Departamento de Matemática da Universidade de Coimbra		
2017/2018	**Métodos de Programação I**	**Folha 1** **(v2018/04/27; v477)**

Leitura e Escrita

```
int printf(char *format, arg1, arg2, ...)
int scanf(char *format, ...)
```

uma sequência de caracteres que especifica o formato de saída (entrada) e depois uma lista de argumentos.

Cada especificação de conversão (ver Tabela 1) começa com % e termina com um carácter de conversão. Entre o % e o carácter de conversão pode-se incluir, em ordem:

- Um sinal de menos, que especifica o ajuste à esquerda do argumento convertido.
- Um número que especifica a largura mínima do campo. O argumento será impresso em um campo, pelo menos, com essa amplitude. Se necessário será preenchido à esquerda (ou à direita, se o ajuste esquerdo for chamado) para compensar a largura do campo.
- Um ponto final '.', que separa a largura do campo da precisão.
- Um número, a precisão, que especifica o número máximo de caracteres a serem impressos a partir de uma «string», ou o número de dígitos após o ponto decimal de um valor de vírgula flutuante, ou o mínimo número de dígitos para um número inteiro.
- Um 'h' se o número inteiro for impresso como um «short», ou 'l' (éle) se for um «long».

1 Edite, compile e execute os seguintes programas:

1. Escrever a frase (sequência de caracteres) "Olá Mundo".

```c
// Olá mundo
#include <stdio.h>

int main() {
  printf("Olá Mundo!\n");
  return 0;
}
```

2. Escrever o inteiro "23" e o resultado de uma expressão com inteiros, alinhados à direita.

```c
// Escrita de inteiros (alinhados à direita)
#include <stdio.h>

int main() {
  printf("%5d\n",23);
  printf("%5d\n",2+3);
  return 0;
}
```

3. Escrever o inteiro "23" e o resultado de uma expressão com inteiros, alinhados à esquerda.
4. Escrever o real "23,5" e o resultado de uma expressão com reais, alinhados à esquerda.
5. Escrever o real "23,5" e o resultado de uma expressão com reais, alinhados à esquerda. Reais em formato cientifico.

Carácter	**Tipo de Argumento**	**Escrito como**
d,i	`int`	número decimal
o	`int`	número octal sem sinal (sem zeros à esquerda).
x, X	`int`	número hexadecimal sem sinal (sem '0x' ou '0X' à esquerda), utiliza-se 'abcdef' ou 'ABCDEF' para 10,..., 15
u	`int`	número decimal sem sinal
c	`int`	um carácter
s	`char *`	escreve os caracteres de uma sequência de caracters (*string*) até o carácter '\0' ou até ao números de caracteres dado pela precisão
f	`double`	[-]m.dddddd, aonde o número de 'ds' é dado pela precisão (por omissão, 6)
e,E	`double`	[-]m.ddddde±xx ou [-]m.dddddE±XX, aonde o número de 'ds' é dado pela precisão (por omissão, 6)
g,G	`double`	usar %e ou %E se o expoente é menor do que -4, ou maior ou igual do que a precisão; em caso contrário usar %f. Os zeros, assim como o ponto decimal, são omitidos quando no fim do número
p	`void *`	ponteiro (representação dependente da implementação)
%		sem argumento, escrever um %.
lf	double	**leitura** de um real do tipo `double`.
Lf	long double	**leitura** de um real do tipo `long double`.

Tabela 1: Especificadores de Formato

6. Escrever o resultado de uma expressão com inteiros formatando a saída de forma a mostrar a expressão, e.g. "$2 + 3 = 5$".

7. Ler dois argumentos do tipo inteiro e escrever o resultado da sua soma.

8. Escrever uma mensagem a pedir a dois argumentos do tipo inteiro e escrever o resultado, formatado, da sua soma.

9. Leia três caracteres e escreva-os por ordem inversa.

10. Leia um inteiro, um carácter, um real e de seguida escreva-os pela mesma ordem.