Оглавление

Предисловие к оригинальному изданию

Замысел «Кода» я вынашивал лет десять. И тогда, и во время работы над рукописью, и даже когда книга вышла из типографии многие спрашивали: «О чем она?»

Я всегда отвечал уклончиво, бормотал что-нибудь в духе: «Необычная экскурсия по истории цифровых технологий, сформировавших современную эпоху» — в надежде, что этого будет достаточно. Но в какой-то момент мне пришлось признать: «Код» — это книга о том, как устроены компьютеры.

Как я и опасался, отклики были неблагожелательными. На возражение в духе: «А-а, у меня уже есть такая книга» — я немедленно парировал: «Отнюдь, такой — нет». И по-прежнему так считаю. «Код» не похож на прочие книги «о компьютерах». В нем нет больших цветных иллюстраций с дисководами, где стрелками показано, как данные поступают в компьютер, нет рисунков, где паровозик в товарных вагончиках везет нули и единички. Метафоры и сравнения чудесны в своей буквальности, но они ни на что не годны, лишь затмевают красоту технологий.

Мне говорили: «А кому интересно, как работают компьютеры?» Верное замечание. Мне, например, нравится вникать в устройство приборов, но я хочу сам решать, когда это делать. Так, описать, как работает мой холодильник, я смогу лишь под пыткой.

Однако окружающие часто задают вопросы, свидетельствующие об их интересе к внутреннему устройству компьютера. Типичный пример: «Чем отличается оперативная память от дисковой?» Естественно, это важный вопрос. Такие понятия составляют основу маркетинга ПК. Даже начинающему пользователю требуется знать, сколько *мегов* одного и *гигов* другого потребуется для конкретного приложения. Кроме того, новичок должен представлять, что такое файл, как он загружается с диска в память, а затем сохраняется там.

На вопрос о дисковой и оперативной памяти принято отвечать: «Память похожа на столешницу, а диск — на ящики стола». В принципе неплохой ответ, но мне он кажется неудовлетворительным. Создается впечатление, будто архитектура компьютера разрабатывалась по образу и подобию бюро. На самом деле

разница между оперативной и дисковой памятью — искусственная и обусловлена отсутствием единого энергонезависимого и при этом быстро работающего носителя. Так называемая архитектура фон Неймана, доминирующая в компьютерной индустрии уже более 50 лет, возникла в результате этого технического изъяна. Когда меня спрашивали, как запускать программы для Macintosh под Windows, я впадал в ступор, осознавая, что для ответа придется затронуть массу технических тонкостей, которые собеседник явно сразу не поймет.

Хочу, чтобы с помощью «Кода» вы научились разбираться во всех этих вещах настолько, чтобы смогли потягаться с электротехниками и программистами. Надеюсь, вы оцените, каким достижением является компьютер среди технологий XX века, и прочувствуете его красоту саму по себе, без метафор и сравнений.

По сути, компьютеры иерархичны: на самом нижнем уровне располагаются транзисторы, а венчает все информация, которая выводится на монитор. В книге мы будем придерживаться этой иерархии. В принципе, книга и структурирована от уровня к уровню. И этот путь не столь сложен, как может показаться. Да, в современном компьютере происходит масса всякой всячины, но это самые обычные и простые операции.

Хотя в настоящее время компьютеры сложнее, чем четверть или полвека назад, они не изменились фундаментально. Вот почему изучать историю техники так здорово: чем сильнее углубляешься в прошлое, тем проще становятся технологии. Именно поэтому легко добраться до точки, где понятно решительно все.

В книге «Код» я заглянул настолько далеко в прошлое, насколько смог. Сам поразился, что удалось добраться до XIX века и на примере первых телеграфных машин объяснить устройство компьютера. Теоретически все, о чем говорится в первых 17 главах, легко собирается из простейших электрических компонентов, которые в ходу уже более века.

Думаю, благодаря всей этой винтажной технике при чтении вы испытаете некоторую ностальгию. Книгу «Код» невозможно было бы озаглавить «Еще быстрее, еще технологичнее» или «Сверхскоростной бизнес на цифровых нейронах»: определение бита дается лишь на 79-й странице, байта — на 199-й. Транзисторы впервые упоминаются на 156-й странице, и то вскользь.

Итак, пусть «Код» и весьма основательно объясняет устройство компьютера (найдется немного других книг, где описано, например, как именно работает процессор), стиль книги вполне развлекательный. Несмотря на глубину темы, я старался устроить читателю максимально комфортную прогулку. Без всяких вагончиков с нулями и единицами.

Чарльз Петцольд
16 августа 2000 года

[Код]

3a. Система сигналов для представления букв и цифр при передаче сообщений.

b. Система символов, букв или слов, которым присваиваются некоторые произвольно подобранные значения. Используется для передачи сообщений в случаях, когда требуется добиться конфиденциальности или краткости.

4. Система символов, применяемая для представления компьютерных команд…

Словарь английского языка американского наследия

Глава 1
Лучшие друзья

Вам десять лет. Ваш лучший друг живет на другой стороне улицы, напротив. Даже окна ваших спален обращены друг к другу. Каждый вечер родители объявляют отбой в безбожно ранний час, а вам еще хочется пообщаться, поделиться мыслями, наблюдениями, секретами, сплетнями, шутками и мечтами. Никто не вправе вас за это упрекнуть. В конце концов, стремление к коммуникации — одно из наиболее характерных человеческих качеств.

Пока в спальнях горит свет, можно помахать друг другу из окон и, полагаясь на примитивный язык тела, жестикулируя, обменяться парой мыслей. Однако передавать таким образом сложную информацию вряд ли удастся. И как только родители скомандуют: «Погаси свет!» — ситуация кажется безнадежной.

Как общаться? Может, по телефону? А был ли у вас в комнате телефон, когда вам было десять? Даже если так, то, где бы он ни находился, вас подслушают. Если ваш домашний компьютер подключен к телефонной линии, возможно, через него удастся поболтать бесшумно, но, опять же, компьютера в комнате нет.

Однако у вас с другом есть карманные фонарики. Все знают, что такой фонарик изобрели специально для чтения книжки под одеялом, но он отлично подходит для ночной коммуникации. Такая связь практически бесшумна, а луч света бьет прицельно, и, пожалуй, его не заметишь в щель под дверью. Бдительные домашние ничего не заподозрят.

Можно ли общаться при помощи вспышек? Попробовать точно стоит. В первом классе вы учились писать на бумаге слова и буквы, поэтому кажется уместным экстраполировать эти знания на обмен сигналами. Просто встаньте у окна и попытайтесь рисовать буквы светом. Чтобы написать *O*, включите фонарик, опишите им круг в воздухе, а потом выключайте. *I* — это вертикальная палочка. Но, как вы вскоре убедитесь, этот метод просто не работает. Наблюдая за фонариком друга, которым тот выводит в воздухе буквы, вы поймете, как сложно мысленно скомпоновать эти штрихи во что-то цельное. Завитушки и мазки света не слишком *точны*.

Наверняка вы видели в фильмах, как два морехода сигнализировали друг другу над водой мерцающими фонариками, один шпион покачивал зеркальцем, направляя свет сообщнику в другую комнату. Кажется, вот решение.

Сначала разработаем простой метод. Каждая буква алфавита соответствует последовательности бликов. Таким образом, один блик будет означать *А*, два — *Б*, три — *В* и т. д. Для *Я* уже понадобятся 33 блика. Слово «ГДЕ» — 4 блика + 5 бликов + 6 бликов, которые передаются с небольшими паузами, чтобы не перепутать эту серию с 15 бликами, то есть с *Н*. Паузы-пробелы между словами должны быть чуть длиннее.

Но вот что скажу: махать фонариком в воздухе больше не понадобится. Достаточно направить его куда нужно и нажимать на кнопочку. Но здесь возникает другая проблема: одно из первых сообщений, которое вы решите отправить («Как дела?»), растянется на 44 вспышки! Более того, придется забыть о пунктуации, ведь неизвестно, сколько бликов соответствуют вопросительному знаку.

Однако вы уже у цели. Вы предполагаете, что кто-то уже сталкивался с такой проблемой. Рано утром вы отправляетесь в библиотеку на поиски и узнаёте о чудесном изобретении под названием «азбука Морзе». *Вот* то, что нужно, пусть даже теперь придется переучиваться, как пишутся все буквы алфавита.

В чем разница: в изобретенной вами системе каждой букве алфавита соответствует определенное количество бликов, от 1 для *А* до 33 для *Я*. В азбуке Морзе два вида бликов: краткие и долгие. Разумеется, при этом код Морзе получается сложнее, но на практике оказывается гораздо эффективнее. Теперь словосочетание «Как дела?» состоит всего из 24 бликов, а не из 44, причем *с учетом* кода вопросительного знака.

Обсуждая принцип работы азбуки Морзе, принято говорить не о долгих и кратких бликах, а о точках и тире, поскольку при помощи этих знаков удобно изображать код на печатной странице. В азбуке Морзе каждой букве алфавита соответствует краткая серия точек и тире, показанная в таблице на следующей странице.

Пусть азбука Морзе и не связана с компьютером, она помогает познать суть кода, а это важная предпосылка для глубокого понимания тайных языков и внутреннего устройства компьютерного харда и софта.

В этой книге слово «*код*» обычно означает систему передачи информации между людьми и машинами. Иными словами, код обеспечивает коммуникацию. Иногда покажется, что код — это шифр, но большинство кодов таковыми не являются, хотя и должны быть понятными, поскольку лежат в основе человеческого общения.

В начале романа «Сто лет одиночества» Габриэль Гарсия Маркес вспоминает времена, когда «мир был еще таким новым, что многие вещи не имели

названия, и на них приходилось показывать пальцем». Создается впечатление, что лексемы присваиваются понятиям совершенно произвольно. Сложно понять, почему собаку называют собакой, а кошку — кошкой. Можно сказать, что словарь — это своеобразный код.

A	• —	R	• — •	А	• —	Р	• — •
B	— • • •	S	• • •	Б	— • • •	С	• • •
C	— • — •	T	—	В	• — —	Т	—
D	— • •	U	• • —	Г	— — •	У	• • —
E	•	V	• • • —	Д	— • •	Ф	• • — •
F	• • — •	W	• — —	Е	•	Х	• • • •
G	— — •	X	— • • —	Ё	•	Ц	— • — •
H	• • • •	Y	— • — —	Ж	• • • —	Ч	— — — •
I	• •	Z	— — • •	З	— — • •	Ш	— — — —
J	• — — —			И	• •	Щ	— — • —
K	— • —			Й	• — — —	Ъ	— — • — —
L	• — • •			К	— • —	Ы	— • — —
M	— —			Л	• — • •	Ь	— • • —
N	— •			М	— —	Э	• • — • •
O	— — —			Н	— •	Ю	• • — —
P	• — — •			О	— — —	Я	• — • —
Q	— — • —			П	• — — •		

Звуки, которые мы произносим и складываем в слова, — код, понятный любому, кто слышит наш голос и понимает язык, на котором мы говорим. Этот код называется говорением, или речью. Существуют и другие коды для записи слов на бумаге (камне, дереве, в воздухе, например когда самолет выводит рекламные надписи в небе). Такой код — это и рукописные и печатные символы, которые мы видим в книгах, журналах или газетах. Мы называем его письменной речью, текстом. Во многих языках речь и текст согласуются друг с другом. Например, в английском буквы и буквосочетания (в большей или меньшей степени) соответствуют произносимым звукам.

Для глухих или немых был разработан иной код, облегчающий межличностное общение, — язык жестов, состоящий из движений рук, передающих отдельные буквы, слова или целые концепции. Для слепых письменный текст заменяется азбукой Брайля — системой выпуклых точек, соответствующих буквам, буквосочетаниям или целым словам. Когда приходится быстро записывать речь, удобно пользоваться стенографией или сокращениями.

При общении мы пользуемся различными кодами, поскольку одна кодировка удобнее других. Например, устную речь невозможно хранить на бумаге,

и ее заменяет письмо. Тихо передавать информацию на расстоянии невозможно ни при помощи речи, ни на бумаге. Удобная альтернатива — азбука Морзе. Далее мы увидим, что в компьютерах применяются различные типы кодов для передачи чисел, звуков, музыки, изображений и видео. Компьютер не может работать непосредственно с человеческими кодами: машина не в состоянии сымитировать работу человеческих глаз, ушей, рта и пальцев. Недавно* в компьютерной технике наметилась такая тенденция: настольные ПК собирают и хранят различные виды информации, используемой при человеческом общении, и имеют возможность манипулировать такой информацией и ее отображениями. Это визуальная (текст, картинки) и акустическая (речь, звуки, музыка) информация, их комбинация (анимация или кино). Для всех этих типов требуются собственные коды, точно так же как при разговоре используются одни органы (рот и уши), а при письме и чтении — другие (руки и глаза).

Даже сама таблица с азбукой Морзе (с. 14) — в некотором роде код. В таблице каждая буква представлена последовательностью точек и тире. Но как передать точки и тире? Получается, они соответствуют бликам. Для обозначения точки мы быстро перещелкиваем кнопку фонарика (короткий блик), тире — задерживаем фонарик включенным чуть дольше. Так, чтобы передать *A*, мы быстро перещелкиваем фонарик, а потом включаем и выключаем его более медленно. Перед отправкой следующего символа делаем небольшую паузу. Принято, что тире должно быть примерно втрое длиннее точки. Так, если точка длится одну секунду, то тире — три (на самом деле азбука Морзе транслируется гораздо быстрее). Адресат видит короткий сигнал, затем длинный и понимает, что это *A*.

Паузы между точками и тире в азбуке Морзе критически важны. Так, при передаче *A* фонарик должен быть выключен между точкой и тире в течение периода, по длительности примерно равного одной точке. (Если точка длится одну секунду, то промежуток между точкой и тире также длится секунду.) Между буквами в слове выдерживаются более долгие паузы, сравнимые по длительности с тире (в данном случае по три секунды). Например, вот так на азбуке Морзе будет «привет» (обратите внимание на паузы между буквами).

• — — • • — • • • • — — • —

Между словами выдерживается период длительностью примерно два тире (шесть секунд, если тире — три секунды). Вот код фразы «как дела».

— • — • — — • — — • • • • — • • • —

* Книга написана в 2000 году, и автор отталкивается от реалий того периода. *Прим. перев.*

Длительность периодов, в течение которых фонарик остается включен или выключен, не фиксируется. Все периоды отсчитываются относительно длительности точки, а эта длина зависит от того, как быстро удается перещелкивать фонарик, насколько быстро отправитель азбуки Морзе успевает вспомнить код для той или иной буквы. Тире у быстрого отправителя может получиться таким же коротким, как точка у неторопливого. Из-за этой небольшой проблемы расшифровка сообщений может усложняться, но после первых двух-трех букв адресат обычно успевает сориентироваться, где точка, а где тире.

На первый взгляд, определение кода Морзе — под *определением* в данном случае я понимаю соответствие различных последовательностей точек и тире буквам алфавита — кажется столь же произвольным, как и раскладка клавиатуры на пишущей машинке. Если присмотреться, не все так однозначно. Сравнительно простые и краткие коды присваиваются более частотным буквам алфавита, например *E* и *T**. Любители игр «Эрудит» и «Поле чудес» могли это сразу приметить. У редких букв (например, *Q* и *Z* на латинице, за которые в «Эрудите» присваивается по 10 очков) коды длиннее.

Практически каждый хоть немного знает азбуку Морзе. Три точки, три тире, три точки — SOS, международный сигнал бедствия. SOS не аббревиатура. Это просто код из азбуки Морзе, который легко запоминается. Во время Второй мировой войны Британская радиовещательная компания предваряла некоторые передачи первыми нотами из Пятой симфонии Бетховена: ТА-ТА-ТА-ТАММММ! Сочиняя эту музыку, Людвиг ван Бетховен еще не мог знать, что именно такая последовательность сигналов (точка-точка-точка-тире) в азбуке Морзе будет соответствовать букве *V*, с которой начинается английское слово victory — «победа».

Один из недостатков азбуки Морзе в том, что в ней нет капитализации букв. Однако она позволяет передавать не только буквы, но и цифры, которым соответствуют свои последовательности по пять точек и тире.

1	• — — — —	6	— • • • •
2	• • — — —	7	— — • • •
3	• • • — —	8	— — — • •
4	• • • • —	9	— — — — •
5	• • • • •	0	— — — — —

Эти коды как минимум чуть более регулярны, чем буквенные. Для большинства знаков препинания используются по пять, шесть или семь точек и тире.

* В русском языке (по массивам текстов) это буквы *О, Е, А*. *Прим. науч. ред.*

.	• • • • • •	'	• — — — — •
,	• — • — • —	(	— • — — • —
?	• • — — • •	)	— • — — • —
:	— — — • • •	=	— • • • —
;	— • — • —	+	• — • — •
-	— • • • • —	$	• • • — • • —
/	— • • — •	¶	• — • — • •
"	• — • • — •	–	• • — — • —

Кроме того, существуют дополнительные коды для букв с диакритическими знаками из некоторых европейских языков и специальные последовательности-сокращения. Одно из таких сокращений — код SOS. Его следует посылать непрерывно, делая между каждой тройкой символов паузу в одну точку.

Вы убедитесь, что общаться с другом азбукой Морзе гораздо удобнее, если вооружиться специальным фонариком. Кроме обычного переключателя-ползунка, на такой фонарик монтируется кнопочный переключатель, который мы нажимаем и отпускаем, и фонарик зажигается и гаснет. Напрактиковавшись, вы, вероятно, научитесь передавать и принимать по пять-десять слов в минуту, что все равно гораздо медленнее, чем речь (при разговоре в минуту укладывается около 100 слов*), но вполне неплохо.

Когда вы с другом наконец-то выучите азбуку Морзе (а иначе общение при помощи этих сигналов не построить), вы сможете пользоваться таким словарем и в обычной речи. Для максимально быстрого общения произносите точку как «*дих*» («*дит*», если это последняя буква в слове), а тире — как «*дах*». Подобно тому как азбука Морзе позволяет сократить письмо до точек и тире, устный код редуцирует речь всего до двух слогов.

В данном случае ключевой элемент — *двойка*. Два типа бликов, два слога. Два любых феномена, если они разные, в правильных комбинациях подходят для передачи информации.

---

* При речи на английском языке. В русском языке темп речи (скорость произнесения ее элементов) медленнее, поскольку слова на 20–30% длиннее. *Прим. науч. ред.*

Глава 2
Коды и комбинации

Азбуку Морзе придумал Сэмюэл Финли Бриз Морзе (1791–1872). Это изобретение неотделимо от создания телеграфа, о работе которого нам также предстоит узнать. Азбука Морзе послужила хорошим вводным материалом для знакомства с сущностью кода, а телеграф — такой же удобный пример, иллюстрирующий аппаратное обеспечение компьютера.

Многим кажется, что азбуку Морзе проще передавать, чем принимать. Даже если вы не знаете ее на память, можете просто сверяться с таблицей, где буквы для удобства расставлены по алфавиту (с. 14).

Принимать азбуку Морзе и переводить ее в обычные слова значительно сложнее и дольше, поскольку вы работаете в обратном порядке: выясняете, какая буква соответствует конкретной кодовой последовательности точек и тире. Например, если вы получите сигнал «тире-точка-тире-тире», придется заглянуть в таблицу и просмотреть почти все буквы одну за другой, пока не выяснится, что перед вами *Ы*.

Проблема в том, что у нас есть таблица для следующего перевода:

буква алфавита → последовательность азбуки Морзе, состоящая из точек и тире.

Однако нет обратной таблицы:

последовательность азбуки Морзе, состоящая из точек и тире, → буква алфавита.

В начале изучения азбуки Морзе такая таблица, безусловно, пригодилась бы. Правда, не вполне понятно, как ее составить. Точки и тире не допускают никакого подобия алфавитного порядка.

Давайте забудем об алфавите. Пожалуй, разумнее сгруппировать коды таким образом, чтобы их расстановка зависела от количества точек и тире в той

или иной букве. Так, последовательность из азбуки Морзе, содержащая одну точку и одно тире, может означать всего одну из двух букв: *E* или *T*.

E	•	E	•
T	—	T	—

Комбинации, в которых содержится по два знака (либо точки, либо тире), дают нам уже четыре буквы: *И, A, H* и *M*.

I	• •	И	• •
A	• —	A	• —
N	— •	H	— •
M	— —	M	— —

Паттерн из трех символов, точек или тире, дает нам восемь букв: *C, Д, У, K, P, Г, O, B*.

S	• • •	D	— • •	C	• • •	Д	— • •
U	• • —	K	— • —	У	• • —	K	— • —
R	• — •	G	— — •	P	• — •	Г	— — •
W	• — —	O	— — —	B	• — —	O	— — —

Наконец (если мы хотим прекратить это упражнение, пока не перешли к цифрам и знакам препинания), четырехзначные последовательности точек и тире дают нам еще 16 символов.

H	• • • •	B	— • • •	X	• • • •	Б	— • • •
V	• • • —	X	— • • —	Ж	• • • —	Ь	— • • —
F	• • — •	C	— • — •	Ф	• • — •	Ц	— • — •
Ü	• • — —	Y	— • — —	Ю	• • — —	Ы	— • — —
L	• — • •	Z	— — • •	Л	• — • •	З	— — • •
Ä	• — • —	Q	— — • —	Я	• — • —	Щ	— — • —
P	• — — •	O	— — — •	П	• — — •	Ч	— — — •
J	• — — —	Ş	— — — —	Й	• — — —	Ш	— — — —

Всего в этих таблицах содержится 2 + 4 + 8 + 16 кодов суммарно для 30 букв; это на четыре кода больше, чем требуется для полной латиницы, состоящей из 26 букв. Именно поэтому четыре кода в последней таблице отведены под буквы с диакритическими знаками.

Эти четыре таблицы помогут с легкостью переводить любые сообщения, передаваемые азбукой Морзе. Получив код конкретной буквы, вы считаете, сколько в нем точек и тире, и решаете, с какой из таблиц сверяться. Каждая таблица устроена так, что код, состоящий из одних точек, располагается в верхнем левом углу, а код из одних тире — в нижнем правом углу.

Замечаете закономерность в *размерах* четырех таблиц? Обратите внимание: в каждой следующей таблице вдвое больше кодов, чем в предыдущей. Это логично: в последующей таблице содержатся все коды из предыдущей «плюс точка», а также все коды из предыдущей «плюс тире».

Эту тенденцию можно резюмировать следующим образом.

Количество точек и тире	1	2	3	4
Количество кодов	2	4	8	16

Каждая из четырех таблиц содержит вдвое больше кодов, чем предшествующая ей таблица, так что если в первой таблице 2 кода, то во второй — 2 × 2 кодов, в третьей — 2 × 2 × 2 кодов. Вот как еще можно это представить.

Количество точек и тире	1	2	3	4
Количество кодов	2	2 × 2	2 × 2 × 2	2 × 2 × 2 × 2

Разумеется, при умножении числа самого на себя можно использовать степени. Так, 2 × 2 × 2 × 2 можно записать как 2^4 (2 в четвертой степени). Числа 2, 4, 8 и 16 являются степенями двойки, поскольку представляют произведения, которые можно получить умножением двойки самой на себя. Итак, нашу таблицу можно переписать и так.

Количество точек и тире	1	2	3	4
Количество кодов	2^1	2^2	2^3	2^4

Таблица сильно упростилась. Количество кодов равно просто 2 в степени <количество точек и тире>. Можно резюмировать табличные данные в виде простой формулы:

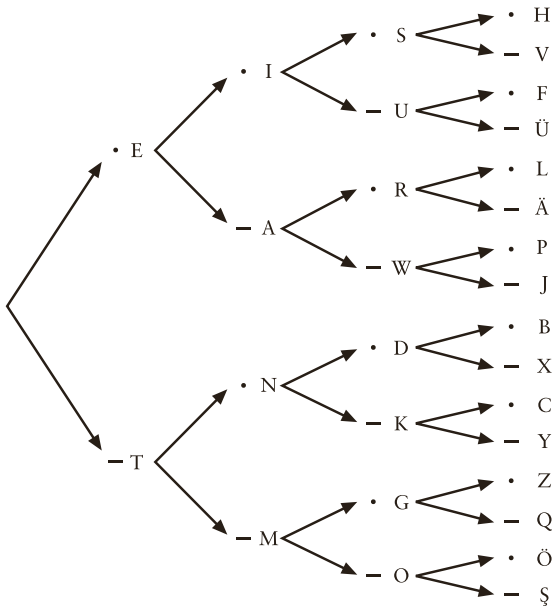
$$\text{Количество кодов} = 2^{\text{количество точек и тире}}.$$

Степени двойки часто используются в различных кодах (другой пример рассмотрим в следующей главе).

Чтобы еще сильнее упростить расшифровку кода Морзе, давайте попробуем построить большую древовидную схему на следующей странице.

На схеме показано, какие буквы получаются при постепенном усложнении последовательностей точек и тире. Чтобы расшифровать конкретную последовательность, идите по стрелкам слева направо. Допустим, мы хотим выяснить, какая буква соответствует коду «точка-тире-точка». Начинаем слева, берем точку; далее идем по стрелкам, выбираем тире, а затем еще одну точку. Получаем букву *R*, расположенную около последней точки.

Такая схема необходима прежде всего для того, чтобы определить код Морзе. Во-первых, она страхует от тупой ошибки: не дает присвоить двум разным буквам один и тот же код. Во-вторых, вы гарантированно задействуете все возможные коды, не выстраивая чрезмерно длинных последовательностей из точек и тире.



Рискуя получить схему, которая не поместится на печатной странице, мы могли бы расширить ее и добавить туда пятизначные коды из точек и тире. Последовательность из пяти точек и тире даст нам 32 ($2 \times 2 \times 2 \times 2 \times 2$, или 2^5) дополнительных кода. Как правило, этого достаточно не только для букв, но и для 10 цифр и 18 знаков препинания, включаемых в азбуку Морзе: цифры действительно кодируются пятизначными последовательностями точек и тире. Правда, многие другие пятизначные коды зарезервированы не за знаками препинания, а за буквами с диакритическими знаками.

Чтобы система учитывала все знаки препинания, в нее нужно включить последовательности из шести точек и тире. Таким образом получим

64 ($2 \times 2 \times 2 \times 2 \times 2 \times 2$, или 2^6) дополнительных кода для суммарного множества из 2 + 4 + 8 + 16 + 32 + 64, или 126, символов. Для азбуки Морзе этого слишком много, поэтому большинство таких длинных кодов остаются неопределенными. Слово «*неопределенный*» в данном контексте указывает на код, который ничего не означает. Если бы вы, принимая азбуку Морзе, получили неопределенный код, то могли бы почти не сомневаться, что кто-то просто допустил ошибку.

У нас хватило смекалки построить эту небольшую формулу:

$$\text{Количество кодов} = 2^{\text{количество точек и тире}}.$$

Так давайте продолжим нашу таблицу и посмотрим, сколько кодов получится из более длинных последовательностей точек и тире.

Количество точек и тире	**Количество кодов**
1	$2^1 = 2$
2	$2^2 = 4$
3	$2^3 = 8$
4	$2^4 = 16$
5	$2^5 = 32$
6	$2^6 = 64$
7	$2^7 = 128$
8	$2^8 = 256$
9	$2^9 = 512$
10	$2^{10} = 1024$

К счастью, нет необходимости выписывать все возможные коды, чтобы определить, сколько их будет. Достаточно умножать двойку на себя нужное количество раз.

Код Морзе называется *двоичным* (что буквально означает «два на два»), поскольку любой его элемент включает только два компонента: точку и тире. Такой код подобен монете, которая может упасть только решкой или орлом. Двоичные объекты (например, монеты) и двоичные коды (например, азбука Морзе) всегда можно описать в виде степеней двойки.

Проделанный нами анализ двоичных кодов — это простое упражнение в одной математической дисциплине, которая называется *комбинаторикой,* или *комбинаторным анализом*. Традиционно комбинаторный анализ особенно активно используется в теории вероятностей и статистике, поскольку связан с выявлением количества вариантов комбинаций различных объектов (например, монет или игральных костей). Он также помогает понять, как составляются и разбираются коды.

Глава 3
Брайль и двоичные коды

Сэмюэл Морзе не был первым, кому успешно удалось транслировать буквы письменного языка в интерпретируемый код. Он не был первым и среди тех, чья фамилия запомнилась как название кода, а не имя собственное. Такая честь выпала слепому французскому подростку, родившемуся примерно через 18 лет после Морзе, но оставившему след в истории гораздо раньше. О жизни Луи Брайля известно немногое, но это захватывающая история.

Луи Брайль родился в 1809 году во французском городке Кувр**е**, в 40 километрах к востоку от Парижа. Отец мальчика был шорником. Будучи трех лет от роду (а в таком возрасте дети не должны играть в отцовской мастерской), Луи случайно ткнул себе в глаз шорным ножом. В ране начался процесс заражения, инфекция распространилась и на второй глаз, и мальчик полностью ослеп. Наверняка его ждала жизнь в невежестве и бедности (как и большинство слепцов в те времена), но Луи проявил незаурядный ум и тягу к знаниям. Благодаря участию деревенского пастора и школьного учителя Луи ходил в сельскую школу вместе с другими ребятами, а в возрасте десяти лет отправился в Парижский государственный институт для слепых детей.

Разумеется, одна из главных сложностей при обучении незрячих в том, что они не могут читать печатные книги. Основатель этой парижской школы Валентин Гаюи (1745–1822) изобрел систему тисненых выпуклых букв для чтения их на ощупь. Но пользоваться системой было сложно, и вышло только несколько книг, напечатанных таким методом. Гаюи не смог посмотреть глубже. Для него буква *A* оставалась буквой *A*. Она должна была выглядеть (ощущаться) как *A*. (Общаясь на языке световых сигналов, мы пробовали рисовать буквы в воздухе и убедились, что такой прием неработоспособен.) Вероятно, Гаюи не догадался, что некий код, сильно отличающийся от печатного алфавита, оказался бы для незрячих удобнее.

Прообраз такого альтернативного кода возник в достаточно необычном контексте. Шарль Барбье, капитан французской армии, изобрел систему записи под названием écriture nocturne, или «ночная азбука». В ней использовались

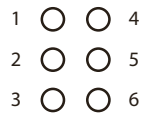
узоры выпуклых точек и тире на плотной бумаге. Предполагалось, что солдаты могли бы обмениваться в темноте такими записками, когда требовалось соблюдать тишину. Писать точки и тире можно было специальным стилусом, вроде шила. Затем выпуклые точки можно было читать на ощупь. Недостаток системы Барбье заключался в ее чрезмерной сложности. Комбинации точек и тире соответствовали звукам, а не буквам алфавита, поэтому одно слово часто могло шифроваться разными кодами. Система хорошо работала для обмена короткими сообщениями в полевых условиях, но решительно не подходила для сравнительно крупных текстов, тем более книг.

Луи Брайль познакомился с системой Барбье в двенадцатилетнем возрасте. Ему понравились выпуклые точки не только потому, что они легко читались на ощупь, но и потому, что их было просто *писать*. Ученик в классе, вооружившись бумагой и стилусом, в самом деле мог записывать и читать такие сообщения. Луи Брайль постарался усовершенствовать эту систему, и через три года (когда ему было пятнадцать) в общих чертах составил собственную, основы которой применяются и сегодня. Много лет такая система использовалась лишь в школах, но постепенно вошла в широкое употребление. В 1835 году Брайль подхватил туберкулез, от которого и умер в 1852 году, в возрасте 43 лет.

Сегодня усовершенствованные варианты системы Брайля соперничают с аудиокнигами, обеспечивая незрячим доступ к письменной информации. Тем не менее шрифт Брайля по-прежнему незаменим и является единственной письменностью, доступной слепоглухим. Шрифт Брайля применяется даже в общественных местах, например в лифтах и банкоматах.

В этой главе мы препарируем код Брайля и разберемся, как он работает. Мы не будем *учить* код Брайля или что-то запоминать. Мы лишь попробуем на этом примере лучше понять его природу.

В шрифте Брайля каждый символ, присутствующий в обычном письменном языке, то есть буквы, цифры и знаки препинания, кодируется в виде одной или нескольких точек в клетке размером две на три точки. Как правило, точки в клетке нумеруются от 1 до 6.



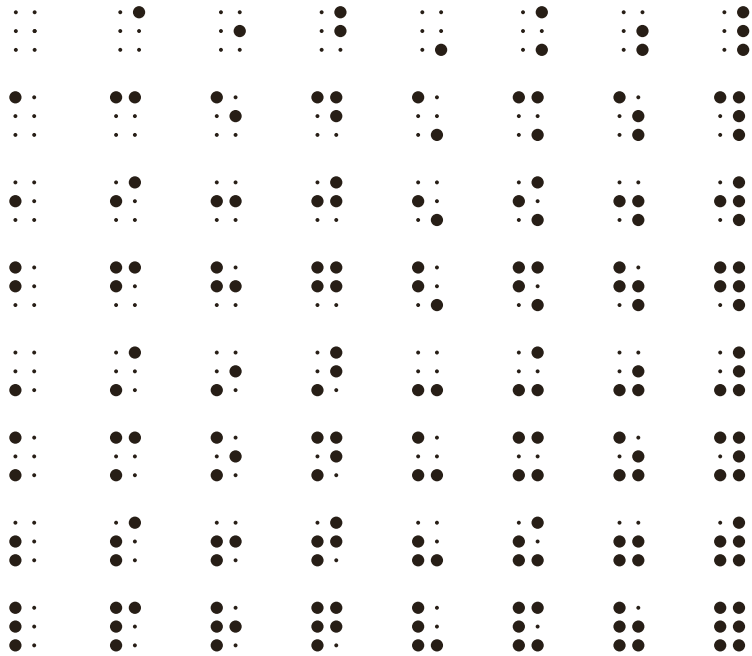
В настоящее время существуют специальные пишущие машинки — брайлевские принтеры, выбивающие точки брайлевского шрифта на бумаге.

Поскольку книга получилась бы запредельно дорогой, если бы хоть пару страниц набрали шрифтом Брайля, я пользовался нотацией, традиционно применяемой для передачи азбуки Брайля при печати. В такой нотации отображаются

все шесть точек в клетке. Жирные точки — это выпуклости на бумаге, мелкие — плоские элементы клетки. Например, в следующем брайлевском символе точки 1, 3 и 5 выпуклые, а 2, 4 и 6 — нет.

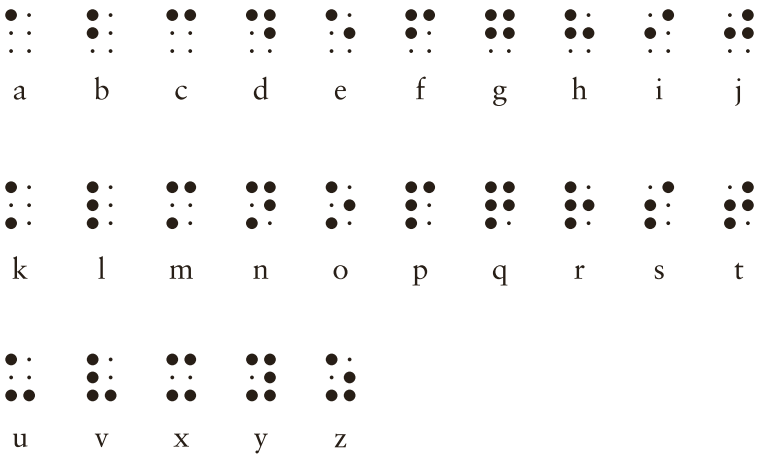
На данный момент нас должно заинтересовать, что эти точки *двоичны*. Любая точка может быть либо выпуклой, либо плоской. Таким образом, шрифт Брайля подчиняется тем же принципам, которые знакомы нам из азбуки Морзе и комбинаторного анализа. Известно, что в клетке шесть точек, и каждая точка может быть плоской или выпуклой, поэтому общее число комбинаций, которые складываются из шести плоских или выпуклых точек, равно $2 \times 2 \times 2 \times 2 \times 2 \times 2$, или 2^6, или 64.

Как видите, в системе Брайля можно представить 64 уникальных кода.



Если в шрифте Брайля используется менее 64 кодов, логично спросить, почему не все возможные варианты в ходу. Если в шрифте Брайля найдется более 64 возможных кодов, значит, сбоит либо наш разум, либо фундаментальные математические истины из разряда «два плюс два равно четырем».

Приступая к изучению шрифта Брайля, рассмотрим, как в нем записываются строчные буквы латиницы.



Например, английская фраза You and me* записывается следующим образом.

Важно: между клетками, соответствующим буквам в слове, ставятся небольшие пробелы; более широкий пробел (в сущности, целая клетка, в которой нет выпуклых точек) соответствует пробелу между словами.

Именно такова основа шрифта Брайля в редакции самого Брайля — как минимум что касается латиницы. Луи Брайль также разработал коды для букв с диакритическими знаками (они часто встречаются во французском языке). Обратите внимание: здесь нет кода для буквы *w*, которая в классическом французском не используется. (Не волнуйтесь, и эта буква вскоре появится.) Пока мы учли всего 25 из 64 возможных кодов.

Внимательно присмотревшись к вышеприведенным строчкам, можно заметить, что в них прослеживается закономерность. В первой строчке (от *a* до *j*) в каждой клетке используются лишь четыре верхние точки: 1, 2, 4 и 5. Второй ряд точно такой же, как первый, но в нем есть и точка 3. Третий ряд подобен первым двум, но в нем мы видим не только точку 3, но и точку 6.

Со времени Луи Брайля его шрифт дополняли различным образом. Современная система, при помощи которой обычно записываются подобные английские тексты, называется «сокращенный Брайль». В сокращенном Брайле много упрощений, помогающих беречь деревья и ускорять чтение. Например, если код некоторой буквы стоит обособленно, то он означает распространенное слово. В следующих трех рядах приведены такие коды для целых слов.

---

* «Ты и я». *Прим. перев.*

(none) (отсутствует)	but но	can может	do делать	every каждый	from от, из	go идти	have иметь	(none)	just только, сейчас

knowledge знание	like как, любить	more больше	not не	(none)	people люди	quite весьма	rather вполне	so так	that так что

us нам, нас, нами	very очень	it оно	you ты	as как, в качестве	and и	for для	of (предлог)	the (артикль)	with с

Таким образом, фразу You and me сокращенным Брайлем можно записать так.

Вот мы и описали 31 код: пробел без точек, который ставится между словами, и три строки по десять кодов, используемых для обозначения букв и слов. Мы до сих пор и близко не израсходовали 64 теоретически доступных кода. Как мы убедимся, в сокращенном Брайле ни один не остался без дела.

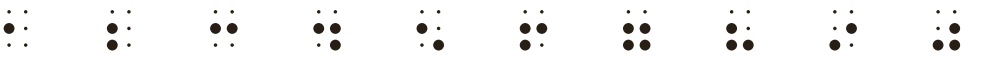
Во-первых, можно использовать коды букв *a* — *j*, добавляя к каждому из них выпуклую точку 6. Эти коды применяются в основном для сокращения в слове букв, для буквы *w* и другого сокращения слов.

ch	gh	sh	th	wh	ed	er	ou	ow	w (или «will»*)

** Will — вспомогательный глагол для образования будущего времени.*

Например, слово about* можно записать сокращенным Брайлем вот так.

Во-вторых, можно взять коды букв *a* — *j* и «опустить» их так, чтобы использовались лишь точки 2, 3, 5 и 6. Этими кодами обозначаются некоторые знаки препинания и сокращения, в зависимости от контекста.



ea	bb	cc	dis	en	to к	gg	his его	in в	was был(а)
,	;	:	.		!	()	“		”

Первые четыре приведенных кода — это запятая, точка с запятой, двоеточие и точка. Обратите внимание: как открывающая, так и закрывающая скобки обозначаются одним и тем же кодом, а вот коды для открывающей и закрывающей кавычки отличаются.

Пока мы использовали 51 код. Далее приведены шесть кодов, представляющих различные незадействованные комбинации точек 3, 4, 5 и 6. С их помощью записывают сокращения и некоторые дополнительные знаки препинания.

st /	ing	ble #	ar	'	com -

Код ble очень важен: если это не часть слова, то он означает, что следующие далее коды должны интерпретироваться как числа. Числовые коды точно такие же, как и для букв *a* — *j*.

1	2	3	4	5	6	7	8	9	0

Следовательно, нижеприведенная последовательность означает 256.

* О, около, про. *Прим. перев.*

Если вы следите за нитью повествования, то помните, что до максимума (64) нам остается еще семь кодов. Вот они.

Первый код (выпуклая точка 4) — индикатор ударения. Остальные используются в качестве префиксов при некоторых сокращениях, а также в иных целях. Например, при выпуклых точках 4 и 6 (пятый код в этом ряду) код может означать либо десятичную запятую (для чисел), либо логическое ударение — в зависимости от контекста.

Наконец (если вам не терпится узнать, как в шрифте Брайля записываются заглавные буквы), у нас есть выпуклая точка 6. Это индикатор заглавной буквы. Следующая после такого символа буква будет в верхнем регистре. Например, имя создателя этой системы записывается так.

Здесь индикатор заглавной буквы, буква *l*, буквосочетание *ou*, буквы *i* и *s*, пробел, еще один индикатор заглавной буквы, а далее — буквы *b, r, a, i, l, l* и *e* (на практике эта запись может быть еще короче: отбрасываются две последние буквы, так как они не произносятся).

Итак, мы рассмотрели, как шесть двоичных элементов (точек) дают 64 возможных кода — и не больше. Получается, что многие из этих кодов выполняют двойную работу в зависимости от контекста. Особенно интересны «числовой» и «буквенный» индикаторы (при этом второй отменяет первый). Эти коды меняют семантику других кодов — тех, что следуют за ними: с букв на цифры и обратно с цифр на буквы. Подобные коды часто именуются кодами *старшинства* или *переключения*. Они меняют семантику всех последующих кодов до тех пор, пока переключение не будет отменено.

Индикатор заглавной буквы означает, что следующая (и только следующая) буква должна быть в верхнем, а не в нижнем регистре. Такой код принято называть *экранирующим*, и он «защищает» последовательность других кодов от банальной, рутинной семантики и обеспечивает им новую интерпретацию. Читая следующие главы, убедимся, что коды переключения и экранирующие коды постоянно используются в ситуациях, когда письменный язык нужно представить в двоичном виде.

Глава 9
За битом бит

Когда в 1973 году Тони Орландо в своей песне попросил, чтобы любимая «повязала желтую ленточку вокруг старого дуба», он не сопроводил свою просьбу ни подробными объяснениями, ни долгими рассуждениями. Никаких «если», «и», «но». Несмотря на сложные чувства и эмоции, сопровождавшие ситуацию, что разворачивалась в реальной жизни и легла в основу песни, мужчине хотелось получить простой ответ: «да» или «нет». Он знал, что, если на старом дубе появится желтая ленточка, это будет означать: «Да, хотя ты и наворотил дел и провел три года в тюрьме, я все равно хочу, чтобы ты вернулся и мы жили под одной крышей». А отсутствие желтой ленточки скажет: «Даже не думай здесь останавливаться».

Здесь есть две четкие взаимоисключающие альтернативы. Тони Орландо не пел: «Повяжи половину желтой ленточки, если тебе нужно время на размышление» или «Повяжи голубую ленточку, если больше не любишь меня, но по-прежнему хочешь остаться друзьями». Нет, он все сформулировал очень просто.

Не менее информативно, нежели отсутствие или наличие желтой ленточки (разве что не столь поэтично), сработал бы дорожный знак, поставленный во дворе, например «Путь открыт» или «Въезд запрещен». Или табличка на двери «Закрыто» или «Открыто». Или фонарик на окне — включенный или выключенный.

Если вам нужно просто сказать «да» или «нет», то способов хватает. Для этого не надо произносить ни одной фразы, слова, даже буквы. Необходим всего один *бит*, то есть 0 или 1.

Как мы узнали в предыдущих главах, десятеричная система, которой мы пользуемся при подсчете предметов, на самом деле ничем не примечательна. Ясно, что мы пользуемся системой с основанием 10, так как у нас 10 пальцев на руках. Мы могли бы с тем же успехом использовать систему с основанием 8 (будь мы мультяшками), 4 (будь мы омарами) или даже 2 (если бы мы были дельфинами).

Однако двоичная система счисления *все-таки* особенная. Дело в том, что это *простейшая* возможная система счисления. В двоичной системе всего две

цифры: 0 и 1. Если нам нужно что-то проще двоичной системы, придется избавиться от 1, и останется только 0. Имея всего лишь 0, ничего не сделаешь.

Слово «бит» — сокращение от английского выражения binary digit («двоичная цифра»). Пожалуй, это одно из симпатичнейших слов в компьютерной терминологии. В английском языке у слова bit есть и общеупотребительное значение — «кусочек, небольшая часть», и это значение нам отлично подходит, поскольку один бит — двоичная цифра, мельчайший фрагмент информации.

Иногда, если изобретается новое слово, ему присваивается новое значение. В данном случае все именно так. Слово «бит» — это не только двоичные цифры, при помощи которых удобно считать дельфинам. В компьютерную эпоху это слово приобрело значение «*мельчайший первичный фрагмент информации*».

Да, смелое утверждение. Естественно, информация передается не только при помощи битов. Буквы, слова, азбука Морзе и шрифт Брайля, а также десятеричные цифры тоже переносят информацию. Суть бита заключается в том, что он передает очень *мало* информации. Один бит — это минимально возможное количество информации. Меньше бита — отсутствие информации.

Поскольку бит выражает минимальный возможный объем информации, более сложную информацию можно передать некоторым количеством битов. (Говоря, что бит передает мало информации, я совершенно не имею в виду, что эта информация граничит с бессмыслицей. Желтая ленточка *очень* важна для тех, кто в ней заинтересован.)

«Запомните, дети, — слышал весь мир, // Как в полночь глухую скакал Поль Ревир…» — писал Генри Лонгфелло. Возможно, он исторически недостоверно описал поступок Поля Ревира, предупредившего американских колонистов о вторжении британцев, однако эти стихи замечательно демонстрируют, как можно передать важную информацию при помощи битов.

Он другу сказал: «Я сигнала жду.
Коль ночью из города наступать
Начнут британцы, ты дай мне знать,
На Северной церкви зажги звезду, —
Одну, если сушей, а морем — две».*

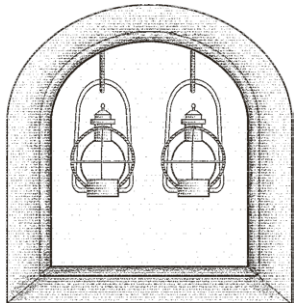
Итак, Ревир дал другу два фонаря. Если британцы вторгнутся по суше, друг зажжет на колокольне один, если высадятся с моря — то зажжет оба.

Правда, Лонгфелло не дает явного указания на все случаи жизни. Он умалчивает о *третьей* возможности, означающей, что британцы пока не вторгаются.

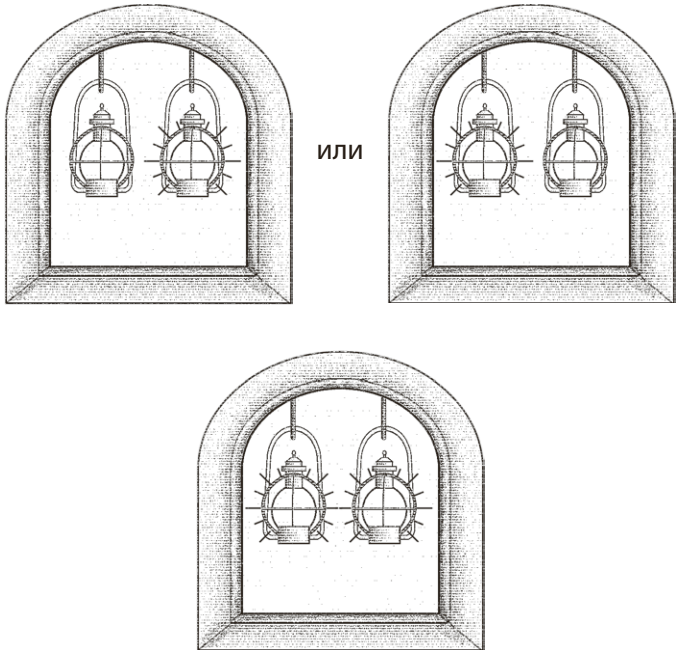
* Здесь и далее стихотворение приводится в переводе М. А. Зенкевича.

Лонгфелло подразумевает, что тогда ни одного фонаря на колокольне гореть не будет.

Допустим, два фонаря висят на колокольне всегда. Как правило, они не горят.



Значит, британцы пока не наступают. Если зажжен один фонарь, значит, британцы наступают по суше, если горят оба, — высаживаются с моря.



Каждый фонарь — это один бит. Зажженный фонарь равен единичному биту, незажженный — нулевому биту. Тони Орландо показал, что для описания любой из двух возможностей достаточно всего одного бита. Если бы Поль Ревир должен был оповестить людей лишь о том, вторгаются британцы или нет (а не откуда именно), то хватило бы и одного фонаря. Фонарь горел бы в случае нападения и был бы потушен, если бы предстоял еще один спокойный вечер.

Чтобы описать одну из трех возможностей, требуется два фонаря. При наличии второго фонаря и двух битов уже можно описать четыре возможности:

00 = британцы пока не наступают;
01 = британцы наступают с суши;
10 = британцы наступают с суши;
11 = британцы наступают с моря.

На самом деле Поль Ревир, ограничившись всего тремя возможностями, поступил весьма хитроумно. В телекоммуникационной терминологии можно сказать, что он задействовал *избыточность*, позволяющую бороться с *шумом*. Термин «шум» в теории телекоммуникаций означает любые помехи, осложняющие передачу информации. Типичный пример шума — помехи на телефонной линии. Тем не менее, даже несмотря на помехи, общение по телефону обычно проходит успешно, поскольку устная речь крайне избыточна. Не требуется четко слышать каждый слог в каждом слове, чтобы понять, что сказали.

В случае с фонарями на колокольне к «шуму» можно отнести такие факторы, как темнота и расстояние от Поля Ревира до колокольни, которые могут помешать отличить первый фонарь от второго. Вот важнейший отрывок из стихов Лонгфелло:

Вдруг видит — звездой огонек замигал,
То дан с колокольни желанный сигнал!
В седло он вскочил, сжал повод в ладонь.
Под ним захрапел в нетерпенье конь.
Второй сигнал! Он коня погнал!

Скорее всего, Поль Ревир был не в состоянии определить, какой фонарь зажегся первым.

Из этого следует важный вывод: *информация — выбор из двух или более возможностей*. Например, когда мы говорим с кем-либо, любое произносимое слово выбирается из словаря. Если бы мы пронумеровали все слова из словаря от 1 до 351 482, то могли бы столь же четко вести беседу, называя номера, а не слова (естественно, обоим собеседникам понадобились бы словари, где все слова пронумерованы одинаково, а также терпение).

Верно и обратное: *любую информацию можно свести к выбору между двумя или более возможностями*, а сами возможности выразить при помощи битов. Излишне говорить, что существует множество таких вариантов человеческой коммуникации, где выбор между конкретными возможностями

не предоставляется, и такие формы коммуникации также жизненно необходимы. Вот почему люди не флиртуют с компьютерами (по крайней мере, хочется на это надеяться). Если некоторую информацию невозможно выразить при помощи слов, картинок или звуков, значит, ее нельзя закодировать при помощи битов. Да и не возникает такого желания.

Жесты «палец вверх» или «палец вниз» содержат по одному биту информации. Еще есть жесты «два пальца вверх» и «два пальца вниз». Два последних жеста нравились ныне покойным кинокритикам Роджеру Эберту и Джину Сискелу, которые таким образом выносили окончательные вердикты по новейшим фильмам. Итак, в данном случае мы следим только за их большими пальцами. Получается четыре возможности, которые можно представить в виде двух битов:

00 = обоим не понравилось;
01 = Сискелу не понравилось, Эберту понравилось;
10 = Сискелу понравилось, Эберту не понравилось;
11 = обоим понравилось.

Первый бит относится к Сискелу; таким образом, 0 означает, что кино не понравилось Сискелу, а 1 — что кино понравилось Сискелу. Аналогично второй бит описывает вкусы Эберта. Итак, если друг спросит, что решили Сискел и Эберт о фильме Impolite Encounter, вы, вместо того чтобы ответить: «С точки зрения Сискела — большой палец вверх, с точки зрения Эберта — большой палец вниз» или «Сискелу понравилось; Эберту — нет», можете просто сказать: «Один ноль». Если друг знает, какой бит относится к Сискелу, а какой — к Эберту, а 1 означает «палец вверх», 0 — «палец вниз», то ваш ответ будет понятен. Просто вы должны знать код.

Можно сразу условиться, что бит 1 означает «палец вниз», а 0 — «палец вверх». Это может показаться нелогичным. Естественно, мы привыкли считать, что 1 означает нечто «утвердительное», а 0 — наоборот. На самом деле такое соответствие произвольное. Требуется всего лишь, чтобы все, кто пользуется кодом, знали значения бита 0 и 1.

Значение конкретного бита или совокупности битов всегда понимается в контексте. Вероятно, смысл желтой ленточки, повязанной на конкретном дубе, понятен лишь тому, кто ее туда повесил, и тому, для кого предназначен этот знак. Достаточно изменить цвет, дерево, дату — и значение исчезнет, останется просто тряпочка. Чтобы извлечь какую-либо полезную информацию из жестикуляции Сискела и Эберта, нужно как минимум понимать, о каком фильме идет речь.

Если вы вели список фильмов, отрецензированных Сискелом и Эбертом, и фиксировали их голоса, можно добавить в систему еще один бит, который будет выражать ваше мнение. В таком случае количество возможностей доходит до восьми:

000 = Сискелу не понравилось; Эберту не понравилось; мне не понравилось;
001 = Сискелу не понравилось; Эберту не понравилось; мне понравилось;
010 = Сискелу не понравилось; Эберту понравилось; мне не понравилось;
011 = Сискелу не понравилось; Эберту понравилось; мне понравилось;
100 = Сискелу понравилось; Эберту не понравилось; мне не понравилось;
101 = Сискелу понравилось; Эберту не понравилось; мне понравилось;
110 = Сискелу понравилось; Эберту понравилось; мне не понравилось;
111 = Сискелу понравилось; Эберту понравилось; мне понравилось.

Один из плюсов при представлении этой информации в виде битов таков: мы уверены, что учли все возможности, мы знаем, что существует восемь и только восемь возможностей — ни больше, ни меньше. Имея три бита, можно сосчитать лишь от нуля до семи. Трехзначных двоичных чисел больше нет.

Итак, при описании таких битов Сискела и Эберта вас, возможно, уже занимает следующий непростой вопрос: «А что насчет сборника рецензий Leonard Maltin's Movie & Video Guide?» Ведь Леонард Малтин оценивает фильмы не такими «пальцевыми» жестами, а более традиционно — при помощи системы звезд.

Чтобы определить, сколько битов Малтина требуется, сперва необходимо разобраться в его системе. Малтин оценивает фильм в некоторое количество звезд (от одной до четырех), причем звезда может делиться и пополам. (Кстати, одну звезду Мартин никогда не присваивает; вместо этого такой фильм получает рейтинг КРАХ.) Вот семь возможностей, и это означает, что для представления любой оценки достаточно трех бит:

000 = КРАХ;
001 = ★★;
010 = ★★;
011 = ★★★;
100 = ★★★;
101 = ★★★★;
110 = ★★★★.

«Что же насчет 111?» — могли бы спросить вы. Да, этот код ничего не значит. Он не определен. Если бы мы использовали двоичный код 111 в рейтингах Малтина, это бы свидетельствовало об ошибке. (Вероятно, ошибся компьютер, ведь люди никогда не ошибаются!)

Напомню, что, когда мы представляли оценки Сискела и Эберта при помощи двух битов, левый бит относился к Сискелу, правый — к Эберту. Есть ли в данном случае какое-либо значение у отдельных битов? Да, в некотором роде. Если взять числовое значение такого битового кода, прибавить к нему 2, а затем разделить результат на 2, то получится количество звезд. Это возможно лишь потому, что мы определяли коды продуманно и непротиворечиво. Мы вполне могли определить коды и так:

000 = ★★★;
001 = ★½;
010 = ★★½;
011 = ★★★★;
101 = ★★★½;
110 = ★★;
111 = КРАХ.

Этот код столь же адекватен, как и предыдущий, если всем понятно его значение.

Если бы Малтину попался фильм, не заслуживающий даже единственной звезды, он мог бы присвоить ему рейтинг в половину звезды. Определенно, кодов на это ему бы хватило. В таком случае коды следовало бы перераспределить так:

000 = БОЛЬШОЙ КРАХ;
001 = КРАХ;
010 = ★½;
011 = ★★;
100 = ★★½;
101 = ★★★;
110 = ★★★½;
111 = ★★★★.

Однако если бы затем нашелся такой провальный фильм, который даже половины звезды не заслуживает (МЕГАКРАХ?), то понадобился бы еще один бит: 3-битных кодов больше не осталось.

В журнале Entertainment Weekly оцениваются не только фильмы, но и телешоу, CD, книги, сайты. Оценки варьируются от A+ до F (правда, есть ощущение, что такой чести удостаиваются лишь фильмы Поли Шора). Если подсчитать все возможные оценки, их наберется тринадцать. Для представления этой системы понадобится четыре бита:

0000 = F;
0001 = D–;
0010 = D;
0011 = D+;
0100 = C–;
0101 = C;
0110 = C+;
0111 = B–;
1000 = B;
1001 = B+;
1010 = A–;
1011 = A;
1100 = A+.

У нас осталось три неиспользованных кода: 1101, 1110 и 1111, а всего их шестнадцать.

Рассуждая о битах, мы часто говорим о конкретном *числе* битов. Чем больше битов у нас в распоряжении, тем больше возможностей удается с их помощью описать.

Естественно, с десятеричными числами складывается точно такая же ситуация. Например, сколько всего региональных телефонных кодов? Региональный телефонный код в США состоит из трех цифр. Если все эти коды будут задействованы (на самом деле пока не все, но мы это проигнорируем), получится 10^3, или 1000 таких кодов, — от 000 до 999. Сколько семизначных телефонных кодов может быть в зоне действия регионального кода 212? 10^7, или 10 000 000. Сколько телефонных номеров может быть в зоне действия регионального кода 212, причем с префиксом 260? 10^4, или 10 000.

Аналогично в двоичной системе количество возможных кодов всегда равно двойке в некоторой степени, где степень — количество битов.

Количество битов	Количество кодов
1	$2^1 = 2$
2	$2^2 = 4$
3	$2^3 = 8$
4	$2^4 = 16$
5	$2^5 = 32$
6	$2^6 = 64$
7	$2^7 = 128$
8	$2^8 = 256$
9	$2^9 = 512$
10	$2^{10} = 1024$

Каждый дополнительный бит удваивает количество кодов.

Если известно, сколько нужно кодов, можно ли рассчитать, сколько для этого потребуется битов? Иными словами, как считать в противоположном направлении по вышеприведенной таблице?

Используемый при этом метод иногда называется «*логарифм по основанию два*». Логарифм — феномен, противоположный возведению в степень. Известно, что 2 в седьмой степени равно 128. Логарифм 128 по основанию 2 равен 7. В более строгой математической нотации получается, что выражение

$$2^7 = 128$$

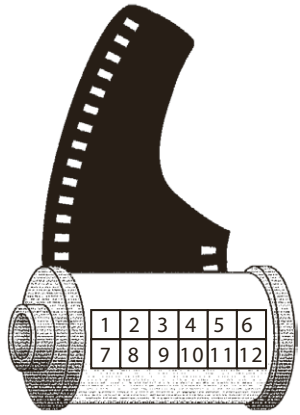
эквивалентно выражению

$$\log_2 128 = 7.$$

Итак, если логарифм 128 по основанию 2 равен 7, а логарифм 256 по основанию 2 равен 8, то каков логарифм 200 по основанию 2? На самом деле он равен примерно 7,64, но нам этого знать и не требуется.

Как правило, биты скрыты от поверхностного взгляда в глубинах электронных устройств. Вы не увидите их на компакт-диске, в электронных часах или внутри компьютера. Лишь иногда биты различимы вполне отчетливо.

Приведу пример. Если у вас есть обычный фотоаппарат с 35-миллиметровой пленкой, возьмите кассету и поверните ее, как показано на рисунке.



Перед вами предстанет набор серебристых и черных квадратов, отдаленно напоминающий шахматную доску. На рисунке я пронумеровал их от 1 до 12. Этот набор называется DX-кодировкой. Эти 12 квадратов в действительности представляют собой 12 бит. Серебристый квадрат соответствует 1, черный — 0. Квадраты 1 и 7 всегда серебристые (1).

Что значат эти биты? Вы, возможно, знаете, что пленки различаются по светочувствительности. Пленка высокой чувствительности считается «быстрой», поскольку на ней можно производить съемку с очень короткими экспозициями. Чувствительность пленки измеряется в единицах ASA (American Standards Association, Американская ассоциация по стандартам), причем наиболее популярны пленки с чувствительностью 100, 200 и 400 единиц*. Чувствительность в единицах ASA не только напечатана на упаковке, но и закодирована на кассете.

Различают 24 стандартные степени светочувствительности для фотопленки.

25	32	40
50	64	80
100	125	160
200	250	320
400	500	640
800	1000	1250
1600	2000	2500
3200	4000	5000

* Сегодня в большинстве стран мира используется система ISO (International Organization for Standartization, Международная организация по стандартизации). Обозначения светочувствительности пленки по системам ASA и ISO совпадают. *Прим. науч. ред.*

Сколько битов нужно, чтобы закодировать чувствительность пленки? Ответ прост: пять. Мы знаем, что $2^4 = 16$, а это слишком мало. А вот $2^5 = 32$ — больше чем достаточно.

Соответствие между квадратами на кассете и чувствительностью пленки показано в таблице.

Квадрат 2	Квадрат 3	Квадрат 4	Квадрат 5	Квадрат 6	Светочувствительность пленки
0	0	0	1	0	25
0	0	0	0	1	32
1	0	0	1	0	50
1	0	0	0	1	64
1	0	0	1	1	80
0	1	0	1	0	100
0	1	0	0	1	125
0	1	0	1	1	160
1	1	0	1	0	200
1	1	0	0	1	250
1	1	0	1	1	320
0	0	1	1	0	400
0	0	1	0	1	500
0	0	1	1	1	640
1	0	1	1	0	800
1	0	1	0	1	1000
1	0	1	1	1	1250
0	1	1	1	0	1600
0	1	1	0	1	2000
0	1	1	1	1	2500
1	1	1	1	0	3200
1	1	1	0	1	4000
1	1	1	1	1	5000

Эти коды используются в большинстве современных 35-миллиметровых фотоаппаратов. Если на вашем фотоаппарате выдержка или тип пленки устанавливаются вручную, эти коды в нем не применяются. Если же коды все-таки считываются, присмотритесь к фотоаппарату, когда будете вставлять пленку. Вы увидите шесть металлических контактов, соответствующих квадратам с первого по шестой на кассете. Серебристые квадраты — это просто открытая металлическая поверхность кассеты, которая является проводником. Черные квадраты покрыты краской, которая не проводит электричество.

Электрическая схема фотоаппарата построена так, что ток подводится к первому квадрату на кассете (он всегда серебристый). Этот ток будет (или не будет) проведен пятью контактами на квадратах со второго по шестой в зависимости от того, окрашены они изолирующей краской или нет. Так, если ток

присутствует на контактах 4 и 5, но отсутствует на контактах 2, 3 и 6, в фотоаппарат вставлена пленка 400 ASA. При съемке выдержка будет установлена автоматически.

В бюджетных фотоаппаратах считываются только квадраты 0 и 3, а чувствительность пленки считается равной 50, 100, 200 или 400 единицам ASA.

Квадраты с восьмого по двенадцатый в большинстве аппаратов также не используются. В квадратах 8, 9 и 10 зашифровано число кадров на пленке, а квадраты 11 и 12 содержат сведения о том, черно-белая или цветная пленка, позитивная или негативная.

Вероятно, чаще всего вам приходилось сталкиваться с двоичными числами в коде UPC (Universal Product Code, универсальный код продукта), или просто штрихкоде, — наборе черных полос, который сегодня присутствует практически на любой упаковке. Штрихкод — наглядный символ повсеместного проникновения компьютеров в нашу жизнь.

Хотя у некоторых людей штрихкод вызывает приступы паранойи, это совершенно безобидная вещь, изобретенная для автоматизации розничной торговли и учета товаров. Со своей задачей он справляется вполне успешно. Благодаря ему, например, современные кассовые аппараты выдают покупателю чек, в котором подробно расписаны все его покупки, чего без штрихкода сделать нельзя.

Нас же в первую очередь интересует, что код UPC является двоичным, хотя на первый взгляд этого не скажешь. Давайте разберемся, как устроен штрихкод и как он работает.

Чаще всего встречается штрихкод, состоящий из нескольких цифр и 30 вертикальных полосок различной толщины, разделенных пустыми интервалами переменной толщины. В качестве примера рассмотрим штрихкод, нанесенный на банку куриного супа с вермишелью фирмы Campbell.

Сразу хочется разделить код UPC на тонкие и жирные полоски, узкие и широкие промежутки, и это действительно помогло бы разобраться в его структуре. Черные полоски и пустые промежутки штрихкода бывают различной ширины (всего четыре полоски).

Конечно, удобнее трактовать UPC как набор битов. Имейте в виду, что сканирующему устройству нет нужды просматривать штрихкод целиком, тем более прибор не может интерпретировать цифры в его основании, поскольку это потребовало бы применения сложной компьютерной технологии распознавания символов (Optical Character Recognition, OCR). Сканеру достаточно «увидеть» тонкий срез штрих-кода. Код UPC делают таким большим просто для того, чтобы кассиру легче было нацелить на него сканер. Срез, попадающий в сканер, выглядит следующим образом.

Почти как азбука Морзе, правда?

Сканируя эту информацию слева направо, компьютер присваивает бит 1 первой встреченной черной полоске и бит 0 первому промежутку. Следующие промежутки и штрихи считываются как последовательности одного, двух, трех или четырех битов в зависимости от ширины штриха или промежутка. В битовом представлении этот штрихкод выглядит так.

```
10100011010110001001100100011010001101000110101010111001011001101101100100111011001101000100101
```

Итак, весь UPC — просто последовательность из 95 бит. В данном случае их можно сгруппировать.

Биты	Значение
101	Левый шаблон-ограничитель
0001101	Цифры слева
0110001	
0011001	
0001101	
0011001	
01010	Центральный шаблон-разделитель
1110010	Цифры справа
1100110	
1101100	
1001110	
1100110	
1000100	
101	Правый шаблон-ограничитель

Первые три бита — всегда 101. Они называются левым шаблоном-ограничителем и нужны для того, чтобы настроить сканирующее устройство. По шаблону-ограничителю сканер определяет ширину штриха и промежутка, соответствующую одному биту. Иначе на всех упаковках код UPC пришлось бы делать одинакового размера.

За левым шаблоном-ограничителем следует шесть групп по семь бит в каждой. В них закодированы десятичные цифры от 0 до 9, в чем мы убедимся чуть позже. Затем идет 5-битовый центральный шаблон-разделитель — фиксированная группа битов (всегда 01010), используемая как встроенная контрольная система. Не найдя центрального шаблона-разделителя в нужном месте, сканер считает штрихкод неверным. В частности, так выявляют плохо пропечатанные или поддельные штрихкоды.

За центральным шаблоном-разделителем всегда идут еще шесть групп по семь бит каждая, а за ними — правый шаблон-ограничитель, всегда равный 101. Позже я расскажу, почему благодаря наличию правого шаблона-ограничителя штрихкод можно сканировать и в обратном направлении, то есть справа налево.

Всего в коде UPC зашифровано 12 десятичных цифр. Шесть из них закодированы с его левой стороны, по семь бит в каждой. Для их расшифровки применяется таблица.

Левосторонние коды

0001101 = 0	0110001 = 5
0011001 = 1	0101111 = 6
0010011 = 2	0111011 = 7
0111101 = 3	0110111 = 8
0100011 = 4	0001011 = 9

Обратите внимание: каждый 7-битовый код начинается с 0 и заканчивается 1. Встретив 7-битовый код, который начинается с 1, а заканчивается 0, сканер «понимает», что код UPC либо неверно прочитан, либо подделан. Кроме того, в каждом коде группы единиц встречаются лишь дважды. Это значит, что каждая десятичная цифра в коде UPC зашифрована двумя вертикальными штрихами.

Еще одна особенность кодов в этой таблице — нечетное количество единиц в каждом из них. Она также позволяет проверить корректность штрихкода — так называемый контроль четности (parity). Группа битов обладает *четным паритетом*, если в ней четное количество битов-единиц, и *нечетным паритетом*, если в ней нечетное количество битов-единиц.

Для расшифровки битов в правой части штрихкода применяется таблица.

Правосторонние коды

1110010 = 0	1001110 = 5
1100110 = 1	1010000 = 6
1101100 = 2	1000100 = 7
1000010 = 3	1001000 = 8
1011100 = 4	1110100 = 9

Эти коды дополняют коды из предыдущей таблицы. Там, где в левосторонних кодах был 0, теперь стоит 1, и наоборот. Правосторонние коды всегда начинаются с 1 и заканчиваются 0. Кроме того, число битов 1 в них всегда четное, что можно применять для контроля четности. Вот мы и готовы к расшифровке UPC. С помощью двух приведенных выше таблиц можно определить 11 цифр, зашифрованных на банке Campbell Soup.

0 51000 01251 7

Какая досада! Да, это те самые цифры, что напечатаны под штрихкодом. На самом деле это очень удобно: если сканер по каким-то причинам не смог прочитать код, кассир может ввести его вручную. Вы наверняка видели, как это бывает. Конечно, получается, что весь наш труд по расшифровке штрихкода был напрасным, к тому же никакой секретной информации мы так и не получили: просто 30 вертикальных штрихов превратились в 12 цифр.

Первая цифра (в данном случае 0) характеризует тип кода.

0 означает, что перед нами обычный код UPC. Если код нанесен на упаковку с товаром переменного веса, например с мясом или овощами, он начинается с 2. Товары со скидкой обозначаются цифрой 5.

Следующие пять цифр — код производителя. В нашем примере код 51000 соответствует компании Campbell Soup. Он есть на всех продуктах марки Campbell. За ними следует пятизначный (01251) код конкретного продукта этой компании, в нашем случае код банки с куриным супом. Код продукта информативен лишь в сочетании с кодом производителя. У куриного супа с вермишелью, выпущенного другой компанией, будет другой код продукта, в свою очередь код 01251 может значить нечто совершенно иное у другого производителя.

Вопреки распространенному мнению, в код UPC не включается цена товара. Информация о ней извлекается из компьютерной базы данных, используемой в кассовых аппаратах наряду со сканерами.

Последняя цифра (здесь — 7) называется *символом проверки остатка* и тоже используется для исключения ошибок. Чтобы проверить его на практике, присвоим букву каждой из первых 11 цифр (наш пример 0 51000 01251).

A BCDEF GHIJK

Теперь вычислим:

$$3 \times (A + C + E + G + I + K) + (B + D + F + H + J).$$

Вычтем результат из ближайшего большего числа, кратного десяти. Полученное число и будет символом проверки остатка для куриного супа с вермишелью Campbell:

$$3 \times (0 + 1 + 0 + 0 + 2 + 1) + (5 + 0 + 0 + 1 + 5) = 3 \times 4 + 11 = 23.$$

Ближайшее большее число, кратное десяти, — 30. Значит, 30 – 23 = 7.

Это число — результат проверки остатка — напечатано под штрихкодом и зашифровано в нем. Такая проверка — одна из форм избыточности. Если остаток, вычисленный по штрихкоду, не совпадет с остатком, явно указанным в нем, штрихкод будет сочтен недействительным.

Как правило, для представления десятичной цифры от 0 до 9 достаточно четырех бит. В штрихкодах используется по семь бит на цифру. Целыми 95 бит закодировано всего 11 значимых десятичных цифр. Если учесть, что UPC с обеих сторон ограничен пустым пространством, эквивалентным девяти нулевым битам, получается, что во всем штрихкоде 11 цифр закодировано 113 бит, по 10 бит на цифру!

Такая избыточная надежность отчасти требуется для защиты от ошибок. Код товара был бы не слишком полезен, если бы покупатель мог в два счета подправить его фломастером.

Кроме того, UPC удобен, поскольку его можно считывать в обоих направлениях. Если в первых считанных цифрах количество единиц четно, сканер распознает, что код читается справа налево. Для расшифровки правосторонних цифр компьютер использует следующую таблицу.

Правосторонние коды в обратном направлении

0100111 = 0	0111001 = 5
0110011 = 1	0000101 = 6
0011011 = 2	0010001 = 7
0100001 = 3	0001001 = 8
0011101 = 4	0010111 = 9

Вот таблица левосторонних кодов.

Левосторонние коды в обратном направлении

1011000 = 0	1000110 = 5
1001100 = 1	1111010 = 6
1100100 = 2	1101110 = 7
1011110 = 3	1110110 = 8
1100010 = 4	1101000 = 9

Эти 7-битовые коды отличаются от кодов, считываемых слева направо. Никакой путаницы не возникает.

Наше знакомство с кодами началось с азбуки Морзе, состоящей из точек, тире и промежутков между ними. Азбука Морзе, на первый взгляд, имеет мало общего с нулями и единицами, на деле же сводится именно к ним.

Вспомните устройство азбуки Морзе. Тире втрое длиннее точки. Точки и тире в пределах одной буквы разделены паузами продолжительностью в одну точку. Промежутки между буквами по длительности равны одному тире. Слова разделяются паузами в два тире.

Чтобы немного упростить анализ, допустим, что длина тире превышает длину точки не в три, а в два раза. Это означает, что точка соответствует одному единичному биту, а тире — двум единичным битам.

Паузы состоят из нулевых битов.

Вот простейшая таблица с азбукой Морзе из главы 2.

A	• —	H	• • • •	O	— — —	U	• • —
B	— • • •	I	• •	P	• — — •	V	• • • —
C	— • — •	J	• — — —	Q	— — • —	W	• — —
D	— • •	K	— • —	R	• — •	X	— • • —
E	•	L	• — • •	S	• • •	Y	— • — —
F	• • — •	M	— —	T	—	Z	— — • •
G	— — •	N	— •				

А вот та же таблица, преобразованная в биты.

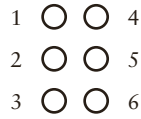
A	101100	H	101010100	O	1101101100	U	10101100
B	1101010100	I	10100	P	10110110100	V	1010101100
C	11010110100	J	101101101100	Q	110110101100	W	101101100
D	11010100	K	110101100	R	10110100	X	11010101100
E	100	L	1011010100	S	1010100	Y	110101101100
F	1010110100	M	1101100	T	1100	Z	11011010100
G	110110100	N	110100				

Обратите внимание: все коды начинаются с 1 и кончаются парой 0, представляющей паузу между буквами в пределах одного слова. Кодом пробела между словами является дополнительная пара 0. Таким образом, на азбуке Морзе фраза Hi there выглядит так.

Представив ее в битах, мы получим нечто очень похожее на срез штрихкода.

```
1010101001010000110010101010010010110100100000
```

В битовом выражении азбука Брайля гораздо проще азбуки Морзе. Шрифт Брайля является 6-битовым кодом. Каждый символ — набор из шести точек, каждая из которых может быть выпуклой или плоской. Как говорилось в главе 3, обычно точки нумеруются от 1 до 6.



Например, вот как записывается шрифтом Брайля слово code, где крайний левый бит соответствует первой позиции, а крайний правый — шестой.

Позже мы узнаем, что с помощью битов можно зашифровать не только коды товаров, чувствительность пленки, художественную ценность фильма, способ наступления британской армии или послание любимой женщине, но и любые слова, изображения, звуки, музыку и кино. В своей основе биты — это числа. Для представления информации в форме битов достаточно пересчитать количество доступных возможностей. Это количество определяет, сколько битов понадобится для того, чтобы присвоить каждой возможности уникальный номер.

Биты также играют важную роль в логике, находящейся на стыке философии и математики; ее главная цель — определение истинности или ложности некоего утверждения. Истину и ложь также можно обозначить через 1 и 0.