

# Понятие информация

- Информация — одно из самых фундаментальных понятий в современной науке, наряду с веществом, энергией, пространством, временем. А фундаментальное, т.е. первичное, понятие невозможно строго определить через вторичные, или производные понятия.

# Информация в быту

- Под **информацией в быту** понимают любые сведения об окружающем мире и протекающих в нем процессах, воспринимаемые человеком (с помощью органов слуха, зрения, осязания, обоняния, вкуса) или специальными устройствами.

# Информация в технике

- Под **информацией в технике** понимают любые сообщения, которые зафиксированы в виде знаков и могут передаваться в виде сигналов.

# Информация в теории управления

Под информацией в теории управления (менеджменте) понимают сообщения, уменьшающие существующую до этого неопределенность в той предметной области, к которой они относятся, и использующиеся для совершения активного действия, например, управленческого решения.

# ключевые атрибуты информации.

- 1. **Достоверность.** информация свободна от ошибок, чьей-либо пристрастности и отражает истинное положение дел. Часто организации применяют независимые источники информации, чтобы анализируя их, уменьшать фактор пристрастности в принимаемом решении или в распространяемой производной информации.
- 2. **Оперативность.** Доставка информации получателям в рамках необходимых временных границ. Например, вчерашняя газета сегодня, запоздавшая котировка акций. Своевременность просто означает, что адресат должен получить информацию, когда ему нужно.
- 3. **Актуальность**, т.е. важность, существенность для настоящего времени. Точная и своевременная информация может в то же время быть неактуальной, более того информация, актуальная для одного получателя, не обязательно актуальна для другого.
- 4. **Полнота.** Информация должна содержать все важные данные, которые ожидают от нее пользователи, и ее должно быть достаточно для понимания и принятия решения.
- 5. **Полезность.** Полезность (ценность) информации определяется по тем задачам, которые можно решить с ее помощью.
- 6. **Понятность** означает, что информация может быть представлена в ясном и понятном для потребителя формате. Потребитель информации – лицо, принимающее решение, должен как можно меньше времени тратить на дополнительные уточнения поступившей информации.

# Информация в узком и широком смысле

- в узком смысле информацией можно назвать сведения о предметах, фактах, понятиях некоторой предметной области.
- С середины XX века **информация** рассматривается в **широком смысле** как общенаучное понятие, включающее в себя как совокупность сведений об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, так и обмен сведениями между людьми, человеком и автоматом, автоматом и автоматом, обмен сигналами между живой и неживой природой, в животном и растительном мире, а также генетическую информацию.

информацию можно подразделить на:

1) **структурную** (или связанную) присущую объектам неживой и живой природы естественного или искусственного происхождения. Эти объекты (орудия труда, предметы быта, произведения искусства, научные теории и т.п.) возникают путем опредмечивания циркулирующей информации, то есть благодаря и в результате целенаправленных управленческих процессов;

2) **оперативную** (или рабочую), циркулирующую между объектами материального мира и используемую в процессах управления в живой природе, в человеческом обществе

# Данные, знания

- Сведения, полученные путем измерения, наблюдения, логических или арифметических операций, и представленные в форме, пригодной для постоянного хранения, передачи и обработки получили название **данные**.
- Совокупность полезной информации, правил и процедур ее обработки, необходимая для получения новой информации о какой-либо предметной области называют **знанием**.



# Свойства знаний

- 1. **Внутренняя интерпретируемость** знаний (понятность знания его носителю).
- 2. **Структурированность знаний.** Информационные единицы должны обладать гибкой структурой. Принцип «матрешки» – рекурсивная вложимость знаний. Возможность произвольного установления и перенастройки отношений (включения) между информационными единицами.
- 3. **Связность.** Отношения между элементами: структурные, функциональные, казуальные и семантические. Структурные задают иерархию, функциональные задают процедурную информацию, позволяющие находить одни элементы через другие, каузальные задают причинно-следственные связи, семантические охватывают все остальные виды отношений.
- 4. **Ассоциативность знаний** – наличие семантической метрики в сфере знаний. Отношение релевантности на множестве информационных единиц характеризует ситуационную близость элементов (силу ассоциативной связи). Позволяет находить знания, близкие к уже найденным.
- 5. **Активность знаний** – наличие у знаний побуждающей и направляющей функции, что фактически превращает знания в квазипотребности. Актуализации тех или иных действий способствуют имеющиеся в системе знания.

- Введение информации в научно-технический и хозяйственный оборот привело к необходимости ее количественной оценки, т.е. к введению меры сравнения. В простейшей комбинаторной форме эта мера была предложена Р. Хартли в 1928 году.

- **Пример 1.** Как определить, какая из двух монет фальшивая, если на вид они одинаковы, но известно, что фальшивая легче. Нет ничего проще, скажете вы. Проводим одно взвешивание на чашечных весах, и все становится ясно. Таким образом, до взвешивания у вас *была неопределенность* по поводу того, какая из монет фальшивая, а после взвешивания вы *сняли эту неопределенность*, получив *информацию*. Иными словами, вы получили сообщение в элементарном альтернативном выборе между двумя событиями («фальшивая – не фальшивая», «да - нет», «истина – ложь», «0–1»).

# Понятие бита

- бит – это и двоичный знак, и единица измерения количества информации, определяемая как ***количество информации в выборе с двумя взаимоисключающими равновероятными исходами.***

- **Пример 2.** А если монет 8? Тогда делим их на две равные части и взвешиваем их. Ту часть, которая легче, снова делим на две части и снова взвешиваем и т.д. За три взвешивания мы определим фальшивую монету.
- За три выбора мы уменьшили существующую неопределенность в 2, 4, 8 раз, получив таким образом 3 бита информации.

**Пример 3.** Перейдем от монет к картам. Пусть в колоде из 32 карт необходимо угадать определенную карту, например, туза пик. Для этого необходимо и достаточно получить ответы «да» и «нет» на **пять** вопросов. Вопросы, ответы на которые позволяют выбрать одну из альтернатив, называют двоичными, или бинарными. Ответами на эти вопросы мы уменьшаем неопределенность в 2, 4, 8, 16, 32 раз. В конце неопределенности не остается. Количество полученной информации равно 5 бит

Вопрос	Ответ	Бинарный ответ
1.Карта красной масти?	Нет	0
2.Трефы?	Нет	0
3.Одна из четырех старших?	Да	1
4.Одна из двух старших?	Да	1
5.Король?	Нет	0
Значит, задуманная карта была туз пик.		14

- В этих примерах процесс получения информации рассматривается как выбор одного сообщения из конечного наперёд заданного множества из  $n$  **равновероятных** сообщений. Легко подметить следующую закономерность: количество информации  $I$ , содержащееся в выбранном сообщении, определяется как двоичный логарифм  $n$ :

$$I = \lg(n)$$

- Справедливо утверждение Хартли: если во множестве  $X=\{x_1, x_2, \dots, x_n\}$  выделить произвольный элемент  $x_i \in X$ , то, чтобы его найти, необходимо получить не менее  $\lg(n)$  единиц информации.
- Недостаток формулы Хартли заключается в том, что она не учитывает **неравновероятность** различных рассматриваемых состояний.



# Вероятности отдельных букв в русском языке (с учетом пробела)

Буква	—	О	Е,Ё	А	И	Т	Н	С
$P_i$	0,175	0,090	0,072	0,062	0,062	0,053	0,053	0,045
Буква	Р	В	Л	К	М	Д	П	У
$P_i$	0,040	0,038	0,035	0,028	0,026	0,025	0,023	0,021
Буква	Я	Ы	З	Ь,Ъ	Б	Г	Ч	Й
$P_i$	0,018	0,016	0,016	0,014	0,014	0,013	0,012	0,010
Буква	Х	Ж	Ю	Ш	Ц	Щ	Э	Ф
$P_i$	0,009	0,007	0,006	0,006	0,004	0,003	0,003	0,002 <sub>17</sub>

# Частоты букв (в процентах) ряда европейских языков

Буква алфавита	Французск ий язык	Немецкий язык	Английский язык	Итальянский язык
А	7,68	5,52	7,96	11,12
В	0,80	1,56	1,60	1,07
С	3,32	2,94	2,84	4,11
Д	3,60	4,91	4,01	3,54
Е	17,76	19,18	12,86	11,63
F	1,06	1,96	2,62	1,15
G	1,10	3,60	1,99	1,73
Н	0,64	5,02	5,39	0,83
І	7,23	8,21	7,77	12,04
J	0,19	0,16	0,16	-
К	-	1,33	0,41	-
L	5,89	3,48	3,51	5,95
М	2,72	1,69	2,43	2,65
N	7,61	10,20	7,51	7,68
О	5,34	2,14	6,62	8,92
Р	3,24	0,54	1,81	2,66

- Для неравновероятных процессов американский учёный Клод Шеннон предложил (1948 г.) другую формулу определения количества информации, которая учитывает возможную неодинаковую вероятность сообщений во множестве сообщений.
- *Вероятность* – это численная мера достоверности случайного события, которая при большом числе испытаний близка к отношению числа случаев  $m$ , когда событие осуществилось (положительных исходов), к общему числу случаев  $n$ :

$$P_i = \lim_{n \rightarrow \infty} (m / n)$$

Например, если много раз подбрасывать монетку, то она упадёт орлом вверх примерно в половине случаев. Это значит, что вероятность выпадения орла равна 0,5, или 50%.

Вероятность любого события – это число, принадлежащее отрезку  $[0;1]$ . Событие с вероятностью 0 называют невозможным, а с вероятностью 1 – достоверным.

Свойство вероятностей:

$$\sum_{i=1}^n P_i = 1$$

- Можно представить что для того, чтобы получить какой то символ от источника сообщения нужно перебрать по крайней мере (с вероятностью близкой к 1)  $n$  символов, где  $n=1/p$ .  $p$  –вероятность появления символа. Чтобы получить из этих символов необходимый нам, нужно сделать двоичный логарифм переборов  $\lg(n)$ .

$$\longrightarrow K_i = \lg(1 / P_i) \text{ бит}$$

**мера Шеннона** количества информации (формула Шеннона).

- Если найти среднее значение количества таких переборов для всех символов получим формулу Шеннона:

$$H = \sum_{i=1}^n P_i I_i = \sum_{i=1}^n P_i \lg\left(\frac{1}{P_i}\right) = -\sum_{i=1}^n P_i \lg(P_i)$$

Эта величина получила название **информационная энтропия**, или энтропия источника сообщений.

Энтропия характеризует информационную мощность данного множества (ансамбля) сообщений и является мерой неопределенности, которая имеется в этом множестве.

$$H = \sum_{i=1}^n P_i I_i = \sum_{i=1}^n P_i \lg\left(\frac{1}{P_i}\right) = -\sum_{i=1}^n P_i \lg(P_i)$$

- Из формулы непосредственно вытекают свойства энтропии:
- энтропия заранее известного сообщения равна 0;
- во всех других случаях  $H > 0$ .

Чем больше энтропия системы, тем больше степень ее неопределенности. Поступающее сообщение полностью или частично снимает эту неопределенность. Поэтому *количество информации можно измерять тем, насколько понизилась энтропия системы после поступления сообщения*:

Уменьшая энтропию, мы получаем информацию – в этом и заключается смысл научного познания!

# Кодирование источника сообщений

- Как уже отмечалось, результат одного отдельного альтернативного выбора может быть представлен как 0 или 1. Тогда выбору всякого сообщения (события, символа т.п.) в массиве сообщений соответствует некоторая последовательность двоичных знаков 0 или 1, то есть *двоичное слово*. Это двоичное слово называют *кодировкой*, а множество кодировок источника сообщений – *кодом источника сообщений*.

# Кодирование – замена информационного слова на кодовое

Пример.

Информационное слово	Кодовое слово
000	0000
001	0011
010	0101
011	0110
100	1001
101	1010
110	1100
111	1111

- Если количество символов представляет собой степень двойки ( $n = 2^N$ ) и все знаки равновероятны  $P_i = (1/2)^N$ , то все двоичные слова имеют длину  $L=N=\text{ld}(n)$ . Такие коды называют **равномерными кодами**.
- Более оптимальным с точки зрения объема передаваемой информации является **неравномерное кодирование**, когда разным сообщениям в массиве сообщений назначают кодировку разной длины. Причем, часто происходящим событиям желательно назначать кодировку меньшей длины и наоборот, т.е. учитывать их вероятность.



### Кодирование словами постоянной длины

Буква	$a$	$b$	$c$	$d$	$e$	$f$	$g$
Кодирование	000	001	010	011	100	101	110

$\text{Id}(7) \approx 2,807$  и  $L=3$ .

. Проведем кодирование, *разбивая исходное множество знаков на равновероятные подмножества*, то есть так, чтобы при каждом разбиении суммы вероятностей для знаков одного подмножества и для другого подмножества были одинаковы. Для этого сначала расположим знаки в порядке уменьшения их вероятностей

Символ	Вероятность, $P_i$	Кодировка	Длина, $L_i$	Вероятность×Длина, $P_i \times L_i$
$a$	0,25	00	2	0,5
$e$	0,25	01	2	0,5
$f$	0,125	100	3	0,375
$c$	0,125	101	3	0,375
$b$	0,125	110	3	0,375
$d$	0,0625	1110	4	0,25
$g$	0,0625	1111	4	0,25

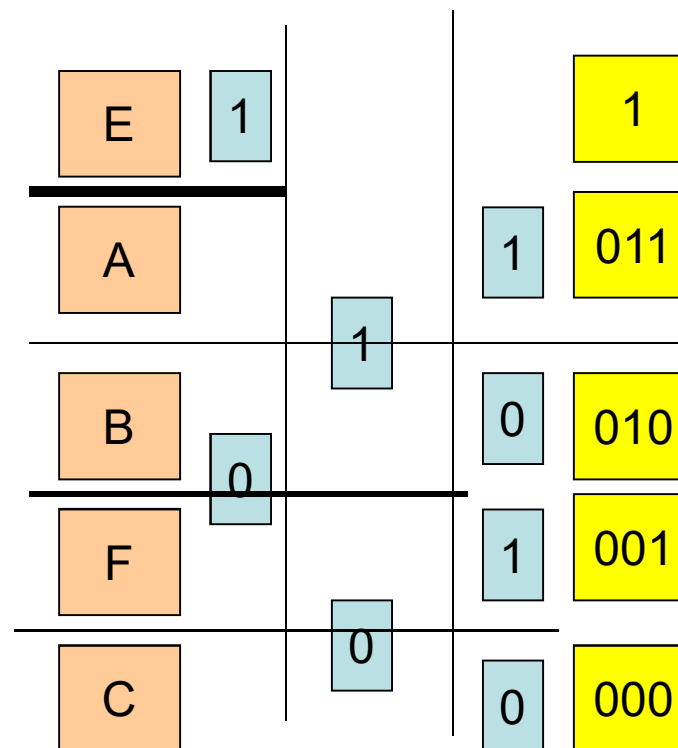
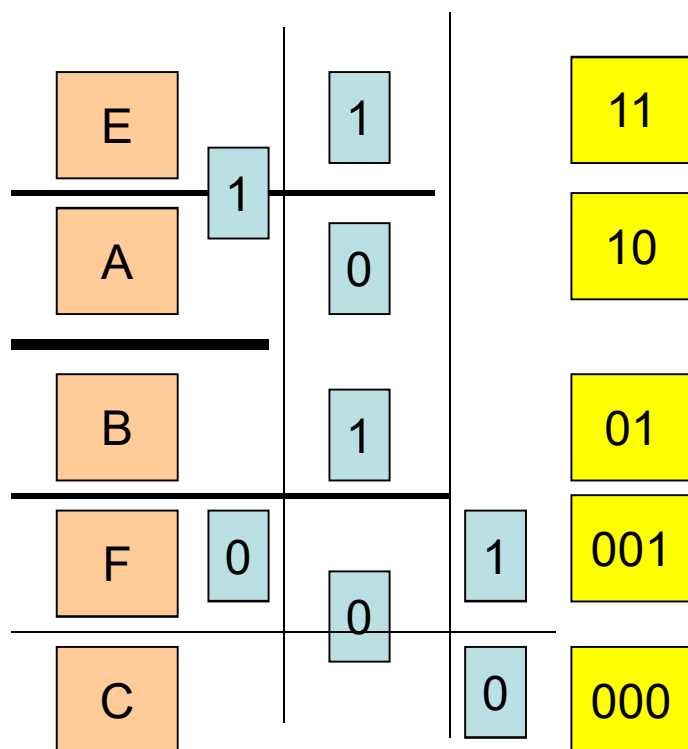
В общем случае **алгоритм построения оптимального кода Шеннона-Фано** выглядит следующим образом:

1. сообщения, входящие в ансамбль, располагаются в столбец по мере убывания вероятностей;
2. выбирается основание кода  $K$  (в нашем случае  $K=2$ );
3. все сообщения ансамбля разбиваются на  $K$  групп с суммарными вероятностями внутри каждой группы как можно близкими друг к другу.
4. всем сообщениям первой группы в качестве первого символа присваивается 0, сообщениям второй группы – символ 1, а сообщениям  $K$ -й группы – символ  $(K-1)$ ; тем самым обеспечивается равная вероятность появления всех символов  $0, 1, \dots, K$  на первой позиции в кодовых словах;
5. каждая из групп делится на  $K$  подгрупп с примерно равной суммарной вероятностью в каждой подгруппе. Всем сообщениям первых подгрупп в качестве второго символа присваивается 0, всем сообщениям вторых подгрупп – 1, а сообщениям  $K$ -ых подгрупп – символ  $(K-1)$ .
6. процесс продолжается до тех пор, пока в каждой подгруппе не окажется по одному сообщению.

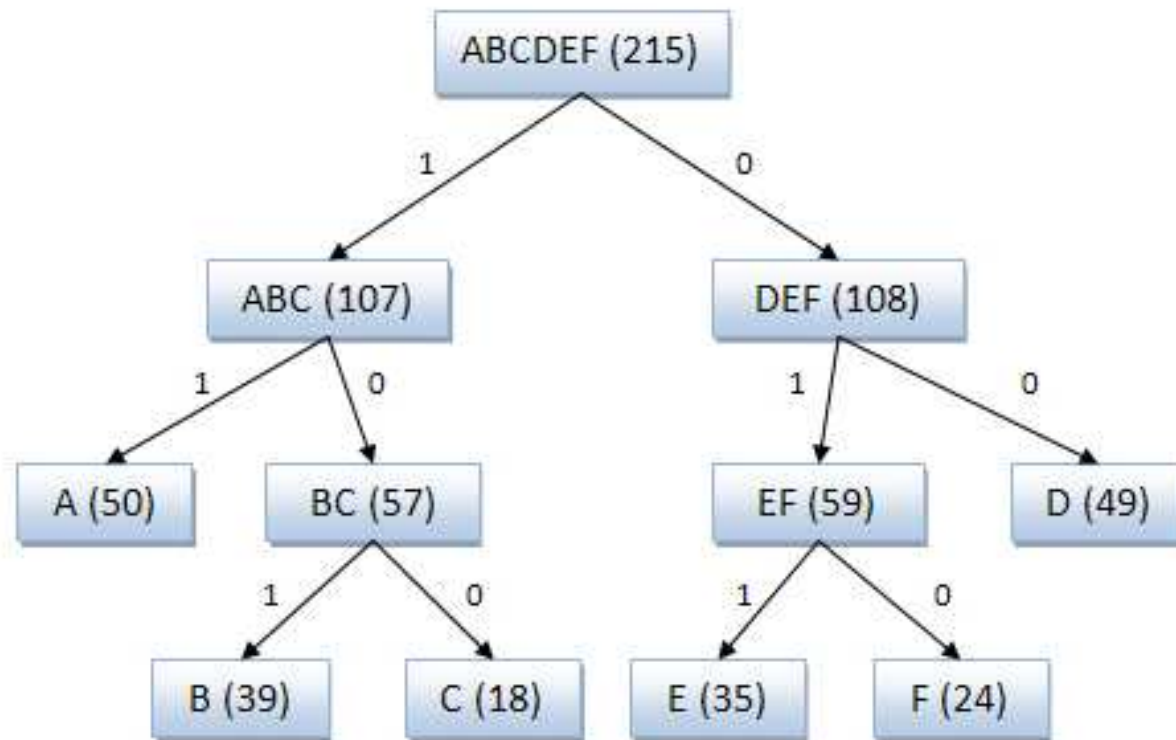
Символ	Вероятность, $P_i$
$A$	0,2
$E$	0,4
$F$	0,125
$C$	0,125
$B$	0,15



Символ	Вероятность, $P_i$
$E$	0,4
$A$	0,2
$B$	0,15
$F$	0,125
$C$	0,125



A (частота встречаемости 50)  
B (частота встречаемости 39)  
C (частота встречаемости 18)  
D (частота встречаемости 49)  
E (частота встречаемости 35)  
F (частота встречаемости 24)



Полученный код: A — 11, B — 101, C — 100, D — 00, E — 011, F — 010.

## Метод Хаффмана

Классический алгоритм Хаффмана на входе получает таблицу частот встречаемости символов в сообщении. Далее на основании этой таблицы строится дерево кодирования Хаффмана (H-дерево).

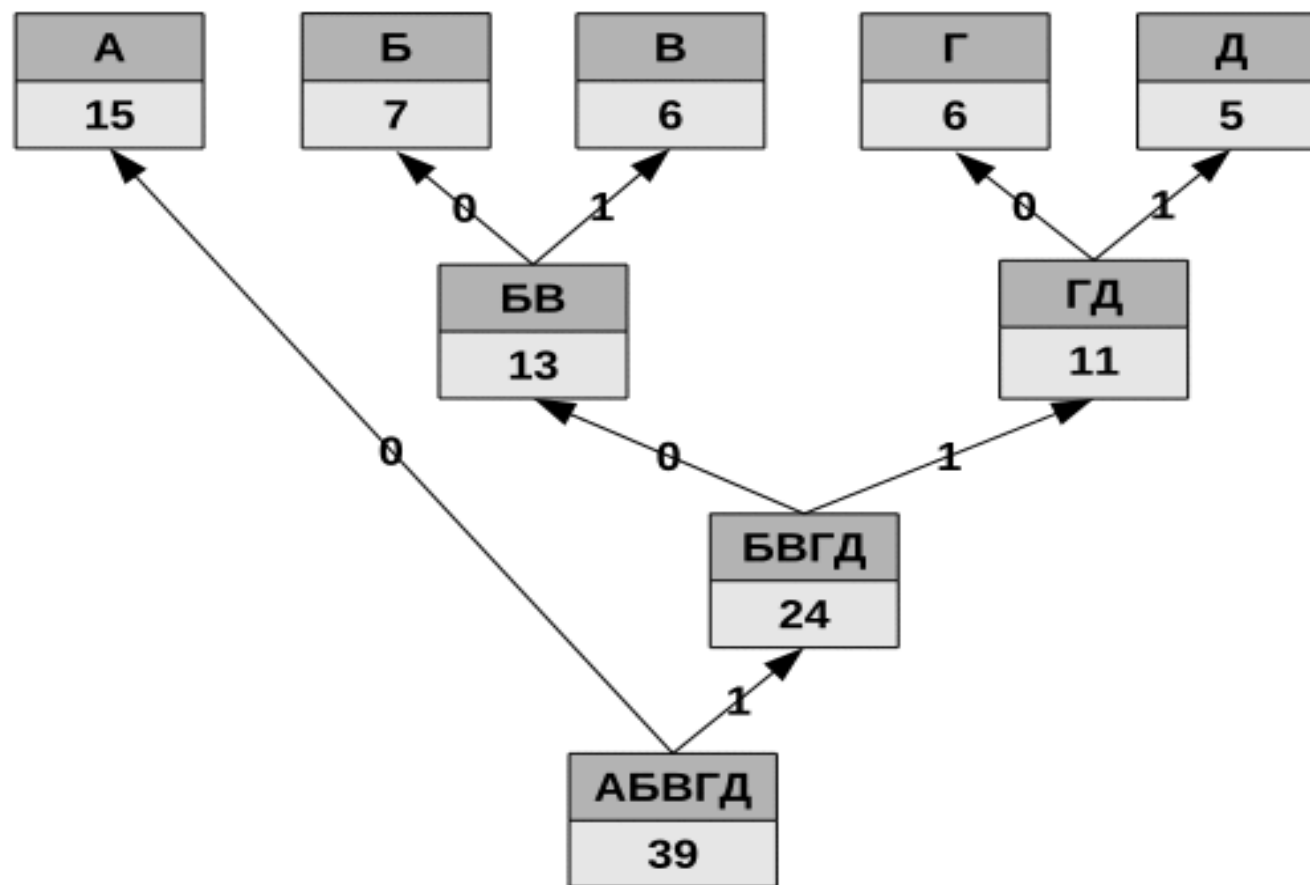
1. Символы входного алфавита образуют список свободных узлов. Каждый лист имеет вес, который может быть равен либо вероятности, либо количеству вхождений символа в сжимаемое сообщение.

2. Выбираются два свободных узла дерева с наименьшими весами. Создается их родитель с весом, равным их суммарному весу.

4. Родитель добавляется в список свободных узлов, а два его потомка удаляются из этого списка.

5. Одной дуге, выходящей из родителя, ставится в соответствие бит 1, другой — бит 0. Битовые значения ветвей, исходящих от корня, не зависят от весов потомков.

Шаги, начиная со второго, повторяются до тех пор, пока в списке свободных узлов не останется только один свободный узел. Он и будет считаться корнем дерева.



Итого:

А	Б	В	Г	Д
0	100	101	110	111

При **неравномерном** кодировании вводят среднюю длину кодировки, которая определяется по формуле

$$L = \sum_{i=1}^n P_i L_i$$

В общем же случае связь между средней длиной кодового слова  $L$  и энтропией  $H$  источника сообщений дает следующая **теорема кодирования** Шеннона:

имеет место неравенство  $L \geq H$ , причем  $L = H$  тогда, когда набор знаков можно разбить на точно равновероятные подмножества;

всякий источник сообщений можно закодировать так, что разность  $L - H$  будет как угодно мала.

Разность  $L - H$  называют **избыточностью кода** (мера бесполезно совершаемых альтернативных выборов).

следует не просто кодировать каждый знак в отдельности, а рассматривать вместо этого двоичные кодирования для  $nk$  групп по  $k$  знаков.

Тогда средняя длина кода  $i$ -го знака  $x_i$  вычисляется так:

$L = (\text{средняя длина всех кодовых групп, содержащих } x_i)/k.$

Символ	Вероятность	Кодировка	Длина	В×Д
<i>A</i>	0,7	0	1	0,7
<i>B</i>	0,2	10	2	0,4
<i>C</i>	0,1	11	2	0,2

Средняя длина слова:  $L = 0,7 + 0,4 + 0,2 = 1,3$ .

Среднее количество информации, содержащееся в знаке (энтропия):

$$H = 0,7 \times \lg(1/0,7) + 0,2 \times \lg(1/0,2) + 0,1 \times \lg(1/0,1) = 0,7 \times 0,515 + 0,2 \times 2,322 + 0,1 \times 3,322 = 1,1571.$$

Избыточность  $L - H = 1,3 - 1,1571 = 0,1429$ .



## Кодирование пар

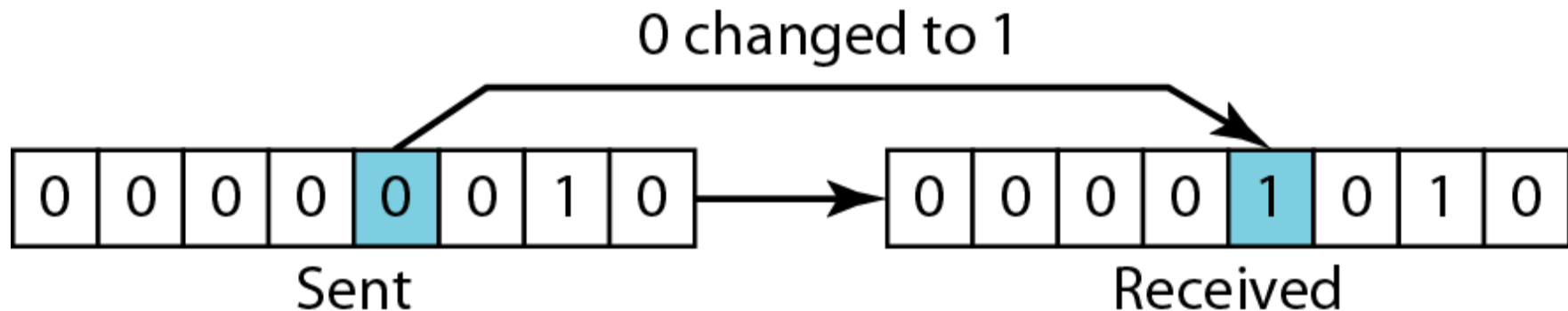
Пары	Вероятность	Кодировка	Длина	В×Д
<i>AA</i>	0,49	0	1	0,49
<i>AB</i>	0,14	100	3	0,42
<i>BA</i>	0,14	101	3	0,42
<i>AC</i>	0,07	1100	4	0,28
<i>CA</i>	0,07	1101	4	0,28
<i>BB</i>	0,04	1110	4	0,16
<i>BC</i>	0,02	11110	5	0,10
<i>CB</i>	0,02	111110	6	0,12
<i>CC</i>	0,01	111111	6	0,06
Средняя длина кодовой группы из 2-х символов равна				2,33

Средняя длина кода одного знака равна  $2,33/2=1,165$  – уже ближе к энтропии. Избыточность равна  $L - H = 1,165 - 1,1571 \approx 0,008$ .

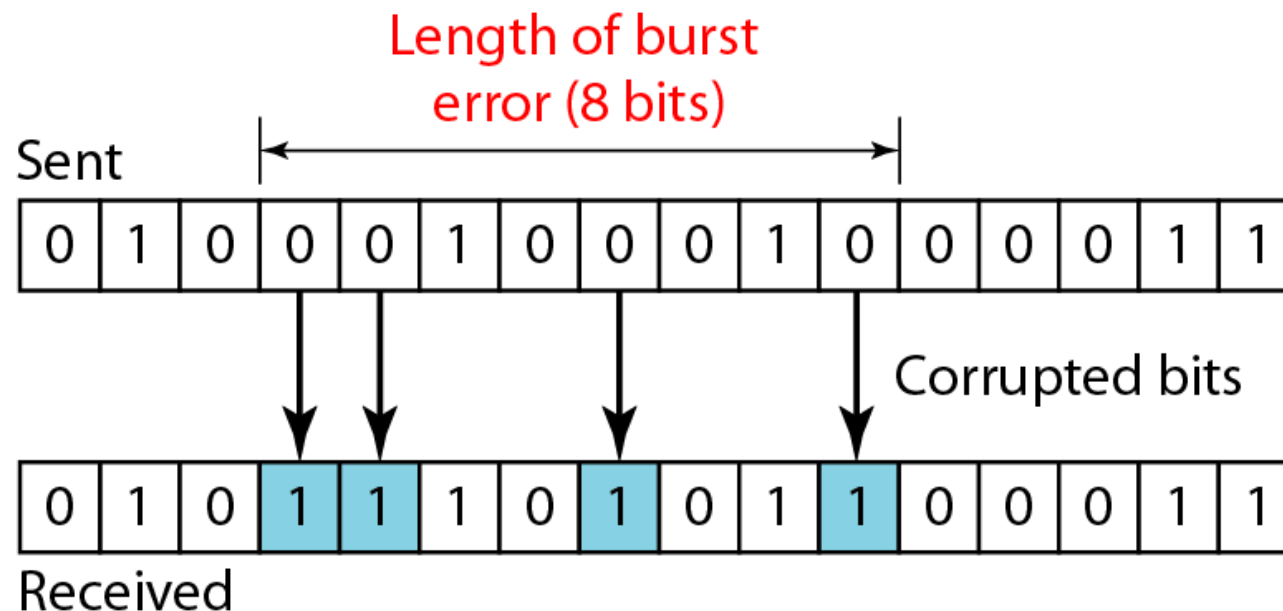
# Помехоустойчивое кодирование

## Введение избыточности

Ошибка в одном разряде



Пакет ошибок длины 8



# Модель ошибки

- *Ошибка* – замена в двоичном сообщении 0 на 1 и\или наоборот, замена 1 на 0
- Пример:  
ИСХОДНОЕ  
СЛОВО: 00010100
- ОШИБОЧНЫЕ  
СЛОВА:  
00**1**10100,  
000**0**100,  
00**101**100

Стирающий канал

1111101 ->111101

Канал со вставками

1111101->**0**1111101

Расстояние Хэмминга между двумя словами есть число разрядов, в которых эти слова различаются

- 1. Расстояние Хэмминга  $d(000, 011)$  есть 2*
- 2. Расстояние Хэмминга  $d(10101, 11110)$  равно 3*

## Декодирование – исправление ошибки, если она произошла

- Множество кодовых слов  
{00000, 01101, 10110, 11011}
- Если полученное слово 10000, то декодируем в «ближайшее» слово 00000
- Если полученное слово 11000 – то только обнаружение ошибки, так как два варианта: 11000 – в 00000 или 11000 – в 11011

-

# Самокорректирующиеся коды

- Коды, в которых возможно автоматическое исправление ошибок, называются **самокорректирующимися**. Для построения самокорректирующегося кода, рассчитанного на исправление одиночных ошибок, одного контрольного разряда недостаточно.

количество контрольных разрядов  $k$  должно быть выбрано так, чтобы удовлетворялось неравенство:

$$2^k \geq k + m + 1$$

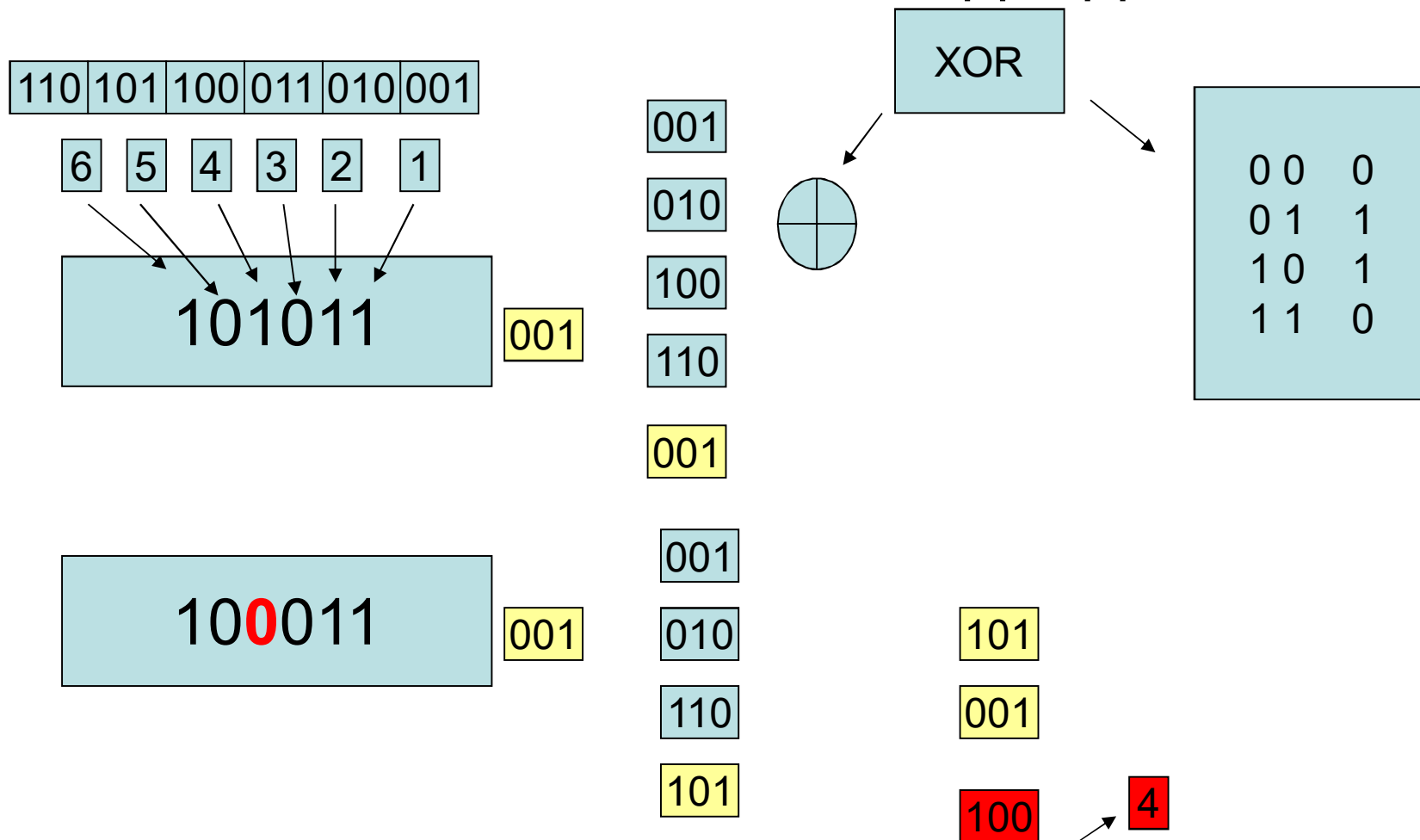
$$k \geq \log_2(k + m + 1)$$

где  $m$  — количество основных двоичных разрядов кодового слова

Минимальные значения  $k$  при заданных значениях  $m$

Диапазон $m$	$k_{\min}$
1	2
2-4	3
5-11	4
12-26	5
27-57	6

# Код Хэмминга, восстановление одного искажения или обнаружение двойного, неклассический подход





Для Примера рассмотрим классический код Хемминга  
Сгруппируем проверочные символы следующим образом:

$$r_1 = i_1 \oplus i_2 \oplus i_3$$

$$r_2 = i_2 \oplus i_3 \oplus i_4$$

$$r_3 = i_1 \oplus i_2 \oplus i_4$$

Получение кодового слова выглядит следующим образом:

$$(i_1 \ i_2 \ i_3 \ i_4) \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = (i_1 \ i_2 \ i_3 \ i_4 \ r_1 \ r_2 \ r_3)$$

# Декодирование

На вход декодера поступает кодовое слово

$$V = (i'_1, i'_2, i'_3, i'_4, r'_1, r'_2, r'_3)$$

В декодере в режиме исправления ошибок строится последовательность синдромов:

$$S_1 = r_1 \oplus i_1 \oplus i_2 \oplus i_3$$

$$S_2 = r_2 \oplus i_2 \oplus i_3 \oplus i_4$$

$$S_3 = r_3 \oplus i_1 \oplus i_2 \oplus i_4$$

Получение синдрома выглядит следующим образом:

$$(i_1 \ i_2 \ i_3 \ i_4 \ r_1 \ r_2 \ r_3) \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = (S_1 \ S_2 \ S_3)$$

$$\sim i_2 + i_2 = 1$$

Нули в синдроме показывают отсутствие ошибок, отличный от нуля код соответствует какой-либо единичной ошибке, например для 111, это ошибка в  $i_2^4$

# Получение кода хэмминга для кодов большей длины

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	1	1	1	0	1	
1	X		X		X		X		X		X		X		X		X		X		X	1
0		X	X			X	X			X	X			X	X			X	X			2
1				X	X	X	X					X	X	X	X					X	X	4
0								X	X	X	X	X	X	X	X							8
0																X	X	X	X	X	X	16

Каждую последовательность суммируем по модулю 2 (операция xor), получая код:

$$0+0+1+0+0+0+0+1+1+1+1=1$$

$$0+0+0+0+1+0+0+1+1+1=0$$

$$0+1+0+0+0+0+0+1+0+1=1$$

$$0+0+1+0+0+0+0+1=0$$

$$0+1+1+1+0+1=0$$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
1	0	0	1	1	0	0	0	0	1	0	0	0	0	1	0	1	1	1	0	1	

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
1	0	0	1	1	0	0	0	0	1	<del>1</del>	0	0	0	1	0	1	1	1	0	1	
X		X		X		X		X		X		X		X		X		X		X	1
	X	X			X	X			X	X			X	X			X	X			2
			X	X	X	X					X	X	X	X					X	X	4
							X	X	X	X	X	X	X	X							8
															X	X	X	X	X	X	16

$$\begin{aligned}
 1+0+1+0+0+1+0+1+1+1+1 &= 11 \\
 0+0+0+0+1+1+0+1+1+1 &= 12 \\
 1+1+0+0+0+0+0+1+0+1 &= 04 \\
 0+0+1+1+0+0+0+1 &= 18 \\
 0+1+1+1+0+1 &= 016 \\
 &11
 \end{aligned}$$

# Понятие системы счисления

Система счисления — это способ записи чисел с помощью заданного набора специальных знаков.

# Два вида систем счисления

## ПОЗИЦИОННЫЕ

В позиционных системах счисления вес каждой цифры **изменяется в зависимости от ее положения** (позиции) в последовательности цифр, изображающих число

Пример: В десятичном числе 757,7 первая семерка означает 7 сотен, вторая – 7 единиц, третья – 7 десятых долей единицы.

## НЕПОЗИЦИОННЫЕ

В непозиционных системах вес цифры (т.е. тот вклад, который она вносит в значение числа) **не зависит от ее позиции** в записи числа.

Пример: В римской системе счисления в числе XXXII (тридцать два) вес цифры X в любой позиции равен просто десяти.

Степень двойки	Десятичное	Восьмеричное
$2^0$	1	1
$2^1$	2	2
$2^2$	4	4
$2^3$	8	10
$2^4$	16	20
$2^5$	32	40
$2^6$	64	100
$2^7$	128	200
$2^8$	256	400
$2^9$	512	1000
$2^{10}$	1024	2000
$2^{11}$	2048	4000
$2^{12}$	4096	10000

при работе с двоичными кодами удобны недесятичные системы счисления, а основания кратные степеням двойки.

**Любая позиционная система счисления характеризуется своим основанием**

Любое двоичное число, состоящее из 1 с несколькими нулями, является степенью двойки. Показатель степени равен числу нулей.

Таблица степеней двойки демонстрирует это правило наглядно.

Степень двойки	Десятичное	Восьмеричное	Четверичное	Двоичное
$2^0$	1	1	1	1
$2^1$	2	2	2	10
$2^2$	4	4	10	100
$2^3$	8	10	20	1000
$2^4$	16	20	100	10000
$2^5$	32	40	200	100000
$2^6$	64	100	1000	1000000
$2^7$	128	200	2000	10000000
$2^8$	256	400	10000	100000000
$2^9$	512	1000	20000	1000000000
$2^{10}$	1024	2000	100000	10000000000

Переводимое число необходимо записать в виде суммы произведений цифр числа на основание системы счисления в степени, соответствующей позиции цифры в числе.

5 4 3 2 1 0 -1 -2

$$111000.11_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2}$$

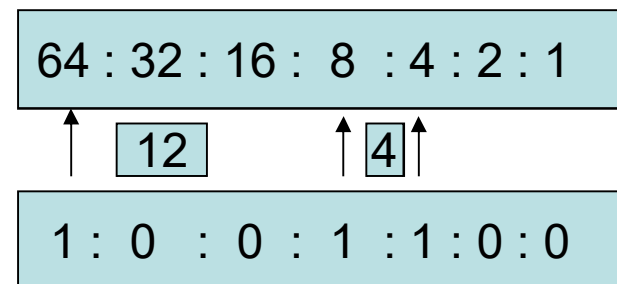
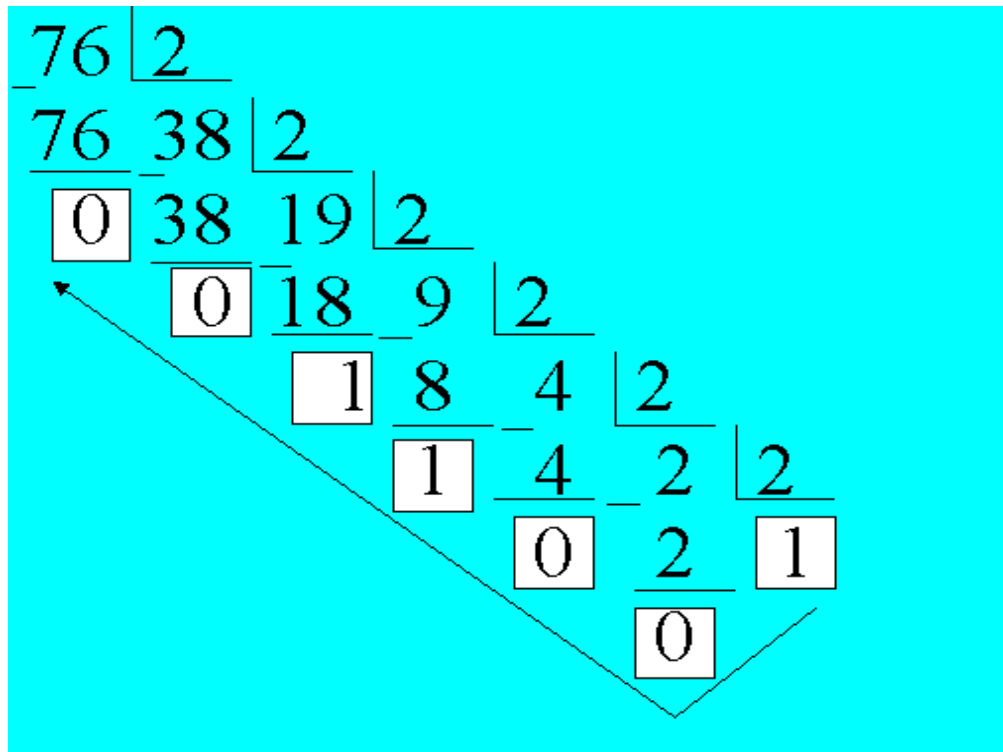
=

$$= 32 + 16 + 8 + \frac{1}{2} + \frac{1}{4} =$$

$$= 56,75_{10}$$



$1001100_2$



$$\begin{array}{r}
 \times 0,375 \\
 \hline
 \times \boxed{0},750 \\
 \hline
 \times \boxed{1},500 \\
 \hline
 \times \boxed{1},000 \\
 \hline
 \end{array}$$

↓

### Дробная часть

получается из целых частей (0 или 1) при ее последовательном умножении на 2 до тех пор, пока дробная часть не обратится в 0 или получится требуемое количество знаков после разделительной точки.

$$0,375_{10} = 0,011_2$$

# Пример перевода из восьмиричной системы счисления

2 1 0 -1

$$\begin{aligned} 421.5_8 &= 4 \cdot 8^2 + 2 \cdot 8^1 + 1 \cdot 8^0 + 5 \cdot 8^{-1} = \\ &= 256 + 16 + 1 + 5/8 = \\ &= 273,625_{10} \end{aligned}$$

# Пример перевода из шестнадцатиричной системы счисления

1 0 -1

$$\begin{aligned} A7.C_{16} &= 10 \cdot 16^1 + 7 \cdot 16^0 + 12 \cdot 16^{-1} = \\ &= 160 + 7 + 12/16 = \\ &= 167,75_{10} \end{aligned}$$

# Запись в десятичной, двоичной, восьмеричной и шестнадцатеричной системах счисления первых двух десятков целых чисел

10 - я	2 - я	8 - я	16 - я
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9

10 - я	2 - я	8 - я	16 - я
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13

# Примеры перевода из двоичной системы счисления в восьмеричную

$100110111.001_2 = 100\ 110\ 111.001_2$

$100110111.001_2 = 4\ 6\ 7.1_8$

$10100101110.11_2 = 010\ 100\ 101\ 110.110_2$

$10100101110.11_2 = 2\ 4\ 5\ 6.6_8$

# Перевод из восьмеричной системы счисления в двоичную

Такой перевод осуществляется путем подстановки: каждая 8-ричная цифра заменяется на соответствующие ей три двоичных.

$$74.6_8 = 111\ 100.110_2$$

$$310.5_8 = 011\ 001\ 000.101_2$$

# Примеры перевода из двоичной системы счисления в шестнадцатеричную

$100110111.001_2 = 0001\ 0011\ 0111.0010_2$

$100110111.001_2 = 1\ 3\ 7.2_{16}$

$10100101110.11_2 = 0101\ 0010\ 1110.1100_2$

$10100101110.11_2 = 5\ 2\ E.C_{16}$



# Перевод из шестнадцатеричной системы в двоичную

Такой перевод осуществляется путем обратной подстановки: каждая 16-ричная цифра заменяется на соответствующие ей четыре двоичных.

$$C1B.3_{16} = 1100 \ 0001 \ 1011. \ 0011_2$$

$$AF0.1_{16} = 1010 \ 1111 \ 0000. \ 0001_2$$

## Кодировки символов

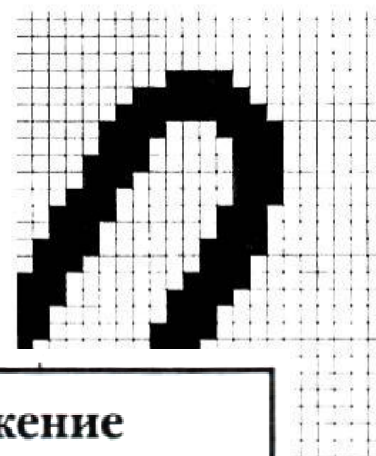
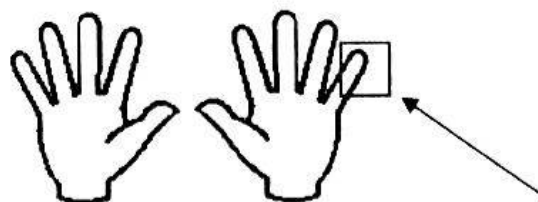
Двоичный код	Десятичный код	КОИ8	CP1251	CP866	Mac	ISO
00000000	0					
.....						
00001000	8	Удаление последнего символа (клавиша Backspace)				
.....						
00001101	13	Перевод строки (клавиша Enter)				
.....						
00100000	32	Пробел				
00100001	33	!				
.....						
01011010	90	Z				
.....						
01111111	127	П				
10000000	128	.	ь	А	А	к
.....						
11000010	194	б	В	.	.	т
.....						
11001100	204	л	М			ь
.....						
11011101	221	щ	Э	-	Ё	н
.....						
11111111	255	ь	я	Нераздел. пробел	Нераздел. пробел	п

# Двоичное кодирование графической информации



В простейшем случае (черно-белое изображение без градаций серого цвета). Каждая точка экрана может иметь лишь два состояния – «черная» или «белая», т.е. для хранения ее состояния необходим 1 бит.

**Восьмицветная палитра**  
(на основе базовых цветов)

R	G	B	Цвет
0	0	0	черный
0	0	1	синий
0	1	0	зеленый
0	1	1	голубой
1	0	0	красный
1	0	1	розовый
1	1	0	коричневый
1	1	1	белый



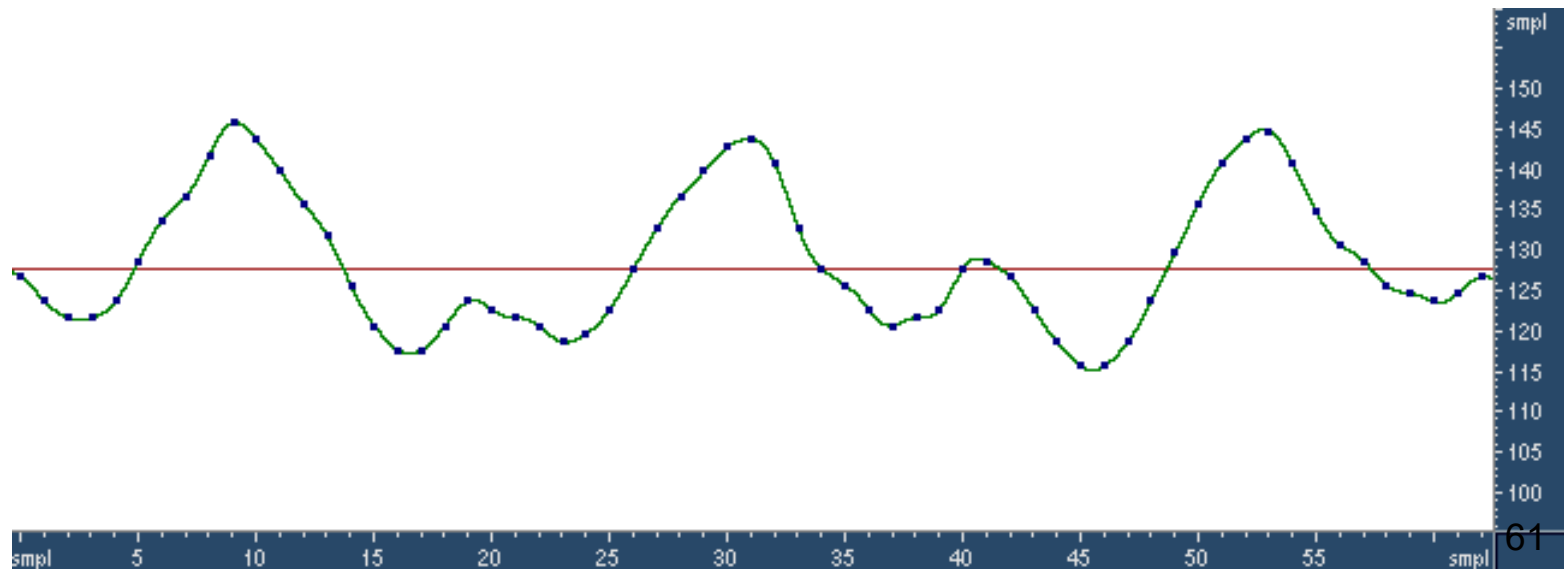
**Монохромное изображение**  
(черно-белый монитор)

	0	1 бит видеопамяти
	1	

R	G	B	Цвет
255	0	0	Красный
0	0	255	Синий
0	255	0	Зеленый
0	0	0	Черный
255	255	255	Белый

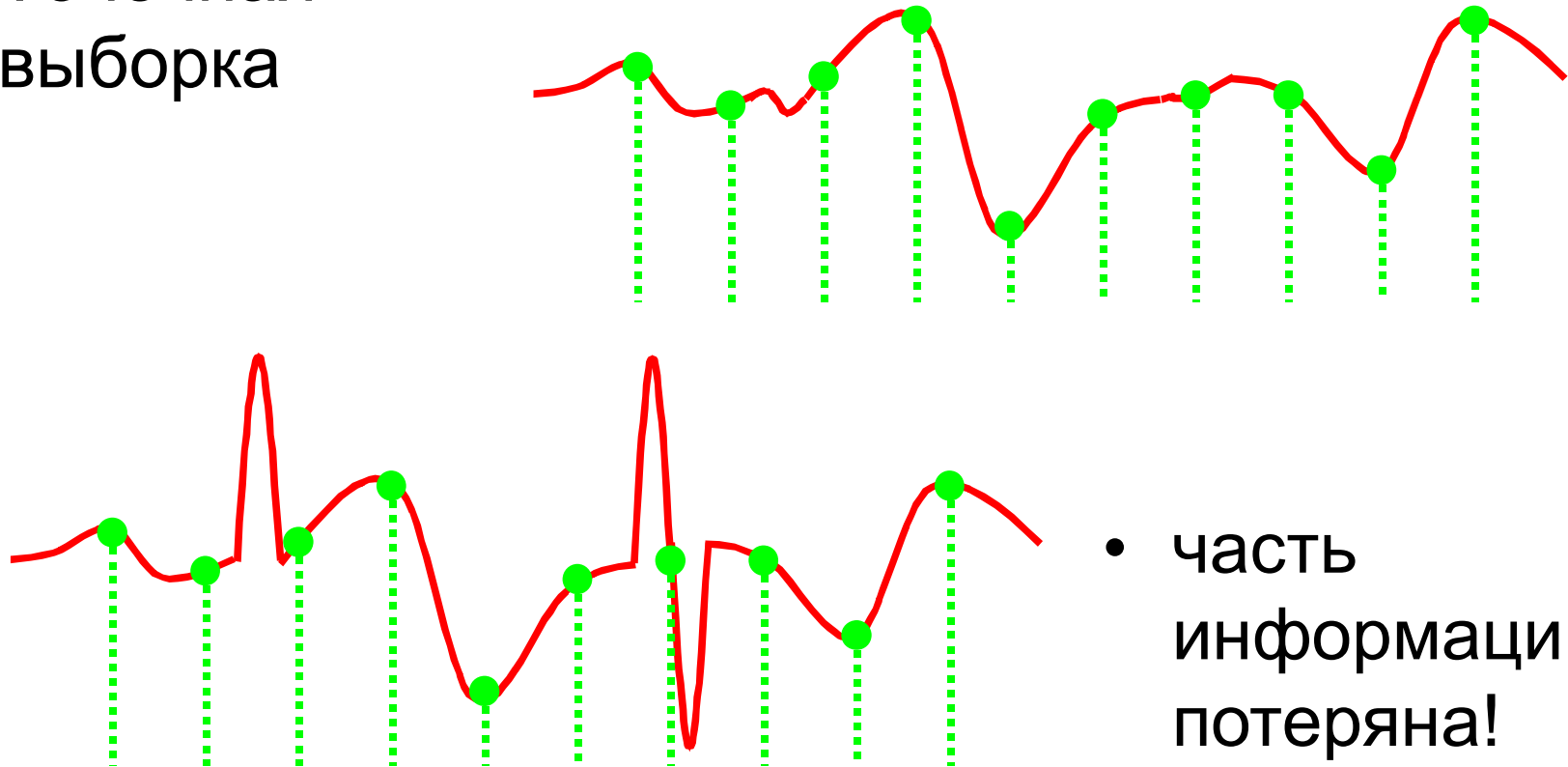
# Мультимедийная информация

- Звук
  - Запись и оцифровка
  - Частота и разрядность дискретизации
  - Артефакты оцифровки



# Выборка

- Точечная выборка



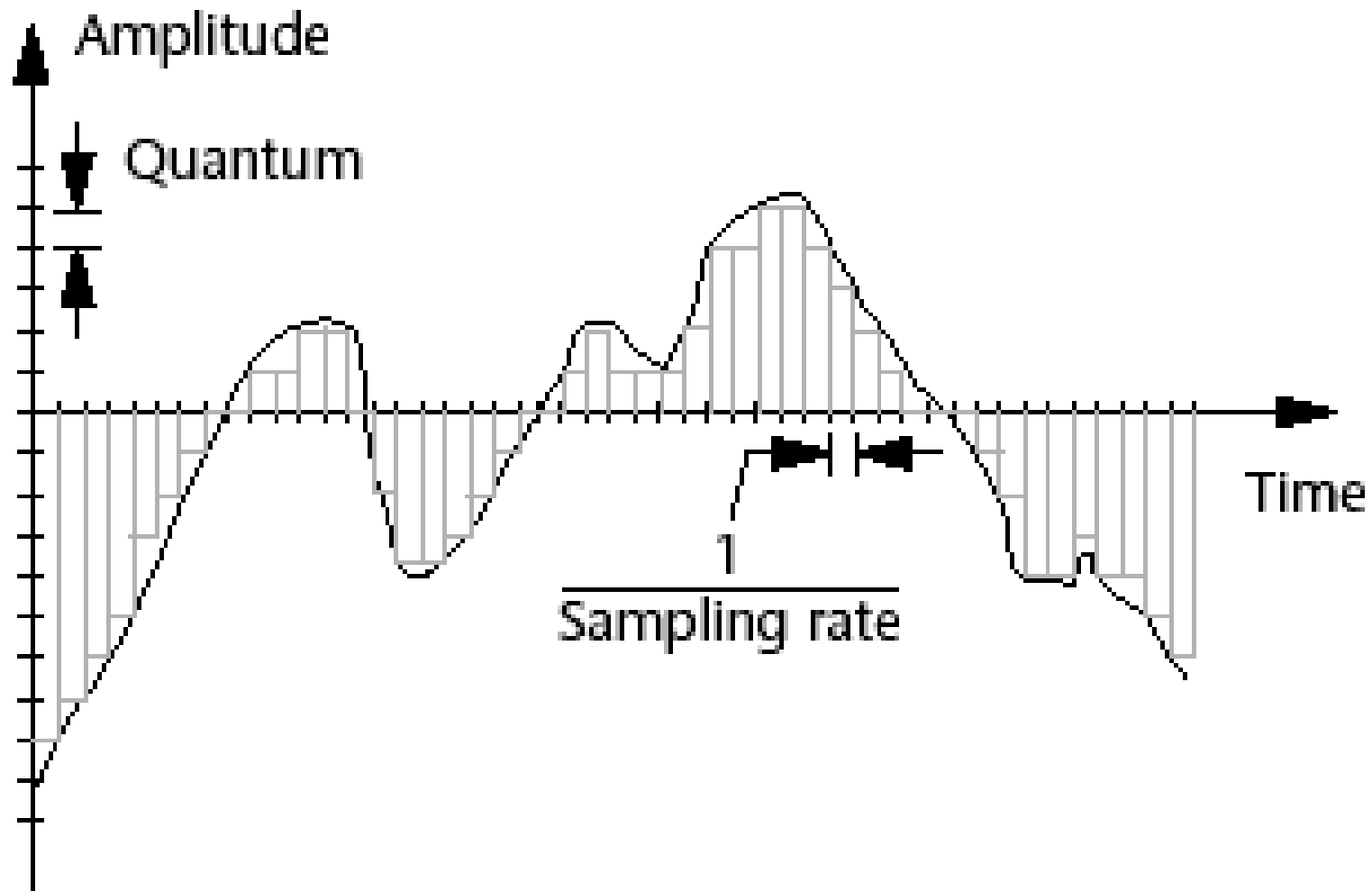
- часть информации потеряна!

# Квантование

Определение: Преобразование чисел высокой точности в числа низкой точности

- *Зачем?*
  - Экономия памяти
  - Вывод на двоичные устройства
- *Как?*
  - Минимизация ошибки (скорее, ошибки восприятия)

# Дискретизация и квантование звуковой волны





# Скорость передачи

- Пример
- 256 уровней квантования
- Значит для кодирования надо 8 бит
- Частота дискретизации 8000 Гц, значит 8000 раз в секунду делаются отчеты
- Скорость передачи –  $8000 * 8 = 64$  кбит/с

Количество бит \* Частоту дискретизации [бит/с]

# Целые числа со знаком

Для хранения **целых чисел со знаком** отводится две ячейки памяти (16 битов).

**Старший разряд** числа определяет его знак.  
Если он равен 0, число положительное,  
если 1, то отрицательное.

$$51_{10} = 110011_2$$

0	0	1	1	0	0	1	1
---	---	---	---	---	---	---	---

$$-51_{10} = -110011_2$$

1	0	1	1	0	0	1	1
---	---	---	---	---	---	---	---

Такое представление чисел в компьютере называется **прямым кодом**.

# Дополнительный код

- Число полученное путем вычитания из числа с числом разрядов больше на 1, и со значением 1 в старшем разряде и 0 младших. Пример 1000.
- Для числа 70, дополнительный код  $100-70=30$
- наиболее распространённый способ представления отрицательных целых чисел в компьютерах. Он позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и беззнаковых чисел, чем упрощает архитектуру ЭВМ.

- $59-41 = ? \quad 18$
- Доп код 41,  $100-41 = 59$
- Можно представить как:
- $59-(100-59) = 59+59 - 100 = 118-100 = 18$
- В двоичной системе
- Доп кол получается как
- $10000-1001\dots$
- Что такое 10000, это  $1111+1$
- $1111-1001$  получается путем инвертирования 0110
- Остается добавить 1, чтобы получить доп код
- 0111

# Целые числа со знаком

Для представления отрицательных целых чисел используется **дополнительный код**.

Алгоритм получения дополнительного кода отрицательного числа:

1. Число записать **прямым кодом** в  $n$  двоичных разрядах.
  1. Получить **обратный код** числа, для этого значения всех битов инвертировать, кроме старшего разряда.
  2. К полученному обратному коду **прибавить единицу**.

Представить число  $-2014_{10}$  в двоичном виде в шестнадцатибитном представлении в формате целого со знаком.

Прямой код	$-2014_{10}$	10000111 11011110 <sub>2</sub>
Обратный код	Инвертирование	11111000 00100001 <sub>2</sub>
	Прибавление единицы	11111000 00100001 <sub>2</sub> 00000000 00000001 <sub>2</sub>
Дополнительный код		11111000 00100010 <sub>2</sub>

# Целые числа со знаком

**Пример 1.** Найти разность  $13_{10} - 12_{10}$  в восьмибитном представлении.

	$13_{10}$	$-12_{10}$
Прямой код	00001101	10001100
Обратный код	–	11110011
Дополнительный код	–	11110100

$$\begin{array}{r} 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1 \\ +\ 1\ 1\ 1\ 1\ 0\ 1\ 0\ 0 \\ \hline 10\ 0\ 0\ 0\ 0\ 0\ 0\ 1 \end{array}$$

Так как произошел перенос из знакового разряда, первую единицу отбрасываем, и в результате получаем 00000001.

# Целые числа со знаком

**Пример 2.** Найти разность  $8_{10} - 13_{10}$  в восьмибитном представлении.

	$8_{10}$	$-13_{10}$
Прямой код	00001000	10001101
Обратный код	–	11110010
Дополнительный код	–	11110011

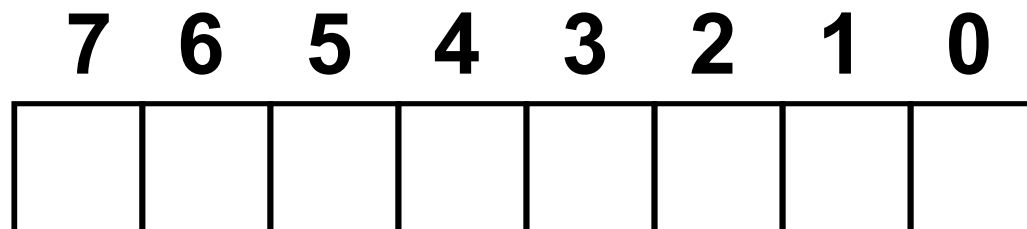
$$\begin{array}{r}
 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0 \\
 +\ 1\ 1\ 1\ 1\ 0\ 0\ 1\ 1 \\
 \hline
 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1
 \end{array}$$

# Пример 1

---

Представить число  $21_{10}$  и  $-21_{10}$  в  
однобайтовой разрядной сетке.

1. Переведем число  $21_{10}$  в двоичную систему счисления.  $21_{10} = 10101_2$ .
2. Нарисуем восьмиразрядную сетку (1 байт = 8 бит).





# Пример 1

---

**3.** Впишем число, начиная с младшего разряда.

			1	0	1	0	1
--	--	--	---	---	---	---	---

**4.** Заполним оставшиеся разряды нулями.

0	0	0	1	0	1	0	1
---	---	---	---	---	---	---	---

# Пример 1

---

**5.** Инвертируем

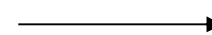
1	1	1	0	1	0	1	0
---	---	---	---	---	---	---	---

**6.** Прибавляем 1

знак	1	1	1	0	1	0	1	1
------	---	---	---	---	---	---	---	---

Проверка

0	0	0	1	0	1	0	0
0	0	0	1	0	1	0	1



21

- Расширение числа, например, от байта до слова (два байта) или до двойного слова (четыре байта) делается путем добавления единиц слева, если это число отрицательное в дополнительном коде и нулей если число в прямом коде

1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1

# Вещественные числа

Вещественные числа хранятся и обрабатываются в компьютере в формате *с плавающей запятой*, использующем экспоненциальную форму записи чисел.

$$A = M \times q^n$$

$M$  – мантисса числа (правильная отличная от нуля дробь),

$q$  – основание системы счисления,

$n$  – порядок числа.

Диапазон ограничен максимальными значениями  $M$  и  $n$ .

# Вещественные числа

Например,  $123,45 = 0,12345 \cdot 10^3$

Порядок указывает, на какое количество позиций и в каком направлении должна сместиться десятичная запятая в мантиссе.

Число в формате с плавающей запятой может занимать в памяти 4 байта (*обычная точность*) или 8 байтов (*двойная точность*).

При записи числа выделяются разряды для хранения знака мантиссы, знака порядка, порядка и мантиссы.

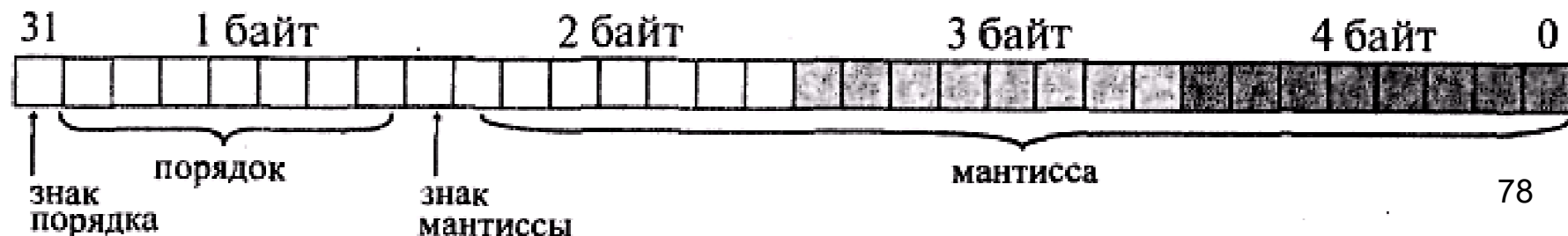
Мантисса **M** и порядок **n** определяют диапазон изменения чисел и их точность.

# Кодирование вещественных чисел

---

Число в форме с плавающей точкой занимает в памяти компьютера **четыре** (число обычной точности) байта или **восемь** (число двойной точности) байта.

Для записи чисел в разрядной сетке выделяются разряды для знака порядка и мантиссы, для порядка и для мантиссы.



## Пример 3

---

Представить число  $250,1875_{10}$  в формате с плавающей точкой в 4-байтовой разрядной сетке:

**1.** Переведем число в двоичную систему счисления с 23 значащими цифрами:  
 $250,1875_{10} = 11111010,0011000000000000_2;$

**2.** Нормализуем мантиссу:  
 $11111010,0011000000000000 =$   
 $0,111110100011\ 000000000000 \cdot 10^{1000};$

# Пример 3

---

**3.**  $0,111110100011000000000000 \cdot 10^{1000};$   
(мантисса положительная)  
(порядок положительный)

**4.** Запишем число в 32-разрядной сетке:

