



**Exercícios de Fundamentos de Algoritmos de Computação I - Professor Leonardo Vianna  
[2021/2]**

**Questão 01:**

Analise o código apresentado a seguir e forneça todos os valores exibidos durante a execução do programa:

```
void funcao01 (int v[], int tam, int pos) {
    int i;
    for (i=tam-1;i>=pos;i-=2) {
        v[i] = v[i]*2;
    }
}

void funcao02 (int x) {
    x++;
}

void funcao03 (int v[], int tam) {
    int i;
    for (i=0;i<tam;i++) {
        printf ("%d  ", v[i]);
    }
    printf ("\n");
}

int main () {
    int i, vet[5] = {1,2,3,4,5};
    int quant = 5;

    for (i=0;i<quant;i++) {
        if (i%2==0) {
            funcao01 (vet, quant, i);
        }
        else {
            funcao02 (vet[i]);
        }
        funcao03 (vet, quant);
    }
}
```

**Questão 02:**

Considere a existência de dois conjuntos numéricos  $A$  e  $B$  contendo  $n_1$  e  $n_2$  elementos, respectivamente. Pede-se o desenvolvimento de uma função que determine se um dos conjuntos está contido no outro, retornando os seguintes códigos:

- 1, se  $A$  estiver contido em  $B$ ;
- 2, se  $B$  estiver contido em  $A$ ;
- 0, caso contrário.

**Questão 03 [ENADE 2011]:**

Considerando a execução do algoritmo abaixo, responda ao que se pede nos itens a e b.

```
01 algoritmo Vetores
02 variaveis
03   vetA[1..10], vetB[1..10], i: inteiro
04 inicio
05   para i ← 1 ate 10 passo 1 faça
06     vetB[i] ← 0
07     se resto(i,2) = 0 entao
08       vetA[i] ← i
09     senão
10       vetA[i] ← 2 * i
11     fimse
12   fimpara
13   para i ← 1 ate 10 passo 1 faça
14     enquanto(vetA[i] > i)
15       vetB[i] ← vetA[i]
16       vetA[i] ← vetA[i] - 1
17     fimenquanto
18   fimpara
19 fimalgoritmo
```

- a) Apresente os dados dos vetores  $vetA$  e  $vetB$  ao término da execução da linha 12.
- b) Apresente os dados dos vetores  $vetA$  e  $vetB$  ao término da execução da linha 19.

**Questão 04 [ENADE 2008]:**

```
1 funcao busca(V[0..9] : inteiro, K : inteiro):
  inteiro
2 variaveis
3   C, F, K, M : inteiro
4 inicio
5   F ← 9
6   [ ]
7   enquanto ((V[M] <> K) ou (F > C))
8     [ ]
9     se (K < V[M]) entao
10       F ← M - 1
11     senao
12       [ ]
13   fim enquanto
14   se (V[M] <> K) entao
15     retorne (0)
16   senao
17     retorne (M)
18   fim se
```

19 fim

O algoritmo representado pelo pseudocódigo acima está incompleto, pois faltam 3 linhas de código. A função *busca* desse algoritmo recebe um vetor ordenado de forma crescente e um valor a ser pesquisado. A partir disso, essa função verificará se o número armazenado no ponto mediano do vetor é o número procurado. Se for o número procurado, retornará o índice da posição do elemento no vetor e encerrará a busca. Se não for, a função segmentará o vetor em duas partes a partir do ponto mediano, escolherá o segmento no qual o valor procurado está inserido, e o processo se repetirá. A partir dessas informações, assinale a opção que contém os comandos que completam, respectivamente, as linhas 6, 8 e 12 do algoritmo.

- a ( )  $C \leftarrow 0$       $M \leftarrow (C + F)/2$       $C \leftarrow M + 1$   
b ( )  $C \leftarrow 1$       $M \leftarrow (C + F)/2$       $C \leftarrow M - 1$   
c ( )  $C \leftarrow 0$       $C \leftarrow M + 1$       $M \leftarrow (C + F)/2$   
d ( )  $C \leftarrow 1$       $C \leftarrow M + 1$       $M \leftarrow (C + F)/2$   
e ( )  $C \leftarrow 1$       $M \leftarrow (C + F)/2$       $C \leftarrow M + 1$

**Questão 05:**

Implementar uma função que, dado um vetor contendo números reais, determine o maior e o segundo maior elementos.

*Nota:* considerar que não há repetição de elementos no vetor.

**Questão 06 [ENADE 2014 - adaptada]:**

Observe o programa classificador ("sort"), em pseudocódigo, apresentado abaixo.

```
1  início
2  variável texto nome[5]
3  variável real nota[5]
4  variável inteiro i, j
5  variável real aux
6  variável texto naux
7  para i de 1 até 5
8    escrever "Nome ", i, " = "
9    ler nome[i-1]
10   escrever "Nota ", i, " = "
11   ler nota[i-1]
12 fimpara
13 para i de 0 até 4
14   para j de i+1 até 4
15     se nota[i] <= nota[j] então
16       aux ← nota[i]
17       nota[i] ← nota[j]
18       nota[j] ← aux
19     naux ← nome[i]
20     nome[i] ← nome[j]
21     nome[j] ← naux
```

22 fimse

23 fimpara

24 fimpara

25 para i de 1 até 5

26 escrever nome[i-1], " : ", nota[i-1], "\n"

27 fimpara

28 fim

Este programa classifica, em ordem:

- a ( ) decrescente, notas de alunos e nomes de alunos de mesma nota.  
b ( ) alfabética crescente, nomes e notas de alunos de mesmo nome.  
c ( ) decrescente, notas de alunos.  
d ( ) alfabética crescente, nomes de alunos.  
e ( ) crescente, notas de alunos.

**Questão 07 [ENADE 2014]:**

Considere uma situação em que um professor que queira saber se existem alunos cursando, ao mesmo tempo, as disciplinas A e B, tenha implementado um programa que:

- 1) Inicializa um array *a* de 30 posições que contém as matrículas dos alunos da disciplina A;
- 2) Inicializa um array *b* de 40 posições que contém as matrículas dos alunos da disciplina B;
- 3) Imprime a matrícula dos alunos que estão cursando as disciplinas A e B ao mesmo tempo.

Considere, ainda, que os arrays foram declarados e inicializados, não estão necessariamente ordenados, e seus índices variam de 0 a *n*-1, sendo *n* o tamanho do array.

```
1 for ( i = 0 to 29 ) {
2   for ( j = 0 to 39 ) {
3
4
5
6   }
7 }
```

Com base nessas observações, conclui-se que o trecho a ser incluído nas linhas 3, 4 e 5 do código acima, para que o programa funcione corretamente, é:

- a ( ) 3. if (a[i] == b[j]) {  
4. print(a[i]);  
5. }  
b ( ) 3. if (a[j] == b[i]) {  
4. print(a[j]);  
5. }

c ( ) 3. if (a[i] == b[j]) {  
4. print(a[j]);  
5. }

d ( ) 3. if (a[i] == b[i]) {  
4. print(a[i]);  
5. }

e ( ) 3. if (a[j] == b[j]) {  
4. print(a[j]);  
5. }

**Questão 08:**

Faça uma função que, dado um vetor de números inteiros, exiba para cada um de seus elementos a quantidade de vezes que o mesmo aparece no vetor.

Exemplo:

Vetor = {3,5,1,3,2,5,7,3,4,7,6,1}

Saída:

3: 3 vezes

5: 2 vezes

1: 2 vezes

2: 1 vez

7: 2 vezes

4: 1 vez

6: 1 vez