Exercícios de programação em C

Ligações úteis:

- Slides de *Visão geral da linguagem C* disponibilizados no moodle;
- Apontamentos de Programação Imperativa (CC1003), em particular:
  - Apontadores;
    Programação com apontadores;
- The C Book;
- Everything you need to know about pointers in C;

---

1. Considere o seguinte bloco de código:

   ```
   int i = 5;
   int *p = &i;
   ```

   - O que é a variável **p** e qual é o seu valor inicial?
   - Qual é o resultado de executar `p++;`?
   - E qual é o resultado de executar `*p++ = 0;`?

2. Tendo como referência o exemplo abaixo, escreva um programa que use apontadores para cada um dos três tipos de dados básicos (char; short, int ou long; e float ou double). Comece por declarar e iniciar uma variável para cada tipo básico juntamente com um apontador para essa variável. De seguida, utilize o operador sizeof para imprimir o tamanho de cada variável, o tamanho do seu endereço de memória, o tamanho de cada apontador e o tamanho do conteúdo apontado por cada apontador. Por fim, imprima o endereço e o conteúdo de cada uma das variáveis.

   ```
   char c, *cptr;
   c = 'a';
   cptr = &c;
   printf("tamanho de um char: %lu\n", sizeof(c));
   printf("tamanho do endereço de um char: %lu\n", sizeof(&c));
   printf("tamanho de um apontador para um char: %lu\n", sizeof(cptr));
   printf("tamanho do conteúdo apontado por um apontador para um char: %lu\n",
sizeof(*cptr));
   printf("Os valores apontados pelos endereços '%p' e '%p' são '%c' e '%c'\n", &c,
cptr, c, *cptr);
   ```

3. Considere a seguinte declaração:

   ```
   int x[3] = {23, 41, 17};
   ```

Qual é o valor das expressões que se seguem:

- x[0]
- x[1]
- x[2]
- x
- *x
- x+1
- *(x+1)
- x+2
- *(x+2)
- &(x[0])
- *&(x[0])
- &*(x[0])

4. Vectores de caracteres versus strings.
   - Complete o exemplo abaixo de forma a imprimir os valores e os endereços de um vector de caracteres com as 10 primeiras letras do alfabeto.

```
#include <stdio.h>
#define SIZE 10

void show_vector(char *a, int n);

int main() {
  char v[SIZE] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j'};
  show_vector(v, SIZE);
  return 0;
}

void show_vector(char *a, int n) {
  int i;
  for (i = 0; i < n; i++) {
    /* imprima aqui os valores e endereços de cada elemento do vector */
  }
}
```

   - Que alterações são necessárias fazer ao programa para se poder imprimir o conteúdo do vector v com a instrução:

```
printf("%s", v);
```

5. Escreva o programa **mat2file** que pede ao utilizador que introduza os elementos (inteiros) de uma matriz, e que os armazena num ficheiro (utilizando como separador o caracter espaço). As dimensões da matriz e o nome do ficheiro são passados através da linha de comando (e.g., `mat2file 2 3 matriz.txt`), sendo os elementos pedidos, um a um, ao utilizador. No ficheiro, os dois primeiros elementos são o número de linhas e de colunas, sendo os restantes os elementos da matriz. Por exemplo, a matriz:

```
10 20 30
40 50 60
```

deve ser armazenada como:

```
2 3 10 20 30 40 50 60
```

Funções a ter em conta: **fopen**, **fclose**, **fprintf**, **fscanf**, e **atoi**.

6. Escreva o programa **file2mat** que lê um ficheiro criado pelo **mat2file**, cujo nome deve ser passado na linha de comando (e.g., `file2mat matriz.txt`) e imprime o seu conteúdo no ecrã, com os elementos da mesma linha e de diferentes colunas separados por tabs (`\t`).