Министерство образования и науки Российской Федерации
Южно-Уральский государственный университет
Высшая школа электроники и компьютерных наук

519.2(07)
А458

Алеев Р.Ж.

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ЗАЩИТЫ ИНФОРМАЦИИ и ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

Учебное пособие

Челябинск
Издательский центр ЮУрГУ
2017

УДК 519.21(075.8) + 519.22(075.8)
А458

Одобрено
учебно-методической комиссией
Высшей школы электроники и компьютерных наук

Рецензенты:
Кафедра алгебры и фундаментальной информатики Уральского федерального университета (зав. кафедрой М.В. Волков),
доктор физ.-мат. наук В.В. Кораблёва

А458 **Алеев, Р.Ж.**

Математические основы защиты информации и информационной безопасности: учебное пособие/ Р.Ж. Алеев. – Челябинск: Издательский центр ЮУрГУ, 2017. — 121 с.

В пособии рассмотрены основные подходы применения математики для защиты информации и информационной безопасности. Изложение сопровождается примерами, предлагаются вопросы и задачи для тестов. В дополнениях приведены результаты, которые не излагаются на лекциях. Пособие рассчитано на читателей, знакомых с основными понятиями математики в объёме специальностей и направлений инженерного и прикладного профилей. Оно основана на лекциях по дисциплине “Математические основы защиты информации и информационной безопасности”, которые читались автором для магистрантов кафедры системного программирования.

УДК 004.0568)

Оглавление

Предисловие

Сначала несколько общих слов. Последние мировые события показывают насколько велико влияние информации и её распространения на все аспекты современной жизни. Поэтому защита информации и информационная безопасность выходят во многих вопросах на первый план. Магистр по информатике просто ***обязан*** иметь правильное и достаточно полное представление о работе с информацией. С другой стороны, совсем не обязательно, что он станет специалистом по защите информации и информационной безопасности, поскольку это требует специализированного обучения. Так, например, на специальности «Компьютерная безопасность» обучение идёт 5,5 лет. Поэтому при изучении математических основ защиты информации и информационной безопасности приходится сосредотачиваться только на основных моментах.

При переходе на кафедру системного программирования я взялся читать курс «Математические основы защиты информации и информационной безопасности». Причин для этого было несколько. Укажу лишь две основные. Во-первых, фактически среди курсов кафедры системного программирования больше не было математических курсов, а я — математик. Во-вторых, в бытность работы на кафедре алгебры я руководил магистерской программой по подобному направлению. Выпускники той программы разлетелись по стране, и некоторые из них работают в очень престижных компаниях. Поэтому я считал, что имею неплохой опыт в подготовке по такому направлению.

Теперь несколько слов о структуре курса. Когда я начинал читать этот курс, на него было выделено всего 18 часов лекций. Пришлось очень тщательно выбирать материал для лекций. Затем число часов для лекций было увеличено, но состав слушателей быстро охладил пыл по увеличению объёма читаемого материала. Фактически, все слушатели имели очень слабое математическое образование, и потому попытки углубления в сторону лучшего математического обоснования были заранее обречены на провал. Поэтому было принято “соломоново решение”. Нужно разбить лекции на две части: базовую, которой и посвящено это пособие, и вариативную, которая определяется желаниями слушателей и их возможностями, которые могут реализованы, как доклады на занятиях. Некоторые из докладов впоследствии показываются в виде слайдов следующим поколениям слушателей. Например, однажды среди слушателей (как магистрант кафедры ЭММиС) оказалась Бабина Ольга Ивановна, доцент, кандидат филологических наук. Она прочитала доклад о лингвистической стеганографии, окончание которого, не сговариваясь, слушатели сопроводили аплодисментами, а я сам был потрясён и изумлён глубиной подхода к рассмотренной теме.

Автор выражает глубокую благодарность рецензенту Кораблёвой В.В. за ценные замечания, способствовавшие улучшению изложения.

Введение

Что такое безопасность? Безопасность — это отсутствие опасности.

Что такое опасность? Опасность — это отсутствие безопасности.

Либо считаем безопасность первичным, неопределяемым понятием, либо надо сослаться на что-то иное. Например, опасность — наличие угроз. Безопасность — отсутствие угроз. (Что такое угроза? Пока утаим . . .)

Подойдём с другой стороны. Какие виды безопасности можно предложить?

Для человека: 1) личная, 2) общественная, 3) государственная . . .

Для страны: 1) военная, 2) морская, 3) воздушная, 4) космическая . . .

Для экономики:1) промышленная, 2) энергетическая, 3) сельскохозяйственная . . .

Для потребления: 1) продовольственная, 2) производства товаров (электроника, хозяйственные, одежда . . .) . . .

Особняком стоит, чтобы там не говорилось — ***информационная безопасность!***

Ведь настоящее время — это время информации. Влияние информации огромно. Её объёмы колоссальны, её проникновение носит всеобщий характер.

Однако, что такое информация? Можно сказать, что это то, что содержится в данных. А что такое данные? Это то, что содержит информацию! Снова замкнутый круг. Поэтому считаем информацию и данные неопределяемыми первичными понятиями.

Отметим, что важно кому-то, необязательно важно другому и наоборот. Например дело о Fiat и ВАЗе[1]. Ведь $5,6\%$ сами по себе не несут никакой информации.

Часто хорошо известно и понятно, какие данные нужны. Например, для финансовой отчётности важны вполне определённые числовые показатели, но совершенно не важно, кем они составлялись. Лишь очень немногие интересуются практически всей информацией. Было сообщение, что CIA=ЦРУ до 80% нужной им информации добывало из открытых советских источников таких, как, например, местные газеты.

---

[1] `http://gubernya63.ru/dostoprimechatelnosti/madein/auto-zhiguli.html?template=95` Наши требовали 5 процентов – и не больше, хотя в то время в банках Италии минимальная ставка по кредиту составляла 7 процентов. В результате после долгих препирательств обе стороны согласились на ставку 5,6 процента. В переводе на иностранную валюту по курсу того времени выигрыш Советского Союза при заключении сделки составил немалую сумму – 38 миллионов долларов.

Подписание контракта с FIAT с такой кредитной ставкой стало возможным благодаря блистательной операции, проведенной офицером КГБ Леонидом Колосовым. Работая в Италии под видом журналиста, Колосов сумел через организованную им агентурную сеть выяснить тот предел, до которого итальянская сторона могла бы опустить ставку. Более того, он даже добыл совершенно секретный документ из правительства Италии, подтверждающий эту информацию.

Итак, будем исходить из того, что мы знаем, какая информация у нас есть и где она содержится. Нас более будут интересовать вопросы, связанные с работой с имеющейся информацией.

Основные требования к информации.

1. Достоверность.
2. Доступность.
3. Целостность.
4. Конфиденциальность.
5. Актуальность.

Чтобы эти условия соблюдались, информацию нужно защищать! Так как сейчас основной способ работы с информацией — это компьютеры (в широком смысле), то на первый план выходит — ***компьютерная безопасность!***

В компьютерной безопасности выделяют три ветви.

1. Программная (Soft).
2. Аппаратная (Hard).
3. Программно-аппаратная.

Не будем касаться 2 и 3, этим занимаются очень специальные люди. Только скажем, что с компьютера можно улавливать *многое* из того, что на нём делается. Приведём два примера. Можно получать изображение экрана, находясь в другой комнате. В журнале Chip № 2 (2014) написано, что “из шума работающего компьютера извлекаются ключи шифрования”!

Для достоверности и целостности информации часто применяется её кодирование. Не секрет, что сейчас почти вся информация хранится в цифровом виде, и даже в двоичном. Даже если повреждён только 1 бит, то можете получить совсем не то, что хотите. Легко понять, что при передаче на большие расстояния по кабелю и/или воздуху и/или из космоса помехи *огромны*! Поэтому крайне важно уметь находить ошибки, их исправлять. Теория кодирования давно стала развитой частью математики, она тесно связана с алгеброй.

Теперь рассмотрим свойства доступности и актуальности. Эти свойства, в основном, связаны с передачей и хранением информации, которые тоже можно на программном уровне решать при помощи кодирования. Отметим, что здесь на первый план выходят аппаратные решения. В самом деле “перекачка” гигантских объёмов информации стала возможной благодаря созданию высокоскоростных широкополосных линий передачи и устройств, способных передавать, принимать и обрабатывать такие объёмы данных.

Мы пока ничего не говорили о свойстве конфиденциальности. Совершенно понятно, что не всегда и не всем нужно знать некоторую информацию. К примеру, очень важной для каждого человека является неприкосновенность его личной жизни. Конечно, много есть информации с грифами секретности. Сейчас много информации пересылается в электронном виде, поэтому часто

важно, чтобы информация, попавшая в чужие руки, оказывалась недоступной. Есть много способов сокрытия информации, которые рассмотрим позже, например:

1) сокрытие смысла (шифр Аве Мария),
2) сокрытие в других объектах (стеганография) и др.

Однако надо исходить из того, что информация тщательно скрытая всё равно может быть найдена, поэтому представление её должно быть таким, что "чужими руками" ею трудно было пользоваться.

Для решения поставленных задач, связанных с информацией, применяются разные математические методы. Если немножечко подумать, то легко сообразить, что вопрос о применении математики в лучшем случае — праздный. Ведь нет ничего более надёжного и бесспорного, чем математика. Поэтому, если чего-то собираетесь утверждать по защите информации и информационной безопасности, то надо иметь в виду чёткое и обоснованное представление о задаче, которое может дать только математика. Кроме того, если собираетесь обосновать важность предложенного решения, то это можно сделать только с помощью математики. Поэтому среди специалистов по защите информации и информационной безопасности очень много крупных математиков, которые, как раз и обеспечивают надёжность принимаемых решений по проблемам защиты информации и информационной безопасности.

В этом курсе невозможно охватить все направления математики, которые важны для защиты информации и информационной безопасности. Основное внимание будет уделено теории чисел и алгебре. Например, за бортом останутся важнейшие вопросы, связанные с теорией вероятностей и математической статистикой.

Возникает естественный вопрос: почему выбраны эти разделы математики, а не другие?

Поскольку мы работаем с числами, то надо их изучать, как правильно с ними работать.

Чуть сложнее дело обстоит с алгеброй. Алгебраический аппарат казался необычайно полезен, поскольку именно с помощью его удаётся построить, проверить и доказать системы защиты информации. Это связано с тем, что именно алгебра первой из математических дисциплин столкнулась с проблемами обоснованности доказательств, их трудности, утверждений о не возможности решений и т.п., что было обусловлено тесными связями алгебры с математической логикой.

Не последнюю роль также играют личные привязанности автора, а они у меня как раз к алгебре и теории чисел.

1. Математические вопросы

1.1. Числа

Должны быть известны следующие классы чисел.

1) $\mathbf{N} = \{1, 2, 3, \dots\}$ — натуральные числа.
2) $\overline{\mathbf{N}} = \{0, 1, 2, 3, \dots\}$ — натуральные числа с нулем (whole numbers).
3) $\mathbf{Z} = \{0, \pm 1, \pm 2, \pm 3, \dots\}$ — целые числа, ими занимается теория чисел.
4) $\mathbf{Q} = \left\{ \frac{p}{q} \mid p \in \mathbf{Z},\ q \in \mathbf{N} \right\}$ — рациональные числа.
5) $\mathbf{R}$ — действительные числа.
6) $\mathbf{C}$ — комплексные числа.

Это далеко не все используемые в математике числа. Например, ещё есть p-адические числа, кватернионы, числа Кэли (октонионы).

В компьютерах не бывает бесконечно много чисел, для информационной безопасности достаточно целых.

Леопольд Кронеккер[2] сказал так:

> "Натуральные числа создал Господь Бог, а все остальное творение рук человеческих".

Хорошо известны следующие свойства натуральных и целых чисел, которые мы будем использовать.

1. Натуральных чисел бесконечно много.
2. Их можно складывать и умножать.
3. Целые числа можно вычитать (натуральные не всегда, например, $2 - 3$ — ненатуральное число).
4. Целые числа можно делить с остатком.

Последнее свойство мы определим, поскольку оно не столь популярно и очевидно, как первые три.

Определение. Деление с остатком:

> Пусть $n, q \in \mathbf{Z}$ и $q \neq 0$. Тогда существуют и единственные целые s, r, для которых
>
> $$sq + r = n, \text{ где } r \in \{0, 1, \dots, |q| - 1\}$$

s — называется ***(неполным) частным***, а r — остатком.

[2] Кронеккер знаменит, в частности, своим знаменитым ***символом Кронеккера*** —

$$\delta_{x,y} = \begin{cases} 1 & \text{при } x = y, \\ 0 & \text{при } x \neq y. \end{cases}$$

Замечание. Иногда используют наименьший по модулю остаток. Ниже приведём соответствующий пример.

Примеры.

1. Пусть $n = -5,\ q = 3$. Тогда

$$-5 = 3(-2) + 1 \text{ — верно}, \quad -5 = 3(-1) - 2 \text{ — неверно}.$$

2. Пусть $n = 5,\ q = 3$. Тогда

$$5 = 3 \cdot 1 + 2,$$
$$5 = 3 \cdot 2 - 1 \text{ — деление с наименьшим по модулю остатком.}$$

Замечания.

1. Деление с остатком — очень медленная операция, особенно для чисел за разрядной сеткой, потому что оно выполняется, как последовательность вычитаний, коих может быть очень много. На этом замечании основан один из приёмов хакерства. Если в ассемблере много раз вычитание, то это скрыто деление.
2. Другим примером длительной операции является возведение в большую степень целого числа. Для действительных чисел обычно используют такой приём. Запишем
$$a^b = e^{b \ln a}.$$
Здесь $\ln a$, $b \ln a$ и $e^{b \ln a}$ можно посчитать очень быстро и с хорошей точностью.
Однако это не годится для целых чисел, поскольку результатом должно быть целое число. Если натуральные числа a, b очень большие, то возникают трудности не только в выполнении, но и в хранении промежуточных и итогового результатов.

Приведённые два почти тривиальных замечания имеют далеко идущие последствия, в том числе и для защиты информации.

Законы для целых чисел Z

1. Ассоциативность $+$ и $\cdot$. Для любых целых чисел a, b, c выполняется

$$(a + b) + c = a + (b + c),$$
$$(ab)c = a(bc).$$

2. Коммутативность $+$ и $\cdot$. Для любых целых чисел a, b выполняется

$$a + b = b + a,$$
$$ab = ba.$$

3. Существование 0 **и** 1. Существуют такие целые числа 0 и 1, что для любого целого числа a

$$a + 0 = a,$$
$$a \cdot 1 = a.$$

4. Существование противоположного. Для любого целого числа a существует такое целое число $-a$, что

$$a + (-a) = 0.$$

С абстрактной точки зрения эти законы утверждают, что $\mathbf{Z}$ относительно сложения и умножения является

ассоциативным коммутативным кольцом с единичным элементом 1.

Приведённые законы являются основополагающими для многих других конструкций, связанных с числами. Отметим, что эти законы выполняются для рациональных чисел $\mathbf{Q}$, действительных чисел $\mathbf{R}$ и комплексных чисел $\mathbf{C}$. Стоит отметить, что для $\mathbf{Q}$, $\mathbf{R}$ и $\mathbf{C}$ выполняется ещё один важный закон.

5. Существование обратного. Для любого числа $a \neq 0$ существует такое число a^{-1}, что

$$a \cdot a^{-1} = 1.$$

Теперь перейдём к изложению конструкции, основанной на целых числах, и которая играет большую роль во многих вопросах защиты информации и информационной безопасности.

Зафиксируем натуральное число n.

Определение. Два целых числа a и b назовём ***эквивалентными или сравнимыми по модулю*** n и будем писать $a \sim b$, если n делит $a - b$ символически $n \mid a - b$. Другими словами, a и b имеют одинаковые остатки от деления на n.

Возникают классы эквивалентности,

0) $[0] = [0]_n$ — класс, состоящий из чисел, которые при делении на n дают остаток 0;

1) $[1] = [1]_n$ — класс, состоящий из чисел, которые при делении на n дают остаток 1;

$\vdots$

$n-1)$ $[n-1] = [n-1]_n$ — класс, состоящий из чисел, которые при делении на n дают остаток $n-1$.

Таким образом, получается множество $\mathbf{Z}_n = \mathbf{Z}/n\mathbf{Z}$, состоящее точно из n классов:

$$\mathbf{Z}_n = \mathbf{Z}/n\mathbf{Z} = \{[0], [1], \ldots, [n-1]\}.$$

На этом множестве классов $\mathbf{Z}_n$ введём операции $+$ (сложение) и $\cdot$ (умножение). Для любых целых чисел a, b

$$[a] + [b] = [a+b],$$
$$[a][b] = [ab].$$

Замечания.

1. $+$ и $\cdot$ определены корректно на $\mathbf{Z}_n$. Это означает, что если $[a] = [a_1]$ и $[b] = [b_1]$, то

$$[a] + [b] = [a+b] = [a_1 + b_1] = [a_1] + [b_1],$$
$$[a][b] = [ab] = [a_1 b_1] = [a_1][b_1].$$

2. Если $n = 1$, то $\mathbf{Z}_1 = \{0\}$. Этот случай не интересен.

На множестве $\mathbf{Z}_n$ относительно введённых операций сложения и умножения классов выполнены условия.

1. Ассоциативность $+$ и $\cdot$. Для любых классов $[a], [b], [c]$

$$([a] + [b]) + [c] = [a] + ([b] + [c]), \quad ([a][b])[c] = [a]([b][c]).$$

2. Коммутативность $+$ и $\cdot$. Для любых классов $[a], [b]$

$$[a] + [b] = [b] + [a], \quad [a][b] = [b][a].$$

3) Существование 0 и 1. Существуют такие классы $[0]$ и $[1]$, что для любого класса $[a]$

$$[a] + [0] = [a], \quad [a] \cdot [1] = [a].$$

4. Существование противоположного. Для любого класса $[a]$ существует такой класс $-[a]$, что

$$[a] + (-[a]) = [0].$$

Определение. Множество $\mathbf{Z}_n$ относительно введённых операций $+$ сложения и $\cdot$ умножения называется ***кольцом классов вычетов по модулю*** n.

Замечание. Есть некоторые традиционные соглашения при работе с кольцами классов вычетов. Так, например, когда в $\mathbf{Z}_n$

$$[a] = [b],$$

то пишут

$$a \equiv b \pmod{n}$$

читается как a сравнимо с b по модулю n.

Допускается даже запись

$$a = b \pmod{n}.$$

Поэтому пишут, к примеру, для модуля 4

$$[2]_4 + [3]_4 = [1]_4, \text{ или } 2 + 3 \equiv 1 \pmod{4}, \text{ или } 2 + 3 = 1 \pmod{4}.$$

Кольцо $\mathbf{Z}_4$

Воспользуемся теми соглашениями, которые были приведены в последнем замечании. Так будем считать, что по модулю 4

$$2 + 3 = 1 \quad \text{и} \quad 2 \cdot 3 = 2.$$

Получаются следующие таблицы сложения и умножения

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

$\cdot$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Обратим внимание, что $2 \cdot 2 \equiv 0 \pmod{4}$.

Кольцо $\mathbf{Z}_3$

Получаются следующие таблицы сложения и умножения

$+$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

$\cdot$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Кольцо $\mathbf{Z}_2$

Имеем таблицы сложения и умножения

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Замечание. Отметим, что в таблицах умножения для $\mathbf{Z}_2$ и $\mathbf{Z}_3$ ***нет*** ненулевых элементов, произведение которых равно нулю, а для $\mathbf{Z}_4$ есть. Это объясняется тем, что 2 и 3 — простые числа, а 4 не является простым.

Определение. Натуральное число $p > 1$ называется ***простым***, если оно делится только на 1 и p.

Начальное множество простых чисел

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, \ldots.$$

Теорема (Евклид)**.** *Множество простых чисел бесконечно.*

Следствие. *Существуют сколь угодно большие простые числа.*

Теорема (Эйлер)**.** *Ряд*

$$\sum_{p\ -\ простое} \frac{1}{p}$$

расходится.

Отсюда, также следует бесконечность множества простых чисел.

Теорема (Основная теорема арифметики целых чисел)**.** *Пусть $a > 1$ — натуральное число. Тогда существуют единственные простые числа*

$$p_1 < p_2 < \cdots < p_n$$

и единственные натуральные числа

$$\alpha_1, \alpha_2, \ldots, \alpha_n,$$

для которых

$$a = p_1^{\alpha_1} p_2^{\alpha_2} \ldots p_n^{\alpha_n}.$$

Таким образом,

> Всякое натуральное число большее 1 разложимо в произведение простых.

Возникает трудная проблема.

Проблема (факторизации)**.** *Разложение произвольного натурального числа на простые множители.*

Если натуральное число a — большое и его простые множители

$$p_1, \ldots, p_n$$

— тоже большие, то трудно найти даже один простой делитель данного числа a, ибо для этого нужно очень много делений, а это требует много времени. Ситуация усугубляется тем, что простых чисел бесконечно много и на составление сверхбольших таблиц простых чисел невозможно.

Отметим, что решение проблемы факторизации трудно даже в случае, когда известно, что

$$a = pq \text{ и } p, q \text{ — простые.}$$

Об этом позже более подробно.

Обозначение. Для каждого действительного положительного числа x обозначим через $\pi(x)$ количество простых чисел, меньших x.

Пример. Имеем

$$\pi(2) = 0, \pi(3) = 1, \pi(4) = 2, \pi(10) = 4, \pi(100) = (25),$$
$$\pi(1\,000) = 168, \pi(1\,000\,000) = 78\,498, \pi(10^9) = 50\,847\,534, \ldots.$$

Теорема (Асимптотический закон распределения простых чисел (Ж. Адамар, Ш. Валле-Пуссен, А. Сельберг, П. Эрдёш)[3])**.**

$$\lim_{x\to\infty} \frac{\pi(x)}{x/(\ln x)} = 1.$$

Следствие. *При больших* x

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

Эта оценка очень грубая, но даёт представление о порядке роста функции $\pi(x)$.

[3]Эту теорему впервые независимо доказали в 1896 году Адамар и Валле-Пуссен с использованием комплексного анализа, в 1949 без использования комплексного анализа доказательство теоремы получили Сельберг и Эрдёш, но последний использовал один важный результат Сельберга.

Пример. Подсчитаем примерное количество простых чисел на промежутке $[10^{20}, 10^{30})$, которое оценим с помощью следствия как

$$T = \pi(10^{30}) - \pi(10^{20}) \approx \frac{10^{30}}{\ln(10^{30})} - \frac{10^{20}}{\ln(10^{20})}.$$

Имеем

$$\begin{aligned}
\frac{10^{30}}{\ln(10^{30})} &= \frac{10^{30}}{69,077552789821} \approx 1,4476482730108 \cdot 10^{28},\\
\frac{10^{20}}{\ln(10^{20})} &= \frac{10^{20}}{46,051701859881} \approx 2,1714724095163 \cdot 10^{18},\\
T &\approx \frac{10^{30}}{\ln(10^{30})} - \frac{10^{20}}{\ln(10^{20})} > 10^{28} - 3 \cdot 10^{18} =\\
&= 10^{18}(10^{10} - 3) > 10^{27}.
\end{aligned}$$

В действительности, $\pi(10^{20}) = 2\,220\,819\,602\,560\,918\,840$.

Кроме того, возникает очень сложная подпроблема.

Задача. *Для данного натурального числа определить является ли оно простым.*

Теорема (2002, Agrawal, Kayal, Saxena)**.** *Существует полиномиальный алгоритм проверки числа на простоту.*

Однако, пока этот алгоритм не реализуем, так как требует слишком много памяти. Для проверки простоты числа используют либо экспоненциальные, либо вероятностные алгоритмы.

Кроме проблемы, связанной с делением, отмечена была другая проблема — возведение в большую степень. Эта проблема приводит к другой проблеме.

Проблема (дискретного логарифма)**.** *Для данных натуральных a, b и n найти такое натуральное число x, что*

$$a^x \equiv b \pmod{n}.$$

Определение. Это число x называется ***дискретным логарифмом числа*** b ***по основанию*** a ***для модуля*** n или ***индексом числа*** b ***по основанию*** a ***для модуля*** n.

Замечание. Не всегда есть решение данной задачи. Например, для модуля 4 она не имеет решения для $b \in \{0, 2\}$, если $a \notin \{0, 2\}$. Обычно выдвигается дополнительное требование

$$a \text{ и } b \text{ взаимно просты с } n.$$

Теорема (1801, Гаусс)**.** *Пусть p — простое число. Тогда существует такое натуральное число $g \in \{1, 2, \ldots, p-1\}$, что для любого $b \in \{1, 2, \ldots, p-1\}$ найдётся такое $x \in \{1, 2, \ldots, p-1\}$, что*

$$g^x \equiv b \pmod{p}.$$

Определение. Это число g называется ***примитивным (первообразным) корнем по модулю*** p.

Приведём таблицу некоторых простых чисел и их наименьших примитивных корней.

модуль	примитивный корень
2	1
3	2
5	2
7	3
23	5
71	7
109	6
191	19
311	17
313	10
409	21
439	15

1.2. Алгебра

1.2.1. Операции

Определение. Пусть X — непустое множество. Тогда всякое отображение

$$X \times X \to X$$

называется ***бинарной алгебраической операцией на*** X.

Таким образом, бинарная алгебраическая операция сопоставляет упорядоченной паре (x, y) элементов из $X \times X$ элемент из X. В алгебре очень часто рассматриваются две операции.

Определения.

1. Операция $\cdot$ и её обычно называют ***умножение***. Результат умножения записывают, как

$$x \cdot y, \text{ а не } \cdot(x, y) \text{ как в анализе.}$$

Часто, если это не вызывает путаницы пишут

$$x \cdot y = xy.$$

2. Операция $+$ и её обычно называют ***сложение***. Результат сложения записывают, как

$$x + y, \text{ а не } +(x, y) \text{ как в анализе.}$$

1.2.2. Полугруппы

Определение. Непустое множество S с операцией $\cdot$ называется ***полугруппой,*** если выполнено условие *ассоциативности*

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c$$

для любых элементов $a, b, c \in S$

Теорема. *Нет алгоритма для определения в произвольной полугруппе равенства двух элементов.*

Эта теорема доказана в 1947 г. Марковым Андреем Андреевичем (младшим) (1903–1979) и Постом (Post Emil Leon, 1897–1954). Она решает ***Проблему Туэ*** (Axel Thue).

Это первый пример «внутриматематической» неразрешимой массовой алгоритмической проблемы.

Пример. В 1956 г. ученик Маркова Цейтин построил следующий простой пример полугруппы, удовлетворяющей теореме Маркова–Поста.

В полугруппе S с порождающими

$$a, b, c, d, e$$

и соотношениями

$$ac = ca, ad = da, bc = cb, bd = db, abac = abace, eca = ae, edb = be$$

нет алгоритма для решения проблемы равенства слов.

Замечание. Поясним, что это означает. Полугруппа S является набором слов в алфавите a, b, c, d, e. Два слова в S равны, если от одного к другому можно перейти последовательным применением конечного числа указанных соотношений. Если получили при пересылке два слова, то мы не можем узнать равны ли они друг другу!

1.2.3. Группы

Определение. Непустое множество G, на котором задана алгебраическая операция $\cdot$, называется ***группой***, если выполнены следующие условия.

1. Ассоциативность. Для любых элементов $a, b, c \in G$

$$a \cdot (b \cdot c) = (a \cdot b) \cdot c.$$

2. Существование единичного (нейтрального) элемента. Существует такой элемент $1 \in G$, что для любого $a \in G$

$$1 \cdot a = a \cdot 1 = a.$$

3. Существование обратного элемента. Для любого элемента $a \in G$ существует такой элемент $a^{-1} \in G$, что

$$a \cdot a^{-1} = a^{-1} \cdot a = 1,$$

где элемент $1 \in G$ выбран согласно условию 2).

Замечание. Чтобы проверить, что некоторое множество является группой необходимо проверить следующие условия.

-1. Непустота.

0. Задание операции. Обычно это сводится к тому, чтобы результат операции для любых двух элементов из множества снова давал элемент данного множества.

1. Ассоциативность.

2. Существование единичного (нейтрального) элемента.

3. Существование обратного элемента.

Ясно, что условие 2) сразу даёт непустоту, то есть условие -1). Поэтому условие -1) излишне, но, представляется не очень разумным, что-то проверять, а потом обнаружить, что данное множество пусто.

Замечание. Правильно говорить — ***группа относительно операции*** $\cdot$. Однако часто будем говорить просто ***группа***, подразумевая, что операция на группе известна.

Теорема (о простейших свойствах групп)**.**
Пусть G — группа (относительно операции $\cdot$). Тогда:

1) *единичный элемент единствен,*
2) *для любого элемента обратный к нему единствен.*

Доказательство.

1) В самом деле, пусть 1 и e — единичные элементы. Тогда

$$e \overset{\text{2) для 1}}{=} 1 \cdot e \overset{\text{2) для } e}{=} 1, \text{ значит } e = 1.$$

Поясним, что $e \overset{\text{2) для 1}}{=} 1$ означает применение свойства 2) из определения группы к соотношению $e = 1$. В остальных случаях утверждений 1) и 2) такие утверждения понимать нужно также.

2) Пусть a^{-1} и a' — элементы, обратные $a \in G$. Тогда имеем

$$a \cdot a^{-1} = a^{-1} \cdot a = 1 = a \cdot a' = a' \cdot a.$$

Отсюда

$$a' \overset{2)}{=} 1 \cdot a' \overset{\text{3) для } a^{-1}}{=} (a^{-1}a)a' \overset{1)}{=} a^{-1}(aa') \overset{\text{3) для } a'}{=} a^{-1} \cdot 1 \overset{2)}{=} a^{-1}.$$

Поэтому

$$a' = a^{-1}.$$

Теорема доказана. □

Замечание. Доказанная теорема говорит о том, что корректно определена операция — ***взятие обратного***

$$^{-1} : G \to G$$

и кроме того, есть ***выделенный*** элемент, а именно, $1 \in G$.

Определение. Группа G называется ***абелевой (коммутативной)***, если выполнено условие коммутативности

$$ab = ba.$$

для любых $a, b \in G$.

Часто операцию в абелевой группе записывают через $+$.

Определение. Группа называется ***конечной***, если в ней конечное число элементов. Число элементов в конечной группе G называется ***порядком группы*** группы G и обозначается

$$|G|.$$

1.2.4. Перестановки

Зафиксируем непустое множество X.
Обозначим через $S(X)$ — множество всех биекций X на X.

Определение. Отображение

$$f : X \to X$$

называется ***биекцией***, если выполнены следующие два условия.

Сюръективность (отображение на). Для любого $y \in X$ существует такой $x \in X$, что

$$f(x) = y.$$

Инъективность (взаимнооднозначность). Для любых $x \neq y$, $x, y \in X$ выполняется

$$f(x) \neq f(y),$$

или равносильно

$$f(x) = f(y) \longleftrightarrow x = y.$$

Если в качестве операции на $S(X)$ взять композицию (суперпозицию, последовательное выполнение $(f \circ g)(x) = f(g(x))$ для любых $f, g \in S(X)$ и $x \in X$) $\circ$ биекций, то $S(X)$ становится группой относительно данной операции. Эту группу будем называть ***симметрической группой*** на множестве X. В самом деле, имеем.

-1. Непустота. Множеству $S(X)$ всегда принадлежит ***тождественное*** отображение

$$\varepsilon : X \to X, \text{ где } \varepsilon(x) = x \text{ для любого } x \in X.$$

0. Задание операции. Хорошо известно, что композиция (суперпозиция) биекций снова биекция.

1. Ассоциативность. Это общее свойство отображений.

2. Существование единичного (нейтрального) элемента. Тождественное отображение является единичным элементом для композиции (суперпозиции). Действительно, пусть $f \in S(X)$. Тогда для любого $x \in X$

$$(f \circ \varepsilon)(x) = f(\varepsilon(x)) = f(x) \quad \text{и} \quad (\varepsilon \circ f)(x) = \varepsilon(f(x)) = f(x),$$

то есть

$$f \circ \varepsilon = f = \varepsilon \circ f.$$

3. Существование обратного элемента. Хорошо известно, что обратное отображение к биекции снова биекция.

Группы, состоящие из биекций данного множества, играют огромную роль не только в алгебре, но и в многочисленных приложениях в других областях математики, механики, физики, химии.

Отметим, следующий важный факт.

Теорема (о неабелевости симметрической группы)**.** *Если в множестве X, по крайней мере,* 3 *элемента, то симметрическая группа $S(X)$ неабелева.*

Доказательство. Пусть x_1, x_2 и x_3 — три различных элемента множества X. Рассмотрим биекции

$$f_{12}(x) = \begin{cases} x_2 & \text{при } x = x_1, \\ x_1 & \text{при } x = x_2, \\ x & \text{при } x \notin \{x_1, x_2\} \end{cases} \quad \text{и } f_{13}(x) = \begin{cases} x_3 & \text{при } x = x_1, \\ x_1 & \text{при } x = x_3, \\ x & \text{при } x \notin \{x_1, x_3\} \end{cases}.$$

Тогда

$$(f_{12} \circ f_{13})(x_1) = f_{12}(f_{13}(x_1)) = f_{12}(x_3) = x_3,$$

но

$$(f_{13} \circ f_{12}((x_1) = f_{13}(f_{12}(x_1)) = f_{13}(x_2) = x_2.$$

Так как $x_2 \neq x_3$, то

$$f_{12} \circ f_{13} \neq f_{13} \circ f_{12},$$

и симметрическая группа $S(X)$ неабелева.

Теорема доказана. □

Нас будут интересовать группы биекций ***конечного*** множества

$$X = \{x_1, \ldots, x_n\}.$$

Определение. Легко заметить, что для данного числа n (количества элементов в X) все симметрические группы $S(X)$ "устроены одинаково", и потому обозначаются через S_n и называются ***симметрической группой степени*** n.

В самом деле, если задана нумерация элементов множества X, то совершенно не важна природа элемента, важен только его номер. Поэтому всякая биекция полностью определяется, как отображение из множества номеров $\{1, \ldots, n\}$ на себя.

В теории конечных групп принято это записывать так. Если $f \in S_n$, то

$$f = \begin{pmatrix} 1 & \dots & j \dots & n \\ f(1) & \dots & f(j) \dots & f(n) \end{pmatrix} = \begin{pmatrix} 1 & \dots & j \dots & n \\ i_1 & \dots & i_j \dots & i_n \end{pmatrix}.$$

Такая запись называется записью в виде ***подстановки***.

Также ясно, что вторая строчка $(i_1, \dots, i_j \dots, i_n)$ является перестановкой чисел $\{1, \dots, j, \dots, n\}$. С другой стороны, каждой перестановке чисел $\{1, \dots, j, \dots, n\}$ можно сопоставить биективно подстановку. Поэтому элементы симметрической группы S_n называются либо ***перестановками***, либо ***подстановками***.

Отметим, что композицию перестановок из S_n обычно называют ***умножением*** перестановок, результат ***произведением*** этих перестановок.

Пример. Пусть

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 5 & 7 & 3 & 1 & 2 & 8 & 6 & 4 \end{pmatrix}$$

и

$$g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \end{pmatrix}.$$

Тогда

$$f \circ g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 8 & 7 & 6 & 5 & 4 & 3 & 2 & 1 \\ 4 & 6 & 8 & 2 & 1 & 3 & 7 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 4 & 6 & 8 & 2 & 1 & 3 & 7 & 5 \end{pmatrix}.$$

Так на первом месте перестановки может любое число от 1 до n, то для первого места будет n возможностей, а на втором месте может стоять только любое число отличное от числа на первом месте, поэтому будет $n-1$ возможностей и т. д.. Поэтому перестановок чисел $\{1, \dots, j, \dots, n\}$ будет

$$n \cdot (n-1) \cdot \dots \cdot 2 \cdot 1 = n!,$$

а отсюда получаем, что порядок симметрической группы степени n

$$|S_n| = n!$$

В задачах обеспечения защиты информации очень широко используются группы, состоящие из перестановок. Одна из причин состоит в том, что перестановок необычайно много. Более точно. Известна формула Стирлинга

$$n! = \sqrt{2\pi n} \left(\frac{n}{e}\right)^n e^{\theta_n}, \text{ где } \frac{1}{12n+1} < \theta_n < \frac{1}{12n}.$$

Например, в системе шифрования AES используется перестановка на 64 символах, а всего перестановок на 64 символах

$$|S_{64}| = 64! > \sqrt{2\pi 64} \left(\frac{64}{e}\right)^{64} = 8\sqrt{2\pi} \left(\frac{64}{e}\right)^{64} \approx 10^{89}.$$

1.2.5. Числовые группы

Рассмотрим числовые множества $\mathbf{Z}$, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{C}$ и $\mathbf{Z}_n$. Последнее множество состоит не из чисел, а из подмножеств (классов вычетов) чисел, но поскольку числа являются основным строительным материалом для данного множества, мы поместили его в этот раздел.

По сложению

$\mathbf{Z}$, $\mathbf{Q}$, $\mathbf{R}$ и $\mathbf{C}$ — бесконечные абелевы группы, а $\mathbf{Z}_n$ — конечная абелева группа порядка n относительно сложения.

По умножению

Если $\mathbf{Q}^* = \mathbf{Q} \setminus \{0\}$, $\mathbf{R}^* = \mathbf{R} \setminus \{0\}$ и $\mathbf{C}^* = \mathbf{C} \setminus \{0\}$, то $\mathbf{Q}^*$, $\mathbf{R}^*$ и $\mathbf{C}^*$ — бесконечные абелевы группы относительно умножения.

Единственными целыми числами, обратными к которым являются тоже целые, это — $\{1, -1\}$. Поэтому группа по умножению

$$\mathbf{Z}^* = \{1, -1\}.$$

Можно понять, что

$$\mathbf{Z}_n^* = \{[a] \mid \text{числа } n \text{ и } a \text{ взаимно просты}\}$$

является группой по умножению.

Пример. Для рассмотренных ранее примеров имеем:

$$\mathbf{Z}_2^* = \{[1]\}, \qquad \mathbf{Z}_3^* = \{[1], [2]\}, \qquad \mathbf{Z}_4^* = \{[1], [3]\}.$$

1.2.6. Кольца. Матрицы

Определение. ***Ассоциативным кольцом с*** 1 является непустое множество K с двумя операциями $+$ и $\cdot$, имеющее следующие свойства.

K — **коммутативная группа** относительно $+$.

Дистрибутивность. Для любых $a, b, c \in K$ выполняется

$$a(b + c) = ab + ac, \ (a + b)c = ac + bc.$$

Ассоциативность умножения. Для любых $a, b, c \in K$ выполняется

$$a(bc) = (ab)c.$$

Существование 1. Существует такой элемент 1 (единичный элемент) в K, что для любого $a \in K$ выполняется

$$1 \cdot a = a \cdot 1 = a.$$

Примерами колец служат множества с рассмотренными ранее операциями

$$\mathbf{Z},\ \mathbf{Q},\ \mathbf{R},\ \mathbf{C} \text{ и } \mathbf{Z}_n.$$

Обозначим через $M_{m\times n}(K)$ — множество всех матриц размера $m \times n$ над кольцом K.

Тогда $M_{m\times n}(K)$ — абелева группа относительно сложения матриц.

Гораздо сложнее дело обстоит с умножением матриц.

Пусть $M_n(K) = M_{n\times n}(K)$ — множество всех квадратных матриц размера $n \times n$ (квадратных матриц порядка n) над кольцом K. Рассмотрение таких матриц будет нам гарантировать, что произведение двух квадратных матриц порядка n является квадратной матрицей порядка n. Можно понять из свойств операций над матрицами, что относительно сложения и умножения матриц

$$M_n(K) \text{ является ассоциативным кольцом с } 1.$$

Определение. Элемент d ассоциативного кольца K с 1 называется ***обратимым***, если существует такой элемент $d' \in K$, что

$$dd' = d'd = 1.$$

Обозначим множество всех обратимых элементов кольца K через K^*.

Теорема (о мультипликативной группе кольца.)**.** *Пусть K — ассоциативное кольцо с 1. Тогда K^* является группой относительно умножения (более точно относительно ограничения умножения на K^*).*

Определение. K^* называется ***мультипликативной группой ассоциативного кольца K с*** 1.

Доказательство. В самом деле, имеем.

-1. Непустота. $1 \in K^*$.

0. Задание операции. Пусть $d, d_1 \in K^*$ и тогда существуют такие $d', d_1' \in K$, что

$$dd' = d'd = 1, \quad d_1d_1' = d_1'd_1 = 1.$$

Тогда

$$(dd_1)(d_1'd') = d(d_1d_1')d' = d \cdot 1 \cdot d' = dd' = 1$$

и

$$(d_1'd')(dd_1) = d_1'(d'd)d_1 = d_1' \cdot 1 \cdot d_1 = d_1'd_1 = 1.$$

1. Ассоциативность. Выполняется для всех элементов кольца K. Значит, тем более, для его части K^*.

2. Существование единичного элемента. Единичным элементом является $1 \in K^*$.

3. Существование обратного элемента. Так как по определению $dd' = d'd = 1$, то $d' \in K^*$ — обратный элемент для $d \in K^*$.

Теорема доказана. □

Примерами мультипликативных групп колец служат рассмотренные ранее группы

$$\mathbf{Z}^*,\ \mathbf{Q}^*,\ \mathbf{R}^*\ \mathbf{C}^* \text{ и } \mathbf{Z}_n^*.$$

Определение. Пусть K — ассоциативное кольцо с 1. Тогда мультипликативная группа кольца матриц $M_n(K)$ обозначается

$$GL(n, K) = GL_n(K)$$

и называется ***общей (полной) линейной группой степени*** n ***над кольцом*** K***.***

Замечание. Следует обратить внимание на одно исключительное кольцо, а именно, ***нулевое*** кольцо, состоящее из одного элемента $\{0\}$. Например, таковым является кольцо $\mathbf{Z}_1$. Для любого размера $m \times n$ есть только одна матрица над нулевым кольцом — ***нулевая*** $O_{m,n}$. В этом случае для любого натурального n

$$GL(n, \{0\}) = \{O_{n,n}\} \text{ — абелева группа порядка 1.}$$

Если же кольцо ненулевое, то $1 \neq 0$. В самом деле, для любого элемента a кольца

$$0 \cdot a = (0 + 0)a = 0 \cdot a + 0 \cdot a, \text{ что даёт } 0 \cdot a = 0.$$

Поэтому, если $1 = 0$, то для ненулевого элемента a кольца

$$a = 1 \cdot a = 0 \cdot a = 0,$$

противоречие с выбором a. Это рассуждение доказывает следующее утверждение.

> Ассоциативное кольцо с 1 — ненулевое тогда и только тогда, когда $1 \neq 0$.

Если же кольцо K ненулевое, то для любого натурального $n \geqslant 2$ группа $GL(n, K)$ неабелева. Так как

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

то

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \in GL(n, K) \quad \text{и аналогично} \quad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \in GL(n, K).$$

Теперь

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1+1 & 1 \\ 1 & 1 \end{pmatrix} \text{ и } \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1+1 \end{pmatrix}.$$

Если бы эти произведения были равны, то

$$1 + 1 = 1, \text{ что даёт } 1 = 0,$$

что невозможно, как отмечено ранее. Таким образом, доказана неабелевость $GL(2, K)$ над ненулевым кольцом K.

В случае $n > 2$ достаточно рассмотреть матрицы

$$\begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \text{ и } \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Доказательство того, что данная квадратная матрица обратима, может быть очень тяжёлым, если кольцо устроено сложно. Ситуация с обратимостью матриц полностью (ну, почти полностью) проясняется в случае коммутативных колец.

Определение. Кольцо K называется ***коммутативным***, если для любых элементов a и b кольца K выполнено следующее условие ***коммутативности***

$$a \cdot b = b \cdot a.$$

Для квадратных матриц над ассоциативным коммутативным кольцом с 1 работает теория определителей. В качестве непосредственного следствия теоремы об умножении определителей получается следующее свойство.

> Квадратная матрица A над ассоциативным коммутативным кольцом K с 1 обратима тогда и только тогда, когда её определитель $\det A$ обратим, то есть $\det A \in K^*$.

Таким образом, в частности

> квадратная матрица A над $K \in \{\mathbf{Q}, \mathbf{R}, \mathbf{C}\}$ обратима тогда и только тогда, когда её определитель $\det A$ обратим, что равносильно тому, что $\det A \neq 0$.

Интереснее ситуация для $\mathbf{Z}$.

> Квадратная матрица A над $\mathbf{Z}$ обратима тогда и только тогда, когда $\det A \in \{1, -1\}$.

Отсюда получаем любопытное следствие.

> Если квадратная матрица A над $\mathbf{Z}$ с $\det A \neq 0$ необратима над $\mathbf{Z}$, то обратима над $\mathbf{Q}$, а это означает, что среди элементов обратной матрицы A^{-1} над $\mathbf{Q}$ ***обязательно есть нецелое число***.

Пример. Над $\mathbf{Q}$

$$\det \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = -2, \text{ значит } \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \in GL(2, \mathbf{Q}) \setminus GL(2, \mathbf{Z}).$$

В самом деле

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}^{-1} = \frac{1}{-2} \begin{pmatrix} 4 & -2 \\ -3 & 1 \end{pmatrix} = \begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}.$$

1.2.7. Многочлены

Кроме перестановок и матриц в работе с информацией часто используются многочлены (= полиномы).

Пусть K — ассоциативное коммутативное кольцо с 1.

Определение. ***Многочлен*** — это любая конечная сумма

$$f = \sum_{i=0}^{n} a_i x^i, \text{ где } a_i \in K \text{ для любого } i \in \{0, 1, \ldots, n\}.$$

Здесь x является значком, который можно заменить на любую другую букву. Важны только коэффициенты при степенях x. Поэтому с абстрактной точки зрения многочлен можно понимать как бесконечную последовательность

$$f = (a_0, a_1, \ldots, a_n, \ldots),$$

в которой ***только конечное число ненулевых членов***, или как бесконечную сумму

$$f = a_0 \cdot 1 + a_1 x + \cdots + a_n x^n + \cdots = \sum_{i=0}^{\infty} a_i x^i = \sum_{i=0} a_i x^i,$$

в которой ***только конечное число ненулевых коэффициентов.***

Определение. Многочлены можно складывать почленно. А именно, если

$$f = \sum_{i=0} a_i x^i \text{ и } g = \sum_{i=0} b_i x^i, \text{ то } f + g = \sum_{i=0} (a_i + b_i) x^i.$$

Можно проверить, что множество многочленов $K[x]$ является абелевой группой относительно сложения.

Определение. Если для любых целых неотрицательных i и j

$$x^i \cdot x^j = x^{i+j},$$

то многочлены можно перемножать, приводя подобные члены. Более точно

$$f \cdot g = \left(\sum_{i=0} a_i x^i \right) \cdot \left(\sum_{j=0} b_j x^j \right) = \sum_{k=0} \left(\sum_{i+j=k} a_i b_j \right) x^k.$$

Можно проверить, что множество многочленов $K[x]$ является ассоциативным коммутативным кольцом с 1 относительно сложения и умножения. $K[x]$ называется ***кольцом многочленов от одного неизвестного над кольцом*** K.

Далее можно рассуждать как при построении кольца вычетов $\mathbf{Z}_n = \mathbf{Z}/n\mathbf{Z}$. Зафиксируем некоторый многочлен $p \in K[x]$. Пусть

$$pK[x] = \{pf \mid f \in K[x]\}.$$

Определение. Будем считать два многочлена g и h из $K[x]$ ***эквивалентными по модулю*** p и писать

$$g \equiv h \pmod{p}, \text{ если } g - h \in pK[x].$$

Возникают классы эквивалентности по модулю p

$$[f] = \{g \in K[x] \mid f \equiv g \pmod{p}\}.$$

Множество классов эквивалентности по модулю p

$$K[x]/pK[x] = \{[f] \mid f \in K[x]\}$$

становится ассоциативным коммутативным кольцом с 1 относительно сложения и умножения, которые определяются следующим образом

$$[f] + [g] = [f + g] \quad \text{и} \quad [f][g] = [fg].$$

Отметим, что система шифрования AES использует многочлен

$$p = 1 + x^4 \in \mathbf{Z}_2[x].$$

1.3. Поля

Определение. Ассоциативное коммутативное кольцо F с 1 называется ***полем***, если

1) $1 \neq 0$;
2) любой ненулевой элемент обратим, то есть для любого $a \neq 0$ из F существует такой элемент $a^{-1} \in F$, что

$$a \cdot a^{-1} = 1.$$

Замечания.

1. Так как $1 \neq 0$, то в любом поле ***не меньше, чем*** 2 ***элемента.*** Полей из одного элемента не бывает!
2. Так как любой ненулевой элемент обратим, то это означает, что $F^* = F \setminus \{0\}$ — группа по умножению.

Примеры.

1. Полями являются $\mathbf{Q}$, $\mathbf{R}$ и $\mathbf{C}$.
2. $\mathbf{Z}$ не является полем, поскольку, например $2x \neq 1$ для всех $x \in \mathbf{Z}$.
3. $\mathbf{F}_p = \mathbf{Z}_p$ является полем для любого простого p (можно понять, если n — непростое число, то $\mathbf{Z}_n$ не является полем). Особую роль играет поле $\mathbf{F}_2 = \mathbf{Z}_2$ (вспомните о байтах!)

Отметим, что при рассмотрении вопросов, связанных с информацией, важную роль играют конечные поля (поля, состоящие из конечного числа элементов). Теория конечных полей очень богата и разнообразна, мы укажем только два основополагающих факта.

Теорема.

1) Существование. *Пусть n — натуральное число. Поле из n элементов существует тогда и только тогда, когда n — степень простого числа.*

2) Примитивный элемент. *В любом конечном поле* $\mathbf{F}$ *существует такой элемент* g*, что любой ненулевой элемент* $a \in \mathbf{F}$ *является степенью элемента* g *с неотрицательным показателем, то есть*

$$a = g^k \text{ для подходящего целого } k \geqslant 0.$$

Такой элемент g *называют* **примитивным элементом поля F.**

Обозначение. Если $q = p^n$ для простого p и натурального n, то поле из q элементов обозначается $\mathbf{F}_q = GF(q)$.[4]

Замечания.

1. Из первого утверждения теоремы следует, что существуют поля из

   $$2, 3, 4, 5, 7, 8, 9, 11, 13, 16, 17, 19, 23, 25, 27, 29, 31, 32, 37, 41, \ldots$$

   элементов, и НЕ существуют поля из

   $$6, 10, 12, 14, 15, 18, 20, 21, 24, 26, 28, 30, 33, 34, 35, 36, 38, 40, \ldots$$

   элементов.
2. Также следует отметить, что для любого простого p и любого натурального $n \geqslant 2$

   $$\mathbf{F}_{p^n} \neq \mathbf{Z}_{p^n}.$$

3. Так как $\mathbf{Z}_p$ является полем для простого p, то понятие примитивного элемента конечного поля обобщает понятие примитивного (первообразного) элемента по модулю p.

[4] GF — сокращение от Galois Field (поле Галуа).

2. Теория информации и кодирования

Считается, что для работы с информацией используются два вида информации:

- дискретная (цифровая);
- непрерывная (аналоговая), обычно это колебания (волны).

Важную роль играет следующий результат, связывающий эти виды информации.

Теорема (отсчётов)**.** *Пусть заданы положительные действительные числа* v_G *и* t_s*. Пусть функция* $f : \mathbf{R} \to \mathbf{R}$ *определена как интеграл*

$$f(t) = \int_0^{v_G} (a(v)\cos(2\pi vt) + b(v)\sin(2\pi vt))dv. \qquad (*)$$

Тогда, если

$$t_s \leqslant \frac{1}{2v_G}, \qquad (**)$$

то

$$f(t) = \sum_{n\in\mathbf{Z}} f(nt_s)\frac{\sin\left(\frac{\pi t}{t_s} - \pi n\right)}{\frac{\pi t}{t_s} - \pi n} \qquad (***)$$

Замечание. Проясним содержание этой замечательной теоремы.

(***) Даёт нам, что, зная только значения функции f в дискретном наборе аргументов $\{nt_s\}_{n\in\mathbf{Z}}$, мы знаем всё о функции $f(t)$ (*), при условии, что частота отсчёта удовлетворяет

$$\frac{1}{t_s} \leqslant 2v_G \text{ равносильно } t_s \leqslant \frac{1}{2v_G}$$

то есть отсчёты должны быть довольно частыми. v_G называется ***шириной полосы пропускания***.

Итак, непрерывную информацию можно успешно дискретизировать.

Возникает интерполяционная функция

$$\frac{\sin\frac{\pi t}{t_s}}{\frac{\pi t}{t_s}},$$

которая считается равной 1 при $t = 0$ по первому замечательному пределу и равна 0 в остальных точках отсчёта nt_s.

В.А. Котельников (1933) доказал эту теорему и использовал для радио[5]. К. Шеннон (1949) тоже доказал эту теорему и понял её важность для теории информации. Э.Т. Уиттекер (1915) доказал эту теорему как математический факт. Г. Найквист (1924) и К. Кюпфмюллер (1928) в своих трудах по теории связи использовали подобные результаты.

История теоремы отсчётов — это отдельная довольно запутанная история[6].

Приложения теоремы отсчётов

Приведём несколько примеров, когда дискретное явление воспринимается как непрерывное.

1. В живописи такие направления, как пуантилизм, импрессионизм, позволяют воспринимать отдельные кусочки картины, как единое изображение.
2. В телевидении используется построчная развёртка, когда из отдельных строчек формируется цельное изображение[7].
3. В графическом редакторе Photoshop обработка изображения по слоям, из которых формируется рисунок [8].

Рассматриваются следующие действия с информацией:

1) сбор,
2) обмен,
3) накопление,
4) хранение,
5) обработка,
6) выдача,
7) и др.

Соглашение. В основном мы будем иметь двоичную информацию и работать с ней.

---

[5]В 1999 году Международный научный фонд Эдуарда Рейна (Германия) признал приоритет Котельникова в использовании этой теоремы в теории связи, наградив его премией в номинации «за фундаментальные исследования» за впервые математически точно сформулированную и доказанную в аспекте коммуникационных технологий теорему отсчётов.

[6]Интерполяционная формула, как её обычно называют, восходит к работе Эмиля Бореля, датированной 1898 годом, и к работе Эдмунда Уиттекера, датированной 1915 годом. Интерполяционная формула была процитирована из работы сына Эдмунда Уиттекера — Джона Макнейтена Уиттекера, датированной 1935 годом, в виде теоремы отсчётов Найквиста–Шеннона в 1949 году. Автором редакции был Клод Шеннон. До Шеннона данную теорему сформулировал Котельников. Также интерполяционную формулу обычно называют интерполяционной формулой Шеннона, или интерполяционной формулой Уиттекера.

[7]Так как изображение формируется по значениям аргумента, то это можно рассматривать аналог подхода интеграла Римана

[8]Так как изображение формируется послойно по значениям цвета, то это можно рассматривать аналог подхода интеграла Лебега

В силу теоремы отсчётов можно предполагать, что информация представлена в дискретном виде. Как её получить в двоичном виде?

Изложим один из подходов.

Выборочный каскад

Имеем конечное множество. Проделаем несколько разбиений этого множества на части, до тех пор пока в каждой части не будет по одному элементу. При делении каждой предыдущей части на две можно добавлять 0 и 1 для обозначения подчастей.

В приведённом ниже примере делается 3 шага. Дано множество

α	β	γ	δ	ε	σ	τ

Первый шаг

α	β	γ	δ	ε	σ	τ
0			1			

Второй шаг

α	β	γ	δ	ε	σ	τ
00	01		10		11	

Третий шаг

α	β	γ	δ	ε	σ	τ
00	010	011	100	101	110	111

Вот здесь можно остановиться, поскольку каждому элементу исходного множества приписан упорядоченный набор из 0 и 1. Таким образом исходному множеству сопоставлено биективно множество упорядоченных наборов из 0 и 1, которые мы назовём для краткости ***знаками***.

Иногда важно, чтобы каждому элементу исходного множества был приписан упорядоченный набор из одинакового количества знаков. Этот процесс называют выравниванием количества знаков. Продемонстрируем на рассматриваемом примере. Добавим α знак 0, чтобы не путать с δ.

α	β	γ	δ	ε	σ	τ
000	010	011	100	101	110	111

Получим

Символ	Код	Вероятность
α	000	$p_\alpha = \left(\frac{1}{2}\right)^2$
β	010	$p_\beta = \left(\frac{1}{2}\right)^3$
γ	011	$p_\gamma = \left(\frac{1}{2}\right)^3$
δ	100	$p_\delta = \left(\frac{1}{2}\right)^3$
ε	101	$p_\varepsilon = \left(\frac{1}{2}\right)^3$
σ	110	$p_\sigma = \left(\frac{1}{2}\right)^3$
τ	111	$p_\tau = \left(\frac{1}{2}\right)^3$

Вероятности символов считаются так. Если k — число шагов из 0 и 1 для выделения знака, то

$$p = \left(\frac{1}{2}\right)^k.$$

В общем случае, пусть имеется не более чем счётное число знаков (источник), каждый из которых закодирован набором из 0 и 1. Вероятность появления i-того знака равна $p_i \geqslant 0$. Тогда должны иметь равенство для дискретной вероятности

$$\sum_i p_i = 1.$$

Как возникает вероятность появления знака? Снова, если k — число выборов из 0 и 1 для выделения знака, то

$$p = \left(\frac{1}{2}\right)^k.$$

Определения. Число

$$k = \log_{\frac{1}{2}}(p) = -\log_2 p = \log_2 \frac{1}{p}$$

называется ***количеством информации*** в данном знаке.

Величина

$$H = \sum_i p_i \log_2 \frac{1}{p_i} [\text{бит}]$$

называется ***энтропией*** источника информации. Она является средним взвешенным количеством информации на знак.

Чуть по-иному, энтропия является средней длиной двоичных слов для кодирования источника.

В нашем случае

$$H = \frac{1}{4} \cdot 2 + 6 \cdot \frac{1}{8} \cdot 3 = \frac{11}{4} = 2\frac{3}{4}$$

является средней длиной двоичных слов при кодировании источника. Если $n = 2^N$, то можно закодировать так, чтобы $H = N = \log_2 n$, если всё время делить пополам.

В общем случае можно доказать, что $H \leqslant \log_2 n$. Если выбрать N, как $2^{N-1} < n \leqslant 2^N$, то источник можно закодировать за N выборов (у нас в примере $N = 3$). Для этого надо делить так на части, чтобы разница между количествами элементов в частях была не более 1.

Пусть N_i — число символов в слове номером i для любого i. Рассмотрим среднюю длину слов, которую определим, как

$$L = \sum_i p_i N_i.$$

В нашем случае до выравнивания из приведённой выше таблицы получаем

$$L = \frac{1}{4} \cdot 2 + 6 \cdot \frac{1}{8} \cdot 3, \text{ значит } L = H,$$

а после выравнивания

$$L = \frac{1}{4} \cdot 3 + 6 \cdot \frac{1}{8} \cdot 3 = 3, \text{ значит } L > H.$$

Теорема (Шеннона о кодировании)**.** *При кодировании энтропия и средняя длина слов имеют следующие свойства.*

1. $H \leqslant L$.
2. *Можно кодировать так, чтобы разность $L - H$ была сколь угодно малой.*

Определение. Величина $L - H$ называется ***избыточностью кода,*** а величина $1 - \frac{H}{L}$ называется ***относительной избыточностью кода.***

Рассмотрим частоту использования символов в русском языке:

№	Символы	Частота
1	пробелы и знаки препинания	0,175
2	о	0,09
3	е и ё	0,072
⋮		
30	щ	0,003
31	э	0,003
32	ф	0,002

Отметим, что в количестве информации в данном знаке и энтропии источника информации используются логарифмы вероятностей знаков, что находит отражении в психологии.

Пример. В психологии имеется «Закон Меркеля» о времени выбора предмета.

Более точно, пусть T — время реакции на выбор определённого предмета из n имеющихся. Тогда

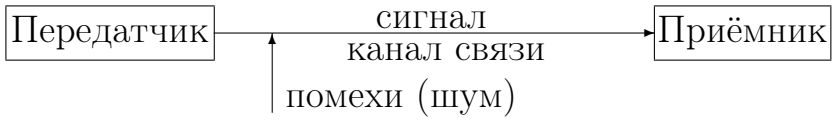
$$T \approx 200 + 180 \log_2 n (\text{мс}).$$

Важно!

Затраты на выбор знака пропорциональны не числу знаков, а его логарифму!

Передача информации

Рис. 1. Простейшая схема передачи информации:



Пусть передаются двоичные слова и вероятность НЕверной передачи 0 равна p_0, вероятность НЕверной передачи 1 равна p_1.

Частные случаи:

1) симметричные помехи: $p_0 = p_1$;
2) односторонние помехи: $p_0 = 0$ или $p_1 = 0$.

2.1. Общая постановка задачи кодирования

Пусть S — исходное множество двоичных слов. Под ***кодированием*** будем понимать сопоставление множеству S некоторого подмножества $C \subseteq \mathbf{F}_q^n$. Здесь $\mathbf{F}_q$ — поле из $q = p^m$ элементов, p — простое число. $\mathbf{F}_q^n$ — множество строчек

$$\{(\alpha_1, \ldots, \alpha_n) | \alpha_i \in \mathbf{F}_q, \forall i\}.$$

Обычно $q = 2$.

Более точно, кодирование может быть:

1) инъективное (взаимно однозначное): образ у каждого слова только один и по образу однозначно восстанавливается исходное слово;

2) неинъективное: образ у каждого слова только один, но по образу неоднозначно восстанавливается исходное слово (меньший битрейт);

3) не являющимся отображением: когда хотя бы у одного слова не один образ (интерполяция в больший битрейт).

Например, для видео и фото кодирование может быть сжатием и интерполяцией.

Мы не будем интересоваться свойствами кодирования, а сосредоточимся на том, каким должно быть множество C, чтобы обеспечивать: нахождение ошибок, их исправление и т. п. при передаче, при работе устройства и т. д.

Соглашение. Далее считать, что $\mathbf{F}_q = \mathbf{F}_2$, то есть у нас коды будут ***двоичными (бинарными)***.

Определение. Любое подмножество C множества $\mathbf{F}_2^n$ будем называть ***кодом***, точнее ***n-кодом***.

Определение. Пусть C — двоичный код. Если для любых $x, y \in C$ имеем $x + y \in C$, то код называется ***линейным***.

Замечание. Если код недвоичный, то для его линейности требуется дополнительно, чтобы $\alpha x \in C$ для любых $\alpha \in \mathbf{F}_q$ и $x \in C$ имеем

Можно рассматривать элементы C как n-мерные векторы, и говорить об их линейной независимости. Отметим, также, что любой код можно дополнить до линейного, добавляя отсутствующие суммы.

Как задать линейный код, чтобы было компактное представление и удобство в работе?

Пусть исходные данные уже выровнены (например, данное сообщение разбито на блоки одной длины) и имеют длину $k \leqslant n$. Векторы будем писать в строчку, а если в столбик, то со знаком транспонирования ^{t}.

Сначала рассмотрим важный частный случай. Будем кодировать так. К каждому слову $\vec{s}$ источника S припишем дополнительный вектор $\vec{y}$, в результате получим кодовое слово $\vec{x}$. Более формально имеем

$$\vec{s} \mapsto (\vec{s}, \vec{y}) = \vec{x} \in \mathbf{F}_2^n, \text{ где}$$
$$\vec{s} = (x_1, \ldots, x_k) \in \mathbf{F}_2^k, \ \vec{y} = (x_{k+1}, \ldots, x_n) \in \mathbf{F}_2^{n-k}.$$

Определение. Выберем матрицу $A \in M_{(n-k)\times k}(\mathbf{F}_2)$, и по ней построим матрицу

$$H = (A \mid E_{n-k}) \in M_{(n-k)\times n}(\mathbf{F}_2),$$

которую назовём ***проверочной***.

Потребуем

$$H\vec{x}^t = \vec{0}^t, \text{ равносильно } (A \mid E_{n-k}) \begin{pmatrix} \vec{s}^t \\ \vec{y}^t \end{pmatrix} = \vec{0}^t \in \mathbf{F}_2^{n-k},$$
$$\text{равносильно } A\vec{s}^t + \vec{y}^t = \vec{0}^t, \text{ равносильно } \vec{y}^t = A\vec{s}^t.$$

Определения. Начальные символы (которые входят в $\vec{s}$) называют ***информационными***, оставшиеся символы (которые входят в $\vec{y}$) называют ***проверочными***.

Далее

$$\begin{cases} \vec{s}^t = E_k \vec{s}^t \\ \vec{y}^t = A\vec{s}^t \end{cases}, \text{ или } \vec{x}^t = \begin{pmatrix} \vec{s}^t \\ \vec{y}^t \end{pmatrix} = \begin{pmatrix} E_k \\ A \end{pmatrix} \vec{s}^t, \text{ или } \vec{x} = \vec{s}(E_k \mid A^t).$$

Определение. Матрица

$$G = (E_k \mid A^t) \in M_{k\times n}(\mathbf{F}_2),$$

называется ***порождающей*** матрицей кода.

Укажем важное свойство произведения проверочной и порождающей матриц

$$G \cdot H^t = (E_k \mid A^t)\begin{pmatrix} A^t \\ E_{n-k} \end{pmatrix} = E_k \cdot A^t + A^t \cdot E_{n-k} = A^t + A^t = 2A^t = \mathbf{O}_{k\times(n-k)},$$

так как для любого $a \in \mathbf{F}_2$ имеем $a + a = 2a = 0$, и также

$$HG^t = (GH^t)^t = \mathbf{O}_{k\times(n-k)} = \mathbf{O}_{(n-k)\times k}.$$

Определения. Число n называют ***длиной*** кода, а k — ***числом информационных символов***.

Абстрактный подход состоит в том, что выбираются произвольные матрицы

$$G \in M_{k\times n}(\mathbf{F}_2) \quad \text{и} \quad H \in M_{(n-k)\times n}(\mathbf{F}_2)$$

с условием

$$GH^t = \mathbf{O}_{k\times(n-k)}.$$

Удобно, если G имеет ранг k. Это означает, что строчки матрицы G линейно независимы.

Теперь можно определить код одним из следующих равносильных способов.

Пусть S — подмножество в $\mathbf{F}_2^k$, которое мы хотим закодировать. Тогда линейный код C определяется так:

$$C = \mathrm{Lin}(\vec{x} = \vec{s}G \mid \vec{s} \in S),$$

здесь Lin — операция дополнения кода до линейного, которая в бинарном случае сводится к тому, что добавляются все возможные суммы векторов $\vec{s}G$, которые не попали во множество

$$\{\vec{s}G \mid \vec{s} \in S\}.$$

Тем самым мы добьёмся, что получим линейный код

Или, по-другому линейный код C определяется так:

$$C = \{\vec{x} \mid H\vec{x}^t = \vec{0}^t\}.$$

Определения. Построенный линейный код C называют $[n, k]$-***кодом*** (иногда заменяют k на $n - k$).

Отношение $\dfrac{k}{n}$ называют ***скоростью***, или ***эффективностью*** кода.

2.2. Метод наибольшего правдоподобия (система исправления ошибок)

Предположим,что по каналу связи передаётся сообщение $\vec{c}$, которое является элементом кода $C \subseteq \mathbf{F}_2^n$, назовём его *исходящим* сообщением. В результате передачи получили сообщение $\vec{f} \in \mathbf{F}_2^n$, назовём его *приходящим* сообщением. При нашем подходе $\vec{c} = (c_1, \ldots, c_n)$, где все $c_i \in \mathbf{F}_2 = \{0, 1\}$. Так как мы рассматриваем линейные коды, то в линейном коде C будет 2^k элементов для $k < n$. Тогда

$$\vec{a} = \vec{f} - c = \vec{f} + \vec{c} \,(\text{над } \mathbf{F}_2)$$

называется ***вектором ошибок***. Множество

$$\vec{a} + C = \{\vec{a} + \vec{c} \mid \vec{c} \in C\}.$$

содержит все полученные сообщения, которые при передаче получили фиксированную ошибку $\vec{a}$. Важно изучать такие множества.

Можно показать, что все различные множества $\vec{a} + C$ образуют разбиение множества $\mathbf{F}_2^n$, то есть, любой элемент из $\mathbf{F}_2^n$ попадает в одно из таких множеств и причём точно в одно. Как описать такие множества?

Определение. Назовём ***весом Хэмминга*** вектора $\vec{x} \in \mathbf{F}_2^n$ число единиц в $\vec{x}$ и будем обозначать его $\mathrm{wt}(\vec{x})$.

Пример. Пусть

$$\vec{x} = (1, 0, 1, 0, 1, 0, 1) \text{ и } \vec{y} = (1, 1, 0, 1, 1, 0, 1).$$

Тогда

$$\mathrm{wt}(\vec{x}) = 4 \text{ и } \mathrm{wt}(\vec{y}) = 5.$$

Наша цель состоит в том, чтобы исходящее сообщение, получив приходящее. Этот процесс называется *декодированием.*

Определение. Во множестве $\vec{a}+C$ выберем вектор $\vec{e}$ с наименьшим весом Хэмминга и назовём его ***лидером множества*** $\vec{a}+C$**.**

Затем декодируем так. Считаем, что исходящее сообщение равно

$$\vec{c}=\vec{f}-\vec{e}=\vec{f}+\vec{e}\ (\text{над } \mathbf{F}_2).$$

Это и есть ***принцип наибольшего правдоподобия*** или ***декодирование в ближайшего соседа.*** Этот способ основан на предположении, что сделано минимальное количество ошибок, что и даёт соответствующее название. Также можно ввести расстояние Хэмминга

$$\mathrm{d}(\vec{x},\vec{y})=\mathrm{wt}(\vec{x}-\vec{y})=\mathrm{wt}(\vec{x}+\vec{y})\ (\text{над } \mathbf{F}_2)$$

— число различающихся компонент векторов $\vec{x}$ и $\vec{y}$.

Важно понять какому лидеру соответствует приходящее сообщение $\vec{f}$?

Определение. ***Синдромом (disorder)*** вектора $\vec{f}$ называется

$$\vec{d}^t=H\vec{f}^t,\ \text{или } \vec{d}=\vec{f}H^t\in\mathbf{F}_2^{n-k},$$

где H — проверочная матрица.

Синдром описывает отклонение от кодового слова. По определению проверочной матрицы

$$\vec{c}\in C \text{ тогда и только тогда, когда } H\vec{c}^t=\vec{0}^t.$$

В нашем случае $\vec{f}=\vec{c}+\vec{e}$ и

$$\vec{d}^t=H\vec{f}^t=H\vec{c}^t+H\vec{e}^t=H\vec{e}^t.$$

Если есть ошибки на местах $i_1,\dots,i_m$ в векторе $\vec{e}^t$, то

$$\vec{d}^t=H^t_{i_1}+\dots+H^t_{i_m},$$

здесь через $H^t_{i_1},\dots,H^t_{i_m}$ обозначены столбцы матрицы H с номерами i_1,..., i_m.

Определение. ***Минимальным расстоянием*** $\mathrm{d}_{\min}(C)$ ***кода*** C называется минимальное расстояние между кодовыми словами кода C,

$$\mathrm{d}_{\min}(C)=\min_{\vec{x}\neq\vec{y}\in C}\mathrm{d}(\vec{x},\vec{y}).$$

Из определения расстояния следует, что минимальное расстояние кода C можно вычислить из равенства

$$\mathrm{d}_{\min}(C)=\min_{\vec{0}\neq\vec{x}\in C}\mathrm{wt}(\vec{x}).$$

Итак, минимальное расстояние между кодовыми словами равно минимальному весу, не равного $\vec{0}$, кодового слова.

Определение. $[n, k, d]$-***кодом*** называется линейный код из слов длины n (код в $\mathbf{F}_2^n$), размерности k (равносильно из 2^k элементов) и с минимальным расстоянием d.

Определение. Будем говорить, что код C ***обнаруживает*** t ***ошибок***, если можно указать в полученном сообщении t ошибок без указания самих конкретных ошибок. Если же код C обнаруживает t ошибок и указывает, какие символы преданы неверно, то говорят, что код C ***исправляет*** t ***ошибок***.

Теорема. *Если минимальное расстояние кода равно d, то он обнаруживает $d-1$ ошибок и исправляет*

$$\left\lfloor \frac{d-1}{2} \right\rfloor = \begin{cases} \frac{d-1}{2} & \text{при нечётном } d, \\ \frac{d}{2} & \text{при чётном } d \end{cases}$$

ошибок.

Доказательство. ***Шаром*** с центром в $\vec{x} \in \mathbf{F}_2^n$ радиуса t назовём множество

$$B(\vec{x}, t) = \{\vec{y} \in \mathbf{F}_2^n \mid \mathrm{d}(\vec{x}, \vec{y}) \leqslant t\} = \{\vec{y} \in \mathbf{F}_2^n \mid \mathrm{wt}(\vec{x} - \vec{y}) \leqslant t\}.$$

Любой шар с центром в $\vec{c} \in C$ радиуса $d-1$ содержит ровно одно одно кодовое слово — центр шара.

Поэтому, если полученное слово $\vec{f}$ содержит не более $d-1$ ошибок, то, сравнивая его с центром шара, мы поймём, сколько ошибок оно содержит.

Если полученное слово $\vec{f}$ содержит не более t ошибок, то $\vec{f} \in B(\vec{c}, t)$. Исправление t ошибок означает, что любые два шара с центрами в двух различных кодовых словах радиуса t не должны пересекаться. Если $\vec{u} \in B(\vec{x}, t) \cap B(\vec{y}, t)$, то

$$d \leqslant \mathrm{d}(\vec{x}, \vec{y}) \leqslant \mathrm{d}(\vec{x}, \vec{u}) + \mathrm{d}(\vec{u}, \vec{y}) \leqslant 2t.$$

Отсюда $t \geqslant \frac{d}{2}$. Поэтому может исправляться не более $\frac{d-1}{2}$ ошибок. Так как есть два кодовых слова точно на расстоянии d друг от друга, то получаем, что теорема доказана. □

Напомним, что для любого натурального n и любого $k \in \{0, 1, \ldots, n\}$

$$C_n^k = \frac{n!}{k!(n-k)!}.$$

Теорема (граница Хэмминга)**.** *Пусть код (не обязательно линейный) содержится в* $\mathbf{F}_2^n$ *и в нём* M *слов. Допустим, что этот код может исправлять* t *ошибок. Тогда*

$$M(1 + C_n^1 + \cdots + C_n^t) \leqslant 2^n.$$

В случае линейного кода размерности k *имеем*

$$1 + C_n^1 + \cdots + C_n^t \leqslant 2^{n-k}.$$

Доказательство. В каждом из M шаров с центрами в кодовых словах должно быть

$$1 + C_n^1 + \cdots + C_n^t$$

векторов. Так как эти шары не должны пересекаться, то получаем следующее.

$$M(1 + C_n^1 + \cdots + C_n^t) \leqslant 2^n,$$

где 2^n — общее число векторов в $\mathbf{F}_2^n$.

Если код линейный размерности k, то в нём $M = 2^k$ элементов. Поэтому

$$2^k(1 + C_n^1 + \cdots + C_n^t) \leqslant 2^n \longleftrightarrow 1 + C_n^1 + \cdots + C_n^t \leqslant 2^{n-k}.$$

Теорема доказана. □

Замечание. С помощью границы Хэмминга можно при данном M оценивать число исправлений t, и наоборот, при данном t оценивать M.

Определение. Код называется ***совершенным***, если достигается граница Хэмминга.

Следующие два результата не будем доказывать.

Теорема (граница Гилберта–Варшамова)**.** *Если*

$$2^{n-k} > \sum_{i=0}^{d-2} C_{n-1}^i,$$

то существует $[n, k, d]$*-код.*

Теорема (граница Плоткина)**.** *Если существует* $[n, k, d]$*-код, то*

$$d \leqslant n \cdot \frac{2^k}{2(2^k - 1)}$$

При маленьких значениях n сильно большого значения d не получится. При больших n и k параметр d может расти.

2.3. Частные случаи кодов

2.3.1. Коды Хэмминга и Рида–Маллера

Определение. Пусть проверочная матрица H состоит из всех ненулевых векторов длины m, таких векторов $2^m - 1$, и они записываются по столбцам. Тогда получаем ***код Хэмминга***. Он будет $[2^m - 1, 2^m - 1 - m, 3]$-кодом.

Определение. Зададим на F_2^n скалярное произведение. Положим для любых $\vec{u}, \vec{v} \in \mathbf{F}_2^n$

$$(\vec{u}, \vec{v}) = ((u_1, \ldots, u_n), (v_1, \ldots, v_n)) = u_1 v_1 + \cdots + u_n v_n = \sum_{i=0}^{n} u_i v_i.$$

Пусть C — произвольный код в $\mathbf{F}_2^n$. Тогда

$$C^{\perp} = \{\vec{u} \in \mathbf{F}_2^n \mid (\vec{u}, \vec{v}) = 0 \ \forall v \in C\}$$

— ***ортогональный (дуальный, двойственный)*** к C код. Он всегда линеен.

Определение. Линейный код C называется ***циклическим***, если для любого $(u_1, u_2, \ldots, u_n) \in C$ имеем $(u_2, u_3, \ldots, u_n, u_1) \in C$.

Код можно удлинять (расширять). Частный случай — проверка на чётность.

Определение. ***Удлинённый (расширенный) код*** строится так. К проверочной матрице H приписывается столбец из 0 (обычно справа), затем приписывается строка из 1 (обычно сверху).

$$\left(\begin{array}{c|c} 1 \ldots 1 & 1 \\ \hline H & \vec{0}^t \end{array}\right) = \left(\begin{array}{c|c} \vec{1}^t & 1 \\ \hline H & \vec{0}^t \end{array}\right).$$

Полученная матрица $\overline{H}$ является проверочной матрицей удлинённого (расширенного) кода.

Определение. Удлиним код Хэмминга, получим $[2^m, 2^m - 1 - m, 4]$-код C. Код $C^{\perp}$, ортогональный к C называется ***кодом Рида–Маллера первого порядка.*** Этот код будет $[2^m, m + 1, 2^{m-1}]$-кодом[9].

[9] В книге [5, с. 270] указывается, что коды Рида–Маллера [32,6,16] использовались для дальней космической связи.

2.3.2. Коды Голея (Golay)

Кодов Голея всего четыре —

$$G_{24},\ G_{23},\ G_{21},\ G_{11},$$

причём G_{24} и G_{23} — бинарные (двоичные, над $\mathbf{F}_2$), а G_{21} и G_{11} — тернарные (троичные, над $\mathbf{F}_3$).

Далее рассмотрим только построение порождающей матрицы кода G_{23}, а порождающая матрица кода G_{24} получается из неё приписыванием столбца и строки из 1. Многочлен $x^{23}+1$ над $\mathbf{F}_2$ раскладывается так:

$$x^{23}+1=(x+1)(x^{11}+x^{10}+x^6+x^5+x^4+x^2+1)\times \\ \times(x^{11}+x^9+x^7+x^6+x^5+x+1).$$

Пусть g — любой из двух полученных многочленов степени 11. Для каждого $i\in\{0,\dots,11\}$ через g_i обозначим коэффициент многочлена g при x^i (например, $g_0=g_{11}=1$, а $g_8=g_3=0$).

Зададим код G_{23} порождающей матрицей G размера 11×23:

$$G=\begin{pmatrix} g_0 & g_1 & \dots & g_{11} & 0\dots & 0 & \\ 0 & g_0 & \dots & g_{10} & g_{11} & \dots & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \dots & 0 & g_0 & \dots & g_{11} \end{pmatrix}$$

Таким образом, получим код G_{23}, который является $[23,12,7]$-код, а код G_{24} является $[24,12,8]$-кодом. Коды Голея имеют много интересных свойств.

Теорема. *Если бинарный код совершенен и исправляет более одной ошибки, то это код Голея G_{23}.*

Замечание. Напомним, что для совершенности двоичного линеейного $[n,k]$-кода необходимо, чтобы выполнялось равенство

$$1+C_n^1+\dots+C_n^t=2^{n-k},$$

где t — исправляемых этим кодом ошибок.

Для $n=23$ и $t=3$ имеем

$$1+C_{23}^1+C_{23}^2+C_{23}^3=1+23+\frac{23\cdot 22}{2}+\frac{23\cdot 22\cdot 21}{2\cdot 3}= \\ =24+253+1771=2048=2^{11}=2^{23-12}.$$

Именно это вычисление натолкнуло Голея на возможность построения кода G_{23}. Таким образом, выполнено необходимое условие существования совершенного $[23, 12, 7]$-код, но само построение такого кода более сложно.

Отметим, так же что расположение 1 в кодовых словах определяется вычетами по модулю 23, которые являются квадратами по модулю 23. Поэтому коды Голея относятся к классу *квадратично-вычетных* кодов.

Совершенные коды, исправляющие 1 ошибку, имеют параметры как у кодов Хэмминга.

В Википедии отмечено, что

> *"Код Голея применялся в ходе программы Вояджер при передаче аппаратами Вояджер-1 и Вояджер-2 цветных изображений Юпитера и Сатурна."*

2.3.3. БЧХ-коды

БЧХ-коды предложены Боузом и Чоудхури (Bose, Chaudhuri) в 1959 г. и Хоквингемом (Hocquenghem) в 1960 г.

Сначала приведём некоторые факты о конечных полях $\mathbf{F}_{2^m}$. Можно доказать, что поле $\mathbf{F}_{2^m}$ из 2^m элементов имеет точно m таких элементов

$$e_1, \ldots, e_m,$$

что

$$\mathbf{F}_{2^m} = \{a_1 e_1 + \cdots + a_m e_m \mid \forall i \, a_i \in \mathbf{F}_2\},$$

причём сложение в поле $\mathbf{F}_{2^m}$ задаётся так:

$$(a_1 e_1 + \cdots + a_m e_m) + (b_1 e_1 + \cdots + b_m e_m) = (a_1 + b_1) e_1 + \cdots + (a_m + b_m) e_m.$$

Указанное представление элементов поля $\mathbf{F}_{2^m}$, является разложением поля $\mathbf{F}_{2^m}$ как векторного пространства над $\mathbf{F}_2$ по базису e_1,…,e_m. Следовательно, для его любого элемента $x \in \mathbf{F}_{2^m}$ выполняется свойство, что $x + x = 0$.

Также отметим, что если фиксированы элементы $(e_1, \ldots, e_m)$, то элемент $a_1 e_1 + \cdots + a_m e_m \in \mathbf{F}_{2^m}$ можно считать строкой

$$(a_1, \ldots, a_m) \in \mathbf{F}_2^m,$$

или столбцом

$$\begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \in \mathbf{F}_2^m.$$

Теперь перейдём к определению бинарных БЧХ-кодов.

Определение. Зафиксируем нечётное число $n \geqslant 3$ и выберем натуральное d с условием

$$2 \leqslant d \leqslant n.$$

Пусть m — минимальное натуральное число со свойством

$$2^m \equiv 1 \pmod{n}.$$

Пусть $\mathbf{F}_{2^m}$ — конечное поле из 2^m элементов. Тогда существует такой элемент $\zeta \in \mathbf{F}_{2^m}$, что $\zeta^n = 1 \in \mathbf{F}_{2^m}$ и $\zeta^l \neq 1$ для любого $l \in \{1, 2, \ldots, n-1\}$. ***БЧХ-код с конструктивным расстоянием*** d задаётся проверочной матрицей:

$$\begin{pmatrix} 1 & \zeta & \ldots & \zeta^{n-1} \\ 1 & \zeta^2 & \ldots & \zeta^{2(n-1)} \\ \vdots & \vdots & & \vdots \\ 1 & \zeta^{d-1} & \ldots & \zeta^{(d-1)(n-1)} \end{pmatrix}$$

Замечания.

1. Для построения бинарного кода нужно заменить каждый элемент $\zeta^k \in \mathbf{F}_{2^m}$ проверочной матрицы БЧХ-кода столбцом высоты m из элементов $\{0, 1\} \in \mathbf{F}_2$ согласно тому, как представляются элементы $\mathbf{F}_{2^m}$ как столбцы из 0 и 1. Получится матрица размера $m(d-1) \times n$ над $\mathbf{F}_2$, и с ней уже можно работать.
2. Можно доказать, что для БЧХ-кода минимальное расстояние $d_{\min} \geqslant d$.

Определение. БЧХ-коды при $n = 2^m - 1$ называются кодами ***Рида–Соломона.***

Замечание. В книге [6] на стр. 306 написано.

> "Практический пример использования БЧХ-кодов доставляют европейская и трансатлантическая информационные системы связи, использующие такие коды в течение многих лет. Сообщения в них имеют длину 231,... длина кодовых слов равна
>
> $$231 + 24 = 255 = 2^8 - 1.$$
>
> Этот код обнаруживает, по меньшей мере, **шесть** ошибок, вероятность неправильного декодирования — **одна 16-миллионная.**"

2.4. Построение БЧХ-кодов

2.4.1. $n = 3$

Так как $2^2 - 1 = 3$, то нужно рассмотреть поле $\mathbf{F}_4$ из $2^2 = 4$ элементов.

В поле $\mathbf{F}_4$ должен быть элемент $\alpha \notin \{0,1\}$. Если бы $1+\alpha \in \{0,1\}$, то $\alpha \in \{1,0\}$, что не так. Если же $1+\alpha = \alpha$, то $1=0$, что невозможно в поле. Итак имеем $\mathbf{F}_4 = \{0,1,\alpha,1+\alpha\}$.

Ненулевые элементы поля $\mathbf{F}_4$ должны образовывать группу $\mathbf{F}_4^* = \{1,\alpha,1+\alpha\}$, в которой должен быть примитивный элемент g, для которого

$$\mathbf{F}_4^* = \{1,\alpha,1+\alpha\} = \{1,g,g^2\}.$$

Если $\alpha = g$, то $\alpha^2 = g^2 = 1+\alpha$. Если $\alpha = g^2$, то $1+\alpha = g$ и $\alpha^2 = g^4 = g^3 g = 1\cdot g = 1+\alpha$.

Итак в любом случае $\alpha^2 = 1+\alpha$ и получаются следующие таблицы сложения и умножения в поле $\mathbf{F}_4$.

$+$	0	1	α	$1+\alpha$
0	0	1	α	$1+\alpha$
1	1	0	$1+\alpha$	α
α	α	$1+\alpha$	0	1
$1+\alpha$	$1+\alpha$	α	1	0

$\cdot$	0	1	α	$1+\alpha$
0	0	0	0	0
1	0	1	α	$1+\alpha$
α	0	α	$1+\alpha$	1
$1+\alpha$	0	$1+\alpha$	1	α

Возьмём $e_1 = 1$ и $e_2 = \alpha$. Тогда

$$\mathbf{F}_4 = \left\{ 0 = \begin{pmatrix}0\\0\end{pmatrix},\ 1 = \begin{pmatrix}1\\0\end{pmatrix},\ \alpha = \begin{pmatrix}0\\1\end{pmatrix},\ 1+\alpha = \begin{pmatrix}1\\1\end{pmatrix} \right\}.$$

Пусть $d = 2$ и $\zeta = \alpha$. Тогда проверочная матрица для БЧХ-кода имеет вид

$$\begin{pmatrix}1 & \zeta & \zeta^2\end{pmatrix} = \begin{pmatrix}1 & 0 & 1\\ 0 & 1 & 1\end{pmatrix}$$

Получаем код Хэмминга, поскольку проверочная матрица состоит из всех ненулевых векторов из $\mathbf{F}_2^2$.

Пусть $d = 3$ и $\zeta = \alpha$. Тогда проверочная матрица для БЧХ-кода

$$\begin{pmatrix}1 & \zeta & \zeta^2\\ 1 & \zeta^2 & \zeta^{2\cdot 2}\end{pmatrix} = \begin{pmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 0\\ 0 & 1 & 1\end{pmatrix} \longleftrightarrow \begin{pmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 0 & 1 & 1\\ 0 & 1 & 1\end{pmatrix} \longleftrightarrow \begin{pmatrix}1 & 0 & 1\\ 0 & 1 & 1\end{pmatrix}.$$

Снова код Хэмминга по тем же причинам, что и ранее.

Для последующего использования отметим ещё раз, что $\alpha^2 = 1+\alpha$ и $\alpha^3 = 1$. Перепишем таблицы сложения и умножения в “мультипликативном

ключе":

$+$	0	1	α	α^2
0	0	1	α	α^2
1	1	0	α^2	α
α	α	α^2	0	1
α^2	α^2	α	1	0

$\cdot$	0	1	α	α^2
0	0	0	0	0
1	0	1	α	α^2
α	0	α	α^2	1
α^2	0	α^2	1	α

2.4.2. $n = 5$

0. Построение поля $\mathbf{F}_{16}$

Так как $2^4 - 1 = 15$, то нужно рассмотреть поле $\mathbf{F}_{16}$.

В отличие от F_4 будем действовать "более индустриально". Возьмём многочлен

$$p = x^2 + x + \alpha \in \mathbf{F}_4[x].$$

Имеем

$$\begin{aligned} p(0) &= 0 + 0 + \alpha = \alpha \neq 0, \\ p(1) &= 1 + 1 + \alpha = \alpha \neq 0, \\ p(\alpha) &= \alpha^2 + \alpha + \alpha = 1 + \alpha \neq 0, \\ p(1+\alpha) &= (1+\alpha)^2 + (1+\alpha) + \alpha = \alpha^4 + \alpha^2 + \alpha = \\ &= \alpha + (1+\alpha) + \alpha = 1 + \alpha \neq 0. \end{aligned}$$

Итак, многочлен p не имеет корней в $\mathbf{F}_4$.

Теперь рассмотрим эквивалентные многочлены по модулю p в $\mathbf{F}_4[x]$. Эквивалентные многочлены будут иметь одинаковые остатки от деления уголком на p, которые будут иметь вид

$$\beta x + \gamma, \text{ где } \beta, \gamma \in \mathbf{F}_4.$$

Понятно, что разные остатки будут соответствовать неэквивалентным многочленам, поэтому классов эквивалентности многочленов будет 16. Можно работать с остатками и учитывать, что

$$p = x^2 + x + \alpha \equiv 0 \pmod{p}, \text{ что равносильно } x^2 \equiv x + \alpha \pmod{p}.$$

Оставив в стороне все тонкости, отметим, что получится поле $\mathbf{F}_{16}$

$$\begin{aligned} \mathbf{F}_{16} = \{&0, 1, \alpha, 1+\alpha, x, x+1, x+\alpha, x+1+\alpha, \alpha x, \alpha x+1, \alpha x+\alpha, \alpha x+1+\alpha, \\ &(1+\alpha)x, (1+\alpha)x+1, (1+\alpha)x+\alpha, (1+\alpha)x+1+\alpha\}. \end{aligned}$$

Нам важно найти примитивный элемент этого поля.

Будем действовать по способу “brute force” (метод «грубой силы», полный перебор), заменяя эквивалентность на равенство.

$$
\begin{aligned}
x^2 &= x+\alpha,\\
x^3 &= (x+\alpha)x = x^2+\alpha x = x+\alpha+\alpha x = (1+\alpha)x+\alpha = \alpha^2 x+\alpha = \\
&= \alpha^2(x+\alpha^2),\\
x^4 &= (x^2)^2 = (x+\alpha)^2 = x^2+2\alpha x+\alpha^2 = x+\alpha+\alpha^2 = x+1,\\
x^5 &= (x+1)x = x^2+x = x+\alpha+x = \alpha,\\
x^6 &= \alpha x,\\
x^7 &= \alpha x^2 = \alpha(x+\alpha),\\
x^8 &= (x^4)^2 = (x+1)^2 = x^2+1 = x+\alpha+1 = x+\alpha^2,\\
x^9 &= (x+\alpha^2)x = x^2+\alpha^2 x = x+\alpha+\alpha^2 x = \alpha x+\alpha = \alpha(x+1),\\
x^{10} &= (x^5)^2 = \alpha^2,\\
x^{11} &= \alpha^2 x,\\
x^{12} &= \alpha^2 x^2 = \alpha^2(x+\alpha),\\
x^{13} &= \alpha^2(x+\alpha x)x = \alpha^2(x^2+\alpha x) = \alpha^2(x+\alpha+\alpha x) = \\
&= \alpha^2(\alpha^2 x+\alpha) = \alpha(x+\alpha^2),\\
x^{14} &= \alpha(x+\alpha^2)x = \alpha(x^2+\alpha^2 x) = \alpha(x+\alpha+\alpha^2 x) = \alpha(\alpha x+\alpha)\\
&= \alpha^2(x+1),\\
x^{15} &= \alpha^2(x+1)x = \alpha^2(x^2+x) = \alpha^2\alpha = \alpha^3 = 1.
\end{aligned}
$$

Таким образом, мы получаем таблицу степеней x

0	1	2	3	4	5	6	7
1	x	$x+\alpha$	$\alpha^2 x+\alpha$	$x+1$	α	αx	$\alpha x+\alpha^2$

8	9	10	11	12	13	14	15
$x+\alpha^2$	$\alpha x+\alpha$	α^2	$\alpha^2 x$	$\alpha^2 x+1$	$\alpha x+1$	$\alpha^2 x+\alpha^2$	1

Итак, x — примитивный элемент поля $\mathbf{F}_{16}$. В качестве нужного нам элемента ζ можно выбрать x^k, где $k \in \{3,6,9,12\}$. Возьмём $\zeta = x^3 = \alpha^2 x+\alpha$.

1. $n = 5$ **и** $d = 2$

В этом случае проверочная матрица H для БЧХ-кода имеет вид

$$H = \begin{pmatrix} 1 & \zeta & \zeta^2 & \zeta^3 & \zeta^4 \end{pmatrix}.$$

Имеем

$$\zeta = x^3 = \alpha^2 x+\alpha, \qquad\qquad \zeta^2 = x^6 = \alpha x,$$

$$\zeta^3 = x^9 = \alpha x + \alpha, \qquad \zeta^4 = x^{12} = \alpha^2 x + 1,$$

поэтому

$$H = \begin{pmatrix} 1 & \alpha^2 x + \alpha & \alpha x & \alpha x + \alpha & \alpha^2 x + 1 \end{pmatrix}.$$

Надо записать матрицу H с помощью нулей и единиц. Сделаем это в два этапа.

1. Сначала запишем матрицу H с помощью элементов поля $\mathbf{F}_4$.
2. Затем полученную матрицу запишем с помощью 0 и 1.

Этап 1. Запись с помощью элементов поля $\mathbf{F}_4$

Пусть $f_1 = 1$ и $f_2 = x$. Тогда элементы матрицы H запишутся как элементы поля $\mathbf{F}_4$ следующим образом:

$$\left\{ 1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ x^3 = \begin{pmatrix} \alpha \\ \alpha^2 \end{pmatrix},\ x^6 = \begin{pmatrix} 0 \\ \alpha \end{pmatrix},\ x^9 = \begin{pmatrix} \alpha \\ \alpha \end{pmatrix},\ x^{12} = \begin{pmatrix} 1 \\ \alpha^2 \end{pmatrix} \right\}.$$

Таким образом, матрица H превращается в следующую матрицу $\mathbf{F}_4$:

$$H = \begin{pmatrix} 1 & \alpha & 0 & \alpha & 1 \\ 0 & \alpha^2 & \alpha & \alpha & \alpha^2 \end{pmatrix}.$$

Этап 2. Запись с помощью элементов поля $\mathbf{F}_2$

Пусть $e_1 = 1$ и $e_2 = \alpha$. Тогда

$$\mathbf{F}_4 = \left\{ 0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ 1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \alpha = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ 1 + \alpha = \alpha^2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}.$$

Получаем матрицу H над $\mathbf{F}_2$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Чтобы найти код, нужно над $\mathbf{F}_2$ решить однородную систему линейных уравнений:

$$H\vec{x}^t = 0.$$

Будем решать систему матричным методом.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \longleftrightarrow$$

$$\longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Отсюда получаем, что

$$x_1 = x_2 = x_3 = x_4 = x_5.$$

Таким образом, код C выглядит так:

$$C = \{(a, a, a, a, a) \mid a \in \mathbf{F}_2\}$$

и является $[5, 1, 5]$-кодом.

2. $n = 5$ и $d = 3$

Теперь проверочная матрица для БЧХ-кода

$$H = \begin{pmatrix} 1 & \zeta & \zeta^2 & \zeta^3 & \zeta^4 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & \zeta^8 \end{pmatrix}.$$

Если

$$\begin{aligned} \zeta &= x^3 = \alpha^2 x + \alpha, & \zeta^2 &= x^6 = \alpha x, \\ \zeta^3 &= x^9 = \alpha x + \alpha, & \zeta^4 &= x^{12} = \alpha^2 x + 1, \\ \zeta^6 &= x^{18} = x^3 = \alpha^2 x + \alpha, & \zeta^8 &= x^{24} = x^9 = \alpha x + \alpha, \end{aligned}$$

то

$$H = \begin{pmatrix} 1 & \alpha^2 x + \alpha & \alpha x & \alpha x + \alpha & \alpha^2 x + 1 \\ 1 & \alpha x & \alpha^2 x + 1 & \alpha^2 x + \alpha & \alpha x + \alpha \end{pmatrix}.$$

Далее будем действовать как в предыдущем примере.

Этап 1. Запись с помощью элементов поля $\mathbf{F}_4$

Пусть $f_1 = 1$ и $f_2 = x$. Получаем матрицу над $\mathbf{F}_4$:

$$H = \begin{pmatrix} 1 & \alpha^2 x + \alpha & \alpha x & \alpha x + \alpha & \alpha^2 x + 1 \\ 1 & \alpha x & \alpha^2 x + 1 & \alpha^2 x + \alpha & \alpha x + \alpha \end{pmatrix} =$$

$$= \begin{pmatrix} 1 & \alpha & 0 & \alpha & 1 \\ 0 & \alpha^2 & \alpha & \alpha & \alpha^2 \\ 1 & 0 & 1 & \alpha & \alpha \\ 0 & \alpha & \alpha^2 & \alpha^2 & \alpha \end{pmatrix}.$$

Этап 2. Запись с помощью элементов поля $\mathbf{F}_2$

Возьмём $e_1 = 1$ и $e_2 = \alpha$. Тогда

$$\mathbf{F}_4 = \left\{ 0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix},\ 1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix},\ \alpha = \begin{pmatrix} 0 \\ 1 \end{pmatrix},\ 1 + \alpha = \alpha^2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right\}.$$

Получаем матрицу над $\mathbf{F}_2$:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Чтобы найти код, нужно над $\mathbf{F}_2$ решить однородную систему линейных уравнений:

$$H\vec{x}^t = 0.$$

Будем решать её матричным методом.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow$$

Отбросили одинаковые строчки.

$$\longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow$$

снова отбрасываем одинаковые строчки

$$\longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow$$

$$\longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Отсюда получаем, что

$$x_1 = x_2 = x_3 = x_4 = x_5.$$

Таким образом код C выглядит так:

$$C = \{(a, a, a, a, a) \mid a \in \mathbf{F}_2\}$$

и является $[5, 1, 5]$-кодом.

3. Элементы криптологии

Криптологией называется наука о создании и анализе систем безопасности информации.

Криптологию принято делить на две части: криптографию и криптоанализ.

Криптографией называется наука о сохранении информации для безопасности.

Криптоанализом называется наука об изучении криптографических методов с точки зрения безопасности (грубо говоря, наука о взломе защищённой информации).

Безопасность информации включает в себя:

1) достоверность,
2) доступность,
3) целостность,
4) конфиденциальность,
5) актуальность,
6) и другие.

Достоверность и целостность информации обеспечивает кодирование.

Доступность и актуальность информации, как правило, обеспечиваются на физическом уровне.

Конфиденциальность в действительности является криптозадачей (другие аспекты безопасности информации также могут быть связаны с криптозадачами).

Шифрованием называется преобразование информации, делающее её не читаемой для посторонних.

Шифрование применяется для хранения важной информации в ненадёжных источниках и передачи её по незащищенным каналам связи. Такая передача данных представляет собой два взаимообратных процесса:

зашифрование,

расшифрование.

Шифрование можно представить следующим образом:

$$\boxed{M} \xrightarrow[\text{кл}_1]{E} \boxed{C} \xrightarrow[\text{кл}_2]{D} \boxed{M'},$$

где

M — исходное сообщение (message);

E — зашифрование (encryption) зависит от ключа кл1, поэтому надо писать $E_{\text{кл1}}$;

C — зашифрованный текст (cipher text);

D — расшифрование (decryption) зависит от ключа кл2, поэтому надо писать $D_{\text{кл2}}$;

M' — полученное сообщение (message).

Ключом называется конкретное секретное состояние параметров алгоритмов шифрования и/или расшифрования.

Необходимо чтобы

$$M = M'.$$

Замечание. В действительности, часто невозможно из-за помех обеспечить $M = M'$, поэтому достаточно, чтобы эти сообщения были хотя бы похожи.

Методы шифрования:

1) симметричное шифрование использует один ключ для зашифрования и расшифрования, кл1 = кл2.

2) асимметричное шифрование использует разные ключи для зашифрования Кл(отк) и расшифрования Кл(зак). Ключ для расшифрования Кл(зак) является значением функции от зашифрования Кл(отк).

Часто ключ зашифрования Кл(отк) не секретен, а доступен, поэтому асимметричное шифрование называют ***шифрованием с открытым ключом.***

Важно, чтобы Кл(зак) вычислялся трудно, например, функция вычисления закрытого ключа должна быть из очень большого набора возможных функций, чтобы её было трудно выбрать, даже зная открытый ключ Кл(отк).

Обычно проблема поиска закрытого ключа приводит к классическим задачам

факторизации целого числа и

дискретному логарифмированию.

Эти задачи очень трудно решать даже на суперкомпьютерах.

Принципы безопасности, важные для шифрования, «выстраданные криптологами — теоретиками и практиками»

1. **Нельзя недооценивать злоумышленника.**

2. **Принцип Керкхоффа (Керкгоффс, Kerkhoffs):** только специалист–криптоаналитик может судить о безопасности шифрования.

3. **Принцип Керкхоффа—Шеннона (Claude Elwood Shannon):** правило разработки криптографических систем, согласно которому в засекреченном виде держится только определённый набор параметров алгоритма, называемый ключом, а сам алгоритм шифрования должен быть открытым. Другими словами, при оценке надёжности шифрования необходимо предполагать, что противник знает об используемой системе шифрования всё, кроме применяемых ключей; более того при асимметричном шифровании злоумышленник может знать всё кроме закрытого ключа.

4. **Принцип Живерже (Жеверже, Marcel Givierge):** поверхностные усложнения криптосистемы могут быть иллюзорны, так как порождают ложные оценки ее криптостойкости.

5. **При оценке шифрования** необходимо учитывать возможные криптографические ошибки и другие нарушения дисциплины безопасности.

Шесть требований Керкгоффса

Огюст Керкгоффс (Auguste Kerckhoffs; 19 января 1835, Нют, Нидерланды — 9 августа 1903, Париж, Франция) — великий нидерландский криптограф, лингвист, историк, математик. Его самым знаменитым учёным трудом была работа под названием «Военная криптография (La Cryptographie Militaire)», там и были сформулированы эти требования к системам шифрования.

1. **Система должна быть** физически, если не математически, невскрываемой.

2. **Нужно, чтобы не требовалось** сохранение системы в тайне; попадание системы в руки врага не должно причинять неудобств (принцип Керкгоффса).

3. **Хранение и передача ключа** должны быть осуществимы без помощи бумажных записей; корреспонденты должны располагать возможностью менять ключ по своему усмотрению.

4. Система должна быть пригодной для сообщения через телеграф.

5. Система должна быть легко переносимой, работа с ней не должна требовать участия нескольких лиц одновременно.

6. Наконец, от системы требуется, учитывая возможные обстоятельства её применения, чтобы она была проста в использовании, не требовала значительного умственного напряжения или соблюдения большого количества правил.

3.1. Примеры симметричного шифрования

3.1.1. Шифр Цезаря

Пример —

$$\text{БАБУШКА} \longleftrightarrow \text{ВБФЩЛБ}.$$

Шифр Цезаря состоит в следующем:

- выписывается алфавит,
- а затем под ним выписывается тот же алфавит, но с циклическим сдвигом на несколько букв влево (в примере сдвиг на 1 букву). Например, для нашего примера:

А	Б	В	Г	Д	Е	...	Э	Ю	Я
Б	В	Г	Д	Е	Ё	...	Ю	Я	А

.

Зашифрование заключается в выборе буквы из первой строки и замену ее на букву второй строки.

Расшифрование представляет собой обратную процедуру.

Этот вид шифрования можно рассматриваться как вид перестановки букв алфавита. Его можно применять в коротких сообщениях. В длинных сообщениях перестановочные виды шифрования вскрываются частотным анализом (частота использования букв). Однако, если перестановка “хитрая”, то расшифрование может быть тяжёлым, часто используют суперкомпютеры для перебора.

3.1.2. Гаммирование

Гаммированием называется симметричный метод шифрования, основанный на «наложении» гамма-последовательности на открытый текст. Под наложением обычно понимается суммирование в каком-либо конечном поле или конечной группе.

Более определённо. Порождается некоторый случайный набор чисел — гамма шифра, и он складывается (обычно поблочно) с сообщением, например по заданному модулю. Ключом является гамма шифра, и если она велика, то шифр очень трудно взломать.

3.1.3. Аналитическое преобразование.

Задаётся функция, которая преобразует текст либо посимвольно, либо по блокам.

Посимвольное шифрование называют ещё поточным методом. Кроме того, при таком шифровании обычно учитывают место символа, так как иначе это просто перестановка символов.

Шифрование по блокам называют также блочным методом. Как правило, шифрование по блокам является частью более сложного шифра, как увидим в дальнейшем.

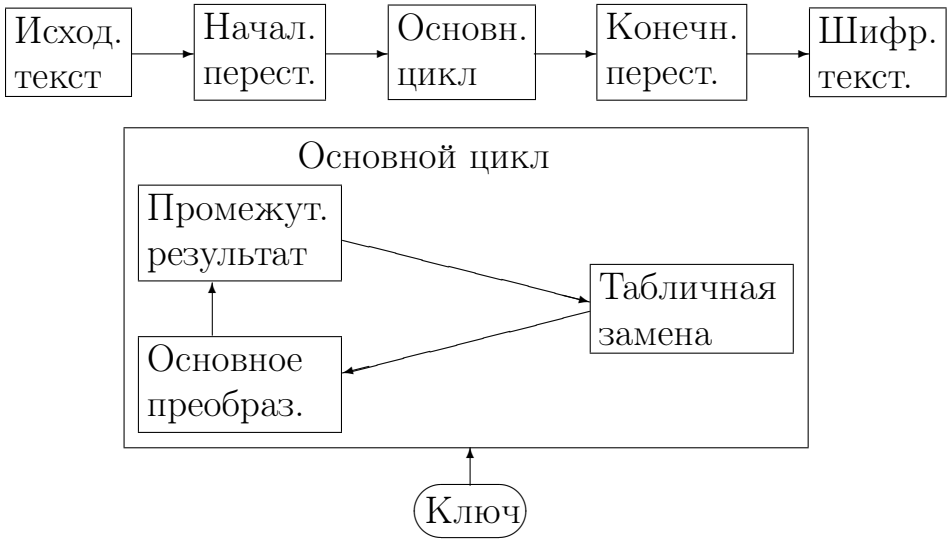
3.2. Схема Шеннона

Нужно использовать два общих принципа.

1. **Рассеивание** состоит в распространении влияния одного знака исходного сообщения на много знаков шифрованного, что позволяет скрыть статистические свойства исходного сообщения (в частности, одинаковые знаки исходного текста должны шифроваться разными символами).

2. **Перемешивание.** Нужно применять такие преобразования, чтобы было сложно установить взаимосвязь статистических свойств исходного и шифрованного текстов (в частности, одинаковые знаки шифрованного текста должны иметь много значений в исходном).

3.3. Схема шифрования



3.4. ГОСТ 28147-89

В качестве примера системы шифрования опишем шифр ГОСТ 28147-89.

0. Информация разбивается на блоки по 64 бита. Каждый из блоков разбивается на два субблока по 32 бита — N_1 и N_2. Далее блоки обрабатываются 16 или 32 раунда (повтора) работы алгоритма.
1. Субблок N_1 обрабатывается, получается новый субблок M_1.
2. Вычисляется $M_1 + N_2 \pmod 2$ побитно, получается новый субблок P_1.
3. Субблоки P_1 и N_2 меняются местами.

Опишем, как происходит обработка на шаге 1.

1. Наложение ключа (гаммирование). Ключ K имеет 256 бит, разбивается на подключи по 32 бита

$$K = (K0\ K1\ K2\ K3\ K4\ K5\ K6\ K7).$$

Вычисляется $N_1 + Kn \pmod{2^{32}}$, где N_1 и Kn рассматриваются, как числа в двоичной записи, и номер $n \in \{0, \ldots, 7\}$ подключа определяется по номеру раунда и режиму (есть несколько режимов).

2. Табличная замена. Субблок N_1 разбивается на 8 частей по 4 бита, значение каждой из которых заменяется согласно некоторой таблице замен.

3. Сдвиг. Выполняется побитовый циклический сдвиг влево на 11 битов.

В результате получается субблок M_1, который на следующем раунде играет ту же роль, что и N_1.

Высокая стойкость обеспечивается двумя свойствами.

1. **Ключ.** Ключ состоит из 256 бит — это много. Можно увеличивать до 320 бит. Кроме того, может быть секретная таблица замены.

2. **32 раунда.** После 8 раундов уже наступает рассеивание.

3.5. Виды шифрования

0. Личное шифрование (для себя).

1. **Архивное шифрование.** Для хранения информации.

2. **Абонентское** — пользовательское шифрование (ключ раздается абонентам).

3. **Прозрачное** — шифрование, которое происходит незаметно для пользователей (защищенные протоколы).

4. **Парное шифрование** (между двумя людьми), когда ключ известен только двум пользователям.

В последнем случае часто используется матрица ключей — набор ключей для парной связи. Матрицу ключей также называют сетевой таблицей.

Главным вопросом при распределении ключей является следующий.

Как сохранить в секрете ключ?

Распределение ключей можно осуществлять следующими способами.

1. **Личный ключ** — только для себя.

2. **Передача ключа лично в руки.**

3. **Матрица ключей для передача по сети.** Имеется набор ключей, называемый матрицей ключей, потом обычно передаётся сообщение, которое позволяет выбирать нужный ключ.

4. **Протокол Ди́ффи–Хе́ллмана).** Этот способ основан на криптографическом протоколе и позволяет двум и более сторонам получить общий секретный ключ, используя незащищенный от прослушивания канал связи. Полученный ключ используется для шифрования дальнейшего обмена с помощью алгоритмов симметричного шифрования.

Схема открытого распределения ключей, предложенная Диффи и Хеллманом, произвела настоящую революцию в мире шифрования, так как снимала основную проблему классической криптографии — проблему распределения ключей.

3.6. Асимметричное шифрование

Как правило, это шифрование с открытым ключом.

Требования Диффи–Хеллмана (Diffie–Hellman, 1976 г.):

1. Открытый ключ и закрытый ключ вычисляются просто для владельца.
2. Каждый пользователь может зашифровать, используя открытый ключ.
3. Расшифрование выполняется легко только для владельца закрытого ключа.
4. Злоумышленнику трудно найти закрытый ключ, зная открытый.
5. Злоумышленник, имея открытый ключ и зашифрованное сообщение, должен быть затруднён в расшифровке.

В этом случае понятие «трудно» означает, что перебор возможностей для ключа и/или возможностей для зашифрованного текста займёт очень много времени.

Для затруднения перебора используются однонаправленные функции.

Функция называется ***однонаправленной,*** если:

1) $y = f(x)$ легко вычисляется для данного x,
2) $x = f^{-1}(y)$ трудно вычисляется для данного y,

то есть это функция, которая легко вычисляется для любого входного значения, но трудно найти аргумент по заданному значению функции.

Широко используются функции с *потайным ходом*: значение $x = f^{-1}(y)$ легко находится, если известна, так называемая, «лазейка» или «потайной вход». В нашем случае это закрытый ключ.

3.7. Система RSA

(аббревиатура от фамилий Rivest, Shamir и Adleman, MIT[10])

Данная система официально предложена в 1978 году. Однако, ещё в августе 1977 года в колонке «Математические игры» Мартина Гарднера в журнале Scientific American, с разрешения Рональда Ривеста появилось первое описание криптосистемы RSA. Читателям было предложено дешифровать английскую фразу, зашифрованную описанным алгоритмом:

9686	9613	7546	2206	1477	1409	2225	4355
8829	0575	9991	1245	7431	9874	6951	2093
0816	2982	2514	5708	3569	3147	6622	8839
8962	8013	3919	9055	1829	9451	5781	5154

[10]MIT — Massachusetts Institute of Technology, Cambridge, Massachusetts. MIT является крупнейшей в мире научно-исследовательской структурой по объёму ежегодных военных заказов на военные исследования

В качестве открытых параметров системы были использованы числа

$$\begin{aligned} n = {} & 114381625757888867669235779976146612010218296721242362562561842935706935245733897830597123563958705058989075147599290026879543541, \\ e = {} & 9007. \end{aligned}$$

Число n состоит из 129 десятичных знаков, или имеет 425 бит в двоичном представлении, также оно известно как ***RSA*-129**.

За расшифровку предложенного сообщения была обещана награда в 100 долларов США. Для осуществления расшифровки нужно было разложить n на простые множители (факторизация). По заявлению Ривеста, для факторизации числа потребовалось бы более 40 квадриллионов лет.

Однако чуть более чем через 15 лет, 3 сентября 1993 года было объявлено о старте проекта распределённых вычислений с координацией через электронную почту по нахождению сомножителей числа RSA-129 и решению головоломки. На протяжении полугода более 600 добровольцев из 20 стран жертвовали процессорное время 1600 машин (две из которых были факс-машинами). В результате были найдены простые множители

$$\begin{aligned} \text{RSA-129} = {} & 3490529510847650949147849619903898133417764638493387843990820577 \\ & \times 327691329932667095499619881908344614131776429679929425397982885 33 \end{aligned}$$

и расшифровано исходное сообщение, которое представляет собой фразу

«THE MAGIC WORDS ARE SQUEAMISH OSSIFRAGE»
(«Волшебные слова — это брезгливый ягнятник»).

Полученную награду победители пожертвовали в фонд свободного программного обеспечения.

В декабре 1997 года была обнародована информация, согласно которой британский математик Клиффорд Кокс (Clifford Cocks), работавший в центре правительственной связи (GCHQ) Великобритании, описал криптосистему, аналогичную RSA в 1973 году.

RSA состоит в следующем:

- берется большое число n,
- блок текста $m < n$,
- ключи $e = k(\text{отк})$ и $d = k(\text{зак})$, $e, d < n$ и $ed \equiv 1 \pmod{n}$.

Зашифровка сообщения с использованием открытого ключа $e = k(\text{отк})$:

$$c \equiv m^e \pmod{n}.$$

Расшифровка сообщения с использованием закрытого ключа $d = k(\text{зак})$:

$$m \equiv c^d \equiv m^{ed} = m^{ed} \pmod{n}.$$

Проблемы:

1) если при нахождении c^d перебирать c^n $(n = 1, 2, \dots)$, то возникает проблема дискретного логарифма;
2) нахождение $d = k(\text{зак})$ зависит от проблемы факторизации.

Такая общая схема не проработана полностью, то есть не изучено всё что даёт трудности, пути их обхода и т.п. Видимо, это невозможно?!

Полностью разобран и проработан следующий важнейший частный случай.

1. $n = p \cdot q$, p и q — простые (большие) числа. Очень трудно разложить, если p и q большие, из-за того, что простых чисел очень много, а деление очень трудоёмко.
2. $e \cdot d = k(\text{отк}) \cdot k(\text{зак}) \equiv 1 \pmod{(p-1)(q-1)}$, поэтому $e = k(\text{отк})$ и $d = k(\text{зак})$ — взаимно простые с $(p-1)(q-1)$.

3.7.1. Недостатки

1. Для небольших n существуют приемы получения разложения $n = pq$.
2. Существуют такие p и q, что можно найти разложение $n = pq$.
3. Существуют такие $e = k(\text{отк})$, что легко найти $d = k(\text{зак})$.

3.7.2. Способы устранения недостатков

1. Особый подбор параметров:
   а) p и q большие;
   б) число $|p - q|$ достаточно большое;
   в) каждое из чисел $p \pm 1$, $q \pm 1$, $e = k(\text{отк})$ и $d = k(\text{зак})$ должно иметь большой простой множитель;
   г) НОД$(p-1, q-1)$ — небольшой.
2. Увеличение n влечет сильный рост вычислений при подборе.
3. При очень больших p и q выбор числа $d = k(\text{зак})$, не взаимно простым с $(p-1)(q-1)$ и меньшим n, имеет вероятность

$$\frac{1}{p} + \frac{1}{q},$$

поэтому возникает очень много случаев перебора для нахождения закрытого ключа.

Теорема (М. Винер (Michael J. Wiener)). *Если*

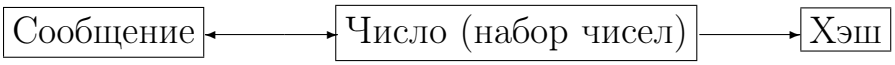
$$q < p < 2q \ \textit{и}\ d = k(\textit{зак}) < \frac{1}{3}(pq)^{\frac{1}{4}} = \frac{1}{3}n^{\frac{1}{4}},$$

то по открытому ключу можно эффективно восстановить закрытый ключ.

Есть предположение, что все $d = k(\text{зак}) < \sqrt{n} = n^{\frac{1}{2}}$ небезопасны.

3.8. Хэширование

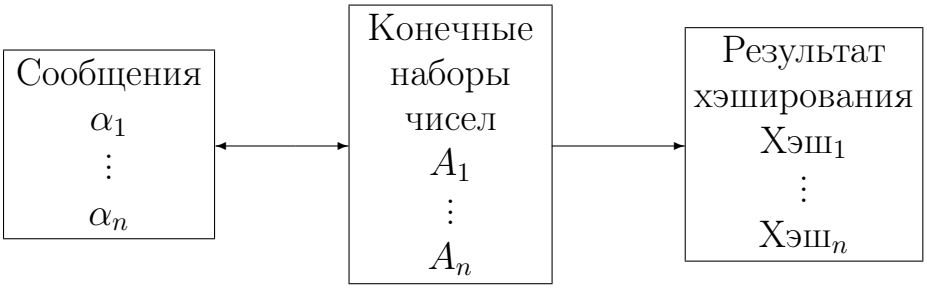
Схема хэширования:



После кодирования сообщение можно рассматривать как набор чисел или просто как число. Смысл хэширования состоит в том, что данному сообщению сопоставляется хэш (хеш), меньшее, как правило, число, которое вполне характеризует данное сообщение.

Ясно, что эта задача бессмысленна теоретически, если рассматривать ***все*** числа.

Однако, реально берётся ограниченное множество сообщений с конечным множеством элементов в каждом сообщении, поэтому появляется смысл.



Известные алгоритмы хэширования:

MD5 — размер хэша 128 бит (16 байт),

SHA1 — размер хэша160 бит (20 байт),

ГОСТ Р 34.11-94 — размер хэша 256 бит (32 байта).

Рассмотрим, как осуществляется хэширование в ГОСТ Р 34.11-94. Выполняется три шага:

1) если длина блока меньше 256 бит, то шаг 3), если больше 256 бит, то шаг 2);
2) последовательно блоки по 256 бит обрабатываются, и добавляется значение предыдущего блока к последующему, пока длина не станет меньше 256 бит.
3) дополняем нулями до 256 бит, обрабатываем как в шаге 2).

Описание обработки на втором шаге довольно громоздко, поэтому мы его опустим.

Требования к хэшу:

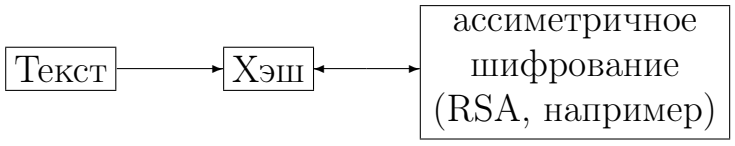
1) известен хэш, но трудно найти исходное сообщение,
2) трудно найти сообщения с одинаковым хэшем,
3) трудно найти по хэшу сообщение, отличное от исходного.

3.9. Электронная шифрования подпись (ЭЦП)

ЭЦП обычно используется для проверки:

1) целостности,
2) авторства,
3) и других целей.

Схема:



Более точно:

1) находим хэш исходного текста,
2) шифруем его с помощью секретного ключа и получаем ЭЦП,
3) вычисляем хэш полученного текста,
4) расшифровываем ЭЦП с помощью открытого ключа и
5) сравниваем с полученным хэшем.

Если хэши совпадают, проверка прошла, если нет, то проверка говорит о нарушениях.

3.9.1. ЭЦП ГОСТ Р 34.10-94

Параметры.

1. Простое число p удовлетворяет неравенствам

$$2^{509} < p < 2^{512} \quad \text{или} \quad 2^{1020} < p < 2^{1024}.$$

2. Простое число q удовлетворяет условиям

$$q \text{ делит } p-1 \quad \text{или} \quad 2^{254} < q < 2^{256}.$$

3. Число a выбирается с условиями

$$a \in \{1, \ldots, p-1\} \quad \text{такое, что} \quad a^q \equiv 1 \pmod{p}.$$

p, q, a не секретны!

Ключи.

1. Выбираем произвольно закрытый ключ

$$0 < k(\text{зак}) < q.$$

2. Находим открытый ключ следующим образом:

$$k(\text{отк}) \equiv a^{k(\text{зак})} \pmod{p}.$$

Создание ЭЦП.

0. Находим хэш h по ГОСТ Р 34.11-94.
1. Находим число $m \in \{1, \ldots, q-1\}$ следующим образом:

$$m = \begin{cases} 1, & \text{если } h \equiv 0 \pmod{q}; \\ m \equiv h \pmod{q}, & \text{если } h \not\equiv 0 \pmod{q}. \end{cases}$$

2. Выбираем случайное число x, которое будет закрытым ключом $k(\text{зак})$

$$x \in \{1, \ldots, q-1\}.$$

3. Находим число r следующим образом:

$$r_1 \equiv a^x \pmod{p} \text{ и } 0 \leqslant r_1 < p,$$

затем

$$r \equiv r_1 \pmod{q} \text{ и } 0 \leqslant r < q.$$

4. Если $r = 0$, то выбор x — плохой, берём другой x (возврат к шагу 2).
5. Находим число s следующим образом:

$$s \equiv k(\text{зак})r + xm \pmod{q}.$$

Полученная пара (r, s) является ЭЦП.

Проверка ЭЦП

1. Ищем хэш h_1 полученного сообщения и вычисляем m_1, как при построении ЭЦП.
2. Надо чтобы $r, s \in \{1, \ldots, q\}$, если нет, то ЭЦП не верна.
3. Делаем приведения по модулям (каждый раз берётся неотрицательное целое число, меньшее модуля) с использованием открытого ключа
$$y = k(\text{отк}) \equiv a^{k(\text{зак})} = a^x \pmod{p}.$$
   3.0) $z_0 \equiv m_1^{q-2} \pmod{q}$.
   3.1) $z_1 \equiv sz_0 \pmod{q}$.
   3.2) $z_2 \equiv -rz_0 \pmod{q}$.
   3.3) $z_3 \equiv a^{z_1} \pmod{p}$.
   3.4) $z_4 \equiv y^{z_2} \pmod{p}$.
   3.5) $z_5 \equiv z_3 z_4 \pmod{p}$.
   3.6) $z_6 \equiv z_5 \pmod{q}$.
4. Окончательная проверка
$$r = z_6?$$

4. Компьютерная безопасность

4.1. Основные понятия

Компьютерная система (КС) — абстракция, под которой могут подразумеваться обычные компьютеры, компьютерные сети (не глобальные), телефоны и смартфоны, гаджеты, управляющие модули и т.д. и т.п.

Доступ — возможность выполнения каких-либо действий в КС.

Защищённость для КС — характеристики КС, описывающие доступ в разных видах.

Политика безопасности (ПБ) — описание защищённости, выраженное в терминах описывающих КС.

4.2. Формализация

$\{S_i\}_{i\in I}$ — множество субъектов (активные сущности).

$\{O_j\}_{j\in J}$ — множество объектов (то, на что могут воздействовать субъекты). Часто считаем, что

$$\{S_i\}_{i\in I} \subseteq \{O_j\}_{j\in J}.$$

$\{F_j\}_{j\in J}$ — множество воздействий на объекты, более точно

$$F_j = \left\{ f_k^{(j)} : S_k \to O_j \right\}_{k\in K}, \text{ где } K \subseteq I.$$

$\{G_{i,j}\}_{j\in I, j\in J}$ — семейство множеств разрешённых воздействий субъектов на объекты, то есть $G_{i,j}$ — множество разрешённых воздействий субъекта S_i на объект O_j и $G_{i,j} \subseteq F_j$.

Замечание. Множества $\{S_i\}_{i\in I}$, $\{O_j\}_{j\in J}$, $\{F_j\}_{j\in J}$ и $\{G_{i,j}\}_{j\in I, j\in J}$ нестационарны, могут меняться во времени.

4.3. Начальные аксиомы защищённости

Аксиома 1. Надо задать семейство множеств $\{G_{i,j}\}_{j\in I, j\in J}$.

Аксиома 2. Безопасность описывается семейством множеств $\{G_{i,j}\}_{j\in I, j\in J}$.

Аксиома 3. Должен существовать субъект S_{cont}, контролирующий операции субъектов над объектами, а именно он разрешает только разрешённые воздействия. Этот субъект отвечает за политику безопасности.

Замечания.

1. Выполнение этих условий должно обеспечивать заданную политику безопасности.
2. Будут ещё аксиомы, которые будут уточнять выполнение политики безопасности.

4.4. Угрозы КС

4.4.1. Направленность угроз

Цель угроз — нарушение основных свойств безопасности информации (достоверность, доступность, целостность, конфиденциальность, актуальность и других.).

Принципы и типы воздействия угроз:

локально — непосредственное воздействие,

удалённо —

пассивно, когда происходит только знакомство с информацией без её изменения, но это нарушение конфиденциальности,

активно, когда происходит изменение информации, это является нарушением целостности, достоверности и др.

4.4.2. Реализация угроз

Реализация угроз через каналы утечки и воздействия не связанные с информацией:

электромагнитные — радио, низкочастотные излучения, электросеть, заземление, линейные (компьютерные сети), электронный шум и др.,

акустические — звук от клавиш клавиатуры, звук от принтеров (особенно матричных), звук от работы других компонентов КС,

визуальные — подсмотр, отражения и др.

Причины реализации угроз:

1) несоответствие ПБ реальной КС,
2) ошибки управления ПБ,
3) ошибки гарантий ПБ,
4) ошибки в программах,
5) ошибки, сбои в аппаратной части КС,
6) ЧЕЛОВЕЧЕСКИЙ ФАКТОР.

4.4.3. Задачи защиты в КС

1. Организационные. Затруднение доступа злоумышленнику к КС.

2. Организационно-технические. Создание специфики работы КС для обеспечения безопасности и предотвращение каналов утечки.

3. Программно-технические. Борьба с утечками на программном и техническом уровнях.

4.5. Модель КС

При построении модели КС надо учитывать следующие факторы.

1. Восприятие информации происходит через субъекты, к которым есть доступ.
2. Угрозы безопасности возникают от субъектов.
3. Субъекты влияют друг на друга.
4. Пользователь управляет субъектами внешним образом, и потому свойства субъектов не зависят от него!

Аксиома 4. *Субъекты порождаются из объектов субъектами,* что будем обозначать так:

$$\text{Create}(S_j, O_i) \longrightarrow S_k,$$

Кроме того, в этом случае будем говорить, что объект O_i *ассоциирован с* субъектом S_j. Отметим, что также есть *уничтожение объектов*

$$\text{Create}(S_j, O_i) \longrightarrow \varnothing = Null \text{ (уничтожение)}.$$

Определение. ***Потоком информации,*** или часто просто ***потоком*** называются операции на объекте O_i, реализуемые субъектом S_j и зависящие от объекта O_m, что будем обозначать так:

$$\text{Stream}(S_j, O_m) \longrightarrow O_i.$$

Примеры потоков — порождение и уничтожение объектов, чтение, запись и т. д.

Определения.

1. ***Доступом*** субъекта S_j к объекту O_i называется порождение потока.
2. ***Легальными потоками (обозначим L)*** называются потоки, разрешённые заданной политикой безопасности, ***несанкционированными потоками (обозначим N)*** называются потоки, неразрешённые заданной политикой безопасности.

3. ***Правилами доступа*** называются потоки из $\boldsymbol{L}$.

Определения.

1. ***Монитором обращений (МО)*** называется субъект, активизирующийся при возникновении потока.
2. ***Монитором безопасности объектов (МБО)*** называется ***МО,*** разрешающий поток из $\boldsymbol{L}$ и запрещающий поток из $\boldsymbol{N}$.
3. ***Монитором безопасности субъектов (МБС)*** называется ***МО***, разрешающий порождение фиксированного множества субъектов для любого объекта–источника.

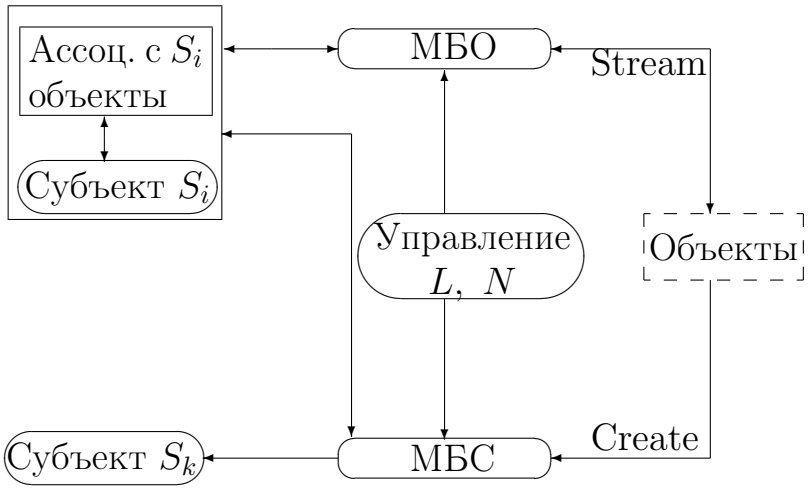
Определения.

1. Будем говорить, что ***субъекты корректны друг относительно друга***, если нет потока между ассоциированными с ними объектами.
2. Будем говорить, что ***субъекты абсолютно корректны друг относительно друга***, если множество потоков между ассоциированными с ними объектами не пересекаются.
3. Будем говорить, что ***множество субъектов изолировано (абсолютно изолировано),*** если существует ***МБС*** и порождаемые субъекты корректны (абсолютно корректны) относительно друг друга и ***МБС***.

Утверждение. *Абсолютная изолированность обеспечивает разрешение только для легальных потоков.*

Определение. ***Изолированной программной средой (ИПС)*** называется механизм реализации в КС абсолютной изолированности.

4.6. Схема ядра безопасности



4.7. Загрузка операционной системы (ОС)

Уровень	Субъект	Локализация	Представление	Реализация
0	Аппаратно-программный	ПЗУ	Сектора	BIOS
1	Первичный загрузчик	Загрузчик ОС	Сектора	BIOS и загрузчик ОС
2	Вторичный загрузчик	Драйверы ОС	Сектора	BIOS и загрузчик ОС
3	ОС	Ядро ОС	Файлы	Драйверы ОС
4	Пользователь	Приложения	Файлы	Ядро ОС

Вывод. ***ИПС нельзя сформировать в начальный момент загрузки!***

Замечание. В DOS и что было на нём (Windows 3.1, 95, 98) можно было работать с секторами, поэтому можно было изменять загруженную ОС.

В WinNT и более поздних (2000, XP, 7, 8, 10) нельзя работать с секторами!

ОДНАКО язык C и его производные (C++, C#, Java (обычно неявно)) позволяют работать с секторами. Кстати, в этом одна из причин использования C для написания ОС.

Утверждение. *Если последовательность загрузки не меняется и объекты при загрузке не меняются, то при окончании загрузки может быть сформирована ИПС.*

Аксиома 5. Порождение ИПС возможно только при неизменности субъектов, активизирующихся до начала контроля целостности объектов и последовательности загрузки. Эта неизменность должна обеспечиваться внешним образом.

Поясним, что значит “внешним образом”. Это может быть одно из следующих.

1. Внешние субъекты: загрузочные диски (дискеты), USB flash и т. п..
2. Локализация — отсутствие доступа извне, обычно отсутствие связи КС с Интернетом, “песочницы” и т. д..
3. Загрузка из другой ОС с системой безопасности, например, виртуальные машины.
4. Другие.

4.8. Опосредованный несанкционированный доступ (ОНСД)

Определение. ***Опосредованным несанкционированным доступом (ОНСД)*** называется следующее:

1) возникновение потоков, создающих новые субъекты, или

2) изменение ассоциированных объектов для активных субъектов, происходящее автономно (без управления пользователем).

Определение. ОНСД приводит к возникновению нелегальных потоков и неразрешённых субъектов. Такие субъекты называют ***разрушающими программными воздействиями (РПВ)*** или ***программными закладками.***

Необходимые условия для возникновения ОНСД:

1) существование субъекта, который может изменять объекты, ассоциированные с другими субъектами или порождать потоки от них к своим ассоциированным субъектам (перехват);

2) внедрение объектов–источников, порождающих субъекты со свойством 1.

Рассмотрим, что происходит при несанкционированных записи и чтении.

№	Санкционированные		Несанкционированные		Вред
	чтение	запись	чтение	запись	
1	0	0	0	0	Нет
2	0	1	0	0	Нет
3	1	0	0	0	Нет
4	1	1	0	0	Нет
5	0	0	0	1	Изменение приложения в памяти
6	0	1	0	1	Изменение вывода приложения
7	1	0	0	1	Изменение ввода приложения
8	1	1	0	1	6+7
9	0	0	1	0	Нет
10	0	1	1	0	Перенос вывода в память

№	Санкционированные		Несанкционированные		Вред
	чтение	запись	чтение	запись	
11	1	0	1	0	Перенос ввода в память
12	1	1	1	0	10 + 11
13	0	0	1	1	Размножение зловреда, не зависящего от приложения
14	0	1	1	1	6
15	1	0	1	1	7
16	1	1	1	1	8

Утверждение. *В ИПС с контролем целостности объектов-источников при порождении субъектов ОНСД невозможен.*

Замечание. Всегда много объектов в потоках информации (драйверы, сетевые средства и др.), поэтому нереально запретить чтение ассоциированных объектов. Следовательно, надо контролировать эти процессы внешне (базовая ОС) и не записывать (по возможности) на внешние носители.

4.9. Доступ пользователя к КС

Определения. Идентификацией называется присвоение пользователю идентификатора, который относится к разрешённым.

Аутентификацией называется проверка принадлежности пользователю идентификатора.

Учётная запись состоит из сочетания ***идентификации и аутентификация***,
также называется ***аккаунтом (account)***.

Аккаунты разделяются на следующие виды:
1) индивидуальные объекты (пропуск, карта, диск, USB flash и т.п.),
2) пароли различных видов,
3) биометрика,
4) другие.

Аутентификация

Аутентификация может быть использовать 2 или 3 стороны.

2 стороны. В этом случае принимающая сторона сама проверяет сведения необходимые для аутентификации. Такой вид аутентификации называется ***непосредственной аутентификацией (direct password authentification)***;

3 стороны. В этом случае добавляется ***доверенная сторона (арбитр = сервер аутентификации) (trusted third party authentification)***, который решает принять или отвергнуть принимающей стороне предлагаемую аутентификацию.

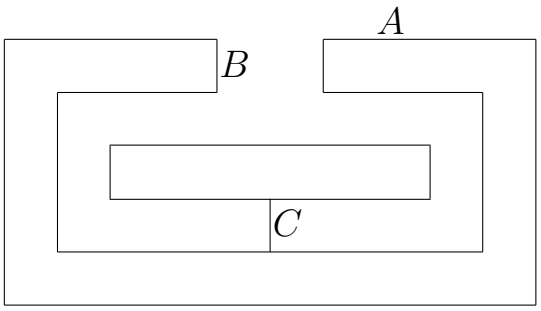
Для аутентификации необходимо выполнение одного или совокупности нескольких следующих сведений.

1. Пароль или его свёртка (plain text equivalent). Для этого должна быть база паролей, следовательно существует угроза её похищения.

2. Проверочное значение (verifier based). Задаётся вопрос или несколько вопросов, в зависимости от ответов происходит доступ к КС. Считается более стойким ко взлому, чем использование паролей.

3. Без непосредственной передачи пароля (zero knowledge). Как пример рассмотрим

Пещера Али Бабы



В C находится дверь с замком.

Проверка осуществляется следующим образом.

1. Проверяющий находится в A, предлагает доказывающему выйти из заданного (правого или левого) коридора. Он не видит, в какой коридор зашёл доказывающий.
2. Доказывающий добирается до C и, если надо, открывает дверь ключом. Если же ключа нет, то поворачивает обратно.
3. Проверяющий переходит в B и смотрит, из какого коридора вышел доказывающий.

Так повторяется несколько раз. Либо проверяющий убеждается, что доказывающий имеет ключ, либо отвергает это. Вероятность неверного ответа за n раундов равна $(0,5)^n$.

Вернёмся к аутентификации.

4. **Криптографический (cryptographic) подход.** За несколько действий вырабатывается ключ для взаимной аутентификации.

5. **Другие.**

Требования для паролей

Парольная аутентификация наиболее популярна. Обязательно нужно учитывать следующие требования.

1. **Минимальная и максимальная длина.** Малая длина паролей облегчает подбор, а при большой длине возможно внедрение вредоносного кода.

2. **Различные группы символов.** Использование различных групп символов расширяет базу для поиска пароля, что вызывает затруднение подбора.

3. **Использование словаря.** В этом случае облегчается проверка и затрудняется подбор.

4. **Минимальный и максимальный срок действия.** Слишком короткий срок действия либо приводит к старым паролям, либо даёт лишнюю возможность для взлома базы паролей, так как в этом случае частое происходит обращение и запись в базу паролей. Слишком длительный срок действия даёт больше возможностей по времени для подбора пароля.

5. **Другие.**

Не подвергается сомнению, что при использовании паролей

главная проблема — человеческий фактор!

Поскольку запоминание абстрактных сложных паролей очень тяжело, а пароли, связанные с конкретным человеком взламываются с использованием данных о нём, например, часто используется дата рождения.

4.10. Криптозащита приложений

Криптозащита приложений делится на два вида.

Наложенной криптозащитой приложений называется отдельный субъект, который должен работать в ОС.

Это приводит к тому, что приложение и защита должны одновременно воспринимать информацию, а это сложно.

Встроенной криптозащитой приложений называется субъект, размещённый внутри приложения.

Свойства	Наложенная	Встроенная
Сопряжение с приложением	при эксплуатации	при проектировании и разработке
Зависимость от приложения	низкая	высокая
Локализация объекта защиты	внешняя	внутренняя
Зависимость от ОС	полная	незначительная

4.11. Защита конфиденциальности информации

Выделяются следующие способы, которые скрывают истинное содержание информации, обеспечивая тем самым защиту конфиденциальности информации.

1. Сокрытие смысла. Шифр AVE MARIA состоит в преобразовании текста в религиозный (можно в спортивный, военный и т. д.). Предложен Иоганнесом Тритемием[12].

Пример. Приходит письмо:

> “Бог простит” или “Святая Богородица поможет”.

Шифр Аве Мария, точнее его часть, состоит в данном случае из следующего соответствия:

[12]Иоганнес Тритемий был аббатом монастыря в городе Вюрцбург. Он заслужил славу чернокнижника своим сочинением «Полиграфия» – в те времена так называлась тайнопись, – написанным в 1499 году и вышедшим в 1518 г. Это была первая печатная книга по криптографии.

Буква	Слово
А	простит
Д	Бог
Е	Богородица
Н	Святая
Т	поможет

Поэтому письмо означает:

“ДА” или “НЕТ”.

Может одна буква обозначаться несколькими словами, а сами сообщения содержать части обычного текста, который не имеет значения для получателя.

2. Употребление жаргонизмов. В этом случае привычные слова могут истолковываться совсем по-иному (например, нож — перо). Наиболее известны, так называемая, “блатная музыка” и использование специфических для некоторого вида слов. Например, спортивные тексты крайне трудны для перевода из-за обилия разнообразных терминов.

3. Стеганография. Стеганографией называется сокрытие одного вида информации в другом (например, текста в картинке или музыке). Здесь основная проблема состоит в том, чтобы понять сам факт стеганографии. Однако уже известны способы, когда даже после этого трудно извлечь скрываемую информацию.

4. Ложная информация. В определённом смысле, это может быть комбинация нескольких способов, когда внутри может быть скрыта истинная информация, например, читаются не все слова, и/или некоторые слова имеют заранее оговорённые значения.

5. Другие.

4.12. Кларк и Вилсон (Clark&Wilson, 1987 г.)

4.12.1. Целостность в КС по Кларку–Вилсону

Кларк и Вилсон выдвинули требования, необходимые для обеспечения целостности информации в КС. Приведём эти требования.

Сначала напомним понятие транзакции. Транзакция — группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще, и тогда она не должна произвести никакого эффекта.

1. Корректность транзакций.

2. Аутентификация пользователей.

3. Минимум привилегий.

4. Разграничение обязанностей.

5. Аудит событий.

6. Контроль.

7. Управление передачей привилегий.

8. Обеспечение работоспособности.

9. Простота использования защиты.

Корректность транзакций определяется следующим образом. Пользователь не должен модифицировать данные произвольно, а только определенными способами, так, чтобы сохранялась целостность данных. Другими словами, данные можно изменять только путем корректных транзакций и нельзя — произвольными средствами. Кроме того, предполагается, что "корректность" (в обычном смысле) каждой из таких транзакций может быть некоторым способом доказана. Принцип корректных транзакций по своей сути отражает основную идею определения целостности данных.

Для выполнения этих задач была разработана соответствующая модель.

4.12.2. Модель Кларка-Вилсона

Элементами этой модели являются данные, операции над данными и правила для этих операций.

Данные разделяются на два класса:

1) контролируемые данные — CD,
2) неконтролируемые данные — UD (не обеспечиваются защитой).

Операции над данными имеются двух видов:

1) контроль целостности информации, обозначаем V,
2) преобразования информации, обозначаем T.

Правила для операций следующие.

C1. Нужно контролировать CD.

C2. Для любых $t \in T$ и $c \in CD$ должно быть

$$V(t(c)) = \text{true},$$

должно быть указано, какие $t \in T$ допустимы для $c \in CD$.

E1. Проверка применимости $t \in T$ к $c \in CD$.

E2. Список разрешений для пользователей $t \in T$ и $c \in CD$.

C3. Список $t \in T$ из C2), удовлетворяющих разграничению обязанностей.

E3. Нужна аутентификация пользователей для применения $t \in T$.

C4. Любое применение $t \in T$ должно влечь регистрацию применения.

C5. Для любых $t \in T$ и $u \in UD$, если $t(u) \in CD$, то это должно быть корректно, то есть не нарушать правила.

E4. Только специальное лицо может изменять списки из C2) и E2), но оно не может изменить требования, регламентирующие эти действия.

4.12.3. Соответствие модели требованиям целостности

В следующей таблице указано, какие требования к целостности удовлетворяются, предложенными правилами.

Правила	C1	E1	C4	C2	C3	C5	E2	E3	E4
Требования	1, 6	1, 2, 3, 4	5	1	4	5	3, 4	2	1

4.13. Защита памяти

Без сомнения защита памяти играет огромную роль в обеспечении безопасности КС. Как пример, можем напомнить опосредованный несанкционированный доступ (ОНСД).

Используются следующие способы защиты памяти.

1. **Барьерные адреса** устанавливают начало пользовательской области памяти, отделяя её от ОС и т.п.

2. **Динамические области** выделяются в процессе исполнения.

3. **Адресные регистры** указывают начало и конец доступной приложению памяти.

4. **Страницы.** В этом случае адресное пространство делится на части, называемые страницами, и есть обращение к страницам. Возможно нахождение на одной странице кода и данных, что не безопасно.

5. Сегменты. Адресное пространство делится на области с подобным содержанием. В этом случае код и данные отделены, что даёт дополнительные возможности для защиты данных.

4.14. Модели безопасности

4.14.1. Матрица доступов HRU, 1971(6) г. (Harrison M., Ruzzo W., Ullman J.)

Для КС задаются:

O — **множество объектов,**

S — **множество субъектов,** причём $S \subseteq O$,

R — **множество прав доступа субъектов к объектам.** Например,

R = {Read (чтение), Write (запись), Exec (выполнение), Own (владение)}.

Определение. ***Матрицей доступов*** называется набор прав доступа $M[s, o] \subseteq R$ субъекта $s \in S$ к объекту $o \in O$ для всех субъектов и объектов .

Замечание. Возникает таблица, строки которой соответствуют субъектам, столбцы — объектам, элементами таблицы являются подмножества правил доступа. Поскольку она похожа на матрицу, это и дало название матрица доступов.

Функционирование системы рассматривается только с точки зрения изменений в матрице доступа. Возможные изменения определяются шестью следующими примитивными операторами:

1) внести право,

2) удалить право,

3) создать субъект,

4) уничтожить субъект,

5) создать объект,

6) уничтожить объект.

Из примитивных операторов создаются команды, преобразующие матрицу доступов. Получается система, характеризующая работу с матрицей доступов.

Определения.

1. Говорят, что ***возможна утечка права***, если вносится право в $M[s, o]$, которое ранее не было в $M[s, o]$.
2. ***Начальное состояние безопасно для права*** $r \in R$, если невозможен переход в состояние с утечкой права r.
3. Система называется ***монооперационной***, если в каждой команде выполняется только один примитивный оператор.

Теоремы.

1. *Существует алгоритм проверки монооперационной системы на безопасность начального состояния для данного права.*
2. *Не существует алгоритма проверки произвольной системы на безопасность начального состояния для данного права.*

4.14.2. Модель Take-Grant, 1976 г.

Для КС задаются:

O — **множество объектов,**

S — **множество субъектов,** причём $S \subseteq O$,

R — **множество прав доступа субъектов к объектам,**

$\{t, g\}$ — t **(take) брать права,** g **(grant) давать права.**

Состояние системы описывается её графом доступов, который определяется следующим образом.

Определение. ***Графом доступов*** $G = (S, O, E)$ называется конечный помеченный ориентированный граф без петель, представляющий текущие доступы в системе:

$S \times O$ — вершины графа,

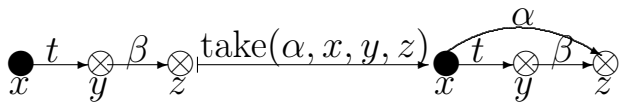
● — субъекты,

$\otimes$ — объекты, НЕ являющиеся субъектами,

$E \subseteq O \times O \times R$ — дуги графа, помеченные непустым подмножеством из R.

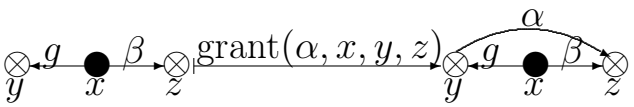
Правила преобразования графа

1. "Брать" — take(α, x, y, z)



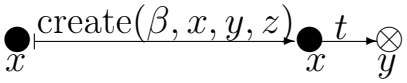
Субъект x берёт у объекта y права $\alpha \subseteq \beta$ на объект z.

2. "Давать" — grant(α, x, y, z)



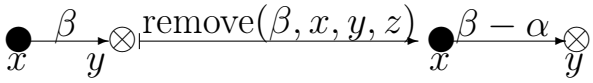
Субъект x даёт объекту y права $\alpha \subseteq \beta$ на объект z.

3. "Создать" — create(β, x, y)



Субъект x создаёт новый объект y с правами на него β.

4. "Удалить" — remove(α, x, y)



Субъект x удаляет права $\alpha \subseteq \beta$ на объект y.

Анализ графа G позволяет судить о возможных утечках прав доступа.

4.14.3. Модель Белла–Лападула (Bell D.E., LaPadulla L.J., 1975 г.)

Наряду с субъектами, объектами, правами доступа вводятся уровни секретности L. Изучаются запросы об L, ответы и действия на них[13]. Удаётся доказать следующий важный факт.

BST (Basic Security Theorem)**.** *В модели Белла–Лападула при определённых условиях КС безопасна.*

[13] В [5,с. 134] отмечается, что полное описание модели Белла–Лападулла недоступно. В интернете поэтому поводу сведения неполные и противоречивые. В частности, отмечается, что эта модель связана с общей моделью информационной безопасности США, в частности, в области обороны.

4.15. Документы по безопасности КС

Первым серьёзным документом по компьютерной безопасности была, так называемая, “Оранжевая книга”, точное название которой следующее:

> Trusted Computer System Evaluation Criteria (TCSEC)
> (критерии оценки безопасности компьютерных систем)
> Министерство обороны США, 1983 г.

В этом документе были впервые формально (хотя и не вполне строго) определены такие понятия, как “политика безопасности”, “корректность” и др. Предложенные в этом документе концепции защиты и набор функциональных требований послужили основой для формирования всех появившихся впоследствии стандартов безопасности.

В “Оранжевой книге” были выдвинуты требования к безопасности.

I. Политика безопасности.

1. Политика безопасности должна быть определена.
2. Объекты должны иметь ***метки***, определяющие доступ к объектам и другие свойства.

II. Аудит. Безопасная КС должна удовлетворять следующим требованиям.

3. Идентификация и аутентификация.
4. Регистрация и учёт.

III. Корректность (гарантии). В безопасной КС должны убыть обеспечены также следующие требования.

5. Контроль корректности работы защиты.
6. Непрерывность защиты.

Также были определены классы защиты.

Класс D — минимальная защита. Есть документация по защите, но без сертифицированного внедрения.

Класс C — дискреционная защита. Дискреционная = избирательная (неполная) защита. 2 подкласса: $C1$ и $C2$. Реализуется разграничение доступа в $C1$ — общее, а в $C2$ — управляемое.

Класс B — мандатное управление защитой. Использование меток безопасности. 3 подкласса: $B1$, $B2$ и $B3$. В $B1$ — просто использование меток, в $B2$ — наличие модели безопасности, в $B3$ — дополнительно монитор ссылок, защищённый от НСД и обрабатывающий все обращения.

Класс A — верифицированная защита. Должна быть математически доказана безопасность.

Развитие

После "Оранжевой книги" было разработано и принято много разнообразных документов по компьютерной безопасности. Мы приведём только некоторые из них

1. В 1991 "Критерии безопасности информационных технологий" (Information Technology Security Evaluation Criteria (ITSEC)) приняты от имени соответствующих органов Германии, Франции, Нидерландов и Великобритании.
2. В 1992 в США были приняты "Федеральные критерии безопасности информационных технологий" (Federal Criteria for Information Technology Security).
3. В 1992 году Гостехкомиссия (ГТК) разработала документы по защите от НСД (несанкционированного доступа).
   В настоящее время ведётся разработка нового комплекта документов по безопасности.

Приложения

П1. Задачи из книги Виноградова

Первая задача не из книги Виноградова, но её использует[14].

Задача. *Число* 123456789 *разложить на простые множители.*

Решение. Сумма цифр этого числа равна 45, значит, число делится на 9 и

$$123456789 = 9 \cdot 13717421.$$

Пусть p — наименьшее простое число, делящее 13717421. Тогда

$$13717421 = p \cdot a, \text{ где } p \leqslant a.$$

Отсюда

$$p \leqslant \sqrt{13717421} = 3703,7036869\ldots.$$

Поэтому $p \leqslant 3703$.

В книге Виноградова на двух последних страницах есть таблица простых чисел меньших 4070. Выпишем оттуда несколько простых чисел, меньших 3703

$$3701, 3697, 3691, 3677, 3673, 3671, 3659, 3643, 3637, 3631,$$
$$3623, 3617, 3613, 3607, 3593, \ldots.$$

Проверим, делится ли 13717421 на эти числа.

$$\frac{13717421}{3701} = 3706,409348 \text{ — нецелое число,}$$
$$\frac{13717421}{3697} = 3710,419529 \text{ — нецелое число,}$$
$$\frac{13717421}{3691} = 3716,451097 \text{ — нецелое число,}$$
$$\frac{13717421}{3677} = 3730,601305 \text{ — нецелое число,}$$
$$\frac{13717421}{3673} = 3734,664034 \text{ — нецелое число,}$$
$$\frac{13717421}{3671} = 3736,698719 \text{ — нецелое число,}$$
$$\frac{13717421}{3659} = 3748,953539 \text{ — нецелое число,}$$

[14]Видимо, это из математического фольклора

$$\frac{13717421}{3643} = 3765,418885 \text{ — нецелое число,}$$
$$\frac{13717421}{3637} = 3771,630739 \text{ — нецелое число,}$$
$$\frac{13717421}{3631} = 3777,863123 \text{ — нецелое число,}$$
$$\frac{13717421}{3623} = 3786,205078 \text{ — нецелое число,}$$
$$\frac{13717421}{3617} = 3792,485761 \text{ — нецелое число,}$$
$$\frac{13717421}{3613} = 3796,684472 \text{ — нецелое число,}$$
$$\frac{13717421}{3607} = 3803 \text{ — целое число!}$$

Из таблицы в книге Виноградова получаем, что 3803 — простое число. Поэтому

$$p = 3607$$

и получаем разложение данного числа в произведение простых

$$123\,456\,789 = 3 \cdot 3 \cdot 3\,607 \cdot 3\,803.$$

□

Задача (Виноградов, Численные примеры к главе IV, 1, b)**.** *Решить сравнение*

$$1215x \equiv 560 \pmod{2755}$$

Решение. Заметим, что коэффициенты в сравнении и модуль делятся на 5. Разделим их на 5. Получим

$$243x \equiv 112 \pmod{551}.$$

Это сравнение равносильно исходному. Решив его, мы достаточно легко запишем решение по модулю 2755. Так как $551 = 19 \cdot 29$, то возникает система сравнений

$$\begin{cases} 243x \equiv 112 \pmod{19} \\ 243x \equiv 112 \pmod{29} \end{cases}.$$

Решим каждое сравнение.

Так как

$$243 = 190 + 53 = 10 \cdot 19 + 3 \cdot 19 - 4 \equiv -4 \pmod{19},$$
$$112 = 95 + 17 = 5 \cdot 19 + 19 - 2 \equiv -2 \pmod{19},$$

то получаем из первого сравнения

$$-4x \equiv -2 \pmod{19}, \text{ или } 4x \equiv 2 \pmod{19}.$$

Поскольку $5 \cdot 4 = 20 \equiv 1 \pmod{19}$, то, умножив полученное сравнение на 5, получим

$$5 \cdot 4x \equiv 5 \cdot 2 \pmod{19}, \text{ или } x \equiv 10 \pmod{19}.$$

Проверим

$$243 \cdot 10 = 2430 = 1900 + 530 = 1900 + 380 + 38 + 112 \equiv 112 \pmod{19}.$$

Получили верное решение $x \equiv 10 \pmod{19}$ сравнения $243x \equiv 112 \pmod{19}$.

Так как

$$\begin{aligned} 243 &= 290 - 47 = 10 \cdot 29 - 2 \cdot 29 + 11 \equiv 11 \pmod{29}, \\ 112 &= 4 \cdot 29 - 4 \equiv -4 \pmod{29}, \end{aligned}$$

то получаем из второго сравнения

$$11x \equiv -4 \pmod{29}.$$

Так как $8 \cdot 11 = 88 = 29 \cdot 3 + 1 \equiv 1 \pmod{29}$, то, умножив полученное сравнение на 8, получим

$$8 \cdot 11x \equiv 8 \cdot -4 \pmod{19}, \text{ или } x \equiv -32 \pmod{29}.$$

Так как $-32 = -2 \cdot 29 + 26 \equiv 26 \pmod{29}$, то получим окончательно

$$x \equiv 26 \pmod{29}.$$

Проверим

$$243 \cdot 26 = (8 \cdot 29 + 11) \cdot 26 = 208 \cdot 29 + 286 = 208 \cdot 29 + 29 \cdot 6 + 112 \equiv 112 \pmod{29}.$$

Получили верное решение $x \equiv 26 \pmod{29}$ сравнения $243x \equiv 112 \pmod{29}$.

Таким образом получили систему

$$\begin{cases} x \equiv 10 \pmod{19} \\ x \equiv 26 \pmod{29}. \end{cases}$$

В общем случае такие системы решаются при помощи, так называемой *"китайской теоремы об остатках (Chinese Remainder Theorem)*[15]*"*, но мы поступим иначе. Полученную систему сравнений перепишем как систему равенств

$$\begin{cases} x = 10 + 19s \\ x = 26 + 29t \end{cases},$$

[15]Это первый результат § 5 главы IV, с. 60, в книге Виноградова [3].

где s и t — целые. Теперь

$$10 + 19s = 26 + 29t, \text{ или } 19s = 16 + 29t.$$

Отсюда по модулю 19 имеем

$$16 + 10t \equiv 0 \pmod{19}, \text{ или } 10t \equiv -16 \equiv 3 \pmod{19}.$$

Так как $2 \cdot 10 = 20 \equiv 1 \pmod{19}$, то имеем

$$t \equiv 20t \equiv 2 \cdot 3 \equiv 6 \pmod{19}.$$

Отсюда

$$t = 6 + 19u$$

для целого u и

$$x = 26 + 29(6 + 19u) = 26 + 174 + 29 \cdot 19u = 200 + 551u$$

и поэтому

$$x \equiv 200 \pmod{551}.$$

Проверим

$$243 \cdot 200 = 48600 = 48488 + 112 = 88 \cdot 551 + 112 \equiv 112 \pmod{551}.$$

Итак, решено сравнение $243x \equiv 112 \pmod{551}$, которое получилось из исходного делением на 5.

Далее поделим целое число u на 5 с остатком

$$u = v + 5w, \text{ где } v \in \{0, 1, 2, 3, 4\} \text{ и } w \text{ — целое.}$$

Отсюда

$$x = 200 + 551u = 200 + 551(v + 5w) = 200 + 551v + 2755w$$

Подставим в это равенство значения $v \in \{0, 1, 2, 3, 4\}$ и получим окончательный ответ

$$\begin{aligned}
&x \equiv 200 \pmod{2755},\\
&x \equiv 200 + 551 = 751 \pmod{2755},\\
&x \equiv 200 + 551 \cdot 2 = 1302 \pmod{2755},\\
&x \equiv 200 + 551 \cdot 3 = 1803 \pmod{2755},\\
&x \equiv 200 + 551 \cdot 4 = 2304 \pmod{2755}.
\end{aligned}$$

□

Замечание. Приведём другое решение. Оно длиннее, но имеет некоторые поучительные моменты. Например, как использовать возведение в степень в сравнениях.

Сначала для модуля, равного 19. Так как $243 = 3^5$ и $112 \equiv 17 \equiv -2 \pmod{19}$, то получаем из первого сравнения

$$3^5 x \equiv -2 \pmod{19}.$$

Поскольку $3 \cdot 6 = 18 \equiv -1 \pmod{19}$, то

$$\begin{aligned} 13 \equiv -6 \equiv 3^{-1} \pmod{19}, \text{ то есть} \\ 13 \cdot 3 \equiv (-6) \cdot 3 \equiv 1 \pmod{19}. \end{aligned}$$

Подсчитаем

$$\begin{aligned} (-6)^5 = (-6)^2 \cdot (-6)^2 \cdot (-6) = 36 \cdot 36 \cdot (-6) \equiv \\ \equiv (-2) \cdot (-2) \cdot (-6) \equiv -24 \equiv -5 \pmod{19}. \end{aligned}$$

Таким образом

$$243 \cdot (-5) \equiv 3^5 \cdot (-6)^5 \equiv (3 \cdot (-6))^5 \equiv 1^5 \equiv 1 \pmod{19}.$$

Проверим

$$243 \cdot (-5) = -1215 = -1216 + 1 = -(19 \cdot 64) + 1 \equiv 1 \pmod{19}.$$

Умножим сравнение $3^5 x \equiv -2 \pmod{19}$ на -5 и получим

$$x \equiv (-5) \cdot (-2) \equiv 10 \pmod{19}.$$

Теперь рассмотрим модуль 29. Так как $243 = 3^5$ и $112 \equiv 25 \equiv -4 \pmod{29}$, то получаем

$$3^5 x \equiv -4 \pmod{29}.$$

Поскольку $3 \cdot 10 = 30 \equiv 1 \pmod{29}$, то

$$10 \equiv 3^{-1} \pmod{29}.$$

Подсчитаем

$$\begin{aligned} 10^5 = 10^2 \cdot 10^2 \cdot 10 = 100 \cdot 100 \cdot 10 \equiv \\ \equiv 13 \cdot 13 \cdot 10 \equiv 169 \cdot 10 \equiv -5 \cdot 10 \equiv -50 \equiv 8 \pmod{29}. \end{aligned}$$

Таким образом

$$243 \cdot 8 \equiv 3^5 \cdot 10^5 \equiv (3 \cdot 10)^5 \equiv 1^5 \equiv 1 \pmod{29}.$$

Проверим

$$243 \cdot 8 = 1944 = 1943 + 1 = 29 \cdot 67 + 1 \equiv 1 \pmod{29}.$$

Умножим сравнение $3^5 x \equiv -4 \pmod{29}$ на 8 и получим

$$x \equiv 8 \cdot -4 \equiv -32 \equiv -3 \equiv 26 \pmod{29}.$$

Проверим

$$243 \cdot (-3) = -729 = -725 - 4 = -25 \cdot 29 - 4 \equiv -4 \pmod{29}.$$

Задача (Виноградов, Численные примеры к главе III, 1, а)**.** *Найти остаток от деления*

$$(12371^{56} + 34)^{28} \text{ } \textit{на} \text{ } 111.$$

Решение. Так как

$$12371 = 111 \cdot 111 + 50 \equiv 50 \pmod{111},$$

то по формуле бинома Ньютона

$$(111 \cdot 111 + 50)^{56} = 111(\dots) + 50^{56}.$$

Следовательно ещё раз по формуле бинома Ньютона

$$(12371^{56} + 34)^{28} \equiv 111(\dots) + (50^{56} + 34)^{28} \equiv (50^{56} + 34)^{28} \pmod{111}.$$

Итак задача свелась к тому, что надо найти остаток от деления

$$(50^{56} + 34)^{28} \text{ на } 111.$$

Переформулируем задачу в терминах сравнений.

Найти такое $x \in \{0, 1, \dots, 110\}$, что

$$x \equiv (50^{56} + 34)^{28} \pmod{111}.$$

Так как

$$111 = 3 \cdot 37,$$

то возникает система сравнений

$$\begin{cases} x \equiv (50^{56} + 34)^{28} \pmod{3} \\ x \equiv (50^{56} + 34)^{28} \pmod{37} \end{cases}.$$

Решим каждое сравнение.

Так как $50 \equiv -1 \pmod 3$ и $34 \equiv 1 \pmod 3$, то получаем

$$(50^{56} + 34)^{28} \equiv ((-1)^{56} + 1)^{28} \equiv (1 + 1)^{28} \equiv (-1)^{28} \equiv 1 \pmod 3.$$

Итак решением сравнения $x \equiv (50^{56} + 34)^{28} \pmod 3$ является

$$x \equiv 1 \pmod 3.$$

Так как $50 \equiv 13 \pmod{37}$ и $34 \equiv -3 \pmod{37}$, то получаем

$$(50^{56} + 34)^{28} \equiv ((13)^{56} - 3)^{28} \pmod{37}.$$

Найдём $13^{56} \pmod{37}$. Имеем

$$\begin{aligned} &13^2 = 169 = 148 + 21 \equiv 21 \equiv -16 \pmod{37}, \\ &13^4 \equiv (-16)^2 = 256 = 222 + 34 \equiv 34 \equiv -3 \pmod{37}, \\ &13^8 \equiv (-3)^2 = 9 \pmod{37}. \end{aligned}$$

Отсюда

$$13^{56} = ((13)^8)^7 \equiv 9^7 \pmod{37}.$$

Так как

$$\begin{aligned} &9^2 = 81 = 74 + 7 \equiv 7 \pmod{37}, \\ &9^4 \equiv 7^2 = 49 = 37 + 12 \equiv 12 \pmod{37}, \end{aligned}$$

то

$$\begin{aligned} 9^7 = 9^4 \cdot 9^2 \cdot 9 \equiv 12 \cdot 7 \cdot 9 = 84 \cdot 9 = (74 + 10) \cdot 9 \equiv \\ \equiv 10 \cdot 9 = 90 = (74 + 16) \equiv 16 \pmod{37}. \end{aligned}$$

Таким образом

$$\begin{aligned} (50^{56} + 34)^{28} \equiv ((13)^{56} - 3)^{28} \equiv (16 - 3)^{28} \equiv 13^{28} \equiv ((13)^4)^7 \equiv (-3)^7 = \\ = (-3)^4 \cdot (-3)^2 \cdot (-3) = 81 \cdot 9 \cdot (-3) = (74 + 7) \cdot 9 \cdot (-3) \equiv \\ \equiv 7 \cdot 9 \cdot (-3) = 63 \cdot (-3) = (74 - 11) \cdot (-3) \equiv (-11) \cdot (-3) = \\ = 33 \equiv -4 \pmod{37}. \end{aligned}$$

Итак решением сравнения $x \equiv (50^{56} + 34)^{28} \pmod{37}$ является

$$x \equiv -4 \pmod{37}.$$

Имеем систему

$$\begin{cases} x \equiv 1 \pmod 3 \\ x \equiv -4 \pmod{37} \end{cases}$$

Полученную систему сравнений перепишем как равенства

$$\begin{cases} x = 1 + 3s \\ x = -4 + 37t \end{cases},$$

где s и t — целые. Теперь

$$1 + 3s = -4 + 37t, \text{ или } 3s = -5 + 37t$$

Отсюда получаем сравнение по модулю 3

$$1 + t \equiv 0 \pmod 3, \text{ или } t \equiv 2 \pmod 3.$$

Следовательно

$$t = 2 + 3u$$

для целого u и

$$x = -4 + 37(2 + 3u) = -4 + 74 + 37 \cdot 3u = 70 + 111u.$$

Надо взять $u = 0$, так как по условию должны иметь $x \in \{0, 1, \ldots, 110\}$. Итак, получаем ответ

$$x = 70.$$

□

Задача (Виноградов, Численные примеры к главе IV, 1, а)**.** *Решить сравнение*

$$256x = 179 \pmod{337}.$$

Решение. 337 — простое число. Найдём такое целое t, что

$$256t \equiv 1 \pmod{337}.$$

Так как $256 = 2^8$, то достаточно найти обратное для 2 по модулю 337 и возвести его в восьмую степень. Поскольку

$$337 = 1 + 336 = 1 + 2 \cdot 168, \text{ или } 2(-168) = 1 - 337,$$

то

$$2^{-1} \equiv -168 \equiv 169 = 13^2 \pmod{337}.$$

Найдём теперь нужное t двумя способами.

Так как

$$168 = 8 \cdot 3 \cdot 7 = 2^3 \cdot 3 \cdot 7,$$

то надо найти $8^8 = 2^{24}$, 3^8 и 7^8 по модулю 337.

Сначала найдём $8^8 = 2^{24} \pmod{337}$. Так как

$$8^3 = 2^9 = 512 \equiv 175 \equiv -162 \pmod{337},$$

то

$$8^8 = 2^{24} = 8^3 \cdot 8^3 \cdot 8^2 = 512 \cdot 512 \cdot 2^6 \equiv 175 \cdot 175 \cdot 2^2 \cdot 2^4 = 350 \cdot 350 \cdot 2^4 \equiv$$
$$\equiv 13 \cdot 13 \cdot 2 \cdot 8 = 169 \cdot 2 \cdot 8 = 338 \cdot 8 \equiv 1 \cdot 8 = 8 \pmod{337}.$$

Теперь найдём $3^8 \pmod{337}$. Так как

$$3^6 = 729 \equiv 55 \pmod{337},$$

то

$$3^8 = 3^6 \cdot 3^2 \equiv 55 \cdot 9 = 495 \equiv 158 \pmod{337}.$$

Наконец найдём $7^8 \pmod{337}$. Так как

$$7^3 = 343 \equiv 6 \pmod{337},$$

то

$$7^8 = 7^3 \cdot 7^3 \cdot 7^2 \equiv 6 \cdot 6 \cdot 7^2 = 6 \cdot 294 \equiv 6 \cdot (-43) = -258 \equiv 79 \pmod{337}.$$

Итак, получили

$$(-168)^8 = (-2)^8 \cdot 3^7 \cdot 7^8 \equiv 8 \cdot 158 \cdot 79 = (2 \cdot 158)(4 \cdot 79) =$$
$$= 316 \cdot 316 \equiv 21 \cdot 21 = 441 \equiv 104 \pmod{337}.$$

Так как $169 = 13^2$, то

$$13^3 = 169(2 \cdot 6 + 1) = 338 \cdot 6 + 169 \equiv 1 \cdot 6 + 337 - 168 = -162 \pmod{337}$$

и

$$169^8 = 13^{16} = (13^3)^5 \cdot 13 \equiv (-162)^5 \cdot 13 = (-2)^5 \cdot 81^5 \cdot 13 =$$
$$= (-2)^5 \cdot 3^{20} \cdot 13 \pmod{337}.$$

Надо найти 3^{20} по модулю 337. Так как

$$3^6 = 729 \equiv 55 \pmod{337},$$

то

$$3^{20} = (3^6)^3 \cdot 3^2 \equiv 55^3 \cdot 9 = 55^2 \cdot 495 \equiv (11 \cdot 55)(5 \cdot 158) = 605 \cdot 790 = \\ = (-69) \cdot 116 = -23) \cdot 348 \equiv (-23) \cdot 11 = -253 \equiv 84 \pmod{337}.$$

Следовательно

$$169^8 = 13^{16} = (-2)^5 \cdot 3^{20} \cdot 13 \equiv -32 \cdot 84 \cdot 13 = (-4 \cdot 84)(8 \cdot 13) = \\ = (-336) \cdot 104 \equiv 1 \cdot 104 = 104 \pmod{337}.$$

Итак

$$256 \cdot 104 \equiv 1 \pmod{337}.$$

Умножим сравнение

$$256x \equiv 179 \pmod{337}$$

на 104 и получим

$$x \equiv 179 \cdot 104 = 358 \cdot 52 \equiv 21 \cdot 52 = 3(7 \cdot 52) = 3 \cdot 364 \equiv 3 \cdot 27 \equiv \\ \equiv 81 \pmod{337}.$$

Итак, ответ

$$x \equiv 81 \pmod{337}.$$

□

П2. Коды, связанные с полем из 8 элементов

П2.1. Построение поля из 8 элементов

Рассмотрим многочлен

$$p = x^3 + x + 1 \in \mathbf{F}_2[x].$$

Многочлен p не имеет корней в $\mathbf{F}_2$, так как

$$p(0) = 0^3 + 0 + 1 = 1 \neq 0,$$
$$p(1) = 1^3 + 1 + 1 = 1 \neq 0.$$

Теперь рассмотрим эквивалентность многочленов в $\mathbf{F}_2[x]$ по модулю p. Любые многочлены g и h над $\mathbf{F}_2$ можно сравнить по модулю p

$$g \equiv h \pmod{p} \text{равносильно } p \text{ делит } g - h.$$

С другой стороны, любой многочлен g можно разделить на p с остатком

$$g = p \cdot q + r, \text{ где } r = 0, \text{ или степень } r \text{ строго меньше } 3.$$

Как в случае чисел, ясно, что эквивалентные многочлены будут иметь одинаковые остатки от деления (уголком) на p. Все остатки многочленов от деления на p будут иметь вид

$$r = a_0 + a_1 x + a_2 x^2, \text{ где } a_0, a_1, a_2 \in \mathbf{F}_2.$$

Понятно, что разные остатки будут давать неэквивалентные многочлены, поэтому классов эквивалентных многочленов будет 8. Можно работать с остатками, учитывая, что

$$p = x^3 + x + 1 \equiv 0 \pmod{p} \text{ и } x^3 \equiv x + 1 \pmod{p}.$$

Остатки можно складывать и перемножать, для этого надо сначала перемножить остатки как обычные многочлены, а затем у произведения взять остаток от деления p.

Остатки при таких определениях сложения и умножения образуют поле $\mathbf{F}_8$ из 8 элементов

$$\left\{0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1\right\}.$$

Построим таблицу умножения в этом поле.

$\cdot$	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		1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
x			x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
$x+1$				x^2+1	x^2+x+1	x^2	1	x
x^2					x^2+x	x	x^2+1	1
x^2+1						x^2+x	x	$x+1$
x^2+x							x	x^2
x^2+x+1								$x+1$

Отметим, что

$$1 = x(x^2+1) = (x+1)(x^2+x) = x^2(x^2+x+1),$$

поэтому все ненулевые элементы обратимы, и действительно получили поле $\mathbf{F}_8$.

Имеем

$$x^3 = x + 1,$$

$$x^4 = x \cdot x^3 = x^2 + x,$$
$$x^5 = x \cdot x^4 = x^3 + x^2 = x^2 + x + 1,$$
$$x^6 = x \cdot x^5 = x^3 + x^2 + x = x^2 + 1,$$
$$x^7 = x \cdot x^6 = x^3 + x = 1.$$

Таким образом

$$\mathbf{F}_8^* = \mathbf{F}_8 \setminus \{0\} = \{1, x, x^2, x^3, x^4, x^5, x^6\}$$

— мультипликативная группа порядка 7, и x — примитивный элемент поля $\mathbf{F}_8$.

Построим таблицу сложения для $F_8 = \{0, 1, x, x^2, x^3, x^4, x^5, x^6\}$

$+$	0	1	x	x^2	x^3	x^4	x^5	x^6
0	0	1	x	x^2	x^3	x^4	x^5	x^6
1	1	0	x^3	x^6	x	x^5	x^4	x^2
x	x	x^3	0	x^4	1	x^2	x^6	x^5
x^2	x^2	x^6	x^4	0	x^5	x	x^3	1
x^3	x^3	x	1	x^5	0	x^6	x^2	x^4
x^4	x^4	x^5	x^2	x	x^6	0	1	x^3
x^5	x^5	x^4	x^6	x^3	x^2	1	0	x
x^6	x^6	x^2	x^5	1	x^4	x^3	x	0

Каждая строчка и каждый столбец содержат все элементы поля $\mathbf{F}_8$.

П2.2. Построение БЧХ-кода

Из поля $\mathbf{F}_8$ (конечное поле из 8 элементов) возникает БЧХ-код, совпадающий с кодом Хэмминга, который является $[2^3-1, 2^3-1-3, 3] = [7, 4, 3]$-кодом. Для построения этого кода надо выбрать базис $\mathbf{F}_8$ над $\mathbf{F}_2$. Выберем базис

$$1, x, x^2.$$

Проверочная матрица нашего кода

$$H = \begin{pmatrix} 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Проверочная матрица определяет для нахождения БЧХ-кода однородную система линейных уравнений

$$H\vec{x}^t = \vec{0}^t \longleftrightarrow \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Запишем эту систему в виде

$$\begin{cases} x_1 + x_4 + x_5 + x_7 = 0 \\ x_2 + x_4 + x_6 + x_7 = 0 \\ x_3 + x_5 + x_6 + x_7 = 0 \end{cases}.$$

По модулю 2 получаем

$$\begin{cases} x_1 = x_4 + x_5 + x_7 \\ x_2 = x_4 + x_6 + x_7 \\ x_3 = x_5 + x_6 + x_7 \end{cases}.$$

Теперь найдём множество решений полученной системы линейных уравнений. Имеем

x_1	x_2	x_3	x_4	x_5	x_6	x_7
1	1	0	1	0	0	0
1	0	1	0	1	0	0
0	1	1	0	0	1	0
1	1	1	0	0	0	1

,

и получаем базисные решения

$$\begin{aligned} c_1 &= (1,1,0,1,0,0,0); \\ c_2 &= (1,0,1,0,1,0,0); \\ c_3 &= (0,1,1,0,0,1,0); \\ c_4 &= (1,1,1,0,0,0,1). \end{aligned}$$

Получаем искомый код

$$C = \{t_1c_1 + t_2c_2 + t_3c_3 + t_4c_4 \mid t_1, t_2, t_3, t_4 \in \mathbf{F}_2 = \{0,1\}\}.$$

Выпишем все элементы этого кода

0) $\vec{0} = (0,0,0,0,0,0,0)$;

1) $c_1 = (1,1,0,1,0,0,0)$;
2) $c_2 = (1,0,1,0,1,0,0)$;
3) $c_3 = (0,1,1,0,0,1,0)$;
4) $c_4 = (1,1,1,0,0,0,1)$;
5) $c_1 + c_2 = (0,1,1,1,1,0,0)$;
6) $c_1 + c_3 = (1,0,1,1,0,1,0)$;
7) $c_1 + c_4 = (0,0,1,1,0,0,1)$;
8) $c_2 + c_3 = (1,1,0,0,1,1,0)$;
9) $c_2 + c_4 = (0,1,0,0,1,0,1)$;
10) $c_3 + c_4 = (1,0,0,0,0,1,1)$;
11) $c_1 + c_2 + c_3 = (0,0,0,1,1,1,0)$;
12) $c_1 + c_2 + c_4 = (1,0,0,1,1,0,1)$;
13) $c_1 + c_3 + c_4 = (0,1,0,1,0,1,1)$;
14) $c_2 + c_3 + c_4 = (0,0,1,0,1,1,1)$;
15) $c_1 + c_2 + c_3 + c_4 = (1,1,1,1,1,1,1)$.

Напишем распределение по весам Хэмминга.

- Вес 0 имеет вектор:
$$\vec{0} = (0,0,0,0,0,0,0).$$

- Вес 3 имеют векторы:
$$c_1, c_2, c_3, c_1 + c_4, c_2 + c_4, c_3 + c_4, c_1 + c_2 + c_3.$$

- Вес 4 имеют векторы:
$$c_4, c_1 + c_2, c_1 + c_3, c_2 + c_3, c_1 + c_2 + c_4, c_1 + c_3 + c_1, c_1 + c_2 + c_3.$$

- Вес 7 имеет вектор:
$$c_1 + c_2 + c_3 + c_4.$$

Подведём итог.

- Код C задаётся строчками (векторами) длины 7.

- Любой вектор из кода C линейно выражается через 4 линейно независимых вектора c_1, c_2, c_3 и c_4.

- В коде C минимальный вес *ненулевых* векторов равен 3, поэтому минимальное расстояние кода равно 3.

Следовательно код C является $[7,4,3]$-кодом, как утверждалось.

Порождающей матрицей кода C является матрица G размера 4×7, строчками которой являются c_1, c_2, c_3, c_4. Итак

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Пусть

$$A = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}.$$

Тогда

$$H = (E_3|A) \quad \text{и} \quad G = \left(A^t|E_4\right)$$

и поэтому

$$GH^t = \left(A^t|E_4\right)\begin{pmatrix} E_3 \\ \hline A^t \end{pmatrix} = \left(A^tE_3 + E_4A^t\right) = \left(A^t + A^t\right) = \mathbf{0}_{4\times 3}.$$

П2.3. Построение расширенного кода

Построим расширенный (удлинённый) код $\overline{C}$ с проверочной матрицей

$$\overline{H} = \left(\begin{array}{c|ccc} 1 & 1 & \dots & 1 \\ \hline \vec{0}^t & & H & \end{array}\right) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Имеем систему

$$\begin{cases} x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 0 \\ \qquad\qquad\quad x_2 + x_5 + x_6 + x_8 = 0 \\ \qquad\qquad\quad x_3 + x_5 + x_7 + x_8 = 0 \\ \qquad\qquad\quad x_4 + x_6 + x_7 + x_8 = 0 \end{cases}.$$

Преобразуя 2–4 уравнения, получим

$$\begin{cases} x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = 0 \\ x_2 = x_5 + x_6 + x_8 \\ x_3 = x_5 + x_7 + x_8 \\ x_5 = x_6 + x_7 + x_8 \end{cases}.$$

Подставим выражения x_2, x_3 и x_5 в первое уравнение системы и получим

$$\begin{aligned} 0 = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 = \\ = x_1 + (x_5 + x_6 + x_8) + (x_5 + x_7 + x_8) + (x_6 + x_7 + x_8) + x_5 + x_6 + x_7 + x_8 = \\ = x_1 + x_6 + x_7 + x_5 = x_1 + x_5 + x_6 + x_7 = 0. \end{aligned}$$

Поэтому

$$x_1 = x_5 + x_6 + x_7.$$

Окончательно получим

$$\begin{cases} x_1 = x_5 + x_6 + x_7 \\ x_2 = x_5 + x_6 + x_8 \\ x_3 = x_5 + x_7 + x_8 \\ x_5 = x_6 + x_7 + x_8 \end{cases}.$$

Теперь опишем множество решений рассматриваемой системы

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
1	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0
1	0	1	1	0	0	1	0
0	1	1	1	0	0	0	1

.

Получаем базисные решения

$$\begin{aligned} d_1 &= (1, 1, 1, 0, 1, 0, 0, 0); \\ d_2 &= (1, 1, 0, 1, 0, 1, 0, 0); \\ d_3 &= (1, 0, 1, 1, 0, 0, 1, 0); \\ d_4 &= (0, 1, 1, 1, 0, 0, 0, 1). \end{aligned}$$

Теперь наш код

$$\overline{C} = \{t_1 d_1 + t_2 d_2 + t_3 d_3 + t_4 d_4 \mid t_1, t_2, t_3, t_4 \in \mathbf{F}_2 = \{0, 1\}\}.$$

Выпишем все элементы кода $\overline{C}$.

0) $\vec{0} = (0, 0, 0, 0, 0, 0, 0, 0)$;
1) $d_1 = (1, 1, 1, 0, 1, 0, 0, 0)$;
2) $d_2 = (1, 1, 0, 1, 0, 1, 0, 0)$;
3) $d_3 = (1, 0, 1, 1, 0, 0, 1, 0)$;
4) $d_4 = (0, 1, 1, 1, 0, 0, 0, 1)$;
5) $d_1 + d_2 = (0, 0, 1, 1, 1, 1, 0, 0)$;
6) $d_1 + d_3 = (0, 1, 0, 1, 1, 0, 1, 0)$;

7) $d_1 + d_4 = (1, 0, 0, 1, 1, 0, 0, 1)$;
8) $d_2 + d_3 = (0, 1, 1, 0, 0, 1, 1, 0)$;
9) $d_2 + d_4 = (1, 0, 1, 0, 0, 1, 0, 1)$;
10) $d_3 + d_4 = (1, 1, 0, 0, 0, 0, 1, 1)$;
11) $d_1 + d_2 + d_3 = (1, 0, 0, 0, 1, 1, 1, 0)$;
12) $d_1 + d_2 + d_4 = (0, 1, 0, 0, 1, 1, 0, 1)$;
13) $d_1 + d_3 + d_4 = (0, 0, 1, 0, 1, 0, 1, 1)$;
14) $d_2 + d_3 + d_4 = (0, 0, 0, 1, 0, 1, 1, 1)$;
15) $d_1 + d_2 + d_3 + d_4 = (1, 1, 1, 1, 1, 1, 1, 1)$.

Напишем распределение по весам Хэмминга.

- Вес 0:
$$\vec{0} = (0, 0, 0, 0, 0, 0, 0, 0).$$
- Вес 8:
$$d_1 + d_2 + d_3 + d_4.$$
- Вес 4 имеют все остальные векторы кода $\overline{C}$.

Теперь становится понятным, почему расширение (удлинение) ещё называют *проверкой на чётность.*

Таким образом построили $[8, 4, 4]$-код.

П.2.4. Построение кода, ортогонального расширенному коду

Построим ортогональный к $\overline{C}$ код

$$\overline{C}^{\perp} = \left\{ v \in F_2^8 \mid (u, v) = 0 \ \forall u \in \overline{C} \right\}.$$

Заметим, что для построения $\overline{C}^{\perp}$ достаточно потребовать

$$(d_i, v) = 0 \text{ для всех } i \in \{1, 2, 3, 4\}.$$

В самом деле, для любого $v \in \overline{C}$ существуют такие $t_1, t_2, t_3, t_4 \in \mathbf{F}_2 = \{0, 1\}$, что

$$u = t_1 d_1 + t_2 d_2 + t_3 d_3 + t_4 d_4.$$

Тогда

$$\begin{aligned}(u, v) = (t_1 d_1 + t_2 d_2 + t_3 d_3 + t_4 d_4, v) = \\ = t_1(d_1, v) + t_2(d_2, v) + t_3(d_3, , v) + t_4(d_4, v) = 0 + 0 + 0 + 0 = 0.\end{aligned}$$

Пусть

$$v = (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8) \in \overline{C}^{\perp}.$$

Возникает система

$$\begin{cases} (d_1, v) = \big((1,1,1,0,1,0,0,0), (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)\big) = \\ \qquad = y_1 + y_2 + y_3 + y_5 = 0 \\ (d_2, v) = \big((1,1,0,1,0,1,0,0), (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)\big) = \\ \qquad = y_1 + y_2 + y_4 + y_6 = 0 \\ (d_3, v) = \big((1,0,1,1,0,0,1,0), (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)\big) = \\ \qquad = y_1 + y_3 + y_4 + y_7 = 0 \\ (d_4, v) = \big((0,1,1,1,0,0,0,1), (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)\big) = \\ \qquad = y_2 + y_3 + y_5 + y_8 = 0 \end{cases}.$$

Итак, получаем

$$\begin{cases} y_5 = y_1 + y_2 + y_3 \\ y_6 = y_1 + y_2 + y_4 \\ y_7 = y_1 + y_3 + y_4 \\ y_8 = y_2 + y_3 + y_4 \end{cases}.$$

Теперь опишем множество её решений

y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8
1	0	0	0	1	1	1	0
0	1	0	0	1	1	0	1
0	0	1	0	1	0	1	1
0	0	0	1	0	1	1	1

.

Получаем базисные решения

$$\begin{aligned} v_1 &= (1,0,0,0,1,1,1,0); \\ v_2 &= (0,1,0,0,1,1,0,1); \\ v_3 &= (0,0,1,0,1,0,1,1); \\ v_4 &= (0,0,0,1,0,1,1,1). \end{aligned}$$

Теперь наш код

$$\overline{C}^{\perp} = \{t_1 v_1 + t_2 v_2 + t_3 v_3 + t_4 v_4 \mid t_1, t_2, t_3, t_4 \in \mathbf{F}_2 = \{0,1\}\}.$$

Выпишем все элементы этого кода

0) $\vec{0} = (0,0,0,0,0,0,0,0)$;
1) $v_1 = (1,0,0,0,1,1,1,0)$;
2) $v_2 = (0,1,0,0,1,1,0,1)$;
3) $v_3 = (0,0,1,0,1,0,1,1)$;

4) $v_4 = (0,0,0,1,0,1,1,1)$;
5) $v_1 + v_2 = (1,1,0,0,0,0,1,1)$;
6) $v_1 + v_3 = (1,0,1,0,0,1,0,1)$;
7) $v_1 + v_4 = (1,0,0,1,1,0,0,1)$;
8) $v_2 + v_3 = (0,1,1,0,0,1,1,0)$;
9) $v_2 + v_4 = (0,1,0,1,1,0,1,0)$;
10) $v_3 + v_4 = (0,0,1,1,1,1,0,0)$;
11) $v_1 + v_2 + v_3 = (1,1,1,0,1,0,0,0)$;
12) $v_1 + v_2 + v_4 = (1,1,0,1,0,1,0,0)$;
13) $v_1 + v_3 + v_4 = (1,0,1,1,0,0,1,0)$;
14) $v_2 + v_3 + v_4 = (0,1,1,1,0,0,0,1)$;
15) $v_1 + v_2 + v_3 + v_4 = (1,1,1,1,1,1,1,1)$.

Напишем распределение по весам Хэмминга:

- Вес 0 имеет вектор:
$$\vec{0} = (0,0,0,0,0,0,0,0).$$
- Вес 8 имеет вектор:
$$v_1 + v_2 + v_3 + v_4.$$
- Вес 4 имеют все остальные вектор.

Получили код *Рида–Маллера* первого порядка с параметрами $[2^m = 8, m+1 = 4, 2^{m-1} = 4]$.

Теперь вскроем тайну!

0) $\vec{0} = (0,0,0,0,0,0,0,0)$;
1) $v_1 = (1,0,0,0,1,1,1,0) = d_1 + d_2 + d_3$;
2) $v_2 = (0,1,0,0,1,1,0,1) = d_1 + d_2 + d_4$;
3) $v_3 = (0,0,1,0,1,0,1,1) = d_1 + d_3 + d_4$;
4) $v_4 = (0,0,0,1,0,1,1,1) = d_2 + d_3 + d_4$;
5) $v_1 + v_2 = (1,1,0,0,0,0,1,1) = d_3 + d_4$;
6) $v_1 + v_3 = (1,0,1,0,0,1,0,1) = d_2 + d_4$;
7) $v_1 + v_4 = (1,0,0,1,1,0,0,1) = d_1 + d_4$;
8) $v_2 + v_3 = (0,1,1,0,0,1,1,0) = d_2 + d_3$;
9) $v_2 + v_4 = (0,1,0,1,1,0,1,0) = d_1 + d_3$;
10) $v_3 + v_4 = (0,0,1,1,1,1,0,0) = d_1 + d_2$;
11) $v_1 + v_2 + v_3 = (1,1,1,0,1,0,0,0) = d_1$;
12) $v_1 + v_2 + v_4 = (1,1,0,1,0,1,0,0) = d_2$;
13) $v_1 + v_3 + v_4 = (1,0,1,1,0,0,1,0) = d_3$;
14) $v_2 + v_3 + v_4 = (0,1,1,1,0,0,0,1) = d_4$;
15) $v_1 + v_2 + v_3 + v_4 = (1,1,1,1,1,1,1,1) = d_1 + d_2 + d_3 + d_4$.

Получили, что коды $\overline{C}$ и $\overline{C}^{\perp}$ совпадают. Такие коды называются *самодуальными (самодвойственными)*.

П3. Примеры RSA

Рассмотрим и обсудим две программы на GAP, которые показывают, как происходит шифрование в системе RSA.

П3.1. Простой пример

```
#Simple Example
gap> s1:=13*17;
221
gap> m1:=128;
128
gap> c1:=(m1^19) mod s1;
15
gap> t1:=12*16; # 12=13-1 and 16=17-1
192
gap> Gcdex(t1,19);
rec( gcd := 1, coeff1 := -9, coeff2 := 91, coeff3 := 19,
coeff4 := -192 )
gap> d1:=Gcdex(t1,19).coeff2 mod t1;
91
gap> c1^d1 mod s1;
128
```

Дадим пояснения к приведённой программе.

1. Прежде всего, значок `#` указывает, что далее в данной строке идёт комментарий, который при переходе на другую строку заканчивается.
2. `s1` — число, которое используется при шифровании как модуль сравнений. Оно является произведением двух простых чисел $p = 13$ и $q = 17$.
3. `m1` — сообщение.
4. `19` — открытый ключ.
5. `c1` — зашифрованное сообщение.
6. `t1` — число, которое нужно для поиска закрытого ключа. Оно равно произведению $(p-1)(q-1)$.
7. `Gcdext` — функция, которая возвращает запись представления наибольшего общего делителя чисел `t1` и `19` в следующем виде:

$$a \cdot t1 + b \cdot 19 = gcd,$$

где gcd — наибольший общий делитель `t1` и `19` и

$$1 = coeff1 \cdot t1 + coeff2 \cdot 19.$$

В самом деле

$$-9 \cdot 192 + 91 \cdot 19 = -1728 + 1729 = 1.$$

Компоненты `coeff3` и `coeff4` не потребуются.

8. `d1:=Gcdex(t1,19).coeff2 mod t1` — вызов компоненты `coeff2`, которая является закрытым ключом.
9. `c1^d1 mod s1` — расшифрование сообщения.

П3.2. Более сложный пример

```
#Our Example
gap> m:=1280321671742241 68;
1280321671742241 68
#Limit for first Prime
gap> a:=RootInt(m);
357815828
#First Prime
gap> p:=2^31-1;
2147483647
#Limit for second Prime
gap> b:=QuoInt(m,p);
59619623
#Base by GOST for second Prime
gap> q:=2^19-1;
524287
#Limit of multiple for second Prime
gap> d:=QuoInt(b,q);
113
#List
gap> l:=[];
[  ]
gap> for n in [57..100] do
> r:=2*n*q+1;
> s:=2^q mod r;
> if (s^(2*n)=1 mod r) and (2^(2*n)<>1 mod r) then
> Append(l,[n,r]); fi;
> od;
```

```
gap> l;
[ 60, 62914441 ]
#second Prime
gap> r:=62914441;
62914441
gap> s:=p*r;
135107733207646327
gap> t:=(p-1)*(r-1);
135107730997248240
#Open Key
gap> e:=2^17-1;
131071
#Encription
gap> c:=m^e mod s;
28761827334326630
#Find Secret Key
gap> Gcdex(t,e);
rec( gcd := 1, coeff1 := 20583, coeff2 := -21216916229496689,
 coeff3 := -131071, coeff4 := 135107730997248240 )
#Secret Key
gap> d:=Gcdex(t,e).coeff2 mod t;
113890814767751551
#Check
gap> d*e mod t;
1
gap> c:=PowerModInt(c,d,s);
128032167174224168
#True!
```

Для этой программы сделаем пояснения только в тех местах, которые отличаются от предыдущего примера. Для RSA нам нужны два простых числа. Сначала находим наибольшее целое, квадрат которого не превосходит `m`. Это будет `a`. Хотим взять первое простое число меньшее чем `a` и в качестве такого выбираем одно из простых чисел Мерсенна `2^{31}-1`.

Хотим найти такое второе простое число `r`, чтобы его произведение с первым было больше `m`. Находим целое число `b`, больше которого должно быть `r`. Это простое число находим по ГОСТ. Имеется в виду, что при генерации простых чисел для ЭЦП в ГОСТ Р 34.10-94 применяется тест простоты. Он изложен как теорема 12.13.2 в книге “Математические и компьютерные основы криптологии” [7].

Теорема. *Пусть q — нечётное простое число и $p = qN+1$, где N — чётное*

число. Пусть $r < (2q+1)^2$ *и выполнены два условия*

1) $2^{qN} \equiv 1 \pmod{r}$,
2) $2^N \not\equiv 1 \pmod{r}$.

Тогда r *— простое число.*

В нашем случае $q = 2^{19} - 1$ и $N = 2n$ для $n \in \{57, \ldots, 100\}$. То, что надо начинать поиск с 57, определяется тем, что должны найти число больше `b`. Такое число находится при $n = 60$, и оно равно `r:=62914441`.

Таким образом получаем число `s:=p*r`, равное 135107733207646327. В качестве открытого ключа выбираем число `e:=2^17-1`, равное 131071.

Шифруем с помощью e и получаем `28761827334326630`. Ищем закрытый (секретный) ключ как в первом примере и расшифровываем текст.

Задание.

Раскодируйте сообщение

128032167174224168.

Подсказка. Использована одна из популярных русских компьютерных кодировок.

П4. Большие простые числа

Этот раздел следует рассматривать как дополнение к первой теме о числах, так и к третьей теме о криптологии, поскольку большие простые числа часто служат основой для многочисленных криптоприложений.

П4.1. Список самых больших простых чисел

На сайте `http://primes.utm.edu/` можно найти 20 самых больших известных на данное время простых чисел. Кроме того, там есть ещё очень много всякой информации. Например, списки больших простых чисел–близнецов, т. е. простые числа различаются только на 2, простых чисел Софи Жермен, т. е. p и $2p+1$ — простые числа, и другие интересные факты.

Приведём список 10 самых больших известных на данное время простых чисел

№	простое	цифр	когда	ссылка
1	$2^{74207281}-1$	22338618	2016	Mersenne 49??
2	$2^{57885161}-1$	17425170	2013	Mersenne 48?
3	$2^{43112609}-1$	12978189	2008	Mersenne 47?
5	$2^{42643801}-1$	12837064	2009	Mersenne 46?
6	$2^{37156667}-1$	11185272	2008	Mersenne 45
5	$2^{32582657}-1$	9808358	2006	Mersenne 44
7	$10223\cdot 2^{31172165}+1$	9383761	2016	
8	$2^{30402457}-1$	9152052	2005	Mersenne 43
9	$2^{25964951}-1$	7816230	2005	Mersenne 42
10	$2^{24036583}-1$	7235733	2004	Mersenne 41

Для того, чтобы визуализировать размер самого большого числа в десятичных цифрах потребуется 5 957 страниц с 75 цифр в каждой строке и 50 строк на странице.

Mersenne — это числа Мерсенна, о которых более подробно будет идти речь далее. Знаки вопроса “?” обозначают, что пока не доказано, что данное число действительно имеет такой номер в списке чисел Мерсенна. Двойной знак вопроса “??” подчёркивает бóльшую неуверенность в этом.

П4.2. Таблица известных чисел Мерсенна[16]

Определение. Пусть p — простое число и $M(p)=2^p-1$. Если $M(p)$ — простое число, то оно называется ***простым числом Мерсенна.***

Изучая приведённую таблицу 10 самых больших простых чисел и более обширные таблицы больших простых чисел, легко заметить, что числа Мерсенна занимают в них особое место, в частности, их там всегда много. Это связано с тем, что для определения простоты чисел $M(p)$ есть эффективный алгоритм Люка, который приведём и обсудим позже.

Среди чисел вида 2^n-1 простыми могут быть только числа Мерсенна (т.е. n обязано быть простым). Действительно, если у n есть делитель $m>1$ и $n/m>1$, то:

$$2^n-1=(2^m-1)(1+2^m+2^{2m}+\cdots+2^{n-m}),$$

и поэтому 2^n-1 не является простым.

Список всех известных простых чисел p, для которых $M(p)$ — простое число Мерсенна можно найти на сайте:

`https://www.mersenne.org/primes/`

[16]Маре́н Мерсе́нн (устаревшая транслитерация Мари́н Мерсе́нн; фр. Marin Mersenne; 8 сентября 1588 — 1 сентября 1648) — французский математик, физик, философ и богослов, теоретик музыки.

№	p (показатель)	цифр в $M(p)$	год	открыватель
1	2	1	—	—
2	3	1	—	—
3	5	2	—	—
4	7	3	—	—
5	13	4	1456	anonymous
6	17	6	1588	Cataldi
7	19	6	1588	Cataldi
8	31	10	1772	Euler (Эйлер)
9	61	19	1883	Первушин (Pervushin)

Замечания. Сделаем два замечания, прежде чем продолжать таблицу.

1. Сначала, используя GAP, приведём примеры не простых чисел $M(p)$.

p	$M(p)$
11	$2^{11} - 1 = 2\,047 = 23 \cdot 89$
23	$2^{23} - 1 = 8\,388\,607 = 47 \cdot 178481$
29	$2^{29} - 1 = 536\,870\,911 = 233 \cdot 1103 \cdot 2089$
37	$2^{37} - 1 = 137\,438\,953\,471 = 223 \cdot 616318177$
41	$2^{41} - 1 = 2\,199\,023\,255\,551 = 13367 \cdot 164511353$
43	$2^{43} - 1 = 8\,796\,093\,022\,207 = 431 \cdot 9719 \cdot 2099863$
47	$2^{47} - 1 = 140\,737\,488\,355\,327 = 2351 \cdot 4513 \cdot 13264529$
53	$2^{53} - 1 = 9\,007\,199\,254\,740\,991 = 6361 \cdot 69431 \cdot 20394401$
59	$2^{59} - 1 = 576\,460\,752\,303\,423\,487 = 179951 \cdot 3203431780337$

2. Девятое простое число Мерсенна $2^{61} - 1$ открыл Первушин. Это очень любопытная фигура в математике. Приведём выдержки из статьи о Первушине в Википедии.

> Иван Михеевич (Иоанн Михеев) Первушин (21 января (2 февраля) 1821, село Архангело–Пашийский завод, Пермский уезд, Пермская губерния —17 (30) июня 1900, село Мехонское, Мехонская волость, Шадринский уезд, Пермская губерния, Российская империя) — российский священник и математик, специалист в области теории чисел.
>
> В 1877 году и в начале 1878 года он представил в Академию Наук работы, в которых доказывалось, что двенадцатое и двадцать третье числа Ферма[17] не являются простыми. А именно он показал, что число $2^{2^{12}}+1$ делится на простое число $7 \cdot 2^{14} + 1 = 114\,689$, а число $2^{2^{23}} + 1$ делится на простое число $5 \cdot 2^{25} + 1 = 167\,772\,161$, предварив французского математика Эдуарда Люка(са), занимавшегося теми же изысканиями.

[17] О числах Ферма будет материал позже.

Академия Наук, в поощрение трудов Первушина, исхлопотала у Святейшего Синода высылки ему математических книг на 190 рублей.

В 1883 году, он доказал, что число

$$2^{61} - 1 = 2\ 305\ 843\ 009\ 213\ 693\ 951$$

(2 квинтиллиона 305 квадриллионов 843 триллиона 009 миллиардов 213 миллионов 693 тысячи 951 единица) является простым, то есть числом Мерсенна. Это число получило название в его честь — число Первушина. Это второе по величине простое число, известное на то время (после числа $2^{127}-1$, найденного Эдуардом Люка́ в 1878 году). Представил на Математический конгресс в Чикаго заметку «О наилучшей проверке арифметических действий над огромными числами при посредстве делений». Опубликовано в трудах 1-го Международного математического конгресса в Цюрихе.

За свои достижения в математике был избран членом–корреспондентом Петербургской, Неаполитанской и Парижской академий наук.

Первушин — член Московского (с 1891), Казанского (с 1889) ученых математических обществ, Уральского общества любителей естествознания (с 1880).

Известна книга

Тимофеев В.П. Священник–математик Иван Михеевич Первушин. — Шадринск, 1996.

Эта книга написана ныне покойным доцентом Челябинского университета Вячеславом Павлиновичем Тимофеевым, который долго жил в Шадринске и был его патриотом. Она была любезно подарена мне автором.

Теперь мы перейдём к продолжению таблицы известных чисел Мерсенна.

Таблица известных простых чисел Мерсенна (продолжение)

№	p (показатель)	цифр в $M(p)$	год	открыватель
10	89	27	1911	Powers
11	107	33	1914	Powers
12	127	39	1876	Lucas
13	521	157	1952	Robinson
14	607	183	1952	Robinson

№	p (показатель)	цифр в $M(p)$	год	открыватель
15	1279	386	1952	Robinson
16	2203	664	1952	Robinson
17	2281	687	1952	Robinson
18	3217	969	1957	Riesel
19	4253	1281	1961	Hurwitz
20	4423	1332	1961	Hurwitz
21	9689	2917	1963	Gillies
22	9941	2993	1963	Gillies
23	11213	3376	1963	Gillies
24	19937	6002	1971	Tuckerman
25	21701	6533	1978	Noll & Nickel
26	23209	6987	1979	Noll
27	44497	13395	1979	Nelson & Slowinski
28	86243	25962	1982	Slowinski
29	110503	33265	1988	Colquitt & Welsh
30	132049	39751	1983	Slowinski
31	216091	65050	1985	Slowinsk
32	756839	227832	1992	Slowinski & Gage
33	859433	258716	1994	Slowinski & Gage
34	1257787	378632	1996	Slowinski & Gage
35	1398269	420921	1996	GIMPS/ Armengaud
36	2976221	895932	1997	GIMPS/ Spence
37	3021377	909526	1998	GIMPS/ Clarkson
38	6972593	2098960	1999	GIMPS/ Hajratwala
39	13466917	4053946	2001	GIMPS/ Cameron
40	20996011	6320430	2003	GIMPS/ Shafer
41	24036583	7235733	2004	GIMPS/Findley
42	25964951	7816230	2005	GIMPS/ Nowak
43	30402457	9152052	2005	GIMPS/ Cooper & Boone
44	32582657	9808358	2006	GIMPS/ Cooper & Boone
45	37156667	11185272	2008	GIMPS/ Elvenich
46?	42643801	12837064	2009	GIMPS/ Strindmo
47?	43112609	12978189	2008	GIMPS/ Smith
48?	57885161	17425170	2013	GIMPS/ Cooper
49??	74207281	22338618	7 Jan 2016	GIMPS/ Cooper

Замечания. Приведём несколько замечаний к этой таблице.

1. Не все кандидаты между $M(37\,156\,667)$ и $M(74\,207\,281)$ были проверены, поэтому это предварительный рейтинг.
2. После того, как 23-е простое число Мерсенна было найдено в University

of Illinois, так отделение математики этого университета было настолько гордо этим, что глава отделения Dr. Bateman изменил почтовый штемпель отделения на

$$2^{11213} - 1 \text{ is prime}$$

на каждом почтовом конверте[18].

Этот штемпель использовался до доказательства в 1976 году теоремы о четырёх красках (the four color theorem).

3. 25-е и 26-е простые числа Мерсенна были найдены студентами Laura Nickel и Landon Curt Noll, которые, несмотря на то, что мало разбирались в математике, использовали тест простоты Люка (Lucas' simple test, об этом тесте чуть позже) на локальном университетском мейнфрейме (CSUH's CDC 174). Открытие ими первого из этих простых чисел было отмечено в новостях национального телевидения и на первой странице New York times.
4. George Woltman — прекрасный программист и организатор, начиная с конца 1995 года, собирал разные базы данных по простым числам Мерсенна и скомбинировал их в одну. Затем он поместил в интернет эту базу данных и свободную высокооптимизированную программу для поиска простых чисел Мерсенна. Это положило начало

   GIMPS (the Great Internet Mersenne Prime Search).

5. За нахождение простого числа Мерсенна $M(43112609)$ проектом GIMPS в 2009 году была получена премия в 100 000 долларов США, назначенная сообществом Electronic Frontier Foundation за нахождение простого числа, десятичная запись которого содержит не менее 10 миллионов цифр.

Пропуски в последовательности простых чисел Мерсенна

Заполненные. Проверено, что простых чисел Мерсенна $2^p - 1$ при

$$p \in \{2, \ldots, 34\,969\,871\} \text{ всего } 44,$$

но простых чисел во множестве

$$\{2, \ldots, 34\,969\,871\} \text{ будет } 2\,145\,061.$$

Незаполненные. Теперь рассмотрим оставшиеся промежутки между последовательными простыми числами Мерсенна в приведённой ранее таблице.

[18]Здесь я хотел вставить фотографию этого штемпеля, но убоялся нарушения авторских прав.

1. Простых чисел во множестве $\{34969871, \ldots, 37156667\}$ будет 125660.
2. Простых чисел во множестве $\{37156667, \ldots, 42643801\}$ будет 313609.
3. Простых чисел во множестве $\{42643801, \ldots, 43112609\}$ будет 26617.
4. Простых чисел во множестве $\{43112609, \ldots, 57885161\}$ будет 833015.
5. Простых чисел во множестве $\{57885161, \ldots, 74207281\}$ будет 906644.

Критерий Люка

На критерии Люка основаны все проверки простоты чисел Мерсенна. Его опубликовал Э. Люка (Lukas) в 1878 году, а в 1930 году усовершенствовал Д.Х. Лемер. Он опирается на следующую теорему.

Теорема (Критерий Люка)**.** *Пусть p — простое число, причем $p \geqslant 3$. Определим последовательность L_n следующим образом:*

$$L_1 = 4, \quad L_{n+1} = L_n^2 - 2.$$

Тогда $M(p) = 2^p - 1$ — простое тогда и только тогда, когда

$$L_{p-1} \equiv 0 \pmod{M(p)},$$

что равносильно тому, что $M(p)$ делит L_{p-1}.

Примеры

1. Вычислим 10 начальных членов последовательности $\{L_n\}_{n=0}^{\infty}$:

```
gap> l:=[];
[  ]
gap> l[1]:=4;
4
gap> for i in [2..10] do
> l[i]:=l[i-1]^2-2;
> od;
gap> l;
[ 4, 14, 194, 37634, 1416317954, 2005956546822746114,
  4023861667741036022825635656102100994,
  16191462721115671781777559070120513661\
  49585901254991585143293087409757880341,
  26216346504927851452605936955756303921\
  36478775595245459119060053495557738312369350 1\
  59562818489334269993079824186649432 7\
  69439016089193966072975851 54,
```

```
687296824066442772388374862317475309\
2424715410864667175219261858308848740579095796 4\
732883069102561043436779663935595172042357306 59\
491634460607456471286807828760805520\
302465835943901758088391097866618587\
571741554108449492650047516738 11\
685059273781818997538392606094522 6\
536527485090187988120371 4 ]
```

2. Пример 1 показывает, что последовательность Люка очень быстро растёт, поэтому её непосредственное применение крайне затруднительно. Следовательно для данного простого p нужно сразу рассматривать эту последовательность по модулю $M(p) = 2^p - 1$.
Теперь рассмотрим $p = 19$.

```
gap> l:=[];
[  ]
# Compute M(19)
gap> m19:=2^19-1;
524287
gap> l[1]:=4;
4
gap> for i in [2..20] do
> l[i]:=(l[i-1]2-2) mod m19;
> od;
gap> l;
[ 4, 14, 194, 37634, 218767, 510066, 386344, 323156, 218526,
  504140, 103469, 417706, 307417, 382989, 275842, 85226,
  523263, 0, 524285, 2 ]
```

Итак, вычислили 20 членов последовательности Люка по модулю $2^{19} - 1 = 52487$, и оказалось, что $19 - 1 = 18$-й член сравним с 0 по модулю $2^{19} - 1 = 52487$. Согласно критерию Люка

$$M(19) = 2^{19} - 1 = 52487 \text{ — простое число Мерсенна.}$$

3. Применим критерий Люка для $p = 23$ и $M(23) = 2^{23} - 1 = 8388607$.

```
gap> l:=[];
[  ]
gap> l[1]:=4;
4
gap> m23:=2^23-1;
```

```
8388607
gap> for i in [2..25] do
> l[i]:=(l[i-1]\^{}2-2) mod m23;
> od;
gap> l;
[ 4, 14, 194, 37634, 7031978, 7033660, 1176429, 7643358,
  3179743, 2694768, 763525, 4182158, 7004001, 1531454,
  5888805, 1140622, 4321431, 7041324, 2756392, 1280050,
  6563009, 6107895, 6243954, 4552058, 5402421 ]
```

Итак, вычислили 25 членов последовательности Люка по модулю $2^{23} - 1 = 8388607$, и оказалось, что $23 - 1 = 22$-й член 6107895 не сравним с 0 по модулю $2^{23} - 1 = 8388607$. Согласно критерию Люка

$$M(23) = 2^{23} - 1 = 8388607 \text{ не является простым числом.}$$

П4.3. Числа Ферма

Числами Ферма называются числа вида $F_n = 2^{2^n} + 1$, где n — неотрицательное целое число.

Числа Ферма для $n = 0, 1, 2, 3, \ldots$ образуют последовательность:

$$F_n : 3, 5, 17, 257, 65537, 4294967297, 18446744073709551617, \ldots$$

(последовательность A000215 в OEIS)

Изучение чисел вида 2^{2^n} начал Ферма, который выдвинул гипотезу, что все они простые.

Замечание. Из интернета:

> Ферма писал Мерсенну 25 декабря 1640 года:
>
> *Если бы я смог определить базисную причину, почему*
>
> $$3, 5, 17, 257, 65537, \ldots,$$
>
> *— простые числа, я чувствую, что я нашёл бы очень интересные результаты, ибо я уже нашёл чудесные вещи [в этом направлении], о которых я сообщу позднее.*

Однако, гипотеза Ферма была опровергнута Эйлером в 1732 году, нашедшим разложение числа F_5 на простые делители:

$$F_5 = 4294967297 = 641 \cdot 6700417.$$

Нужно провести только два пробных деления, чтобы найти множитель 641, потому что Эйлер показал, что любой делитель числа Ферма F_n с $n > 2$ имеет вид $k \cdot 2^{n+2} + 1$. В случае F_5 — это будут $128k + 1$, поэтому мы должны были испробовать 257 и 641 (129, 385 и 513 не просты).

Важность знания чисел Ферма показывает следующий результат.

Теорема Гаусса–Ванцеля

Правильный n-угольник можно построить с помощью циркуля и линейки тогда и только тогда, когда

$$n = 2^r \cdot p_1 \cdot p_2 \cdots p_k,$$

где p_i — различные простые числа Ферма.

Среди чисел вида $2^n + 1$ простыми могут быть только числа Ферма, т.е. n обязано быть степенью 2. Действительно, если у n есть нечётный делитель $m > 1$ и $n/m > 1$, то:

$$2^n + 1 = (2^m + 1)(1 - 2^m + 2^{2m} - \cdots + 2^{n-m}),$$

и поэтому $2^n + 1$ не является простым.

Сейчас известно лишь 5 простых чисел Ферма:

$$3, 5, 17, 257, 65537 \qquad \text{(последовательность A019434 в OEIS.)}$$

Существование других простых чисел Ферма является открытой проблемой. Простая эвристика показывает, что возможно это единственные простые числа Ферма, хотя многие люди, такие как Эйзенштейн [19] думали иначе. На с. 110 отмечалось, что Первушин доказал, что числа Ферма $2^{2^{12}} + 1$ и $2^{2^{23}} + 1$ не просты. Известно, что F_n являются составными при $5 \leqslant n \leqslant 32$.

Числа Ферма попарно взаимно просты, как можно увидеть из следующего тождества:

$$F_0 F_1 F_2 ..... F_{n-1} + 2 = F_n.$$

Также это равенство даёт простое доказательство теоремы Евклида о бесконечности простых чисел.

Десятичная запись чисел Ферма, больших 5, оканчивается на 7.

Простоту чисел Ферма можно эффективно установить с помощью теста Пепина. Тест назван в честь французского математика Феофила Пепина. Тест основывается на следующей теореме.

Теорема. *При $n \geqslant 1$ число Ферма $F_n = 2^{2^n} + 1$ является простым тогда и только тогда, когда*

$$3^{(F_n - 1)/2} \equiv -1 \pmod{F_n}.$$

[19] Фердина́нд Го́ттхольд Макс Э́йзенштейн (нем. Ferdinand Gotthold Max Eisenstein; 16 апреля 1823, Берлин — 11 октября 1852, Берлин) — немецкий математик.

Число 3 в тесте Пепина может быть заменено на 5, 6, 7 или 10 (последовательность A129802 в OEIS), которые также являются первообразными корнями по модулю каждого простого числа Ферма. Известно, что Пепин привел критерий с числом 5 вместо числа 3. Прот и Люка отметили, что можно также использовать число 3. Так как тест Пепина требует $2^n - 1$ возведений в квадрат по модулю F_n, то он выполняется за полиномиальное время от длины числа F_n. Однако, если на вход подается лишь число n, то тест Пепина выполняется за сверхэкспоненциальное время от длины входа $\log n$. Из-за большого размера чисел Ферма, тест Пепина был использован всего несколько раз на числах Ферма, чья простота ещё не была доказана или опровергнута. Майер, Пападопулос и Крэндалл выдвинули предположение о том, что в действительности, должно пройти несколько десятилетий, прежде чем технологии позволят выполнить новые тесты Пепина, поскольку размеры еще неисследованных чисел Ферма очень велики. На март 2017 известны 336 простых делителей чисел Ферма и известно, что 292 числа Ферма составные. Ежегодно находится несколько новых делителей чисел Ферма.

Год	Исследователи	Числа Ферма	Результаты теста Пепина	Делитель найден позже
1905	Джеймс С. Морхед и Альфред Вестерн	F_7	составное	Да (1970)
1909	Джеймс С. Морхед и Альфред Вестерн	F_8	составное	Да (1980)
1952	Рафаэль М. Робинсон	F_{10}	составное	Да (1953)
1960	Г. А. Паксон	F_{13}	составное	Да (1974)
1961	Джон Селфридж и Александр Гурвиц	F_{14}	составное	Да (2010)
1987	Дункан Бьюэл и Джеффри Янг (1988)	F_{20}	составное	Нет
1993	Ричард Крэндалл, Дж. Диньяс, С. Норри и Джеффри Янг (1995)	F_{22}	составное	Да (2010)
1999	Эрнст В. Майер, Джейсон С. Пападопулос и Ричард Крэндалл	F_{24}	составное	Нет

Приведём разложение чисел Ферма на простые множители:

$$F_0 = 2^{2^0} + 1 = 2^1 + 1 = 3,$$
$$F_1 = 2^{2^1} + 1 = 2^2 + 1 = 5,$$

$$
\begin{aligned}
F_2 &= 2^{2^2}+1 = 2^4+1 = 17,\\
F_3 &= 2^{2^3}+1 = 2^8+1 = 257,\\
F_4 &= 2^{2^4}+1 = 2^{16}+1 = 65537,\\
F_5 &= 2^{2^5}+1 = 2^{32}+1 = 4294967297 =\\
&= (5\cdot 2^{5+2}+1)\cdot(52347\cdot 2^{5+2}+1) = 641\cdot 6700417,\\
F_6 &= 2^{2^6}+1 = 2^{64}+1 = 18446744073709551617 =\\
&= (1071\cdot 2^{6+2}+1)\cdot(262814145745\cdot 2^{6+2}+1) =\\
&= 274177\cdot 67280421310721,\\
F_7 &= 2^{2^7}+1 = 2^{128}+1 =\\
&= 340282366920938463463374607431768211457 =\\
&= (116\,503\,103\,764\,643\cdot 2^{7+2}+1)\times\\
&\quad\times(11\,141\,971\,095\,088\,142\,685\cdot 2^{7+2}+1) =\\
&= 59\,649\,589\,127\,497\,217\cdot 5\,704\,689\,200\,685\,129\,054\,721,\\
F_8 &= 2^{2^8}+1 = 2^{256}+1 =\\
&= 115792089237316195423570985008687907853269984665640\backslash\\
&\quad\backslash 564039457584007913129639937 =\\
&= (3853149761\cdot 157\cdot 2^{8+3}+1)\times\\
&\times(10573720467811625362740343546868933296253 29\times\\
&\quad\times 31618624099079\cdot 13\cdot 7\cdot 5\cdot 3\cdot 2^{8+3}+1) =\\
&= 1238926361552897\times\\
&\times 934616397153579777691635581996068965840512375416381\backslash\\
&\quad\backslash 88580280321,\\
F_9 &= 2^{2^9}+1 = 2^{512}+1 =\\
&= 340780792994259709957402499820584612747936582059239 33\backslash\\
&\quad\backslash 77723561443721764030073546976801874298166903427690031\backslash\\
&\quad\backslash 85818648605085375388281194656994643364900608 4097 =\\
&= (37\cdot 2^{9+7}+1)\cdot(43226490359557706629\cdot 1143290228161321\times\\
&\quad\times 82488781\cdot 47\cdot 19\cdot 2^{9+2}+1)\times\\
&\quad\times(16975143302271505426897585653131126520182328037821\backslash\\
&\quad\backslash 729720833840187223\cdot 17338437577121\cdot 40644377\cdot 26813\times\\
&\quad\times 1129\cdot 2^{9+2}+1) =\\
&= 2424833\times\\
&\quad\times 7455602825647884208337395736200454918783366342657\times
\end{aligned}
$$

$$\times 7416400626275308015247871419019374740599407810975190\backslash$$
$$\backslash 2390582131614441575950470500809281871169394073 7.$$

Пример в GAP

```
# Задание F_6
gap> f6:=2^64+1;
18446744073709551617
# Тест Пеппина
gap> a:=(f6-1)/2;
9223372036854775808
gap> b:=PowerModInt(3,a,f6);
11860219800640380469
#не сравнимо с -1 по модулю F_6, значит F_6 --- не простое число
# Факторизация $F_6$
gap> c:=FactorsInt(f6);
[ 274177, 67280421310721 ]
gap> c1:=c[1];
274177
gap> c2:=c[2];
67280421310721
gap> d1:=(c1-1)/256;
1071
gap> FactorsInt(1071);
[ 3, 3, 7, 17 ]
gap> d2:=(c2-1)/256;
262814145745
gap> FactorsInt(d2);
[ 5, 47, 373, 2998279 ]
```

Получаем

$$\begin{aligned} F_6 &= 2^{2^6}+1 = 2^{64}+1 = 18446744073709551617 = \\ &= 274177 \cdot 67280421310721 = \\ &= (1071 \cdot 2^{6+2}+1)\cdot(262814145745 \cdot 2^{6+2}+1) = \\ &= (3^2 \cdot 7 \cdot 17 \cdot 2^{6+2}+1)\cdot(5 \cdot 47 \cdot 373 \cdot 2998279 \cdot 2^{6+2}+1). \end{aligned}$$

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. **Бауэр, Ф.Л.** Информатика: Вводный курс /Ф. Л. Бауэр, Г. Гооз. — М. : Мир, 1976. — 488 с.
2. **Белов, Е.Б.** Основы информационной безопасности: Учебное пособие. /Е. Б. Белов, В. П. Лось, Р. В. Мещеряков. — М. : Горячая линия — Телеком, 2006. — 544 с.
3. **Виноградов, И.М.** Основы теории чисел: Учебное пособие. 12-е изд., стер. /И. М. Виноградов.— СПб. : Издательство «Лань», 2009. — 176 с.
4. **Глухов, М.М.** Введение в теоретико-числовые методы криптографии: Учебное пособие. /М. М. Глухов, И. А. Круглов, А. Б. Пичкур, А.В. Черемушкин. — СПб. : Издательство «Лань», 2011. — 400 с.
5. **Девянин, П.Н.** Теоретические основы компьютерной безопасности: Учеб. пособие для вузов / П.Н. Девянин, О.О. Михальский, Д.И. Правиков, А.Ю. Щербаков.— М.: Радио и связь, 2000. — 192 с.: ил
6. **Лидл, Р.** Прикладная абстрактная алгебра: Учебное пособие. /Р. Лидл, Г. Пильц. — Екатеринбург. : Изд-во Урал. ун-та, 1996. — xx+744 с.
7. **Харин, Ю.С.** Математические и компьютерные основы криптологии: Учебное пособие. /Ю. С. Харин, В. И. Берник, С. В. Агиевич. — Минск. : Новое знание, 2003. — 382 с.
8. **Щербаков, А.Ю.** Введение в теорию и практику компьютерной безопасности. /А. Ю. Щербаков. — М. : издатель Молгачева С.В., 2001. — 352 с.
9. **The GAP Group, GAP - Groups, Algorithms, and Programming,** Version 4.7.7; 2015 (`http://www.gap-system.org`).