GUIÃO 01 – INTRODUÇÃO À LINGUAGEM DE PROGRAMAÇÃO C

A documentação da linguagem C pode ser consultada em https://en.cppreference.com/w/c

1. Usando um editor à sua escolha, crie um ficheiro com o programa seguinte (**`simplest_1.c`**).

```
int main ( void )
{
   // O programa mais simples em C
}
```

Nota algum pormenor estranho neste programa?

Use o seu compilador da linguagem C para compilar o ficheiro; analise eventuais mensagens do compilador. Execute o programa de aplicação resultante.

Deve usar sempre a *flag* de compilação **`-Wall`** que lhe permite obter avisos de compilação adicionais.

Experimente compilar o ficheiro usando diferentes normas da linguagem: **`-std=c17`** , **`-std=c11`** , **`-std=c99`** ou **`-std=c90`**. Analise eventuais mensagens do compilador.

Modifique o ficheiro de acordo com o exemplo seguinte.

```
#include<stdio.h>

int main ( void )
{
   printf ( "%ld\n", __STDC_VERSION__ );

   return 0;
}
```

Compile e execute o programa usando diferentes normas da linguagem. Analise eventuais mensagens do compilador e o resultado do programa.

Qual é o tipo da constante **__STDC_VERSION__** ? Qual é o significado do seu valor? Qual é a norma que o seu compilador usa por omissão?

2. Crie um ficheiro com o programa seguinte (**`hello_world.c`**). Compile e execute o programa.

```
#include<stdio.h>

int main ( void )
{
   puts ( "Hello World!" );

   return 0;
}
```

Experimente usar as funções **`printf()`** e **`fputs()`**. Há diferenças no resultado do programa?

3. Crie um novo programa (**`hello_X.c`**) que peça o nome próprio e o apelido do utilizador e escreva, por exemplo, "**`Hello Joaquim Madeira!`**".

Experimente usar as funções **`gets()`**, **`fgets()`** e **`scanf()`**. Analise eventuais mensagens do compilador.

Crie uma outra versão do exemplo em que o nome próprio e o apelido do utilizador são lidos a partir da **linha de comandos**.

4. Desenvolva um programa que escreva uma **tabela** com os **quadrados** e as **raízes quadradas** dos primeiros números naturais. O utilizador deve indicar quantas linhas tem a tabela.

Formate as colunas da tabela de modo apropriado; as colunas da tabela deverão ter um cabeçalho.

O que é necessário fazer para poder usar a função **`sqrt()`**?

5. Modifique o programa anterior para que escreva uma **tabela** com sucessivos valores do **seno** e do **cosseno**. O utilizador deve indicar o menor valor e o maior valor do ângulo (em **graus**), e o espaçamento entre sucessivos valores intermédios. Por exemplo:

```
ang sin(ang)      cos(ang)
--- ------------- -------------
  0  0.0000000000  1.0000000000
  5  0.0871557427  0.9961946981
 10  0.1736481777  0.9848077530
 15  0.2588190451  0.9659258263
 20  0.3420201433  0.9396926208
```

Use um **menor número de casas decimais** em cada coluna.

Modifique o programa para que a tabela seja escrita num **ficheiro**.

6. Desenvolva um programa que liste o número de bytes usados para representar os tipos primitivos da linguagem C, usando a função **`sizeof()`**.

Por exemplo, numa arquitetura de **32 bits** obteve-se:

```
sizeof(void *) ..... 4
sizeof(void) ........ 1
sizeof(char) ........ 1
sizeof(short) ....... 2
sizeof(int) ......... 4
sizeof(long) ........ 4
sizeof(long long) ... 8
sizeof(float) ....... 4
sizeof(double) ...... 8
```

Compare os resultados acima com aqueles que obtém numa arquitetura de **64 bits**.

7. Considere o programa em **Java** listado de seguida, que apresenta um simples exemplo de utilização de *arrays*.

Usando a **linguagem C**, desenvolva um programa equivalente com o mesmo tipo de funções.

```
/*
Crie um programa em C equivalente a este em Java.
*/

public class ProgA {

    public static void main(String[] args) {
        int[] a = {31,28,31,30,31,30,31,31,30,31,30,31};
        printArray("a", a);

        int[] b = new int[12];
        cumSum(a, b);
        printArray("b", b);
    }

    public static void printArray(String s, int[] a) {
        System.out.println(s + ":");
        for (int i=0; i<a.length; i++) {
            System.out.print(a[i]+" ");
        }
        System.out.println();
    }

    public static void cumSum(int[] a, int[] b) {
        int c = 0;
        for (int i=0; i<a.length; i++) {
            c += a[i];
            b[i] = c;
        }
    }
}
```

Atenção, na linguagem C:

- Uma função tem de ser definida (ou declarada) antes da sua primeira invocação.
- Um *array* é declarado de modo diferente.
- Quando um *array* é argumento de uma função, é também habitual passar como argumento o seu tamanho.