



## Programação 2

2º Semestre

### Folha de Exercícios

#### Conceitos básicos sobre programação

Efectue as tarefas de programação descritas abaixo, usando a linguagem C++ em ambiente Linux.

1. a) Escreva um programa que apresente no écran os argumentos da linha de comandos e o valor das variáveis de ambiente.

b) Modifique o programa anterior por forma a apresentar apenas o conteúdo das variáveis de ambiente HOME, TERM e PATH. Use as funções da biblioteca de C para a manipulação de strings.

2. Escreva um programa que teste a possibilidade de utilização de atexit handlers recorrendo à função `atexit`. O programa deverá registar dois desses handlers, cada um dos quais deverá, simplesmente, imprimir uma mensagem do tipo "Executing exit handler x", em que x é um número diferente para cada handler; o programa principal deverá escrever a mensagem "Main done!", imediatamente antes de terminar. Tire conclusões sobre a ordem de instalação dos handlers. Será possível instalar um handler mais do que uma vez ?

3. Teste o programa:

```
int main ()
{
    char aux[10000000];
    printf ("Ola Mundo\n");
}
```

a) O programa imprime a mensagem? Altere convenientemente o tamanho da variável `aux` de forma a imprimir a mensagem. Em que área da memória se encontra a variável `aux`?

4. Implemente a função `const char * inttoa (int i)` que converte um inteiro para `char *`:

a) usando uma variável global para retornar o resultado. Em que área da memória se encontra a variável?

b) usando uma variável estática para retornar o resultado. Em que área da memória se encontra a variável?

c) usando uma variável alocada dinamicamente para retornar o resultado. Em que área da memória se encontra a variável?

d) usando uma variável automática (local) para retornar o resultado. Funciona? Em que área da memória se encontra a variável?

5) a) Crie o ficheiro cabeçalho (header file) `helper.h` com a declaração da função `const char * inttoa (int i)`

b) Crie o ficheiro `helper.c` com a implementação da função `const char * inttoa (int i)`

c) Crie o ficheiro `teste1.c` com o programa principal. Vamos escrever um programa de teste que pede sucessivamente números ao utilizador e usa a função `inttoa` para imprimir o seu simétrico. Use a directiva de compilação `#include "helper.h"` para ter acesso à declaração da função.

d) Crie um segundo ficheiro `teste2.c` com um segundo programa principal. Este deve imprimir apenas os simétricos dos números positivos lidos.

6) Considere o seguinte excerto de um programa:

```
class TesteClass
{
public:
    TesteClass (){}
    ~TesteClass () {printf("destructor called\n");}
};

void removeLastElementFromVector (vector<TesteClass *>& vec)
{
    vec.pop_back();
}

main ()
{
    vector<TesteClass *> test;
    for(int i = 0; i < 10; i++)
        test.push_back(new TesteClass);
    removeLastElementFromVector (test);
}
```

a) Em que área da memória reside a variável `test`?

b) Em que área da memória residem os elementos do tipo `TesteClass` criados no `main`?

c) A função

`void removeLastElementFromVector (vector<TesteClass *>& vec)`  
não liberta correctamente toda a memória. Complete a função de forma adequada.