Programação 2 _ T1

Apresentação

Representação de dados

Rui Camacho
(slides por Luís Teixeira)

MIEEC 2020/2021

EQUIPA DOCENTE

.: 2 blocos de turmas

Aulas teóricas

- Rui Camacho

Aulas práticas

- Diogo Pernes
- Gil Coutinho
- Isabel Rio-Torto
- João Teixeira
- Pedro Costa

Monitores: Luís Sousa, Sílvia Faria, Andreia Seabra, Leonardo Ribeiro, Fábio Gaspar, Miguel Almeida, João Pires, Tiago Martins, Pedro Teixeira, Rui Coutinho, Fábio Morais, João Pinhal, Ana Cruz, Miguel Macedo, David Viana, Hugo Martins

PROGRAMA (PARTE 1)

1. Programação em C e Metodologias de Desenvolvimento

- Consolidação de conceitos básicos de programação em C
- Compilação separada de ficheiros utilizando ferramentas auxiliares
- Criação e utilização de bibliotecas de funções
- Utilização de técnicas de depuração (debugging)

2. Programação de baixo-nível

- Representação de dados em memória
- Gestão de memória
- Mecanismos de passagem de argumentos a funções
- ~~Noções básicas de desenvolvimento em Assembly~~

PROGRAMA (PARTE 2)

3. Conceitos Fundamentais de Algoritmia

- Análise de complexidade de algoritmos
- Estratégias de concepção de algoritmos
- Algoritmos de ordenação e pesquisa
- Exemplos práticos de aplicação de diferentes estratégias

4. Estruturas de dados

- Estruturas lineares - listas, filas e pilhas
- Árvores
- Tabelas de dispersão
- Heaps
- Grafos

AVALIAÇÃO

Classificação Final (CF)

CF = 0,5 * F + 0,5 * MT

F → Classificação de frequência

MT → Classificação dos minitestes

Mínimos:

45% na classificação de frequência

40% na classificação dos minitestes

CLASSIFICAÇÃO DE FREQUÊNCIA (F)

.: 2 trabalhos práticos

$$F = 0{,}35\ TP1 + 0{,}65\ TP2$$

TP → classificação dos trabalhos práticos

CLASSIFICAÇÃO DE FREQUÊNCIA (F)

Frequência 2019/2020

- pode ser mantida este ano, sendo necessário realizar MTs
- **aconselhável** acompanhar de perto aulas teóricas e práticas

TRABALHOS PRÁTICOS

2 trabalhos práticos

implementação de **biblioteca de funções** de algoritmos e estruturas de dados e **aplicação** que utilize estruturas de dados e algoritmos adequados

a desenvolver numa **aula prática** dedicada e **para além das aulas**

trabalhos em grupos de **2 estudantes**

CLASSIFICAÇÃO DOS MINITESTES (MT)

.: 2 minitestes

MT = 1/3 MT1 + 2/3 MT2

MT1 → perto da Páscoa

MT2 → no final do semestre

componentes **teórica e prática**

exercícios práticos fortemente **relacionados** com trabalhos práticos realizados

CLASSIFICAÇÃO FINAL (CF)

$$CF = 0{,}167\ MT1 + 0{,}33\ MT2 + 0{,}175\ TP1 + 0{,}325\ TP2$$

OBSERVAÇÕES

Qualquer **fraude ou tentativa de fraude** nas avaliações (testes ou trabalhos) será punida com a **anulação** dessa avaliação e será apresentada ao Diretor do Curso.

Os docentes estarão disponíveis para esclarecer dúvidas sobre a matéria.

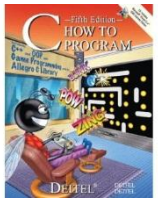
ESTUDO ASSISTIDO

Sessão semanal de 2 horas

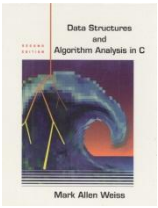
- 4ª feira, 15h-17h
- mais informações no Moodle

BIBLIOGRAFIA

Bibliografia Principal

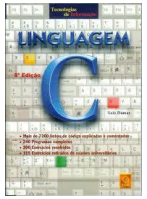


PJ Deitel e HM Deitel, "C How to Program", Prentice Hall

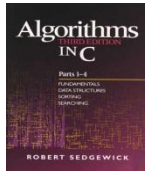


MA Weiss, "Data structures and algorithm analysis in C", Addison-Wesley

Bibliografia Complementar

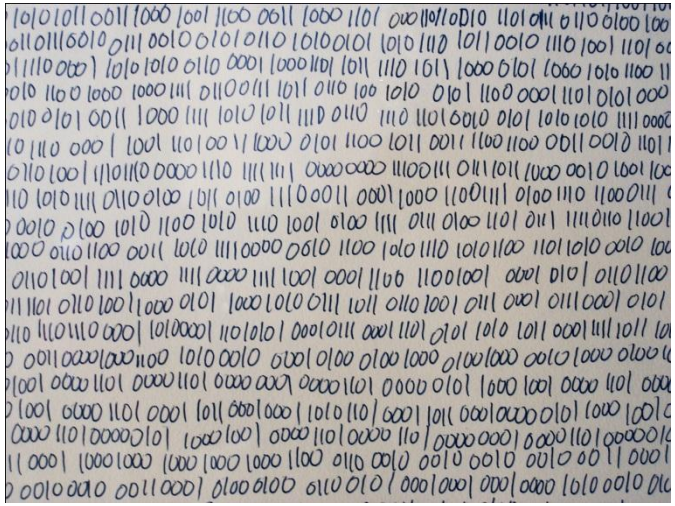


L Damas, "Linguagem C", FCA Editores



R Sedgewick, "Algorithms in C", Addison-Wesley

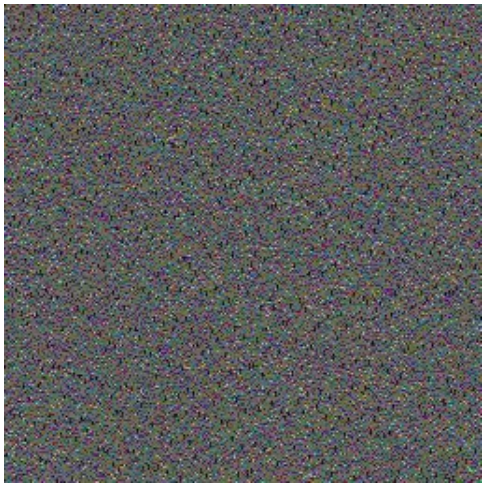
REPRESENTAÇÃO DE INFORMAÇÃO



ESTRUTURA DE DADOS

```
31 0A 0A 41 73 20 61 72 6D 61 73 20 65 20 6F 73 20
62 61 72 C3 B5 65 73 20 61 73 73 69 6E 61 6C 61 64
6F 73 2C 0A 51 75 65 20 64 61 20 6F 63 69 64 65 6E
74 61 6C 20 70 72 61 69 61 20 4C 75 73 69 74 61 6E
61 2C 0A 50 6F 72 20 6D 61 72 65 73 20 6E 75 6E 63
61 20 64 65 20 61 6E 74 65 73 20 6E 61 76 65 67 61
64 6F 73 2C 0A 50 61 73 73 61 72 61 6D 20 61 69 6E
64 61 20 61 6C C3 A9 6D 20 64 61 20 54 61 70 72 6F
62 61 6E 61 2C 0A 45 6D 20 70 65 72 69 67 6F 73 20
65 20 67 75 65 72 72 61 73 20 65 73 66 6F 72 C3 A7
61 64 6F 73 2C 0A 4D 61 69 73 20 64 6F 20 71 75 65
20 70 72 6F 6D 65 74 69 61 20 61 20 66 6F 72 C3 A7
61 20 68 75 6D 61 6E 61 2C 0A 45 20 65 6E 74 72 65
20 67 65 6E 74 65 20 72 65 6D 6F 74 61 20 65 64 69
66 69 63 61 72 61 6D 0A 4E 6F 76 6F 20 52 65 69 6E
6F 2C 20 71 75 65 20 74 61 6E 74 6F 20 73 75 62 6C
69 6D 61 72 61 6D 3B 0A 0A 32 0A 0A 45 20 74 61 6D
62 C3 A9 6D 20 61 73 20 6D 65 6D C3 B3 72 69 61 73
20 67 6C 6F 72 69 6F 73 61 73 0A 44 61 71 75 65 6C
65 73 20 52 65 69 73 2C 20 71 75 65 20 66 6F 72 61
6D 20 64 69 6C 61 74 61 6E 64 6F 0A 41 20 46 C3 A9
```

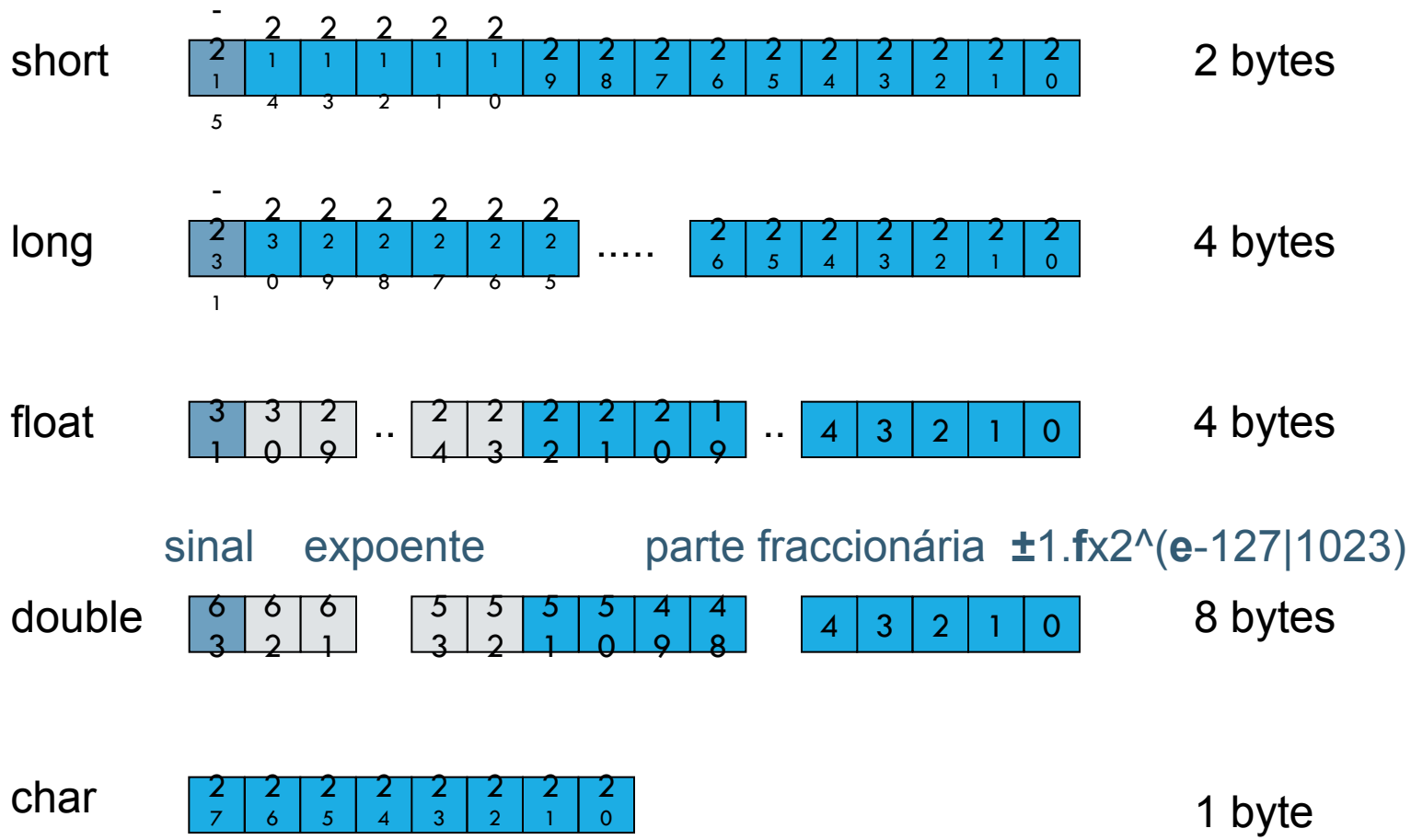
```
1

As armas e os barões assinalados,
Que da ocidental praia Lusitana,
Por mares nunca de antes navegados,
Passaram ainda além da Taprobana,
Em perigos e guerras esforçados,
Mais do que prometia a força humana,
E entre gente remota edificaram
Novo Reino, que tanto sublimaram;

2

E também as memórias gloriosas
Daqueles Reis, que foram dilatando
A Fé, o Império, e as terras viciosas
De África e de Ásia andaram devastando;
E aqueles, que por obras valerosas
Se vão da lei da morte libertando;
Cantando espalharei por toda parte,
Se a tanto me ajudar o engenho e arte.
```



Dados armazenados num conjunto de bytes de forma estruturada

Mesmos dados podem ser interpretados de forma diferente

TIPOS PRINCIPAIS DE DADOS



REPRESENTAÇÃO DE CARACTERES

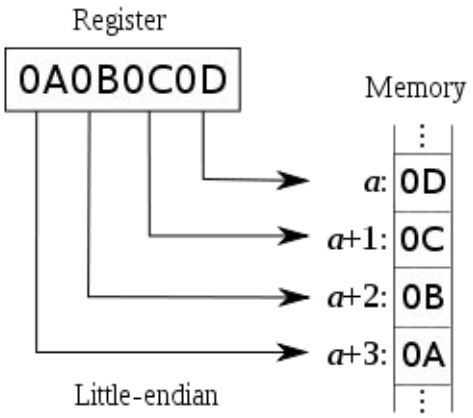
ASCII - American Standard Code for Information Interchange

*	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	TAB	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	{	|	}	~	

REPRESENTAÇÃO EM MEMÓRIA

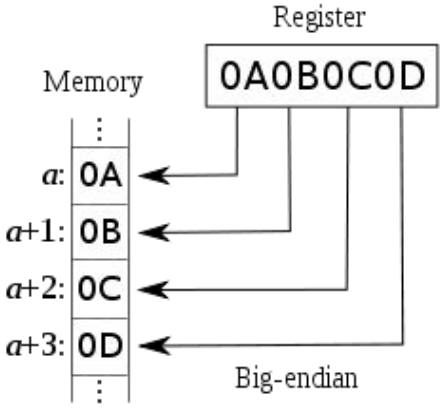
Valores compostos por múltiplos bytes podem ser guardados em memória de diferentes formas

Little-endian (ordem inversa)



- **exemplos:** Linux, Mac OS X e Windows em arquitetura Intel

Big-endian (mesma ordem)



- **exemplos:** Linux e Mac OS, em arquitetura PowerPC, Playstation 3

REPRESENTAÇÃO EM MEMÓRIA

```
/* Exemplo(little-endian) */
#include <stdio.h>
main()
{
   int i = 1211;
   char v[5] = {'m', 'n'};
   char c = 'a';

   printf("i - %p\n", &i);
   printf("v - %p\n", v);
   printf("c - %p\n", &c);
}
```

Resultado:

```
i - 0xbfaf8ca3
v - 0xbfaf8ca7
c - 0xbfaf8cac
```

Endereço		Valor	
0xbfaf8ca	**3**	bb	int i
	4	04	
	5	00	
	6	00	
	7	6d	char v[5]
	8	6e	
	9	00	
	a	00	
	b	00	
	c	61	char c

ARMAZENAMENTO (PERSISTENTE) DE INFORMAÇÃO - FICHEIROS

Em quase todos os Sistemas Operativos modernos os ficheiros estão organizados num **vetor unidimensional de bytes**

São associados a um **nome** através do qual podem ser acedidos

Operações típicas sobre um ficheiro: criar, modificar de atributos, abrir, ler e modificar conteúdo, guardar alterações e fechar

TIPOS DE FICHEIROS

Texto

Inclui apenas bytes que tenham correspondência a carateres que possam ser lidos

Diferentes formatos de codificação

Exemplos: *plain text files*, html, xml, ficheiros de código fonte e de configuração.

Binário

Ficheiro genérico que pode incluir qualquer tipo de informação

Requer um conhecimento da estrutura para que possa ser lido correctamente

Exemplos: imagens, vídeos, áudio, documentos, programas compilados.

FICHEIROS DE TEXTO

Nos ficheiros de texto a informação é guardada sob a forma de linhas de texto, separadas por um caráter terminador de linha, '\n' ("new line") ou, como nos sistemas Windows, pela sequência de carateres '\r"\n' ("return, new line").

FORMATOS DE CODIFICAÇÃO

ISO 646 (ASCII)

ISO 8859

- ISO 8859-1 (Latin-1)
- ISO 8859-9 (Latin-9)

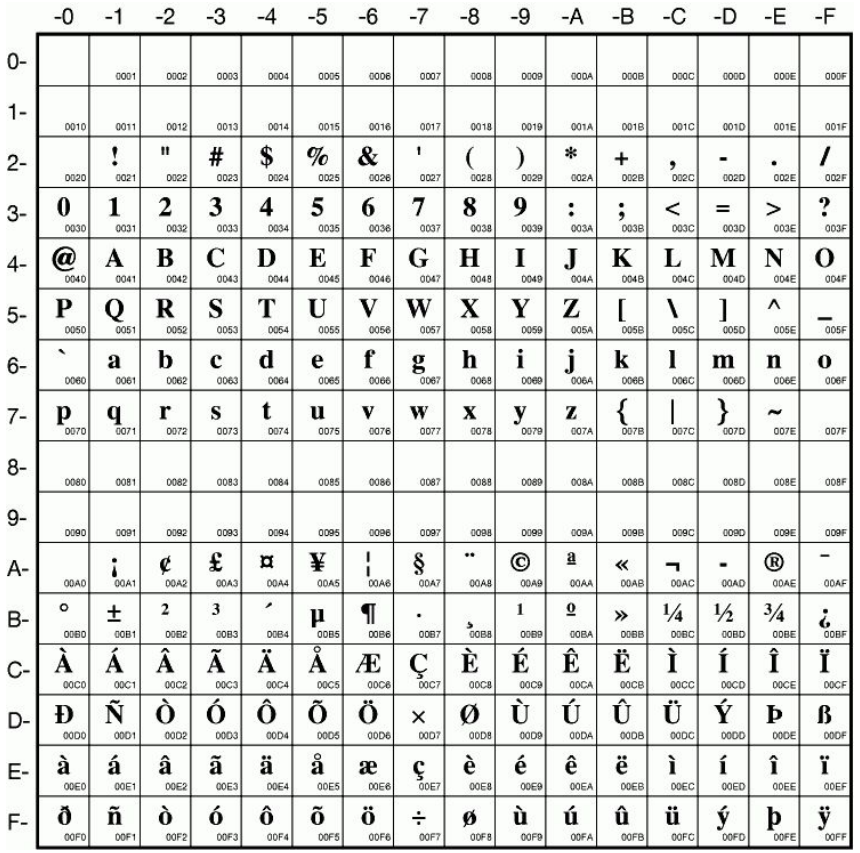
Unicode

- UTF-8, UTF-16

Windows Latin 1

Mac OS Roman

…



	-0	-1	-2	-3	-4	-5	-6	-7	-8	-9	-A	-B	-C	-D	-E	-F
0-	0000	0001	0002	0003	0004	0005	0006	0007	0008	0009	000A	000B	000C	000D	000E	000F
1-	0010	0011	0012	0013	0014	0015	0016	0017	0018	0019	001A	001B	001C	001D	001E	001F
2-	0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3-	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
4-	@ 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
5-	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
6-	` 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
7-	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	007F
8-	0080	0081	0082	0083	0084	0085	0086	0087	0088	0089	008A	008B	008C	008D	008E	008F
9-	0090	0091	0092	0093	0094	0095	0096	0097	0098	0099	009A	009B	009C	009D	009E	009F
A-	00A0	¡ 00A1	¢ 00A2	£ 00A3	¤ 00A4	¥ 00A5	¦ 00A6	§ 00A7	¨ 00A8	© 00A9	ª 00AA	« 00AB	¬ 00AC	- 00AD	® 00AE	¯ 00AF
B-	° 00B0	± 00B1	2 00B2	3 00B3	´ 00B4	µ 00B5	¶ 00B6	· 00B7	¸ 00B8	1 00B9	º 00BA	» 00BB	¼ 00BC	½ 00BD	¾ 00BE	¿ 00BF
C-	À 00C0	Á 00C1	Â 00C2	Ã 00C3	Ä 00C4	Å 00C5	Æ 00C6	Ç 00C7	È 00C8	É 00C9	Ê 00CA	Ë 00CB	Ì 00CC	Í 00CD	Î 00CE	Ï 00CF
D-	Ð 00D0	Ñ 00D1	Ò 00D2	Ó 00D3	Ô 00D4	Õ 00D5	Ö 00D6	× 00D7	Ø 00D8	Ù 00D9	Ú 00DA	Û 00DB	Ü 00DC	Ý 00DD	Þ 00DE	ß 00DF
E-	à 00E0	á 00E1	â 00E2	ã 00E3	ä 00E4	å 00E5	æ 00E6	ç 00E7	è 00E8	é 00E9	ê 00EA	ë 00EB	ì 00EC	í 00ED	î 00EE	ï 00EF
F-	ð 00F0	ñ 00F1	ò 00F2	ó 00F3	ô 00F4	õ 00F5	ö 00F6	÷ 00F7	ø 00F8	ù 00F9	ú 00FA	û 00FB	ü 00FC	ý 00FD	þ 00FE	ÿ 00FF

ISO/IEC 8859-1

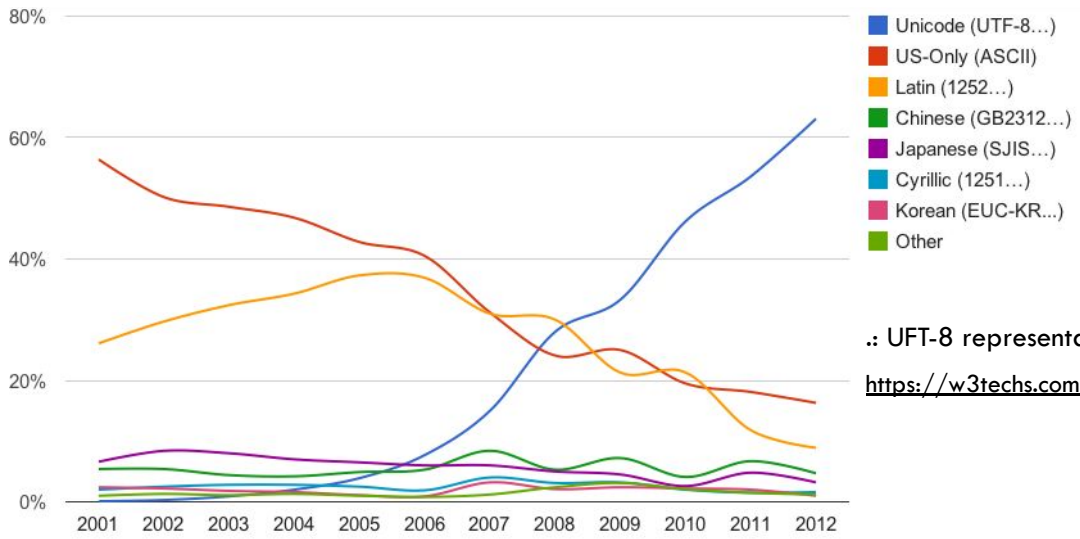
UNICODE

UTF-16/32

- 2/4 bytes por cada caráter
- ordem dos bytes pode variar (little- ou big-endian)

UTF-8

- 1 a 4 bytes por caráter
- compatível com ASCII
- ordem dos bytes fixa



.: UFT-8 representa 93% em Fevereiro de 2019

https://w3techs.com/technologies/history_overview/character_encoding

COMPARAÇÃO ENTRE FORMATOS

UTF-8

```
00000000 31 0A 0A 41 73 20 61 72 6D 61 73 20 65 20 6F 73 20  1..As armas e os
00000011 62 61 72 C3 B5 65 73 20 61 73 73 69 6E 61 6C 61 64  bar..es assinalad
00000022 6F 73 2C 0A 51 75 65 20 64 61 20 6F 63 69 64 65 6E  os,.Que da ociden
00000033 74 61 6C 20 70 72 61 69 61 20 4C 75 73 69 74 61 6E  tal praia Lusitan
00000044 61 2C 0A 50 6F 72 20 6D 61 72 65 73 20 6E 75 6E 63  a,.Por mares nunc
00000055 61 20 64 65 20 61 6E 74 65 73 20 6E 61 76 65 67 61  a de antes navega
00000066 64 6F 73 2C 0A 50 61 73 73 61 72 61 6D 20 61 69 6E  dos,.Passaram ain
```

õ □ C3 B5

Latin-1

```
00000000 31 0A 0A 41 73 20 61 72 6D 61 73 20 65 20 6F 73 20  1..As armas e os
00000011 62 61 72 F5 65 73 20 61 73 73 69 6E 61 6C 61 64 6F  bar.es assinalado
00000022 73 2C 0A 51 75 65 20 64 61 20 6F 63 69 64 65 6E 74  s,.Que da ocident
00000033 61 6C 20 70 72 61 69 61 20 4C 75 73 69 74 61 6E 61  al praia Lusitana
00000044 2C 0A 50 6F 72 20 6D 61 72 65 73 20 6E 75 6E 63 61  ,.Por mares nunca
00000055 20 64 65 20 61 6E 74 65 73 20 6E 61 76 65 67 61 64   de antes navegad
00000066 6F 73 2C 0A 50 61 73 73 61 72 61 6D 20 61 69 6E 64  os,.Passaram aind
```

õ □ F5

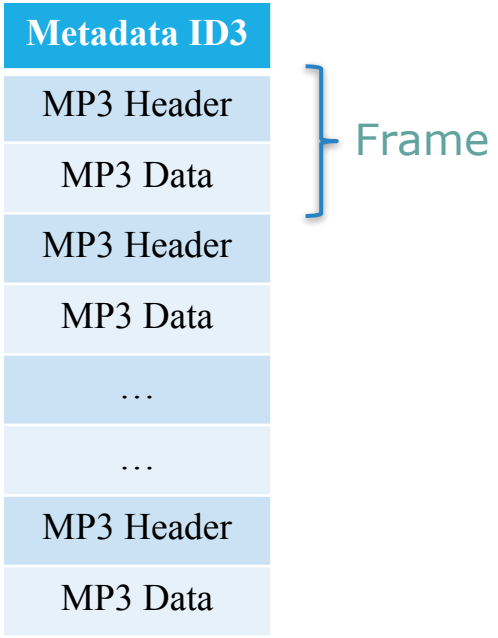
FICHEIROS BINÁRIOS

Os ficheiros binários estão organizados em elementos de *dimensão fixa* que podem ser de tipos simples ou registos (estruturas). Portanto, nos ficheiros binários não há marcas de "fim de elemento", porque o fim de cada elemento é sempre conhecido.

FORMATOS DE FICHEIROS BINÁRIOS

Exemplo: ficheiro MP3

- Composto por diversas frames, cada uma com um cabeçalho e um bloco de dados associados
- Os blocos de dados contêm a informação áudio
- Os cabeçalhos definem parâmetros cada bit tem um significado (em baixo)



FFFBA040 Colour-coding shows binary bit mapping to hex values below

Bits	1 2 3 4 5 6 7 8 9 10 11 12	13	14	15	16	17 18 19 20	21 22	23	24	25 26	27 28	29	30	31 32
Binary	1 1 1 1 1 1 1 1 1 1 1 1	1	0	1	1	1 0 1 0	0 0	0	0	0 1	0 0	0	0	0 0
Hex	F F F		B			A		0			4			0
Meaning	MP3 Sync Word	Version	Layer		Error Protection	Bit Rate	Frequency	Pad. Bit	Priv. Bit	Mode	Mode Extension (Used With Joint Stereo)	Copy	Original	Emphasis
Value	Sync Word	1 = MPEG	01 = Layer 3		1 = No	1010 = 160	00 = 44100 Hz	0 = Frame is not padded	Unknown	01 = Joint Stereo	0 = Intensity Stereo Off / 0 = MS Stereo Off	0 = Not Copy-righted	0 = Copy Of Original Media	00 = None

OPERAÇÕES COM FICHEIROS