Представление текстовой информации

8-БИТНЫЕ КОДИРОВКИ

При представлении текста каждому символу алфавита ставится в соответствии некоторая последовательность бит (*bit pattern*) – двоичный код.

Перечень символов и их двоичных кодов называется *кодировкой* (*encoding*).

В 1963 г. была разработана, а в 1967 г. доработана стандартная кодировка ASCII (American Standard Code for Information Interchange).

Данная кодировка использует 7-битные кодовые последовательности (00_{16}–$7F_{16}$) и определяет коды для следующих символов

- латинских букв;
- цифр;
- знаков препинания;
- некоторых математических символов;
- некоторых специальных символов;
- т. н. управляющих символов (например, CR и LF – символы возврата каретки и перевода строки; HT – горизонтальная табуляция, SP – пробел, ESC – альтернативный регистр, DEL – удаление, BEL – звуковой сигнал).

Кодировка ASCII-1967

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL 00	SOH 01	STX 02	ETX 03	EOT 04	ENQ 05	ACK 06	BEL 07	BS 08	HT 09	LF 0A	VT 0B	FF 0C	CR 0D	SO 0E	SI 0F
1	DLE 10	DC1 11	DC2 12	DC3 13	DC4 14	NAK 15	SYN 16	ETB 17	CAN 18	EM 19	SUB 1A	ESC 1B	FS 1C	GS 1D	RS 1E	US 1F
2	SP 20	! 21	" 22	# 23	$ 24	% 25	& 26	' 27	(28	) 29	* 2A	+ 2B	, 2C	- 2D	. 2E	/ 2F
3	0 30	1 31	2 32	3 33	4 34	5 35	6 36	7 37	8 38	9 39	: 3A	; 3B	< 3C	= 3D	> 3E	? 3F
4	@ 40	A 41	B 42	C 43	D 44	E 45	F 46	G 47	H 48	I 49	J 4A	K 4B	L 4C	M 4D	N 4E	O 4F
5	P 50	Q 51	R 52	S 53	T 54	U 55	V 56	W 57	X 58	Y 59	Z 5A	[5B	\ 5C	] 5D	^ 5E	_ 5F
6	` 60	a 61	b 62	c 63	d 64	e 65	f 66	g 67	h 68	i 69	j 6A	k 6B	l 6C	m 6D	n 6E	o 6F
7	p 70	q 71	r 72	s 73	t 74	u 75	v 76	w 77	x 78	y 79	z 7A	{ 7B	| 7C	} 7D	~ 7E	DEL 7F

в кодировке ASCII коды латинских букв образуют непрерывную возрастающую последовательность: «A» – 41_{16}, «B» – 42_{16}, …, «Z» – $5A_{16}$; аналогично и для строчных букв: «a» – 61_{16}, «b» – 62_{16}, …, «z» – $7A_{16}$. Непрерывную возрастающую последовательность образуют и коды цифр: «0» – 30_{16}, «1» – 31_{16}, …, «9» – 39_{16}.

В дальнейшем ASCII стала международным стандартом под именем ISO/IEC 646:1991.

В этом стандарте часть символов не определены и должны быть заданы в национальных реализациях (в таблице выделены серым цветом).

ISO - Совместного технического комитета по стандартизации

8-БИТНЫЕ КОДИРОВКИ

- 8 битные кодировки каждый символ текста кодируют 8 битным целым числом. Для кодирования управляющих символов, символов латинского алфавита, цифр, знаков препинания и некоторых других символов используется таблица ASCII
- Существуют различные варианты дополнения кодировки ASCII до 8 битной.

Недостаток

- 2 набора символов
- Нет редких символов
- Модифицирующие символы
- Љ®¤Ёа®ўЄ
- Перекодировка

Кодировка ISO 8859-5 (Cyrillic)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	PAD 80	HOP 81	BPH 82	NBH 83	IND 84	NEL 85	SSA 86	ESA 87	HTS 88	HTJ 89	VTS 8A	PLD 8B	PLU 8C	RI 8D	SS2 8E	SS3 8F
9	DCS 90	PU1 91	PU2 92	STS 93	CCH 94	MW 95	SPA 96	EPA 97	SOS 98	SGCI 99	SCI 9A	CSI 9B	ST 9C	OCS 9D	PM 9E	APC 9F
A	NBSP A0	Ё A1	Ђ A2	Ѓ A3	Є A4	S A5	I A6	Ї A7	J A8	Љ A9	Њ AA	Ћ AB	Ќ AC	SHY AD	Ў AE	Џ AF
B	А B0	Б B1	В B2	Г B3	Д B4	Е B5	Ж B6	З B7	И B8	Й B9	К BA	Л BB	М BC	Н BD	О BE	П BF
C	Р C0	С C1	Т C2	У C3	Ф C4	Х C5	Ц C6	Ч C7	Ш C8	Щ C9	Ъ CA	Ы CB	Ь CC	Э CD	Ю CE	Я CF
D	а D0	б D1	в D2	г D3	д D4	е D5	ж D6	з D7	и D8	й D9	к DA	л DB	м DC	н DD	о DE	п DF
E	р E0	с E1	т E2	у E3	ф E4	х E5	ц E6	ч E7	ш E8	щ E9	ъ EA	ы EB	ь EC	э ED	ю EE	я EF
F	№ F0	ё F1	ђ F2	ѓ F3	є F4	s F5	і F6	ї F7	ј F8	љ F9	њ FA	ћ FB	ќ FC	§ FD	ў FE	џ FF

Стандарт ISO 8859 5 этого семейства определяет символы кириллицы и других славянских языков.

8-БИТНЫЕ КОДИРОВКИ

В операционных системах MS-DOS и Windows различные варианты наборов символов для 8-битных кодов 80_{16}–FF_{16} называют *кодовыми страницами* (code page).

В MS-DOS помимо символов национального алфавита кодовая страница содержала *символы псевдографики*, использовавшиеся для отображения графического интерфейса в текстовом режиме –

MS DOS cp866 .

С выходом Windows необходимость в псевдографике отпала, поэтому стали использоваться новые наборы кодовых страниц

- Windows-1251.

В операционных системах Unix используют стандарт –

ISO 8859-5

Покажите, как текст «**Я выучил C++19!**» представляется с использованием 8-битных кодировок:

а) cp866; б) Windows-1251; в) ISO 8859-5. В ответе приведите шестнадцатеричные коды символов.

Решение задания «а»

Найдем код для каждого символа текста. Коды русских букв будем искать по табл. 3 для «верхней» части кодовой страницы **cp86**
коды латинских букв («С»), знаков (пробел, «+», «!») и цифр – по табл1.1 для «нижней» части таблицы ASCII.

Символ	Я	[SP]	в	ы	у	ч	и	л	[SP]	С	+	+	1	9	!
Табл.	3	1	3	.3	3	3	3	3	1	1	1	1	1	1	1
Код	9F	20	A2	EB	E3	E7	A8	AB	20	43	2B	2B	31	39	21

Длина строки – 15 символов, длина кода – 15 байт.

Ответ: **9F 20 A2 EB E3 E7 A8 AB 20 43 2B 2B 31 39 21.**

Найдем код для каждого символа текста. Коды русских букв будем искать по табл. .4 для «верхней» части кодовой страницы **Windows-125**
коды латинских букв («С»), знаков (пробел, «+», «!») и цифр – по табл. .1 для «нижней» части таблицы ASCII.

Символ	Я	[SP]	в	ы	у	ч	л	л	[SP]	С	+	+	1	9	!
Табл.	3	1	3	3	3	3	3	3	1	1	1	1	1	1	1
Код	DF	20	E2	FB	F3	F7	E8	EB	20	43	2B	2B	31	39	21

Длина строки – 15 символов, длина кода – 15 байт.

Ответ: **DF 20 E2 FB F3 F7 E8 EB 20 43 2B 2B 31 39 21.**

Решение задания «в»

Найдем код для каждого символа текста. Коды русских букв будем искать по табл. 2 для «верхней» части кодировки **ISO 8859-5**,
коды латинских букв («С»), знаков (пробел, «+», «!») и цифр – по табл. 1 для «нижней» части таблицы ASCII.

Символ	Я	[SP]	в	ы	у	ч	и	л	[SP]	С	+	+	1	9	!
Табл.	3	1	3	3	3	3	3	3	1	1	1	1	1	1	1
Код	CF	20	D2	EB	E3	E7	D8	DB	20	43	2B	2B	31	39	21

Длина строки – 15 символов, длина кода – 15 байт.

Ответ: **CF 20 D2 EB E3 E7 D8 DB 20 43 2B 2B 31 39 21.**

Представление строк

Строковые переменные в памяти компьютера обычно хранятся как массивы символов, причем каждый символ представляется своим кодом: либо 8 битным, либо 16 битным UTF 16.

Для задания длины строки могут использоваться следующие схемы.

- Паскалевская строка
- Нуль-терминированная строка
- Гибридная строка

Для заданной строки **«М7,Елшвщпй2~»** (все буквы – кириллица), Windows 1251; определите последовательность кодов символов строки в рамках заданной 8 битной кодировки. Покажите, как полученная последовательность кодов хранится в памяти компьютера.

Решение:

«М7,Елшвщпй2~» Длина строки – 12 символов.

Найдем код для каждого символа текста.

Коды русских букв будем искать по «верхней» части кодовой страницы Windows-1251, знаков («~», «,») и цифр для «нижней» части таблицы ASCII.

Символ	**М**	**7**	**,**	**Е**	**л**	**ш**	**в**	**ш**	**п**	**й**	**2**	**~**
Код	**CC**	**37**	**2C**	**5C**	**EB**	**F8**	**E2**	**F9**	**EF**	**E9**	**32**	**7E**

В паскалевском формате строка дополняется префиксом, представляющим ее длину. Т. к., значения байт памяти мы записываем в шестнадцатеричном виде (требование задачи), длину строки представим в шестнадцатеричном виде: **$12 = 0C_{16}$.**

Строка имеет вид: **0C** CC 37 2C C5 EB F8 E2 F9 EF E9 32 7E.

В формате нуль-терминированной сроки в конце строки помещается нуль-символ, т. е. символ с кодом 00_{16} в кодировке Windows-1251.

Строка имеет вид: CC 37 2C C5 EB F8 E2 F9 EF E9 32 7E **00**.

В гибридном формате используется как префикс длины строки, так и двойной нуль-символ:

0C CC 37 2C C5 EB F8 E2 F9 EF E9 32 7E **00 00**.

UNICODE

В настоящее время для кодирования, представления и обработки текста используется стандарт Unicode. Этот стандарт каждому символу (*code point*) ставит в соответствие 21-битное числовое значение из диапазона 000000_{16}–$10FFFF_{16}$.

В пространстве Unicode можно выделить *блоки* (blocks) символов – непрерывные последовательности символов, относящиеся к одной «письменности».

Например, блок «Controls and Basic Latin» с кодами 0000_{16}–$007F_{16}$ эквивалентен таблице ASCII ,а блок «Cyrillic» с кодами 0400_{16}–$04FF_{16}$ содержит символы кириллицы.

Символы Unicode будем записывать в виде «**U+*xxxx*», где *xxxx*** – шестнадцатеричное значение кода символа.

Существуют различные варианты представления 21-битных кодов Unicode в памяти (в оперативной или во внешней – файлах на диске).

Рассмотрим следующие: UTF-8, UTF-16, UTF-32.

Unicode

- **Universal Character Set**, UCS
  - 21-битные коды: 000000–10FFFF
  - Символ ←→ Code Point
- **Unicode Transformation Format**, UTF
  - Машинное представление кодов
  - UTF-8, UTF-16, UTF-32

Unicode, UCS

- **Controls and Basic Latin: 0000–007F**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	NUL 0000	SOH 0001	STX 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007	BS 0008	HT 0009	LF 000A	VT 000B	FF 000C	CR 000D	SO 000E	SI 000F
001	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017	CAN 0018	EM 0019	SUB 001A	ESC 001B	FS 001C	GS 001D	RS 001E	US 001F
002	SP 0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
003	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
004	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
005	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
006	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
007	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	DEL 007F

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
040	Ѐ 0400	Ё 0401	Ђ 0402	Ѓ 0403	Є 0404	Ѕ 0405	І 0406	Ї 0407	Ј 0408	Љ 0409	Њ 040A	Ћ 040B	Ќ 040C	Ѝ 040D	Ў 040E	Џ 040F
041	А 0410	Б 0411	В 0412	Г 0413	Д 0414	Е 0415	Ж 0416	З 0417	И 0418	Й 0419	К 041A	Л 041B	М 041C	Н 041D	О 041E	П 041F
042	Р 0420	С 0421	Т 0422	У 0423	Ф 0424	Х 0425	Ц 0426	Ч 0427	Ш 0428	Щ 0429	Ъ 042A	Ы 042B	Ь 042C	Э 042D	Ю 042E	Я 042F
043	а 0430	б 0431	в 0432	г 0433	д 0434	е 0435	ж 0436	з 0437	и 0438	й 0439	к 043A	л 043B	м 043C	н 043D	о 043E	п 043F
044	р 0440	с 0441	т 0442	у 0443	ф 0444	х 0445	ц 0446	ч 0447	ш 0448	щ 0449	ъ 044A	ы 044B	ь 044C	э 044D	ю 044E	я 044F
045	ѐ 0450	ё 0451	ђ 0452	ѓ 0453	є 0454	ѕ 0455	і 0456	ї 0457	ј 0458	љ 0459	њ 045A	ћ 045B	ќ 045C	ѝ 045D	ў 045E	џ 045F
046	Ѡ 0460	ѡ 0461	Ѣ 0462	ѣ 0463	Ѥ 0464	ѥ 0465	Ѧ 0466	ѧ 0467	Ѩ 0468	ѩ 0469	Ѫ 046A	ѫ 046B	Ѭ 046C	ѭ 046D	Ѯ 046E	ѯ 046F
047	Ѱ 0470	ѱ 0471	Ѳ 0472	ѳ 0473	Ѵ 0474	ѵ 0475	Ѷ 0476	ѷ 0477	Ѹ 0478	ѹ 0479	Ѻ 047A	ѻ 047B	Ѽ 047C	ѽ 047D	Ѿ 047E	ѿ 047F
048	Ҁ 0480	ҁ 0481	≠ 0482	◌҃ 0483	◌҄ 0484	◌҅ 0485	◌҆ 0486	◌҇ 0487	◌҈ 0488	◌҉ 0489	Ҋ 048A	ҋ 048B	Ҍ 048C	ҍ 048D	Ҏ 048E	ҏ 048F
049	Ґ 0490	ґ 0491	Ғ 0492	ғ 0493	Ҕ 0494	ҕ 0495	Җ 0496	ж 0497	Ҙ 0498	ҙ 0499	Қ 049A	қ 049B	Ҝ 049C	ҝ 049D	Ҟ 049E	ҟ 049F
04A	Ҡ 04A0	ҡ 04A1	Ң 04A2	ң 04A3	Ҥ 04A4	ҥ 04A5	Ҧ 04A6	ҧ 04A7	Ҩ 04A8	ҩ 04A9	Ҫ 04AA	ҫ 04AB	Ҭ 04AC	ҭ 04AD	Ү 04AE	ү 04AF
04B	Ұ 04B0	ұ 04B1	Ҳ 04B2	ҳ 04B3	Ҵ 04B4	ҵ 04B5	Ҷ 04B6	ҷ 04B7	Ҹ 04B8	ҹ 04B9	Һ 04BA	һ 04BB	Ҽ 04BC	ҽ 04BD	Ҿ 04BE	ҿ 04BF
04C	Ӏ 04C0	Ӂ 04C1	ӂ 04C2	Ӄ 04C3	ӄ 04C4	Ӆ 04C5	ӆ 04C6	Ӈ 04C7	ӈ 04C8	Ӊ 04C9	ӊ 04CA	Ӌ 04CB	ӌ 04CC	Ӎ 04CD	ӎ 04CE	ӏ 04CF
04D	Ӑ 04D0	ӑ 04D1	Ӓ 04D2	ӓ 04D3	Ӕ 04D4	ӕ 04D5	Ӗ 04D6	ӗ 04D7	Ә 04D8	ә 04D9	Ӛ 04DA	ӛ 04DB	Ӝ 04DC	ӝ 04DD	Ӟ 04DE	ӟ 04DF
04E	Ӡ 04E0	ӡ 04E1	Ӣ 04E2	ӣ 04E3	Ӥ 04E4	ӥ 04E5	Ӧ 04E6	ӧ 04E7	Ө 04E8	ө 04E9	Ӫ 04EA	ӫ 04EB	Ӭ 04EC	ӭ 04ED	Ӯ 04EE	ӯ 04EF
04F	Ӱ 04F0	ӱ 04F1	Ӳ 04F2	ӳ 04F3	Ӵ 04F4	ӵ 04F5	Ӷ 04F6	ӷ 04F7	Ӹ 04F8	ӹ 04F9	Ӻ 04FA	ӻ 04FB	Ӽ 04FC	ӽ 04FD	Ӿ 04FE	ӿ 04FF

- **Представление UTF-16**
- Каждый код Unicode представляется в виде 2 байт или 4 байт.
- Символы от U+0000 до U+D7FF и от U+E000 до U+FFFF кодируются 2 байтными значениями («как есть», т. е. код равен численному значению code point).
- Символы от U+D800 до U+DFFF – зарезервированы (см. ниже), они не должны соответствовать реальным символам.
- Символы от U+010000 до U+10FFFF кодируются 4 байтными значениями по следующему алгоритму.

Unicode, UTF

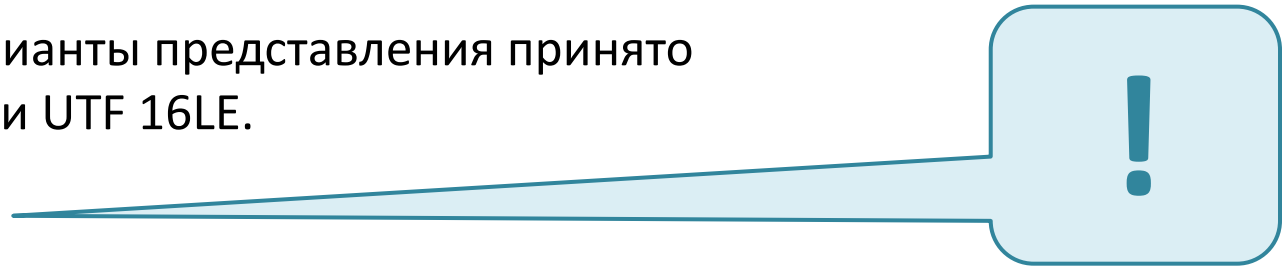
UTF-16

- 1 символ = 2 байта
- «**М**» → U+041C → 04 1C
- «**Г**» → U+0413 → 04 13
- «**Т**» → U+0422 → 04 22
- «**У**» → U+0423 → 04 23
- «**МГТУ**» → 04 1C 04 13 04 22 04 23

- В зависимости от архитектуры аппаратного и программного обеспечения, UTF 16 может записывать байты кода в разном порядке: Big-endian или Little-endian; соответствующие варианты представления принято обозначать UTF 16BE и UTF 16LE.
- При сохранении текста в представлении UTF 16 в файл, в начало файла может добавляться т. н. маркер порядка байт (byte order mark, BOM) – символ U+FEFF «ZERO WIDTH NON-BREAKING SPACE». Этот символ позволяет определить используемый порядок байт: если в начала файла находятся байты FE FF, то при сохранении использовался UTF 16BE, если же байты FF FE, то использовался UTF 16LE. Если BOM отсутствует, то стандарт предполагает использование UTF 16BE (однако многие Windows-приложения предполагают UTF 16LE, т. к. данная кодировка используется в операционной системе по умолчанию).

UTF-16

В зависимости от архитектуры аппаратного и программного обеспечения, UTF 16 может записывать байты кода в разном порядке: Big-endian или Little-endian;
соответствующие варианты представления принято обозначать UTF 16BE и UTF 16LE.

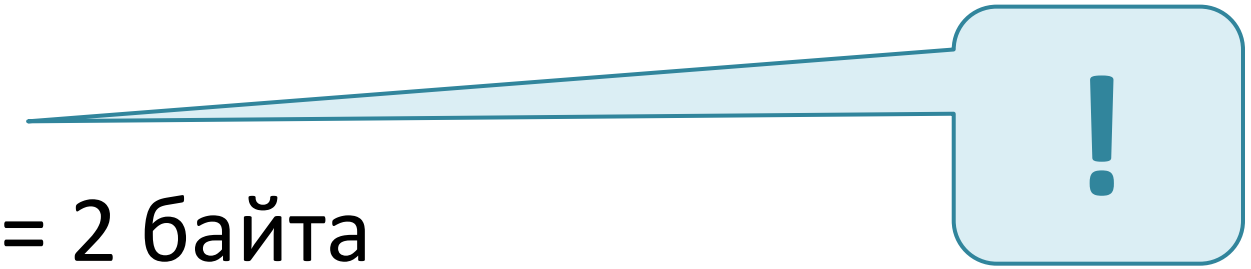


UTF-16BE

- 1 символ = 2 байта
- «**М**» → U+041C → 04 1C
- «**Г**» → U+0413 → 04 13
- «**Т**» → U+0422 → 04 22
- «**У**» → U+0423 → 04 23
- «**МГТУ**» → 04 1C 04 13 04 22 04 23

UTF-16

UTF-16LE



- 1 символ = 2 байта
- «**М**» → U+041C → 04 1C → 1C 04
- «**Г**» → U+0413 → 04 13 → 13 04
- «**Т**» → U+0422 → 04 22 → 22 04
- «**У**» → U+0423 → 04 23 → 23 04
- «**МГТУ**» → 1C 04 13 04 22 04 23 04

UTF-16

UTF-16 LE или BE?

- 04 1C 04 13 04 22 04 23
- «МГТУ» (BE)
- «ᰄጄ∄⌄» (LE)

UTF-16

UTF-16 LE или BE?

- <метка> 04 1C 04 13 04 22 04 23
- U+FEFF, (zero-width non-breaking space)
- **FE FF** 04 1C 04 13 04 22 04 23

При сохранении текста в представлении UTF 16 в файл, в начало файла может добавляться т. н. маркер порядка байт (byte order mark, BOM) – символ U+FEFF «ZERO WIDTH NON-BREAKING SPACE».
Этот символ позволяет определить используемый порядок байт:
если в начала файла находятся байты FE FF, то при сохранении использовался UTF 16BE,
если же байты FF FE, то использовался UTF 16LE. Если BOM отсутствует, то стандарт предполагает использование UTF 16BE (однако многие Windows-приложения предполагают UTF 16LE, т. к. данная кодировка используется в операционной системе по умолчанию).

UTF-16

Символы больше U+FFFF?

- Каждый код Unicode представляется в виде 2 байт или 4 байт.
- Символы от U+0000 до U+D7FF и от U+E000 до U+FFFF кодируются 2 байтными значениями («как есть», т. е. код равен численному значению code point).
- Символы от U+D800 до U+DFFF – зарезервированы, они не должны соответствовать реальным символам.
- Символы от U+010000 до U+10FFFF кодируются 4 байтными значениями по следующему алгоритму.

- *Алгоритм Получение кода UTF-16 для символов от U+010000 до U+10FFFF*

1. Из 21-битного кода символа вычитается значение 10000_{16}. В результате получается 20-битное значение от 0_{16} до $FFFFF_{16}$.
2. К старшим 10 битам (значению от 0_{16} до $3FF_{16}$) результата, полученного в п. 1, прибавляется значение $D800_{16}$. Полученное 16-битное значение (из диапазона $D800_{16}$–$DBFF_{16}$) становится первой половиной кода. Оно называется *старшим суррогатом* (high surrogate или leading surrogate).
3. К младшим 10 битам (значению от 0_{16} до $3FF_{16}$) результата, полученного в п. 1, прибавляется значение $DC00_{16}$. Полученное 16-битное значение (из диапазона $DC00_{16}$–$DFFF_{16}$) становится второй половиной кода. Оно называется *младшим суррогатом* (low surrogate или trailing surrogate).
4. 2-байтные значения, полученные в пунктах 2 и 3, образуют 4-байтный код символа.
5. Посмотрим на примере.

Пример UTF-16

U+10E6D

- Вычесть 10000:
  - 10E6D – 10000 = 0E6D
- 20 бит результата разбить по 10:
  - **0000 0000 1110 0110 1101**
- Старший суррогат: прибавить D800
  - **11011000 00000000 + 00 00000011 = D803**
- Младший суррогат: прибавить DC00
  - **11011100 00000000 + 10 0110 1101 = DE6D**
- Итого: U+10E6D → D8 03 DE 6D

UTF-32

Каждый код Unicode представляется в виде 4 байт, содержащих числовое значение символа.
При хранении и передаче информации в данном представлении, также может использоваться маркер порядка байт, причем правила его использования аналогичны UTF-16.
Так, если первые байты: 00 00 FE FF, то используется UTF-32BE.
Если же первые байты FF FE 00 00, то используется UTF-32LE.

- 1 символ = **4 байта** (всегда)

«**МГТУ**»

- **00 00** 04 1C 00 00 04 13 00 00 04 22 00 00 04 23 (Little Endian)
- 1C 04 00 00 13 04 00 00 22 04 00 00 23 04 **00 00** (Big Endian)

Покажите, как заданная строка хранится в памяти компьютера, если используется кодировка UTF-16 и формат: «паскалевский», нуль-терминированной строки и гибридный (приведите шестнадцатеричные представления байт памяти). Считайте, что под хранение длины строки отводится 4 байта.
«Bmz;яфюксе5=» (первые 3 буквы – латиница), Little-endian;

Решение:
Длина строки – 12 символов.
По таблицам выпишем Unicode-символы строки:

Символ	Символ Unicode
B	0042
m	006D
z	007A
;	003B
я	044F
ф	0444
ю	044E
к	043A
с	0441
е	0435
5	0035
=	003D

Все символы строки принадлежат диапазону от U+0000 до U+D7FF, поэтому они будут иметь 2-байтные коды, совпадающие с их числовыми значениями

в UTF-16LE символы строки представляются следующим образом:
42 00 6D 00 7A 00 3B 00 4F 04 44 04 4E 04 3A 04 41 04 35 04 35 00 3D 00

Исходная строка

42 00 6D 00 7A 00 3B 00 4F 04 44 04 4E 04 3A 04 41 04 35 04 35 00 3D 00

В паскалевском формате строка дополняется префиксом, представляющим ее длину: $12 = 0C_{16} = 0000000C_{16}$. При записи длины строки учитываем порядок байт **Little-endian.** Строка имеет вид:

0C 00 00 00 42 00 6D 00 7A 00 3B 00 4F 04 44 04 4E 04 3A 04 41 04 35 04 35 00 3D 00

В формате **нуль-терминированной** сроки в конце строки помещается нуль-символ U+0000. Строка имеет вид:

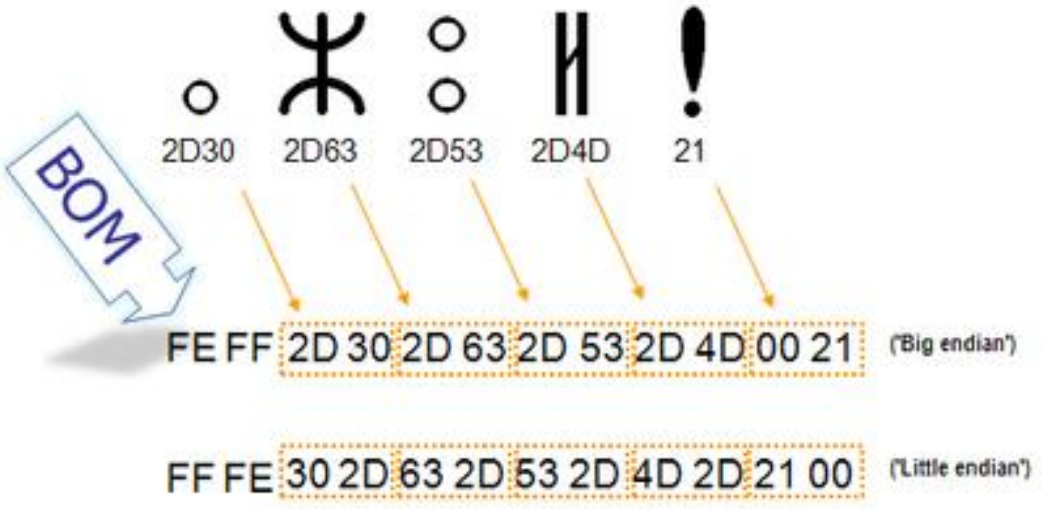
42 00 6D 00 7A 00 3B 00 4F 04 44 04 4E 04 3A 04 41 04 35 04 35 00 3D 00 **00 00**

В гибридном формате используется как префикс длины строки, так и двойной нуль-символ:

0C 00 00 00 42 00 6D 00 7A 00 3B 00 4F 04 44 04 4E 04 3A 04 41 04 35 04 35 00 3D 00 **00 00 00 00**

Маркер последовательности байтов или метка порядка байтов (англ. Byte Order Mark (BOM)) — Юникод-символ, используемый для индикации порядка байтов текстового файла.
Его кодовый символ **U+FEFF.**
По спецификации, его использование не является обязательным, однако, если маркер последовательности байтов используется, то он **должен быть установлен в начале текстового файла.**
Помимо своего конкретного использования в качестве указателя порядка байтов, символ может также указать, какой кодировкой Unicode закодирован текст.

Т.к. использование этого символа (U+FEFF), согласно спецификации Юникод, не **является обязательным,** однако оно широко распространено, так как **позволяет легко избежать неверного раскодирования текстовой информации.**



FF FE 2604 5104 4004 2C00 6D00 6400 6200 6700 6900 6A00 6D00 2E00.

Каков будет размер текстового файла, если этот текст сохранить с использованием представления UTF-8 и записать в начало файла маркер порядка байт (BOM)?

Байты	Символ Unicode	Символ
26 04	U+0426	Ц
51 04	U+0451	ё
40 04	U+0440	р
2C 00	U+002C	,
6D 00	U+006D	m
64 00	U+0064	d
62 00	U+0062	b
67 00	U+0067	g
69 00	U+0069	i
6A 00	U+006A	j
6D 00	U+006D	m
2E 00	U+002E	.

Определим размер текстового файла

В представлении UTF 8 коды **русских букв** имеют длину **2**, а коды латинских букв, цифр и знаков – длину **1 байт.**

В тексте: русских букв **– 3**, латинских букв **– 7**, цифр **– 0**, знаков **– 2.**

Следовательно, общая длина текста составит **15 байт.**

С учетом 3 байтного **BOM размер текстового файла в представлении UTF 8 будет 18 байт.**