

# Tipos básicos

**Programação I**  
**2020.2021**

*Teresa Gonçalves*  
[tcg@uevora.pt](mailto:tcg@uevora.pt)

Departamento de Informática, ECT-UÉ

# Sumário

## Tipos

## Conversão de tipos

# Tipos

# Tipos C

## **int**

Inteiro que pode ser positivo ou negativo

## **float**

Valor numérico com parte decimal

A melhor aproximação aos números reais

## **char**

Indicado entre plicas

# Caracteres

## Caracteres especiais

`\n` : mudança de linha

`\t` : tabulação

## Cadeia de caracteres (string)

Indicada entre aspas

Não é um tipo no C

# Representação

## **int**

Complemento para 2

Ocupa 2, 4 ou 8 bytes

## **float**

Vírgula flutuante

Ocupa 4 bytes

## **char**

Código específico: ASCII

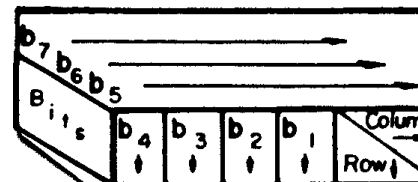
Ocupa 1 byte (8 bits)

# Código ASCII

## ASCII (American Standard Code for Information Interchange)

USASCII code chart

7 bits



					0	1	2	3	4	5	6	7
b4	b3	b2	b1	Column	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	\	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	\$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

## Extended ASCII

8 bits

## Unicode

UTF-8

UTF-16

UTF-32

# Valores booleanos

## Uma expressão com operadores relacionais origina um valor booleano

$x == y \rightarrow \text{True}$  se  $x$  é igual a  $y$

Não confundir com a atribuição

$x != y \rightarrow \text{True}$  se  $x$  é diferente de  $y$

$x < y \rightarrow \text{True}$  se  $x$  é menor que  $y$

$x > y \rightarrow \text{True}$  se  $x$  é maior que  $y$

$x <= y \rightarrow \text{True}$  se  $x$  é menor ou igual a  $y$

$x >= y \rightarrow \text{True}$  se  $x$  é maior ou igual a  $y$

## Em C não existe um tipo para valores booleanos

### São usados valores numéricos

0 : considerado Falso

$<>0$  : considerado Verdade



# Operadores lógicos

## Operadores lógicos

&& (and)

|| (or)

! (not)

## Tabela de verdade

a	b	a && b	a    b	!a
F	F	F	F	T
F	T	F	T	T
T	F	F	T	F
T	T	T	T	F

# Avaliação de expressões lógicas

## A avaliação é “short-circuited”

Apenas avalia o lado direito se necessário

## Comportamento

$a \ || \ b$

Se  $a == \text{Falso}$  então  $b$ , senão  $a$

$a \ \&\& \ b$

Se  $x == \text{Falso}$  então  $a$ , senão  $b$

$! \ a$

Se  $a == \text{Falso}$  então Verdade, senão Falso

# Conversão de tipos

# Conversão de tipos

## Conversão implícita

Se a expressão tiver 2 tipos compatíveis a expressão fica com o tipo mais geral

`char → int`

`int → float`

## Explícita (casting)

`(tipo) variável`

# *Casting*

## **(float) expr\_num**

Converte expr para um valor real

## **(int) expr\_num**

maior inteiro menor que expr (parte inteira do número)

## **(char) expr\_int**

Caracter correspondente a expr\_int (código ASCII)

## **(int) expr\_char**

Código ASCII correspondente a expr\_char

# Exercício 1

## Calcule o perímetro, a área e o volume

de uma circunferência, círculo e esfera (respetivamente) cujo raio é especificado pelo utilizador.

### A saber

pi: 3.14159265

perimetro:  $2 * \pi * r$

area:  $\pi * r^2$

volume:  $4 * \pi * r^3 / 3$

# Perimetro, area e volume

```
int main() {
```

```
float raio, perimetro, area, volume;
```

```
float pi = 3.14159265;
```

```
printf( "Insira o raio: ");
```

```
scanf( "%f", &raio );
```

```
perimetro = 2 * pi * raio;
```

```
area = pi * raio * raio;
```

```
volume = 4 * pi * raio * raio * raio / 3;
```

```
printf( "perimetro=%f\n", perimetro );
```

```
printf( "area=%f\n", area );
```

```
printf( "volume=%f\n", volume );
```

```
return 0;
```

```
}
```

*É possível declarar várias variáveis na mesma instrução*

*É possível declarar e atribuir valor na mesma instrução*

## Exercício 2

**Escreva um programa que lê um número real e escreve-o com 3 casas decimais**

Num: 1234.56789 → resultado: 1234.567



# Número com 3 casas decimais

```
int main() {
```

```
    float x, y;
```

```
    int n;
```

```
    printf( "Insira um número real: " );
```

```
    scanf( "%f", &x);
```

```
    y = x *1000;
```

```
    n = (int) y;
```

```
    y = n / 1000.0;
```

```
    printf("y=%f", y);
```

```
    return 0;
```

```
}
```

*n é convertido  
(implicitamente)  
para float antes  
da divisão*