

# ВМСИС

## Лекция 2

### Представление данных в памяти ЭВМ

# Представление данных в ЭВМ

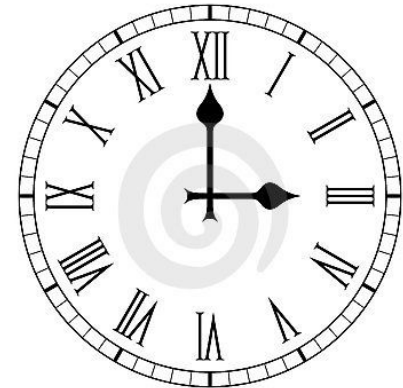
- Все данные хранятся в двоичном виде - 01010110101100011
- Значение данных определяется программой, и изображения и видео и текст в памяти ничем не отличаются
- **Бит** - минимальная единица хранения информации
- **Байт** - минимальная адресуемая единица хранения информации  
*1 байт = 8 бит, и может принимать одно из  $2^8$  значений (обычно)*
- **Слово** - количество байт, которое процессор может передать по шине данных за один такт. Для 16 битных процессоров слово = 2 байта, для 64 битных - 8 байт

# Представление чисел в памяти

- Для хранения чисел используются числа различной разрядности, кратные степени двойки - 1, 2, 4, 8 байт.
- Беззнаковые числа могут принимать значения от 0 до  $2^n-1$ ,  $n$  - разрядность числа
- Знаковые числа хранятся в дополнительном коде и могут принимать значения от  $-2^{n-1}$  до  $2^{n-1}-1$
- Вещественные числа хранятся в формате с плавающей точкой

# Системы счисления

- Непозиционные - значение символа не зависит от позиции
  - Римская
  - и многие другие древние системы
- Позиционные - значение символа зависит от его позиции в числе
  - Десятичная



# Позиционная система счисления

- Основание - количество символов в одном разряде
  - В десятичной системе - основание 10
  - В двоичной - 2
  - В системе счисления часов - 24, минут и секунд - 60
- Цифры - все возможные значения каждого разряда числа
  - Десятичная система - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - Двоичная система - 0, 1
  - Шестнадцатеричная система - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

В вычислительной технике чаще всего применяют двоичную и шестнадцатеричную системы счисления

# Двоичная система счисления

- Алфавит состоит из двух цифр - 0 и 1
- Легко хранить и передавать в вычислительных системах:
  - Нулевого и положительного электрического потенциала
  - Направление магнитного потока
  - И т.д.
- Позволяет закодировать любые данные

0	0
1	1
2	10
3	11
4	100
5	101

# Шестнадцатеричная система счисления

- Алфавит состоит из десятичных цифр и первых 6 букв латинского алфавита: **0123456789ABCDEF**
- Используется для короткой записи двоичных данных
- Все значения 1 Б данных могут быть представлены двумя разрядами
- При записи используется префикс **0x**, либо суффикс **h**

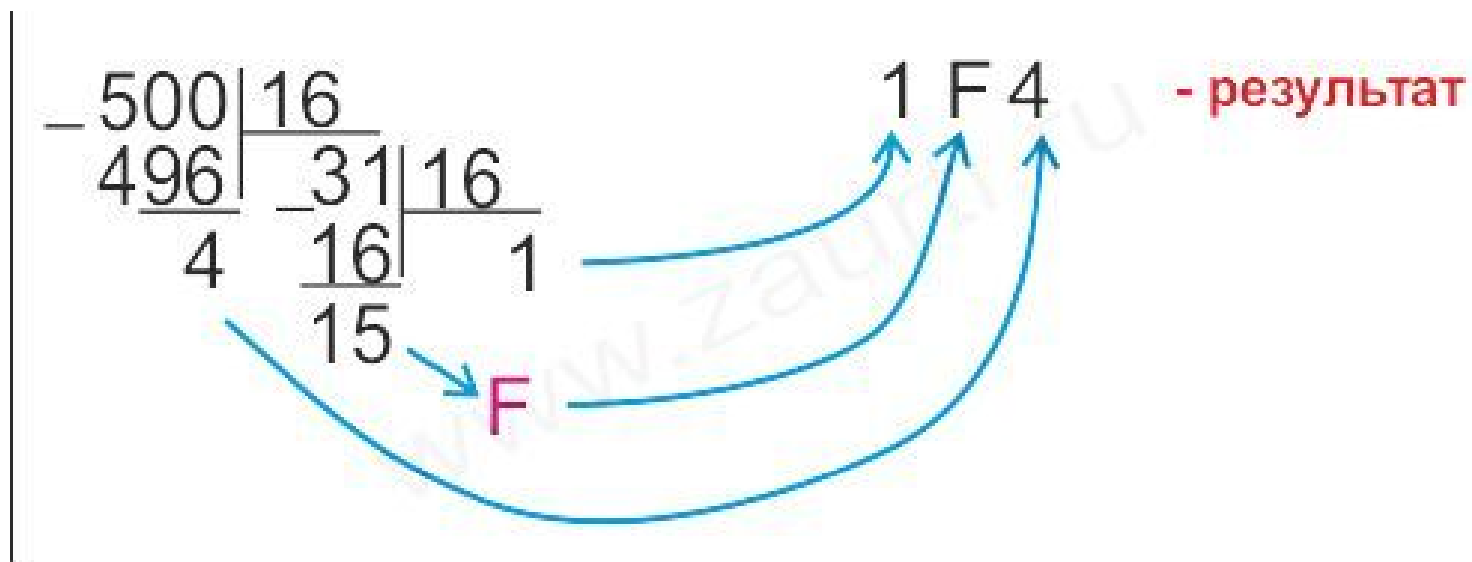
0	0x0
10	0xA
16	0x10
255	0xFF
1024	0x400

# Перевод из одной системы счисления в другую

Для перевода числа  $N$  из системы счисления с основанием  $X$  в систему счисления с основанием  $Y$  необходимо делить это число на  $Y$  до получения остатка меньшего  $Y$ . Полученное частное необходимо снова делить на  $Y$ , до получения остатка меньшего  $Y$  и так до тех пор пока частное будет меньше  $Y$ . Затем нужно записать остатки в порядке их получения. Это и будет искомое число в системе счисления  $Y$



# Перевод из 10 в 16



# Перевод в десятичную систему

Для перевода в десятичную систему необходимо разложить число в сумму степеней номеров разрядов умноженных на цифру разряда:

$$0x1F4 = 1 * 16^2 + 15 * 16^1 + 4 * 16^0 = 256 + 240 + 4 = 500$$

$$\begin{aligned} 0b11101011 &= 1 * 2^7 + 1 * 2^6 + 1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = \\ &= 128 + 64 + 32 + 0 + 8 + 0 + 2 + 1 = 235 \end{aligned}$$

# Перевод из 16 в 2

Десятичные числа	Шестнадцатеричные числа	Двоичные числа
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

$$0xFF = 1111\ 1111 = 15 * 16 + 15 = 255$$

$$0x80 = 1000\ 0000 = 8 * 16 + 0 = 128$$

$$0x10 = 0001\ 0000 = 1 * 16 + 0 = 16$$

$$0xAB = 1010\ 1011 = 10 * 16 + 11 = 171$$

$$0x63 = 0110\ 0011 = 6 * 16 + 3 = 99$$

$$0x93 = 1001\ 0011 = 9 * 16 + 3 = 147$$

$$0xD5 = 1101\ 0101 = 13 * 16 + 5 = 213$$

# Знаковые и беззнаковые целые числа

- Беззнаковые числа **не могут** принимать отрицательные значения
- Знаковые - **могут**
  
- Как хранить отрицательные числа?

# Система со знаком

- Старший разряд используется для указания знака числа
- 0 - положительное число
- 1 - отрицательное число

Пример:

00000001 = 1	00000000 = +0
10000001 = -1	10000000 = -0
10001000 = -8	

- Понятно человеку
- **Не понятно ЭВМ**

# Двоичный дополненный код

- Старший разряд используется для указания знака числа
- 0 - положительное число
- 1 - отрицательное число
- Отрицательное число инвертируется и к нему прибавляется 1

Пример:

00000001 = 1            10000000 = -128

11111110 = -1            01111111 = 127

11111011 = -5

- Не понятно человеку
- **Понятно ЭВМ**

# Вычитание двоичных чисел

- Вычитания нет, есть только сложение
- Вычитаемое заменяется на число в двоично-дополненном коде
- Числа складываются

Пример:

$10 - 7 = 3$	$\begin{array}{r} \phantom{-} 00001010 \\ - 00000111 \\ \hline = 00000011 \end{array}$	$\begin{array}{r} 00001010 \\ + 11111001 \\ \hline = 100000011 \end{array}$
--------------	--	---

# Вещественные числа

- Числа с плавающей запятой
  - одинарной точности - single precision, float
  - двойной точности - double precision
- Числа с фиксированной запятой



# IEEE 754

- Стандарт описывает принципы представления чисел с плавающей запятой
- Поддерживается подавляющим числом ЭВМ и языков программирования

Вещественное число представляется основанием ( $b$ ) и тремя целыми числами:

- $S$  - Знак
  - $Q$  - Порядок (экспонента)
  - $C$  - Мантисса
- $$(-1)^S \times C \times b^Q$$
- $$\begin{aligned} S &= 1 \\ C &= 12345 \\ Q &= -3 \\ (-1)^1 \times 12345 \times 10^{-3} &= 12.345 \end{aligned}$$

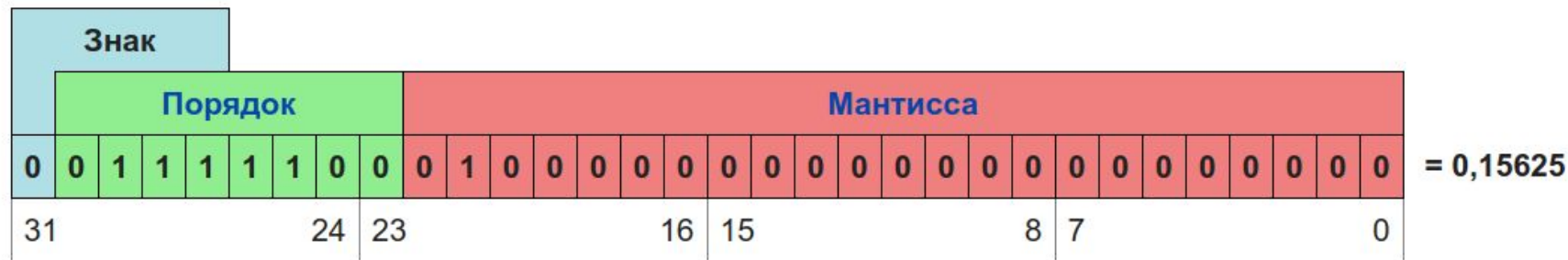
# IEEE 754 - представление в ЭВМ

- Мантисса всегда хранится в нормализованном формате:
  - 1.xxxxxxx
- Ведущая единица отбрасывается для экономии памяти, хранится только часть после запятой
- Экспонента записывается со смещением зависящим от точности числа
  - Одинарная точность:  $2^7 - 1 = 127$
  - Двойная точность:  $2^{10} - 1 = 1023$

# Числа одинарной точности

- В языках программирования часто обозначаются как float
- Описывается стандартом IEEE 754 и имеет размер 4 байта
- Основание  $b=2$
- Смещение экспоненты  $2^7-1 = 127$
- Количество разрядов мантииссы - 24

# Пример



$$S = 0$$

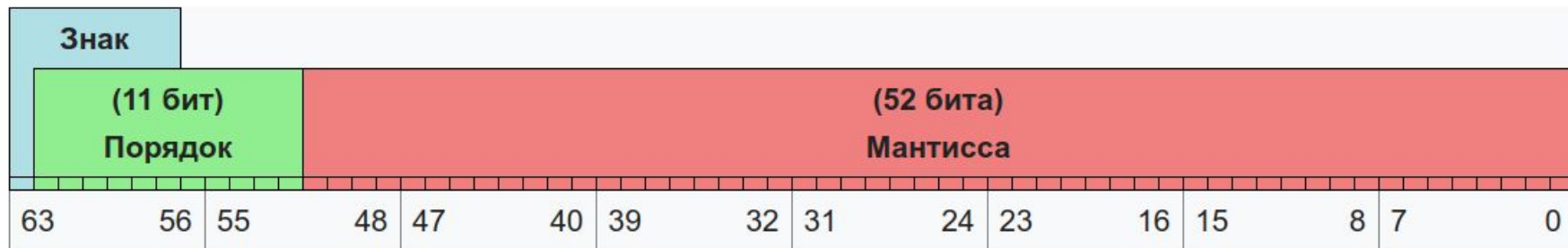
$$C = 1.01_2 = 1.25_{10}$$

$$Q = 1111100_2 - 1111111_2 = -3_{10}$$

$$(-1)^0 \times 1.25 \times 2^{-3} = 0.15625$$

# Числа двойной точности

- В языках программирования часто обозначаются как float
- Описывается стандартом IEEE 754 и имеет размер 8 байта
- Основание  $b=2$
- Смещение экспоненты  $2^{10}-1 = 1023$
- Количество разрядов мантииссы - 52



# Представление текста в памяти ЭВМ

- Как и все данные, текст в ЭВМ хранится в виде 0 и 1
- Мы договорились что 01000001 - 'А', 01000010 - 'В' и т.д.
- Такая договоренность называется “КОДИРОВКА”

Широко используемые кодировки с поддержкой кириллицы:

- Code page 1251 - Кириллическая Windows
- Code page 866 - Кириллическая DOS
- KOI-8 - Кириллическая Unix
- UTF-8 - современные ОС и web

# ASCII

- Разработана в 1963 году
- Кодировает латинские символы и управляющие последовательности 7 битами, старший бит - бит четности
- $2^7 = 128$  позиций

ASCII Code Chart

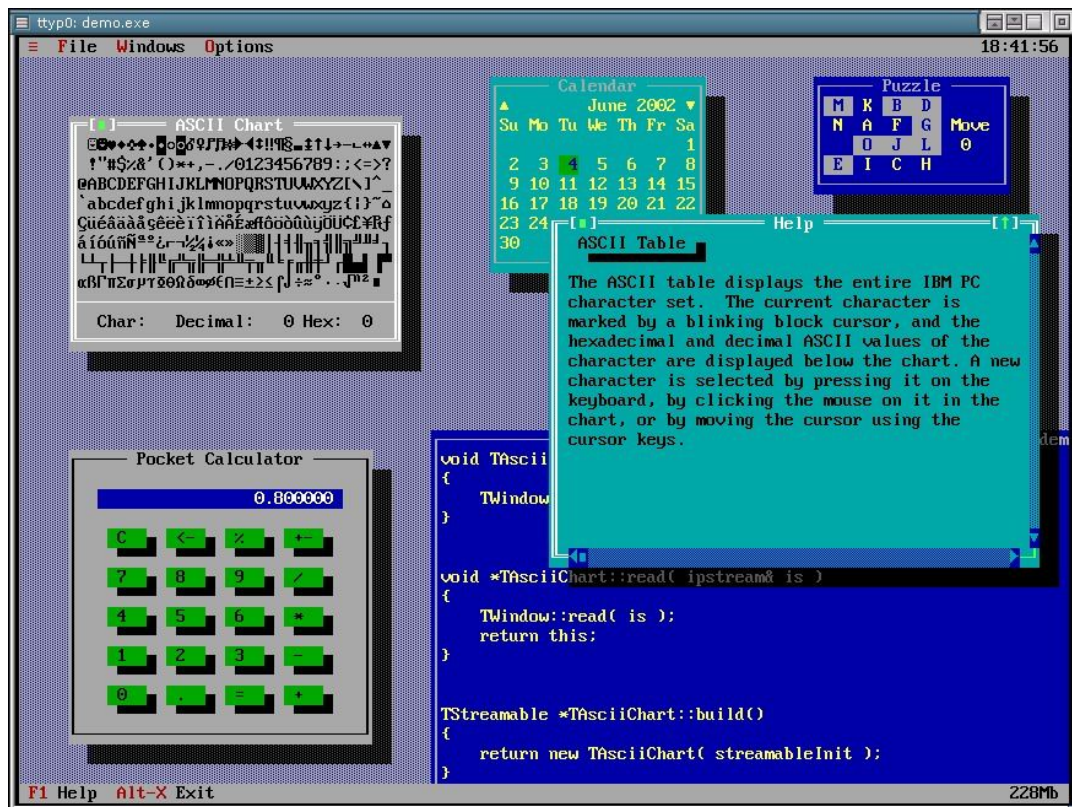
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

# Extended ASCII

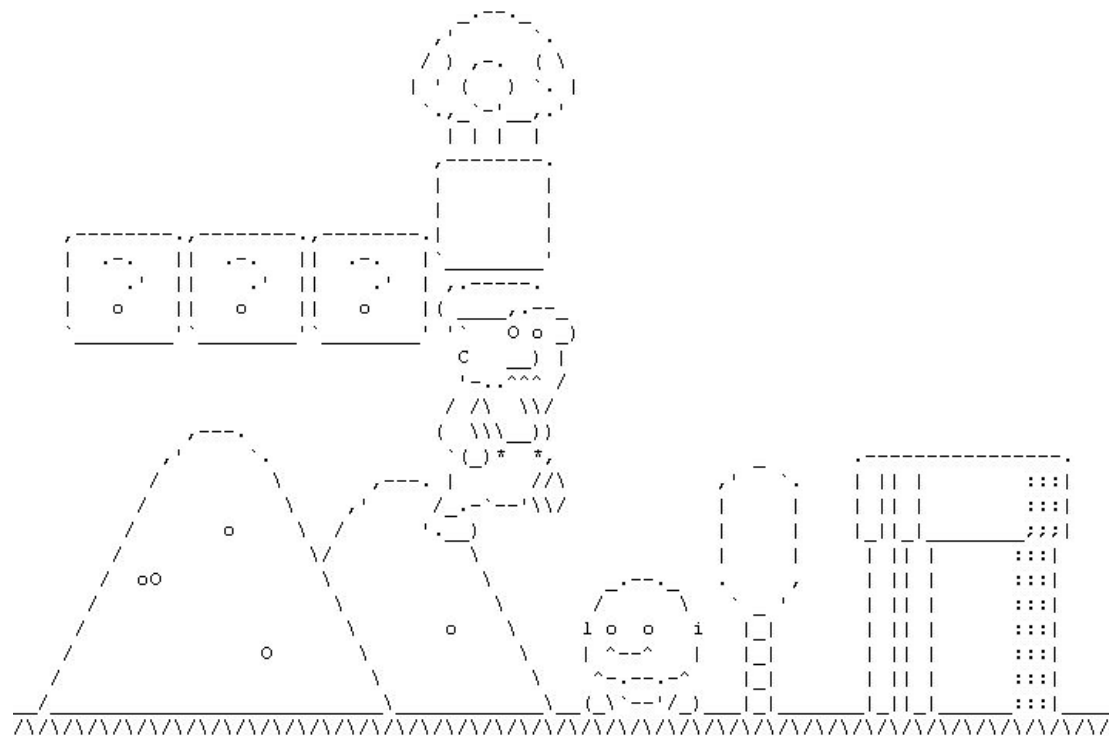
- Кодировывает символы 8 битами
- $2^8 = 256$  позиций
- Дополнительные 128 символов использовались для добавления национальных символов или псевдографики
- Большая часть однобайтовых кодировок относится к расширенному ASCII. В том числе:
  - KOI-8
  - CP1251
  - CP866



# Псевдографика с использованием Extended ASCII



# ASCII art



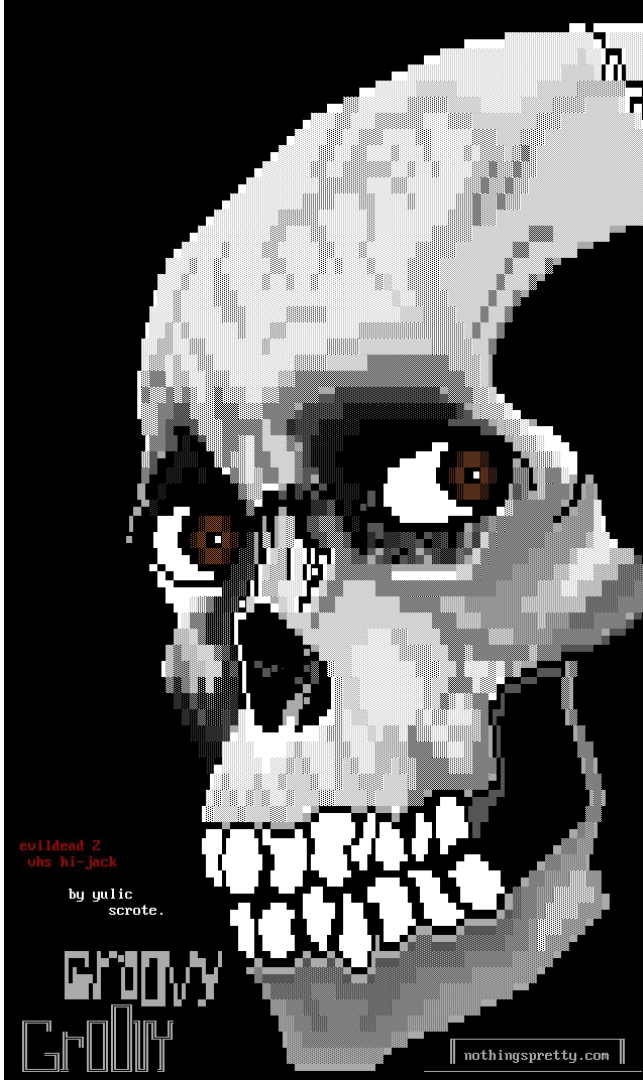
--Curchack

.....  
.....

# ASCII / ANSI art



# ASCII / ANSI art



# Проблемы несовместимости кодировок

□ хъёĥ яю№ĥшĥёп ш ёĥрэютшĥёп эхїшĥрхьĥь я№ш хую шэĥх№я№шĥрішш  
їх№хч эхтх№эġŷ ъюфш№ютъġ:

- □ №ш тĥтюфх т ъюэёюыġ т Windows
- □ №ш эхтх№эю эрёĥ№юхээюь с№орġчх№х
- □ №ш эхя№ортшыġэĥs j№шєĥps
- □ х№етюыġ ъĥю §ĥю ях№хтхфхĥ чріĥġ ях№етġŷ ырсġ схч тĥяюыэхэшп  
□ тю ъэюушs ф№ġушs ёыġірps

# Unicode спешит на помощь

- Единая кодировка для всех
- Кодировует “все” возможные символы
  - Буквы
  - Математические символы
  - Музыкальные ноты
  - И многое другое
- 16 бит - хватит всем!



# Существующие форматы представления Unicode

- Несколько устаревших и не рекомендованных к использованию
- UTF-8
- UTF-16BE
- UTF-16LE
- UTF-32BE
- UTF-32LE

# UTF-8

- Кодировка использует переменную длину символа
- Первые 127 символов полностью повторяют ASCII и кодируются 1 байтом
- Национальные и специальные символы могут иметь длину от 2 до 4 байт
- Кириллические символы кодируются 2 байтами



# BOM - Маркер последовательности байтов

- UTF-8                    EF BB BF
- UTF-16 (BE)            FE FF
- UTF-16 (LE)            FF FE
- UTF-32 (BE)            00 00 FE FF
- UTF-32 (LE)            FF FE 00 00

# Состояние на сегодняшний момент

- 90% WEB-страниц использует кодировку UTF-8
- Все современные ОС поддерживают Unicode в том или ином виде
- Visual Studio с настройками по умолчанию выводит кракозябры в консоли