

# Programação I

## Vetores e matrizes (ficha 8)

Considere as seguintes variáveis para os exemplos mencionados nas perguntas abaixo.

```
a={1,2,2};  
b={1,2,3,4,5};  
c={2,4};  
d={7,8};
```

1. Crie a função `verifica_ordem( int vec[], int sz )` que verifica se os elementos do vetor `vec` estão ordenados por ordem crescente (devolve 1 se estiver ordenada e 0 caso contrário. Por exemplo, `verifica_ordem( a, 3 )` deve devolver 1.
2. Crie a função `conta_elementos( int v1[], int v2[], int sz1, int sz2 )` que permite contar o número de elementos comuns entre 2 vetores. Por exemplo,  
`conta_elementos( b, c, 5, 2 )` devolve 2 e  
`conta_elementos( b, d, 5, 2 )` devolve 0.  
*Sugestão: implemente uma função auxiliar que verifica se um elemento existe num vetor:*  
`int existe_elemento( int el, int vec[], int sz )`.
3. Crie a função `int matriz_identidade( int matriz[][10], int n )` que permite verificar se uma matriz quadrada é a matriz identidade.
4. Implemente um programa para ler valores (não ordenados) correspondentes ao tempo (em segundos) que vários atletas demoraram para correr 100m e mostrar o top3 dos tempos mais baixos. Para isso deve, no programa principal, ler quantos atletas existem e o tempo de cada um, separadamente. O vetor dos tempos deve ser passado como argumento às funções `float primeiro( float v[], int sz )`, `float segundo( float v[], int sz )` e `float terceiro( float v[], int sz )` que calculam os 3 melhores tempos.

```
quandos tempos pretende inserir? 4  
tempo do atleta 1: 11.0  
tempo do atleta 2: 9.0  
tempo do atleta 3: 12.0  
tempo do atleta 4: 11.5  
primeiro: 9.0  
segundo: 11.0  
terceiro 11.5
```

5. Implemente a função `acumulado(int vec[], int sz, int index)`, que devolve o acumulado dos valores dos elementos do vetor entre o índice 0 e `index`. Por exemplo, `acumulado([1,2,3], 3, 1)` devolve 3.
6. Na cadeira de Arquitetura a avaliação é contínua: todas as semanas há um trabalho que recebe classificação entre -1 e 20 (-1 significa que o trabalho não foi entregue). A nota final é a média arredondada às unidades das notas dos trabalhos **descontando a melhor e pior nota**. Escreva um programa que lê a sequência de notas do aluno (guardando-as num vetor) e mostra a nota final. O semestre tem 15 semanas (por isso, esse deve ser o tamanho do vetor). Por exemplo se as notas forem [-1, 10, 11, 12, 13, -1, 15, 8, 9, 15, 10, 12, 19, -1, 20], as 2 notas que não contam para a média são -1 e 20.

*Sugestão: para a média contam sempre 13 notas. Utilize um ciclo para somar todas as notas; no final, subtraia a maior e menor encontradas para o cálculo da média.*

7. Altere o programa anterior de modo permitir que o aluno possa não entregar até 3 trabalhos sem ser penalizado (ou seja, no máximo 3 notas -1 não contam para a média).
8. Assuma que o utilizador introduz uma sequência de números terminando com um número negativo (no máximo, 20 valores). Crie um programa para ler esses números para um vetor e que peça um número para pesquisar o vetor. O programa deve indicar o tamanho da maior sequência que esse número ocorre no vetor. Por exemplo, para a sequência 1,2,3,4,4,1,3,4,-1 e num=4, a função devolve 2. Utilize a função `int contaRepetidos(int v[], int sz, int num)`.
9. Escreva um programa que cria, preenche e faz o somatório de um vetor de números reais mostrando, no final, quer o vetor introduzido, quer o somatório. Assuma que é solicitado ao utilizador quer o n<sup>o</sup> quer os valores a introduzir (para definir o tamanho do vetor, assumo que n<sup>o</sup> máximo de elementos a introduzir é 15). Utilize as seguintes funções para desenvolver o programa:
  - `float calcula_somatorio(float v[], int n)`
  - `void imprimir_vetor(float v[], int n)`
10. Escreva um programa que preenche um vetor com 10 reais, introduzidos pelo utilizador. Depois deverá imprimir:
  - a média dos elementos. Para tal implemente a função `float media(float v[])`
  - o maior elemento. Para tal implemente a função `float minimo(float v[])`
  - o menor elemento. Para tal implemente a função `float maximo(float v[])`
  - o conteúdo do vetor.

Utilize a função `imprimir_vetor(float v[], int n)` implementada no exercício anterior.