 7$

$3^1 = 3 \bmod 7$

$3^2 = 2 \bmod 7$

$3^3 = 6 \bmod 7$

$3^4 = 4 \bmod 7$
$3^5 = 5 \bmod 7$

таблица 6.1

M	1	2	3	4	5	6
$\text{imd}_3 m$	0	2	1	4	5	3

Перепишем таблицу относительно значений индексов и получим следующую таблицу:

таблица 6.2

M	0	1	2	3	4	5
$\text{imd}_3 m$	1	3	2	6	4	5

5.3. Применение индексов к решению сравнений

Таблицы индексов могу быть использованы для решения некоторых сравнений по модулю.

Решение сравнений первой степени по простому модулю.

$$a \cdot x \equiv b \pmod{p}, \text{ где НОД}(a, p) = 1$$

«Индексируем» левую и правую части, получаем:
Imd a + imd x = imd b mod (p – 1)
imd x = imd b – imd a mod (p – 1)
Индексы находят из таблиц
Imd x = imd c mod (p – 1)
$x = c \pmod{p}$

Пример. Решить сравнение $4 \cdot x = 5 \pmod 7$
Решение: НОД(4, 7) = 1, сравнение имеет единственный класс решений.
Индексируем обе части:
imd 4+imd x = imd 5 mod 6
По таблице индексов (как и раньше для основания 3 по mod 7) находим:
$\text{imd}_3 5 = 5$ $\text{imd}_3 4 = 4$.
Тогда $\text{imd}_3 x = 1 \bmod 6$
По обратной таблице находим, что $x \equiv 3 \pmod 7$.
Проверяем: $4 \cdot 3 = 5 \bmod 7$.

Рассмотрим более сложный пример, необходимо решить сравнение:
$2 \cdot x^5 = 3 \bmod 7$.
Проиндексируем обе части сравнения:

imd 2+ 5 imd x = imd 3 mod 6;
5 imd x = imd 3 – imd 2 mod 6, по таблице определяем imd 2 и imd 3;

5 imd x = 1 – 2 mod 6;
5 imd x = – 1 mod 6;
5 imd x = 5 mod 6, НОД(5, 6) = 1;
imd x = 1 mod 6, по таблице определяем imd 1;
x = 3 mod 7.
Проверим: $2 \cdot 3^5 = 486 = 3 \bmod 7$.

Контрольные вопросы

Дайте определение понятия первообразного (примитивного) корня.

Приведите условие существования первообразного корня.

Поясните термин показатель при определении перообразного корня?

Приведите пример нахождения перообразного корня для простого числа.

Приведите пример нахождения перообразного корня для составного числа.

Может ли для поля, заданного каким-либо модулем отсутствовать первообразный корень?

Дайте определение индекса (дискретного логарифма).

Может ли одно и тоже число иметь разные индексы?

Может ли для какого-либо числа по некоторому модулю не существовать индекса?

Каким образом может быть решена задача нахождения индекса?

Для решения каких задач могут использоваться индексы?

6. ЛИНЕЙНЫЕ ДИОФАНТОВЫ УРАВНЕНИЯ

6.1. Общие сведения о диофантовых уравнениях

Определение 6.1. *Диофантовыми уравнениями* называются уравнения вида:

$$P(x_1, x_2, ..., x_m) = 0,$$

где $P(x_1, x_2, ..., x_m)$ – многочлен с целыми коэффициентами. (Решения обычно ищут в целых числах.)

Считается, что первым систематически изучал неопределённые уравнения и описывал методы их решения *Диофант* Διόφαντος *(Diophamtus*) ~ III век н.э. Александрия.

Одно из самых известных диофантовых уравнений

$$x^2 + y^2 = z^2$$

его решения – пифагоровы тройки: (3; 4; 5), (6; 8; 10), (5; 12; 13), (12; 35; 37) …

Доказательство неразрешимости в целых числах диофантового уравнения

$$x^m + y^m = z^m$$

при $m > 2$, что является Великой теоремой Ферма, было закончено Эндрю Уайлсом в 1994 году.

Исследование диофантовых уравнений подразумевает определение:

- имеет ли приведенное уравнение решение в целых числах?
- является ли множество целочисленных решений конечным или бесконечным?
- решения таких уравнений на множестве целых чисел, т. е. нахождения всех его целочисленных решений.
- решения диофантовых уравнений на множестве целых положительных чисел.
- решения таких уравнений на множестве рациональных чисел?

Задача решения подобных уравнений в целых числах полностью завершена только для уравнений с одним неизвестным и для уравнений первой степени. Кроме того, известны методы решения для уравнений второй степени с одним или двумя неизвестными.

Для более сложных уравнений, со степенью выше второй и с двумя или более неизвестными достаточно трудной является даже задача определения существования целочисленных решений. Так по данным на весну 2025 года неизвестно, имеет ли уравнение

$$x^3 + y^3 + z^3 = 114$$

хотя бы одно решение в целых числах, также решения не найдены для чисел (165, 390, 579, 627, 633, 732, 906, 921 и 975 в пределах первой 1000).

Более того, доказано, что в принципе не может существовать единого алгоритма, который бы позволил за конечное число шагов находить решения в целых числах для произвольного диафантова уравнения.

Решения некоторых уравнений найдены лишь в 21 веке:

$x^3 + y^3 + z^3 = 74$ (решено в 2016):

$x = 66\,229\,832\,190\,556$,

$y = -283\,450\,105\,697\,727$,

$z = -284\,650\,292\,555\,885$.

$x^3 + y^3 + z^3 = 33$ (решено в апреле 2019)
$x = 8\ 866\ 128\ 975\ 287\ 528$,
$y = -8\ 778\ 405\ 442\ 862\ 239$,
$z = -2\ 736\ 111\ 468\ 807\ 040$.

$x^3 + y^3 + z^3 = 42$ (решено в сентябре 2019 и потребовало 1 миллиона 300 тысяч часов машинного времени:
$x = -80\ 538\ 738\ 812\ 075\ 974$,
$y = 80\ 435\ 758\ 145\ 817\ 515$,
$z = 12\ 602\ 123\ 297\ 335\ 631$.

6.2. Диофантовы уравнения первой степени с двумя переменными

Определение 6.2. Диофантовым уравнением первой степени с двумя неизвестными называют уравнения, имеющие следующий имеет вид:

$$a \cdot x + b \cdot y = c,$$

где a, b, c — известные целые числа, x и y — неизвестные целые числа.

Применение расширенного алгоритма Евклида позволяется найти одно частное решение такого уравнения.

Пусть даны числа $a \geq 0$ и $b \geq 0$. Расширенный алгоритм Евклида по заданным неотрицательным числам a и b находит их наибольший общий делитель d, а также такие коэффициенты x_d и y_d, что:

$$a \cdot x_d + b \cdot y_d = d.$$

Лемма. Если число c делится на $d = (a, b)$, то диофантово уравнение $a \cdot x + b \cdot y = c$ имеет решение; если число c не делится на d, то диофантово уравнение решений не имеет. В частном случае, когда НОД $(a, b) = 1$, решение всегда существует.

Доказательство. Доказательство следует из того, что линейная комбинация двух чисел a и b должна делиться на общий делитель этих двух чисел. ■

Предположим, что c делится на d, тогда в уравнении $a \cdot x_d + b \cdot y_d = d$ умножим обе части на c/d, получим:

$$a \cdot x_d \cdot (c/d) + b \cdot y_d \cdot (c/d) = c.$$

Таким образом получаем частное решение диофантова уравнения:

$$x_0 = x_d \cdot (c/d)$$
$$y_0 = y_d \cdot (c/d)$$

Данное решение было получено в предположении, что числа a и b неотрицательны. Если одно из чисел или оба числа отрицательны, то можно взять абсолютные значения чисел и применить к ним расширенный алгоритм Евклида, как рассматривалось ранее, а затем изменить знак найденных x_0 и y_0 в соответствии с со знаком чисел a и b соответственно.

Зная одно из *частных решений* (x_0, y_0) уравнения, можно получить все решения диофантова уравнения. Таких решений будет бесконечно много. Пусть d = НОД (a, b), а числа x_0, y_0 удовлетворяют условию: $a \cdot x_0 + b \cdot y_0 = c$.

Прибавляя к числу x_0 число b/d и одновременно отнимая a/d от y_0, не нарушая равенства получаем:

$$a \cdot (x_0 + b/d) + b \cdot (y_0 - a/d) =$$
$$= (a \cdot x_0 + b \cdot y_0 + a \cdot b/d - b \cdot a/d) = \text{c}$$

Процесс добавления и вычитания величины b/d можно повторять бесконечное количество раз, получая таким образом *общее решение* диофантова уравнения, т.е. числа вида:

$$x = x_0 + k \cdot b/d,$$
$$y = y_0 - k \cdot a/d,$$

где k – целое.

Представленное общее решение диофантова уравнения описывает все числа, которые могут являться решениями. В общем случае количество решения является бесконечным, но в конкретной задаче на решение могут быть наложены дополнительные условия, например, решения должны быть неотрицательными числами или находиться в заданном диапазоне.

Пример. Имеются монеты достоинством только в 7 и 12 копеек, необходимо уплатить 43 копейки. Как это сделать? Решаем уравнение:
$7x + 12y = 43$
Поездка на метро стоит 56 рублей, на автобусе 51 рубль. Если всего на поездки был потрачен 1381 рубль, то сколько поездок было совершено на метро, сколько на автобусе.

Составим диофантово уравнение:

$a \cdot x + b \cdot y = c$

В нашем случае $a \cdot$=56, y = 51, c = 1381, подставим:

$56 \cdot x + 51 \cdot y = 1381$

Применяем алгоритм Евклида:

$56 = 51 \cdot 1 + 5$
$51 = 5 \cdot 10 + 1$
$5 = 1 \cdot 5 + 0$

Значит, наибольший общий делитель d чисел 56 и 51 равен 1. Построим линейное выражение для числа d, используя расширенный алгоритм Евклида:

$1 = 51 - 5 \cdot 10 = 51 - (56 - 51) \cdot 10 = 56 \cdot \mathbf{(-10)} + 51 \cdot \mathbf{11}$,
В итоге мы определили коэффициенты Безу: $u = -10$, $v = 11$.

Запишем частное решение:

$x_0 = u \cdot c = (-10) \cdot 1381 = -13810$,
$y_0 = v \cdot c = 11 \cdot 1381 = 15191$.

Данное решение диофантова уравнения говорит о том, что нам надо проехать (-13810) раз на метро, чтобы «заработать» на 15191 поездку на автобусе. Понятно, что данное частное решение не отвечает неявно заданными дополнительными требованиями к решению нашей задачи, а именно требования, чтобы решения были неотрицательными числами. Для получения решения с учетом обозначенных условий найдем общее решение доифантова уравнения и выберем частное, удовлетворяющее нашим условиям.

Общее решение:

$x = x_0 + (b/d) \cdot k = 13810 + 51\,k$;
$y = y_0 - (b/d) \cdot k = 15191 - 56\,k$,

где $d = (a,b) = 1$, k – целое число.

Для получения положительных решений диофантова уравнения, т.е. значений $x \geq 0$ и $y \geq 0$, выбираем k:

для $x \geq 0$: $-13810 + 51k \geq 0 \Rightarrow k \geq 13810/51 \approx 270.784 \Rightarrow k \geq 271$;
для $y \geq 0$: $15191 - 56k \geq 0 \Rightarrow k \leq 15191/56 \approx 271.268 \Rightarrow k \leq 271$.

Таким образом, единственным решением исходной задачи являются числа:

$x = x_0 + (b/d)\cdot k = 13810 + 51\cdot k = 13810 + 51\cdot 271 = 11;$
$y = y_0 - (b/d)\cdot k = 15191 - 56\cdot k = 15191 - 56\cdot 271 = 15.$

В итоге если Вы потратили 1381 рубль при стоимости проезда в метро 56, а в автобусе 51, то Вы проехали на автобусе 15 раз, а на метро 11.

Контрольные вопросы

Какие уравнения называются диофантовыми?

Приведите пример диофантового уравнения.

Поясните понятие Пифагоровых троек, приведите пример.

Какие задачи ставятся при исследовании диофантовых уравнений?

Для каких диофантовых уравнений существуют регулярные методы решений?

Приведите пример линейного диофантового уравнения с двумя неизвестными.

Какой метод позволяет найти решение диофантового уравнения первой степени с двумя неизвестными?

Привести пример линейного диофантового уравнения, которое не имеет решения.

Приведите пример, как на основе частного решения, получить все решения линейного диофантового уравнения.

7. РАЗЛОЖЕНИЕ НА МНОЖИТЕЛИ

Согласно основной теореме арифметики любое натуральное число может быть единственным образом представлено в виде:

$$n = {p_1}^{e_1} {p_2}^{e_2} \dots {p_k}^{e_k},$$

где $p_1, p_2, \dots, p_k$ – простые, $(1 < p_1 < p_2 < \dots < p_k)$, $e_1, e_2, \dots, e_k$ – натуральные числа. Отсюда задача разложения на множители заключается в нахождении соответствующих чисел $p_1, p_2, \dots, p_k$. Для этого достаточно решать более простую задачу о разложении целого числа на два меньших, т.е. решения в целых числах $a > 1$ и $b > 1$, уравнения $M = a\cdot b$. Существующие методы решения данной задачи являются сложными, с точки зрения теории сложности и, в зависимости от конкретного алгоритма, относятся к экспоненциальным или субэкспоненциальным.

7.1. Алгоритм разложения на множители методом пробных делений

Алгоритм пробных делений состоит в последовательном делении заданного числа M на числа (2, 3, 4, 5, …) не превосходящие корня из M. По мере выполнения алгоритма будет либо найден делитель, либо, пройдя все числа до корня из M и так и не обнаружив делителя, мы сможем утверждать, что число M гарантированно простое.

При «ручной» оценки наличия малых делителей числа можно использовать признаки делимости, из которых наиболее известными являются признаки делимости числа на 2, 3, 5, 6, 7, 9, 11. Однако при программном анализе проще выполнить соответствующие деления.

Некоторая возможность оптимизации метода пробных делений заключается в делении не последовательно на каждое из чисел меньших корня из исследуемого числа, а делении только на простые числа. Это намного эффективнее, но требует знать заранее какой-то набор простых чисел. На практике такой подход используют, ограничившись какой-то верхней границей для используемого ряда простых чисел.

7.2. Алгоритм разложения на множители Ферма

Данный алгоритм пытается представить M как разность квадратов двух чисел, т.е. в виде $M = x^2 - y^2$, тогда m раскладывается на $(x - y)(x + y)$. Числа x, y в этом случае принимают значение:

$$x = \frac{a+b}{2}, y = \frac{a-b}{2}$$

$$x^2 - y^2 = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2 = a \cdot b = N$$

Приведем *алгоритм Ферма разложения на множители*:

1) Вычисляем $x = [\mathrm{sqrt}(M)]$ ([…] обозначает взять целое), если $M = x^2$, то x является делителем исходного числа – окончание алгоритма, иначе формируем новое значение: $x = x + 1$.

2) Если $x = (M + 1)/2$, то число M является простым – окончание алгоритма, иначе вычисляем $y = \mathrm{sqrt}(x^2 - M)$.

3) Если y целое, т.е. $y^2 = x^2 - M$, то решение найдено и принимает значение ($M = (x + y)\cdot(x - y)$), иначе формируем новое значение $x = x + 1$ и переходим к пункту 2).

Пример, разложить на множители число M =1342127. $\lceil\sqrt{N}\rceil$=1158.

Таблица 4.1

x	x^2	$x^2 - M$	$y = \text{sqrt}(x^2 - M)$
1158	1340964	–1163	
1159	1343281	1154	33,97058
1160	1345600	3473	58,93216
1161	1347921	5794	76,11833
1162	1350244	8117	90,09439
1163	1352569	10442	102,1861
1164	1354896	12769	113

Т.е. x = 1164, y = 113. M = (1164 + 113) · (1164 – 113) = 1277·1051.

Данный алгоритм является эффективным при близких сомножителях, располагающих вблизи корня исходного числа.

7.3. Другие алгоритмы разложения на множители

Существуют и другие алгоритмы разложения на множители, например, такие как алгоритм Лемана, ρ – метод Полларда, алгоритм цепных дробей, (p–1) метод Полларда, алгоритм Диксона и ряд других. Изучение перечисленных алгоритмов выходит за рамки данного курса, часть из них будет рассмотрены в следующих курсах по информационной безопасности.

7.4. Контрольные вопросы

Приведите признаки делимости числа на числа 2, 3, 4, 5, 6, 7, 8, 9, 10.

К каким классам сложности относятся алгоритмы разложения на множители?

Какое максимальное количество пробных делений необходимо выполнить для нахождения множителей составного числа?

Для каких чисел алгоритм пробных делений позволит определить хотя бы один сомножитель за небольшое число шагов?

Для каких чисел алгоритм Ферма позволит определить хотя бы один сомножитель за небольшое число шагов?

Поясните, чем схожи и чем отличаются задачи нахождения хотя бы одного делителя числа, и задача определения является ли число простым или составным.

8. ПРОСТЫЕ ЧИСЛА

8.1. Простые и составные числа

Попомним, что натуральное число $M > 1$ называется *составным*, если его можно представить в виде произведения двух меньших натуральных чисел. Если такое представление невозможно, число M называется *простым*.

Существует метод, позволяющий сравнительно легко определить список всех простых чисел, до заданной границы M. Этот метод носит название решето Эратосфена.

Решето Эратосфена реализует алгоритм, который находит список всех простых чисел $p_1 < p_2 < ...$ до заданной границы M. Приведем шаги алгоритма.

1. Выпишем все целые числа 2, 3, 4, 5, ..., $M-1$, M. Положим $p_1 = 2$ и начиная с $4 = p_1^2$ будем вычеркивать числа, двигаясь с шагом 2.

2. Пусть $k \geq 2$ и уже определены числа p_1, ..., p_{k-1}. Обозначим через p_k первое невычеркнутое число, следующее за p_{k-1}. Если $p_k^2 > N$ обозначаем через p_{k+1}, p_{k+2}, ... все оставшиеся невычеркнутыми числа, следующие за p_k в порядке возрастания; на этом алгоритм завершает свою работу.

3. Если $p_k^2 \leq M$, вычеркиваем числа, начиная с p_k^2 и двигаясь до M с шагом p_k. Вычеркнутые ранее числа также принимаются в учет, но не вычеркиваются еще раз. По завершении этой процедуры алгоритм увеличивает индекс k на единицу и переходит в шаг 2.

8.2. О распределении простых чисел

Непосредственно из имеющихся таблиц простых чисел усматривается, что простые числа распределены в множестве $\mathbb{N}$ весьма неравномерно. В первой сотне натуральных чисел насчитывается 25 простых чисел, во второй — 21, в сорок девятой — 8, в пятидесятой — 15. Наблюдается явная тенденция к постепенному уменьшению плотности распределения простых чисел.

Вместе с тем в 1845 г. французский математик Бертран высказал предположение, что при любом $m > 1$ между m и $2m$ обязательно найдется хотя бы одно простое число. Бертран использовал эту гипотезу при изучении групп подстановок. Предположение Бертрана было строго доказано П.Л. Чебышевым в 1852 г.

Значительные усилия при изучении простых чисел предпринимались для поиска явных формул, описывающих множество простых чисел. Известно, что значения многочлена с целыми коэффициентами с решениями в виде целых чисел не могут состоять только из простых чисел. Тем не менее в 1976 г. был построен многочлен степени 25 от 26 переменных, у которого

множество положительных значений совпадает с множеством всех простых чисел.

Обозначим функцией $\pi(x)$ количество простых чисел p в интервале $1 < p \leq x$. Четырнадцатилетний Гаусс в 1791 г., анализируя эту функцию при малых x, заметил, что лучше всего ее приближает функция

$$\pi(x) \approx \frac{x}{\ln x}$$

В 1808 г. Лежандр предложил формулу, которая с достаточно большой точностью дает количество простых чисел в диапазоне до 10^6. В 1850 г. П. Л. Чебышев показал, что формула Лежандра неверна, и получил следующие оценки:

$$0{,}921\frac{x}{\ln x} < \pi(x) < 1{,}106\frac{x}{\ln x}$$

В работах, опубликованных в 1849, 1852 гг., он доказал теоремы, из которых следует, что если существуют пределы

$$\lim_{x\to\infty} \frac{\pi(x)}{\frac{x}{\ln x}} \qquad \lim_{x\to\infty} \frac{\pi(x)}{\int_2^x \frac{dt}{\ln x}},$$

то они равны 1. Существование этих пределов было доказано лишь в 1896 г. независимо французским математиком Ж. Адамаром и бельгийским математиком Ш. Валле Пуссеном.

8.3. Критерии простоты

1. **Решето Эратосфена.** Имеется некоторое заданное число m. Для проверки, является ли оно простым, необходим список всех простых чисел p_i в диапазоне от 2 до $\lfloor\sqrt{n}\rfloor$. Для каждого p_i из списка должно выполняться условие

$$(p_i, m) = 1.$$

Если условия выполняются для всего списка, число M является простым. В противном случае оно составное.

Метод даёт точный ответ, но для чисел таких, как 2^{500} или больших, ответа можно и не дождаться.

2. **Критерий Вильсона.** В 1770 г. Э. Варинг опубликовал следующую теорему, приписываемую Д. Вильсону.

Теорема 8.1. (без доказательства). Для любого m следующие два условия эквивалентны:

а) m — простое;

б) $(m-1)! \equiv -1 \pmod{m}$.

Критерий иногда бывает удобен в доказательствах, но применять его для проверки простоты невозможно ввиду большой трудоёмкости.

3. Теорема 8.2. (без доказательства).

Нечетное число m является простым тогда и только тогда, когда сравнения

$x^2 \equiv 1 \pmod{m}$,

$x^2 \equiv 0 \pmod{m}$.

имеют соответственно ровно 2 и ровно 1 решение по модулю m.

4. Теорема 8.3. (без доказательства).

Нечетное число n является простым тогда и только тогда, когда оно единственным образом представляется в виде разности квадратов целых неотрицательных чисел.

Приведенные критерии простоты требуют для своей проверки значительных вычислений. Поэтому на практике ими пользуются только в отдельных частных случаях, когда известна дополнительная информация о числе m.

Эффективные тесты проверки простоты целых чисел используют в настоящее время следующие три критерия простоты числа. Основная идея следующих критериев состоит в использовании канонического разложения числа $n-1$.

5. Критерий Лукаса

Натуральное число m является простым тогда и только тогда, когда существует такое a, $(a, n) = 1$, что:

1) $a^{n-1} = 1 \pmod{n}$;

2) для любого простого делителя q числа $n-1$ выполняется условие $a^{\frac{n-1}{q}} \not\equiv 1 \pmod{n}$.

6. Критерий Эйлера

Натуральное число $n > 1$ является простым числом тогда и только тогда, когда для любого a, $(a, n) = 1$ выполняется сравнение

$$a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) \pmod{n}.$$

7. Критерий Миллера

Пусть n — нечетное натуральное число и выполнены условия $n - 1 = 2^t \cdot u$, $(u, 2) = 1$. Тогда равносильны утверждения:

1) число n является простым;

2) для любого a такого, что $(a, n) = 1$ и $a^u \not\equiv 1 (\mathrm{mod}\ n)$, найдется $k \in \{0, 1, ..., t - 1\}$ со следующим свойством:

$$a^{2^k u} = -1 \ (\mathrm{mod}\ n).$$

Числа Ферма и числа Мерсенна

Числа $F_m = 2^{2^n} + 1$ называются *числами Ферма*. Ферма установил, что для $m \in \{0, ..., 4\}$ числа F_m являются простыми, и предположил, что числа F_m простые для всех m. Однако Эйлер доказал, что число F_5 делится на 641. К настоящему времени известно много составных чисел Ферма, например F_{9448}, и не известны простые числа Ферма при $m > 4$.

Основные свойства чисел Ферма:

1) любой натуральный делитель числа F_m , $m > 1$, имеет вид $k \cdot 2^{m+2} + 1$, $k \geq 0$.
2) если $k < m$, то $(F_k , F_m) = 1$.

Теорема 8.4. (Пепин, 1877). Число F_m , $m \geq 1$, является простым тогда и только тогда, когда

$$3^{\frac{F_n - 1}{2}} \equiv -1 (\mathrm{mod}\ F_n).$$

Рассмотрим числа вида $M_m = 2^m - 1$, m ≥ 1. Число M_m может быть простым только при простом m. Простые числа M_m называются *числами Мерсенна*. На начало 2025 года известно 52 числа Мерсенна, наибольшее из которых имеет 41 024 320 десятичных цифр и равно $2^{136279841}-1$.

8.4. Алгоритмы проверки простоты

8.4.1. Вероятностные тесты простоты

Для определения того, что данное число n является простым или составным достаточно найти его каноническое разложение. Однако для проверки больших чисел n это потребует значительного объема вычислений. Задача факторизации целых чисел является одной из известных вычислительно сложных задача.

В связи с этим для проверки простоты числа были разработаны другие эффективные алгоритмы, которые получили название *тесты простоты*.

Под тестом простоты понимается алгоритм, позволяющий для любого целого $n > 1$, не производя разложения на множители, определить, является ли число n составным. Данный алгоритм может быть, как детерминированным, так и вероятностным.

В основе любого такого теста лежит некоторый критерий простоты числа n, состоящий из конечного набора условий. Проверка выполнения полного набора условий позволяет определить является ли исследуемое число n простым или составным.

Полная проверка всех условий скорее всего окажется не более эффективной, чем разложение числа на множители, т.е. будет слишком трудоемкой. Так что на практике выполняют проверку только части условий.

Выполнение таких проверок приводит к следующему результату. Если хотя бы одно условие в рамках проверок не было выполнено, то число **точно** является составным. Если все проверяемые условия оказались выполненными, то можно говорить, что число n является простым, но при этом лишь **с некоторой вероятностью**.

Алгоритмы, выполняющие часть проверок условий критерия простоты, называют *вероятностными тестами простоты*. Большинство вероятностных тестов простоты относительно просты в обосновании и реализации. Временная сложность таких тестов имеет полиномиальную сложность порядка $\log n$.

8.5. Тест Ферма

Тест Ферма основан на рассмотренной ранее малой теореме Ферма. Малая теорема Ферма утверждает, что если n простое, то для всех $a \in \{2, 3, ..., n - 1\}$ и НОД $(a, n) = 1$ имеет место сравнение

$$a^{n-1} = 1 \pmod{n}.$$

Если сравнение выполняется, то число n является *псевдопростым по основанию a*. Обратное утверждение неверно, указанное условие является лишь необходимым условием простоты числа n.

Из теоремы следует, что если сравнение не выполняется хотя бы для одного числа a из интервала $\{2, ..., n - 1\}$, то n — составное число. Поэтому

На этом основан тест, получивший название теста Ферма. Для выполнения теста выполняется следующий набор шагов:

1) выбрать случайное число a из интервала $\{2, ..., n - 1\}$
2) проверить выполнение условия $(a, n) = 1$ (вычисляется алгоритмом Евклида); если условие не выполняется, то n — составное число;
3) проверить выполнение сравнения: $a^{n-1} = 1 \pmod{n}$, если сравнение не выполнено, то n — составное число;

4) если обе проверки выполнены, то число возможно простое.

Такой набор проверок не может дать точный ответ, можно лишь утверждать, что проверяемое число является псевдопростым по выбранному числу a, но проверки можно повторять, используя другие числа a.

Трудоемкость данного алгоритма определяется величиной $O(\log^3 n)$.

Псевдопростое по основанию a число не обязательно является простым. Например, число 341 является псевдопростым по основанию 2, хотя оно составное: 341 = 11·31.

Оценка вероятности успеха теста Ферма. Обозначим $H_n = \{a \in \mathbb{Z}_n^* \mid a^{n-1} = 1(\mathrm{mod}\, n)\}$ множество псевдопростых чисел в кольце классов вычетов $\mathbb{Z}_n^*$. Это множество является мультипликативной абелевой группой в $\mathbb{Z}_n^*$.

В данной группе выделим подгруппу

$$H_n' = \{a \in \mathbb{Z}_n^* \mid (a,n) = 1, a^{n-1} = 1(mod\ n)\}.$$

Обозначим $G_n = \{a \in \mathbb{Z}_n^* \mid (a,n) = 1\}$.

Это тоже мультипликативная абелева группа. Ясно, что $H_n' \subseteq G_n \subseteq \mathbb{Z}_n^*$. Порядок подгруппы H_n' по известной теореме Лагранжа должен делить порядок G_n.

Пусть *модуль* n не является псевдопростым хотя бы по одному основанию a. Тогда a не принадлежит группе H_n', а находится в некотором смежном классе фактор-группы G_n / H_n' и мощность этого смежного класса равна в точности мощности $|H_n'|$. Так как смежных классов не менее двух, то получаем, что

$$\frac{|H_n'|}{|G_n|} \leq \frac{1}{2}$$

Поскольку $|G_n| \leq |\mathbb{Z}_n^*| = n$ получаем, что

$$\frac{|H_n'|}{|\mathbb{Z}_n^*|} \leq \frac{1}{2}$$

Определим понятие *вероятности успеха* в работе алгоритма. Пусть n – составное число. Тогда под вероятностью успеха работы теста будем понимать вероятность события, состоящего в том, что алгоритм даст ответ «число n составное» для составного числа n, то есть распознает это свойство. Эта вероятность равна

$$p_0 = 1 - \frac{|H_n'|}{n-1}$$

В результате выполнении теста возможны три случая:

1) если число n — простое, тогда тест всегда даст ответ, что число вероятно простое;

2) число n — составное, и n не является псевдопростым хотя бы по одному основанию a, тогда тест даст ответ, что число n — составное. Вероятность p_0 такого исхода составляет:

$$p_0 = 1 - \frac{|H_n'|}{n-1} \geq \frac{1}{2}$$

3) число n — составное и n является псевдопростым числом по всем основаниям из группы G_n. В этом случае

$$p_0 = 1 - \frac{\varphi(n)}{n-1}$$

Третья ситуация возможна, так как существуют составные числа, являющиеся псевдопростыми по всем основаниям $a \in G_n$.

Определение 8.1. Составные числа n, для которых сравнение $a^{n-1} = 1 \pmod{n}$ в тесте Ферма выполняется при всех $a \in G_n$, называются числами Кармайкла.

Замечание к определению: запись $a \in G_n$ обозначает такие числа a, которые взаимно простые с n.

Числа Кармайкла имеют по меньшей мере три простых множителя. Количество чисел Кармайкла, не превосходящих **25 *млрд.*** всего **2163**, а не превосходящих **100 000** всего **16**: **561**, **1105**, **1729**, **2465**, **2821**, **6601**, **8911** и т.д.

Пример. Числа Кармайкла с их разложением на множители:

561 = 3·11·17
1105 = 5·13·17
…
6601 = 7·23·41
41041 = 7·11·13·41
…
321197185 = 5·19·23·29·37·137

Числа Кармайкла являются достаточно редкими, но лишь недавно было доказано, что чисел Кармайкла бесконечно много, а предварительная отбраковка чисел Кармайкла не представляется возможной.

В итоге, хотя на всех составных числах, кроме чисел Кармайкла, вероятность в ходе одиночной проверки определить, что число составное не хуже, чем ½, но выпадение числа Кармайкла становится критическим для данного алгоритма. В итоге, тест Ферма не рекомендуется к практическому применению.

8.6. Тест Соловея-Штрассена

Данный вероятностный тест был предложен в 1977 г., работа его основана на критерии простоты Эйлера.

1) Случайно выбирается число $a \in \{2, ..., n-1\}$. Далее с помощью алгоритма Евклида вычисляется $(a, n) = d$. Если d > 1, то тест завершает работу и формирует ответ: «n — составное число».
2) Проверяется выполнение сравнения

$$a^{\frac{n-1}{2}} = \left(\frac{a}{n}\right) \pmod{n}.$$

Результат определения, является ли число a вычетом или не вычетом должен совпасть со значением символа Якоби. Если n – простое число, они совпадают.

Если сравнение не выполнено, то результат теста – число n составное.

Если сравнение выполняется, ответ теста – «неизвестно».

Трудоёмкость данного алгоритма оценивается величиной $O(\log^3 n)$.

Данный тест аналогичен тесту на основе малой теоремы Ферма, однако имеет отличительную особенность. В отличие от предыдущего теста, при его однократном использовании возникают только две ситуации:

1) если число n — простое и тест выдает ответ «неизвестно» (простое с некоторой вероятностью);

2) если число n — составное. Тогда тест дает ответ «n — составное число» с вероятностью успеха не менее 1/2.

Если n — составное число, то при использовании данного теста для $s > 1$ различных значений a вероятность определения, что число составное оценивается следующим образом:

$$p_0^{(s)} = 1 - (1 - p_0)^s \geq 1 - \frac{1}{2^s}.$$

8.7. Тест Миллера-Рабина

Данный тест использует следующий факт: в простом поле классов вычетов по модулю простого числа n корень из единицы может иметь только два значения – единица и минус единица (или, что тоже самое, $n - 1$). Если получается другое значение корня из единицы, то модуль n – составное число.

Пусть n — нечетное число, тогда можно записать следующее выражение:

$$n - 1 = 2^t \cdot u, (u, 2) = 1.$$

Случайно выбирается число $a \in \{2, ..., n - 1\}$, вычисляется наибольший общий делитель выбранного числа a и проверяемого модуля n: $(a, n) = d$:

- если $d > 1$, то ответ: «n — составное число»;
- если $d = 1$, то провести последовательные вычисления:

$$r_k = a^{2^k u} \pmod{n}, k \in \{0, 1, \dots, t - 1\}.$$

Если $r_0 = 1 \pmod{n}$ или $r_0 = -1 \pmod{n}$ то ответ теста – «неизвестно».

В противном случае продолжаем последовательное вычисление r_k , $k = 1, 2, \dots t - 1$. Получаем последовательность чисел, не равных единице:

- если при вычислении очередного r_k в этой последовательности получаем $r_k = -1$, то ответ теста – «неизвестно» (число вероятно простое);
- если при вычислении очередного r_k в этой последовательности получаем $r_k = 1$, то ответ теста – «n – составное число».

Трудоемкость теста Миллера–Рабина для одного числа a оценивается величиной $O(\log^3 n)$.

В 1980 г. было доказано, что в случае, когда число m составное, то при одном проходе теста Рабина вероятность распознавания данного свойства не менее 0,75.

8.8. Сертификат простоты числа

При заказе большого простого числа обычной практикой для заказчика является требование предоставления доказательства простоты полученного числа или способа проверки простоты с допустимым расходом вычислительных ресурсов, то есть проверки выполнения некоторого критерия простоты. Одним из таких критериев может служить критерий Лукаса.

Пусть известно каноническое представление числа $m - 1$:

$$n - 1 = \prod_{i=1}^{r} q_i^{l_i}$$

Число m является простым, если

1. Найдётся число $a \in \{2, 3, \ldots, n-1\}$ взаимно простое с проверяемым модулем, $(a, n) = 1$.
2. Для него выполняется сравнение
$$a^{n-1} \equiv 1 \pmod{n}$$
3. Для всех q_i, $i \in \{1, 2, \ldots, r\}$ должны выполняться условия
$$a^{\frac{n-1}{q_i}} \not\equiv 1 \pmod{n}$$

В случае ответа «n — простое» найденное значение a вместе с набором чисел q_i, $i \in \{1, ..., r\}$ образует *сертификат простоты числа* n. Действительно, по a и q_i, $i \in \{1, ..., r\}$ простота числа n может быть подтверждена с полиномиальной относительно $\log n$ трудоемкостью.

Кроме вероятностных тестов, которые определяют свойство «число составное» существуют детерминированные тесты, выясняющие наличие свойства простоты числа. Один из них опубликован в августе 2002 г. в работе индийских математиков. Работа алгоритма основывается на очевидном критерии простоты.

Теорема 8.5. Пусть числа a и n взаимно просты. Тогда n — простое в том и только в том случае, когда выполнено сравнение

$$(x - a)^n = (x^n - a) \pmod{n}.$$

Реальная сложность данного алгоритма настолько высока, что он представляет собой пока только теоретический интерес. Другой тест основывается на использовании характеров сумм Гаусса, но также достаточно сложен для практического использования.

8.9. Построение больших простых чисел

8.9.1. Общие подходы

Рассмотрим некоторые способы распознавания простоты, которые проверяют достаточные условия простоты числа. Подобные тесты распознавания дают следующие ответы:

– n составное число;

– не знаю;
– n простое число.

Общая схема построения большого простого числа имеет следующий вид: определяется последовательность $\{n_i\}$ целых чисел заданного вида, среди которых проводится поиск простого. К числам этой последовательности применяется тест. Если ответ теста «n составное число» или «неизвестно», то выбирается очередное число из последовательности. Если получен ответ «n простое число», построение завершено.

Применяемые в настоящее время алгоритмы распознавания простоты числа n состоят из трёх частей или этапов проверки.

Этап 1. Отсеивание испытываемого числа при наличии у него малых простых делителей. Заготавливается таблица первых простых чисел и производится деление числа n на числа из этой таблицы. При получении целого результата деления число n отсеивается, как не выдержавшее испытаний первого этапа, выбирается для проверки следующее число. На данном этапе возможно использование признаков делимости, их применение требует незначительных вычислительных расходов.

Этап 2. Отсеивание испытываемого числа после выявлении у него свойства «число n – составное». При этом рекомендуется использование вероятностного теста Миллера-Рабина, который при одном испытании распознаёт свойство «составное» (если число m составное) с вероятностью 0,75. Трудоемкость данного теста при проведении одного испытания оценивается величиной $O(\log^3 n)$, количество испытаний обычно 1000 и более.

Этап 3. После завершения второго этапа проверки наличия свойства простоты можно сказать, что число n является простым с высокой вероятностью. Содержанием этапа 3 проверки является распознавание наличия некоторого достаточного признака простоты. При этом обычно используются вероятностные тесты.

8.9.2. Критерий Лукаса

В том случае, когда доступно каноническое представление числа $m - 1$, для доказательства простоты числа m удобно использование теоремы Лукаса (или Люка), см. выше. Пусть известно каноническое представление очередного выбранного числа $m - 1$:

$$n - 1 = \prod_{i=1}^{r} q_i^{l_i}$$

Для распознавания простоты этого числа m можно воспользоваться вероятностным алгоритмом:

1) выбираются случайным образом числа $a_1, a_2, \ldots, a_s$ из множества $\mathbb{Z}_n^*$ и проверяются для них условия

а) $a^{m-1} \equiv 1 \pmod{m}$;

б) для любого простого делителя q числа $m - 1$ выполняется условие

$$a^{\frac{n-1}{q}} \not\equiv 1 (mod\ n)$$

2) если нашлось число a, для которого оба условия выполнены, то m – простое. В противном случае ответ теста «неизвестно», следует выбрать для проверки следующее число m из последовательности $\{m_i\}$.

В терминах теории групп можно сказать следующим образом: отыскание во множестве $\mathbb{Z}_n^*$ элемента a порядка $m - 1$, ord (a) = $m - 1$ доказывает, что $\mathbb{Z}_n^*$ – группа, следовательно, m – простое число.

Теорема Лукаса является основой для построения целой группы тестов, позволяющих проверять простоту чисел. Группа таких тестов получила название ($m - 1$) – методов, они предполагают знание полной или частичной факторизации числа $m - 1$.

8.9.3. Теорема Поклингтона

В 1914 г. Поклингтон доказал теорему, позволяющую проверять простоту числа n на основе знания только частичного разложение числа $n - 1$ на множители.

Теорема 8.6. (Поклингтон). Пусть $n = q^k \cdot R + 1$, q — простое число и $(q, R) = 1$. Если существует такое целое число $a \in \mathbb{Z}_n^*$, для которого выполняются соотношения:

1) $a^{n-1} = 1 \bmod n$;

2) $\left(a^{\frac{n-1}{q}} - 1, n\right) = 1$,

то каждый простой делитель p числа n имеет вид $p = q^k \cdot r + 1$, $r \in \{2, \ldots, q - 1\}$.

Рассмотрим следующую теорему, которая является обобщением теоремы Поклингтона.

Теорема 8.7. Пусть $n = F \cdot R + 1$, $0 < R < F$. Если для любого простого делителя q числа F существует такое целое число $a \in \mathbb{Z}_n^*$, для которого выполняются соотношения:

1) $a^{n-1} = 1 \bmod n$;

2) $\left(a^{\frac{n-1}{q}} - 1, n\right) = 1$.

то число n является простым.

Данная теорема показывает, что если удалось частично факторизовать число $n - 1 = FR$ и $F > \sqrt{n}$, то наличие свойства «число n – простое» достаточно просто проверить.

Частные случаи теоремы Поклингтона.

Утверждение 8.1. Пусть $n = 2^k \cdot R + 1$, $(2, R) = 1$, $2^k > R$. Если найдётся целое число a, для которого выполняется условие

$$a^{\frac{n-1}{2}} = -1 (\mathrm{mod}\, n),$$

то число n является простым.

Утверждение 8.2. Пусть $n = 2^k \cdot R + 1$, $(2, R) = 1$, $(3, R) = 1$, $2^k > R$, $k \geq 2$. Число n является простым в том и только в том случае, когда

$$3^{\frac{n-1}{2}} = -1 (\mathrm{mod}\, n).$$

8.9.4. Теорема Диемитко

Рассмотрим теорему Поклингтона. Если в ней заменить второе условие на более слабое:

$$a^{\frac{n-1}{q}} \not\equiv 1 \bmod n$$

то можно получить следующий результат.

Лемма. Пусть $n = q^k \cdot R + 1$, q — простое число, $(q, R) = 1$. Если существует целое число $a \in \mathbb{Z}_n^*$, для которого выполняются соотношения:

1) $a^{n-1} = 1 \bmod n$;

2) $a^{\frac{n-1}{q}} \neq 1 \bmod n$,

то существует простой делитель p числа n вида $p = q^k \cdot r + 1$, $r \geq 1$.

Данный результат слабее утверждений теоремы Поклингтона, но может быть эффективно использован для доказательства простоты чисел.

Теорема 8.8. Теорема Диемитко (без доказательства). Пусть $n = q \cdot R + 1$, q — простое число, $(q, R) = 1$, R – чётное, $R < 4 \cdot (q + 1)$. Если существует такое целое число $a \in \mathbb{Z}_n^*$, для которого выполняются соотношения:

1) $a^{n-1} = 1 \bmod n$;

2) $a^{\frac{n-1}{q}} \neq 1 \bmod n$,

то число n является простым.

Теорема Диемитко лежит в основе алгоритма формирования простых чисел в отечественном стандарте цифровой подписи ГОСТ Р 34.10-94. В нем в качестве свидетеля a выбирают не слишком большие степени двойки, число R подбирается перебором.

Контрольные вопросы

Определите понятия простых и составных чисел.

Поясните алгоритм получения набора простых чисел методом решета Эратосфена.

Как можно приблизительно определить количество простых чисел меньших n?

Приведите критерий Эйлера простоты числа.

Как образом можно определить является ли число простым или составным?

Приведите примеры детерминированных алгоритмов проверки того, что число является простым.

Приведите примеры вероятностных тестов проверки того, является ли число простым.

Приведите последовательность действий в тесте Ферма на простоту.

Поясните основной недостаток теста Ферма.

Какие числа являются числами Кармайкла?

Поясните основную идею теста Рабина-Миллера.

9. АЛГЕБРАИЧЕСКАЯ СТРУКТУРА

9.1. Группа, Кольцо, Поле

Определение 9.1. Непустое множество (чаще всего целых чисел) называется *алгебраической структурой*, если на нём определены какие-либо операции, обладающие определёнными свойствами.

В математике чаще всего рассматриваются такие алгебраические структуры, как *группы*, *поля* и *кольца*.

Определение 9.2. *Группой* называется конечное или бесконечное множество «M», на котором:

- определена операция «*» (например, умножение или сложение), которую можно выполнить, не выходя за пределы M;

- для элементов из «M» выполняется сочетательный (ассоциативный) закон, т.е. для любых $a, b, c \in M$ выполняется:

$(a * b) * c = a * (b * c)$;

- существует так называемый единичный элемент e:

$a * e = a;$

- для каждого элемента a из «M» существует единственный обратный элемент a^{-1} такой, что:

$a * a^{-1} = e;$

Если в группе выполняется переместительный (коммутативный) закон, т.е. для любых $a, b, c \in M$ выполняется:

$a * b = b * a,$

то такую группу называют коммутативной или *абелевой группой*.

Группу, в которой определена операция умножения (·), называют *мультипликативной группой*.

– Существует единичный элемент e и для каждого элемента группы «a» существует единственный обратный элемент a^{-1}, для которых выполняется:

$a \cdot e = e \cdot a = a$
$a \cdot a^{-1} = a^{-1} \cdot a = e$

Если операцией группы является сложение, то группу называют *аддитивной группой*.

– Существует единичный элемент z и для каждого элемента группы a существует единственный противоположный элемент – a, для которых выполняется:

$a + z = z + a = a$
$(–a) + a = a + (–a) = z$

Целые числа и классы вычетов, образуют группу по сложению. Множество положительных действительных чисел является группой по умножению, а множество всех действительных чисел $\boldsymbol{R}$ – группой по сложению (на этом множестве невозможно ввести число, обратное нулю).

У множества комплексных чисел, действительных чисел, рациональных чисел и целых чисел есть общая особенность: в них можно выполнять операции сложения, умножения и вычитания, оставаясь в границах множества.

Определение 9.3. Каждое множество элементов, например, чисел, которое содержит сумму, произведение и разность любых двух своих чисел, называется *кольцом*.

Кольцо образуют, например, чётные числа, это группа по сложению. Нечётные числа не образуют кольцо, так как сумма нечётных чисел – чётное число. Множество классов вычетов по модулю 8 образует кольцо классов вычетов, это группа по сложению.

Кроме того, никакая система положительных чисел не будет кольцом, так как если a и b два различных положительных числа, то либо $a - b$, либо $b - a$ отрицательно. Не будет кольцом и система отрицательных чисел, так как произведение двух отрицательных чисел положительно.

Определение 9.4. Кольцо называется *полем*, если оно содержит частное любых двух своих чисел (делитель предполагается отличным от нуля).

Следовательно, можно говорить о поле рациональных чисел, поле действительных чисел, поле комплексных чисел, в то время как кольцо целых чисел полем не является.

Множество называют полем, если в этом множестве по меньшей мере два элемента и для них:

1) определена операция сложения;

1') определена операция умножения;

2) для сложения выполняется сочетательный (ассоциативный) закон;

2') для умножения выполняется сочетательный (ассоциативный) закон;

3) для сложения выполнятся переместительный (коммутативный) закон;

3') для умножения выполняется переместительный (коммутативный) закон;

4) выполнима операция вычитания;

4') выполнима операция деления, кроме деления на нуль.

Для любых элементов поля a и b найдётся такой элемент x, что $a + x = b$.

Для любых элементов поля a и b найдётся такой элемент y, что $a * y = b$, если $a \neq 0$.

Для поля выполняется распределительный (дистрибутивный) закон умножения (относительно сложения): $(a + b)\cdot c = a\cdot c + b\cdot c$.

Определение 9.5. Если число элементов поля конечно, то такое поле называется *конечным полем* или *полем Галуа* (*Galois field*)**.** Названы в честь французского математика начала XIX века Эвариста Галуа.

Для полей Галуа используют обозначения: *GF(7), GF(q^m), G_{q^n}, GF(2^3).*

Поле Z_p (p – простое число) с операциями сложения и умножения целых чисел по модулю p есть конечное поле. Часто обозначается как $GF(p)$.

Пример. Конечное поле из двух элементов 0 и 1: $Z_2 = GF(2)$

Таблица 9.1

+	0	1		×	0	1
0	0	1		0	0	0
1	1	0		1	0	1

Определим поле $GF(5) = \{0, 1, 2, 3, 4\}$ на множестве Z_5 (5 – простое число) с операциями сложения и умножения как в таблицах:

Таблица 9.2

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Таблица 9.3

×	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Таблица 9.4

Противоположные элементы поля					
a	0	1	2	3	4
$-a$	0	4	3	2	1
Обратные элементы поля					
a	0	1	2	3	4
a^{-1}	0	1	3	2	4

Определение 9.6. Элемент g поля Галуа называется *порождающим* или *примитивным элементом*, если все элементы поля, кроме нулевого, являются степенями g.

Группы с порождающим элементом называют *циклическими*.

Таким образом $G = \{g, g^2, g^3, \ldots, g^m\}$, $g^m = e$, e – нейтральный элемент группы.

Порядок группы G – число элементов группы (обозначение $|G|$).

Порядок элемента $g \in G$ – наименьшее m так что $g^m = e$ (обозначается ord g).

Конечные поля существуют только для порядков $q = p^m$ (p – простое, m – натуральное).

Простое поле порядка p, $GF(p)$, можно трактовать как множество $\{0, 1, \ldots, p - 1\}$ остатков от деления целых чисел на p с операциями сложения и умножения по модулю p.

9.2. Расширенные поля

Определение 9.7. *Расширенное поле* $GF(p^m)$ порядка $q = p^m$ при $m > 1$ представляется множеством остатков от деления полиномов над $GF(p)$ на некоторый неприводимый полином $f(x)$ степени m с операциями сложения и умножения по модулю $f(x)$.

Неприводимый полином нельзя представить в виде произведения других полиномов над этим полем. Такой полином можно определить, как полином, *задающий* расширенное поле.

Поле $GF(p^m)$ можно представить всеми полиномами над простым полем $GF(p)$ степени не выше $m - 1$ с обычным полиномиальным сложением.

Умножение выполняется как обычное умножение полиномов с дальнейшим приведением по модулю и нахождением остатка от деления полученного произведения на неприводимый полином $f(x)$, например, н на многочлен степени m

$$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + a_1x^1 + a_0x^0,$$

где a_i – коэффициенты из некоторого множества (поля). В нашем случае $a_i \in \{0,1\}$.

Представим в таблице, как можно 8-разрядное слово (10011001) представить в виде многочлена.

Таблица 9.5

8 – битовая строка:	1	0	0	1	1	0	0	1
Полином:	$1\cdot x^7 +$	$0\cdot x^6 +$	$0\cdot x^5 +$	$1\cdot x^4 +$	$1\cdot x^3 +$	$0\cdot x^2 +$	$0\cdot x^1 +$	$1\cdot x^0$
Упрощение 1:	$1\cdot x^7 +$			$1\cdot x^4 +$	$1\cdot x^3 +$			$1\cdot x^0$
Упрощение 2:	$x^7 +$			$x^4 +$	$x^3 +$			1
Итоговый полином:	$x^7 + x^4 + x^3 + 1$							

Определим поле $GF(2^2)$, состоящее из 4 двухразрядных слов: {00, 01, 10, 11}. Определим операции сложения и умножения следующим образом (таблично):

Таблица 9.6

+	00	01	10	11		×	00	01	10	11
00	00	01	10	11		00	00	00	00	00
01	01	00	11	10		01	00	01	10	11
10	10	11	00	01		10	00	10	11	01
11	11	10	01	00		11	00	11	01	10

9.3. Задающие полиномы

Пусть задан неприводимый полином степени m над полем $GF(2)$, тогда коэффициенты этого поля будут значения 0 и 1. Такой полином не должен делиться на полиномы степени $m - 1$. Элементами расширенного поля будут все полиномы степени $m - 1$ с коэффициентами из того же поля. Сложение элементов в расширенном поле задается как сложение полиномов с коэффициентами из поля GF(2). Умножение элементов определяется как произведение полиномов с приведением результата по модулю задающего неприводимого полинома. Для элементов существуют противоположный элемент, обратный элемент (для всех элементов, кроме нулевого).

В общем случае полиномиальный алгоритм построения примитивного полинома неизвестен. Существуют таблицы до довольно больших степеней с такими полиномами. Приведем примеры нескольких таких полиномов

Таблица 9.7

Степень	Полиномы
1	$x + 1$; x
2	$x^1 + x + 1$
3	$x^3 + x^2 + 1$; $x^3 + x + 1$
4	$x^4 + x^3 + x^2 + x + 1$; $x^4 + x^3 + 1$; $x^4 + x + 1$
5	$x^5 + x^2 + 1$; $x^5 + x^3 + x^2 + x + 1$; $x^5 + x^4 + x^3 + x + 1$; $x^5 + x^4 + x^3 + x^2 + 1$; $x^5 + x^4 + x^2 + x + 1$

Пример. Полином $f(x) = x^3 + x + 1$ (неприводимый), $\deg(f(x)) = 3$, тогда его можно использовать для построения расширенного поля $GF(2^3) = GF(8)$.

Для $a(x) = x^2 \oplus x \oplus 1$ и $b(x) = x \oplus 1$ сумма в поле $GF(8)$:

$a(x) + b(x) = x^2 \oplus x \oplus 1 \oplus x \oplus 1 = x^2$

произведение в $GF(8)$:

$(x^2 \oplus x \oplus 1) \cdot (x \oplus 1) = x^3 \oplus x^2 \oplus x^2 \oplus x \oplus x \oplus 1 = x^3 \oplus 1,$

после чего разделим полученный результат на $f(x)$ с последующим удержанием только остатка:

$x^3 + 1 = q(x) \cdot f(x) + r(x) = 1 \cdot (x^3 + x + 1) + x.$

Таким образом, $a(x) \cdot b(x) = (x^2 + x + 1) \cdot (x + 1) = x$.

Примечание: $x^3 + 1 = x^3 + 1 + x + x = x^3 + x + 1 + x = x$

Таблица умножения в поле $GF(2^3)$

Исходные значения представляем как элементы поля:
$000 \rightarrow 0$,
$001 \rightarrow 1$,
$010 \rightarrow x$,
$011 \rightarrow x + 1$,
$100 \rightarrow x^2$,
$101 \rightarrow x^2 + 1$,
$110 \rightarrow x^2 + x$,
$111 \rightarrow x^2 + x + 1$.

Таблица 9.8

×	0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
0	0	0	0	0	0	0	0	0
1	0	1	x	$x + 1$	x^2	$x^2 + 1$	$x^2 + x$	$x^2 + x + 1$
x	0	x	x^2	$x^2 + x$	$x + 1$	1	$x^2 + x + 1$	$x^2 + 1$
$x + 1$	0	$x + 1$	$x^2 + x$	$x^2 + 1$	$x^2 + x + 1$	x^2	1	x
x^2	0	x^2	$x + 1$	$x^2 + x + 1$	$x^2 + x$	x	$x^2 + 1$	1
$x^2 + 1$	0	$x^2 + 1$	1	x^2	X	$x^2 + x + 1$	$x + 1$	$x^2 + x$
$x^2 + x$	0	$x^2 + x$	$x^2 + x + 1$	1	$x^2 + 1$	$x + 1$	x	x^2
$x^2 + x + 1$	0	$x^2 + x + 1$	$x^2 + 1$	x	1	$x^2 + x$	x^2	$x + 1$

Построим элементы поля $GF(2^3)$. Рассмотрим поле $GF(2^3)$ с заданным неприводимым многочленом: $x^3 + x + 1$ с коэффициентами из $GF(2)$.

Определение 9.8. Примитивным элементом поля, называется элемент, последовательные степени которого, перечисляют все элементы поля, кроме нулевого.

Рассмотрим примитивный элемент α этого поля. Для построения элементов поля принимаем α = *x*. Получаем все ненулевые элементы поля последовательным возведением α в степень с приведением по модулю $x^3 + x + 1$.

Построение элементов поля $GF(2^3)$ с полиномом: $x^3 + x + 1$ по степеням примитивного элемента α

Таблица 9.9

Степень	Элемент поля		Вектор
–	0	0	000
α^0	1	1	001
α^1	α^1	α^1	010
α^2	α^2	α^2	100
α^3	$\alpha^3 = \alpha^3 + (\alpha^3+\alpha+1) = \alpha+1$	α+1	011
α^4	$\alpha^2+\alpha$	$\alpha^2+\alpha$	110
α^5	$\alpha^3+\alpha^2=\alpha^3 + \alpha^2 + (\alpha^3+\alpha+1)=\alpha^2+\alpha+1$	$\alpha^2+\alpha+1$	111
α^6	$\alpha^3+\alpha^2+\alpha=\alpha^3+\alpha^2+\alpha+ (\alpha^3+\alpha+1)=\alpha^2+1$	α^2+1	101
α^7	$\alpha^3+\alpha=\alpha^3+\alpha+ (\alpha^3+\alpha+1)=1$	**1**	

Рассмотрим сложение элементов в поле и умножение:

$\alpha^3 + \alpha^6 = (1 + \alpha) + (1 + \alpha^2) = \alpha + \alpha^2 = \alpha^4$
$\alpha^3 \cdot \alpha^6 = (\alpha + 1)\cdot(\alpha^2 + 1) = \alpha^3 + \alpha^2 + \alpha + 1 = \alpha^2$ или
$\alpha^3 \cdot \alpha^6 = \alpha^9 = \alpha^2 \cdot \alpha^7 = \alpha^2$

Можно найти обратный элемент в поле : $\alpha^1 \cdot \alpha^6 = \alpha^7 = 1$, $\alpha^2 \cdot \alpha^5 = \alpha^7 = 1$, т.е. обратны полиномы: α и $\alpha^2 + 1$; α^2 и $\alpha^2 + \alpha + 1$.

Пример. Выполним вычисления в *GF*(8). Дано поле $GF(2^3)$ и неприводимый многочлен $f(x) = x^3 + x + 1$. Входные данные в десятичном представлении это числа 2 и 5. Выполним сложение.

- Переводим в двоичный вид и представим в виде полинома:
  $2 = 10_2 \Rightarrow 1\cdot x^1 + 0\cdot x^0 = x$,
  $5 = 101_2 \Rightarrow 1\cdot x^2 + 0\cdot x^1 + 1\cdot x^0 = x^2 + 1$.
- Складываем: $(x^2 + 1) + (x) = x^2 + x + 1 \Rightarrow 111_2$.
- Ответ может быть переведен в десятичное представление: $111_2 = 7$.

Входные данные в десятичном представлении это числа 5 и 7. Выполним умножение.

- Переводим в двоичный вид и записываем в виде полинома: $7 = 111_2 = x^2 + x + 1$.
- Умножаем:
  $(x^2 + 1)\cdot(x^2 + x + 1) = x^4 + x^3 + x^2 + x^2 + x + 1 = x^4 + x^3 + x + 1$, теперь результат умножения необходимо привести по модулю.

$$
\begin{array}{rl|l}
 & x^4 + x^3 + x + 1 & x^3 + x + 1 \\
- & x^4 + x^2 + x & \hline
x + 1 \\
 & x^3 + x^2 + 1 & \\
- & x^3 + x + 1 & \\
\hline
 & x^2 + x &
\end{array}
$$

$(x^4 + x^3 + x + 1) \bmod f(x) = x^2 + x^1 => 110_2$.

Выполняем перевод в десятичное представление: $110_2 = 6$.

Составим таблицу умножения в $GF(8)$ с неприводимым многочленом $f(x) = x^3 + x + 1$.

Таблица 9.10

умножение в GF(8)								
Полиномы		**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	**1**	1	2	3	4	5	6	7
x	**2**	2	4	6	3	1	7	5
$x + 1$	**3**	3	6	5	7	4	1	2
x^2	**4**	4	3	7	6	2	5	1
$x^2 + 1$	**5**	5	1	4	2	7	3	6
$x^2 + x$	**6**	6	7	1	5	3	2	4
$x^2 + x + 1$	**7**	7	5	2	1	6	4	3

деление в $GF(8)$ с неприводимым многочленом $f(x) = x^3 + x + 1$:
$6/5 = 6 \cdot 5^{(-1)} = 6 \cdot 2 => (x^2 + x^1) \cdot x = x^3 + x^2 = x^2 + x + 1 => 111_2 = 7$

$$
\begin{array}{rl|l}
 & x^3 + x^2 & x^3 + x + 1 \\
- & x^3 + x + 1 & \hline
1 \\
\hline
 & x^2 + x + 1 &
\end{array}
$$

$5^{(-1)} = 2$ (по таблице из предыдущего слайда)

В предыдущем примере обратный полином мог быть получен по расширенному алгоритму Евклида для полиномов. Рассмотрим его подробно.

9.4. Расширенный алгоритм Евклида для полиномов

Основные шаги алгоритма:

Входные данные: два многочлена $a(x)$ и $b(x)$ над полем Галуа $GF(p^m)$, где p — простое число, а m — натуральное число.

Инициализация

- $r_0(x) = a(x)$, $r_1(x) = b(x)$, $s_0(x) = 1$, $s_1(x) = 0$, $t_0(x) = 0$, $t_1(x) = 1$

Далее выполняется итерационный процесс:

- Пока $r_i(x) \neq 0$:
  - Вычислить:
    - частное $q(x) = r_{i-2}(x) \div r_{i-1}(x)$ (деление многочленов).
    - остаток: $r_i(x) = r_{i-2}(x) - q(x) \cdot r_{i-1}(x)$.
    - коэффициенты Безу:
      - $s_i(x) = s_{i-2}(x) - q(x) \cdot s_{i-1}(x)$.
      - $t_i(x) = t_{i-2}(x) - q(x) \cdot t_{i-1}(x)$.

Результат:

- НОД многочленов $a(x)$ и $b(x)$ равен $r_{i-1}(x)$.
- Коэффициенты Безу: $s_{i-1}(x)$ и $t_{i-1}(x)$, такие что:
  НОД($a(x)$, $b(x)$) = $s_{i-1}(x) \cdot a(x) + t_{i-1}(x) \cdot b(x)$

Пример. Пусть $a(x) = x^3 + x + 1$ и $b(x) = x^2 + 1$ в поле $GF(2)$.
Инициализация:
$r_0(x) = x^3 + x + 1$, $r_1(x) = x^2 + 1$, $s_0(x) = 1$, $s_1(x) = 0$, $t_0(x) = 0$, $t_1(x) = 1$.

Сведем данные в таблицу

Таблица 9.11

	Q	$r_i(x)$	$s_i(x)$	$t_i(x)$
0		$x^3 + x + 1$	1	0
1		$x^2 + 1$	0	1
2	x	1	$1 - x \cdot 0 = 1$	$0 - x \cdot 1 = x$
3	$x^2 + 1$	0		

Таким образом, НОД многочленов $a(x)$ и $b(x)$ равен 1. Соответствующее соотношение Безу:

$1 = 1 (x^3 + x + 1) + x \cdot (x^2 + 1)$.

Из соотношения Безу может получить значение обратного элемента для $(x^2 + 1)$ по модулю $(x^3 + x + 1)$, это значение коэффициента Безу $t_2(x) = x$. Обратите внимание, что данный ответ совпал с результатом, представленном в таблице 9.10 и соответствующем примером на деление.

Контрольные вопросы

Дайте определение алгебраической структуры.

Дайте определения алгебраической структуры – группа.

Какую группу называют абелевой?

Какую группу называют мультипликативной?

Какую группу называют аддитивной?

Какие элементы являются единичными относительно сложения; относительно умножения?

Дайте определение алгебраической структуры – кольца.

Дайте определение алгебраической структуры – поля.

Дайте определение алгебраической структуры – поля Галуа.

Дайте определение примитивного элемента поля Галуа.

Поясните понятия порядок группы, порядок элемента.

Для каких порядков существуют поля Галуа.

Что является примитивным элементом поля?

Выполните сложение и умножение в поле.

Какая задача может быть решена с применением расширенного алгоритма Евклида для полиномов?

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Герман, О. Н. Теоретико-числовые методы в криптографии : учебник для студ. учреждений высш. проф. образования / О. Н. Герман, Ю. В. Нестеренко. — Москва : Академия, 2012. — 272 с.
2. Виноградов, И. М. Основы теории чисел / И. М. Виноградов. — Москва ; Ижевск : НИЦ «Регулярная и хаотическая динамика», 2003. — 176 с.
3. Коржик, В. И. Основы криптографии / В. И. Коржик, В. А. Яковлев. — Санкт-Петербург : Интермедия, 2016. — 312 с.
4. Черемушкин, А. В. Лекции по арифметическим алгоритмам в криптографии / А. В. Черемушкин. — Москва : МЦНМО, 2002. — 104 с.
5. Глухов, М. М. Введение в теоретико-числовые методы криптографии : учебное пособие / М. М. Глухов, И. А. Круглов, А. Б. Пичкур, А. В. Черёмушкин. — Санкт-Петербург : Лань, 2011. — 400 с.
6. Манин, Ю. И. Введение в теорию чисел / Ю. И. Манин, А. А. Панчишкин // Итоги науки и техники. Современные проблемы математики. Фундаментальные направления. — Москва : ВИНИТИ, 1990. — Т. 49. — 249 с.
7. Дискретная математика : энциклопедия / гл. ред. В. Я. Козлов. — Москва : Большая российская энциклопедия, 2004. — 382 с.
8. Бухштаб, А. А. Теория чисел : учебное пособие для вузов / А. А. Бухштаб. — 8-е изд., стер. — Санкт-Петербург : Лань, 2024. — 384 с.
9. Ростовцев, А. Г. Алгебраические основы криптографии / А. Г. Ростовцев. — Санкт-Петербург : Мир и семья ; Интерлайн, 2000. — 354 с.
10. Коблиц, Н. Курс теории чисел и криптографии / Н. Коблиц ; пер. с англ. — Москва : Научное издательство ТВП, 2001. — 254 с.
11. Коутинхо, С. Введение в теорию чисел. Алгоритм RSA / С. Коутинхо. — Москва : Постмаркет, 2001. — 328 с.
12. Коварцев, А. Н. Алгоритмы и анализ сложности : учебник / А. Н. Коварцев, А. Н. Даниленко. — Самара : Изд-во Самарского ун-та, 2018. — 128 с.
13. Применко, Э. А. Алгебраические основы криптографии : учебное пособие / Э. А. Применко. — Москва : Ленанд, 2015. — 288 с.

УКАЗАТЕЛЬ

Полиномиальный алгоритм 9
алгоритм полиномиальной временной сложности 9
трудноpешаемый алгоритм 9
NP- задачи 9
NP - полные задачи 9
алгоритм 10
асимптотическая сложность 13
мера сложности 15
сложность в среднем 16
оценки сложности алгоритма 15
натуральные числа 22
целые числа 22
простые числа 24
составные числа 24
каноническим разложением 27
наибольший общий делитель 27
взаимно простые числа 29
алгоритм Евклида 30
расширенный алгоритм Евклида 30
соотношением Безу 32
коэффициентами Безу 32
функция Эйлера φ(m) *32*
непрерывная дробь 33
цепная дробь 33
подходящими дробями 34
равноостаточные числа 38
числа, сравнимые по модулю 39
вычет 41
класс вычетов 41
полной системой вычетов 42
приведенная система вычетов 42
теорема Эйлера 43
малая теорема Ферма 44
аддитивная инверсия (противоположный элемент) 44
мультипликативная инверсия (обратный элемент) 45
китайская теорема об остатках 46
линейные сравнения 51
сравнения степени m *54*
квадратичный вычет 54
символ Лежандра 55
критерий Эйлера 55
символ Якоби 57
первообразный (примитивный) корень 61
индекс 64

дискретный логарифм 64
диофантовы уравнения 68
частное решение диофантового уравнения 70
общее решение диофантового уравнения 71
алгоритм Ферма разложения на множители 73
критерий простоты Вильсона 76
критерий простоты Эйлера 77
тест Ферма 79
псевдопростым по основанию a 79
вероятностный тест простоты 79
числа Кармайкла 81
алгебраической структурой 88
группа 88
мультипликативная группа 88
аддитивная группа 89
кольцо 89
конечные поля 90
поля Галуа 90
порождающий, примитивный элемент 91
расширенное поле 91
расширенный алгоритм Евклида для полиномов 96