

## Exercícios – TVC3

Esta lista de exercícios foi elaborada como preparatório para a terceira avaliação que será realizada em breve, abordando matrizes e estruturas, além dos conteúdos de repetições, vetores numéricos e strings vistos anteriormente. Algumas instruções:

- Desenvolva todos os problemas em linguagem C;
- Quando for pedido para desenvolver uma função que resolva um determinado problema, está implícito que a função principal que fará uso desta função também terá que ser desenvolvida;
- Após terminar o programa, faça o teste de mesa com diferentes entradas, para verificar se o seu programa de fato resolve o problema proposto.

**01.** Construa uma estrutura para representar e armazenar os dados contidos na tabela abaixo. Faça um programa para preencher a estrutura com os dados e para, em seguida, imprimir tais dados.

Nome	Avenida/Rua	Nº	Bairro	Compl.	CEP	Cidade	UF	Tel. 1	Tel.2
João	Av. Rio Branco	421	Centro	Apto. 501	36.046-030	Juiz de Fora	MG	(32) 3236-5539	(32) 98415-7873
Carolina	R. Miguel de Frias	125/A	Icaraí	-	36.240-000	Niterói	RJ	(21) 2613-5671	(21) 98856-9091
Jefferson	R. São Clemente	-	Botafogo	Bloco IV	22.250-040	Rio de Janeiro	RJ	-	99194-0921

**02.** Faça um programa principal que crie duas matrizes quadradas, com 4 linhas e 4 colunas, para armazenar valores reais. A primeira matriz deve ser preenchida com valores informados pelo usuário. Desenvolva as funções abaixo para armazenar na segunda matriz, o conteúdo solicitado relativo à primeira matriz.

a) *transposta* (*float mat1[4][4]*, *float mat2[4][4]*): a segunda matriz recebe a transposta da primeira (linhas e colunas invertidas).

b) *diagonal* (*float mat1[4][4]*, *float mat2[4][4]*): a diagonal principal de *mat2* recebe a diagonal principal de *mat1*. Os demais elementos da segunda matriz devem ser nulos.

c) *soma* (*float mat1[4][4]*, *float mat2[4][4]*): a segunda matriz recebe a soma da primeira matriz com ela mesma.

d) *media* (*float mat1[4][4]*, *float mat2[4][4]*): a média dos valores de cada linha da primeira matriz é adicionada à respectiva linha da segunda matriz na posição da diagonal principal. Exemplo: a média dos valores da linha 0 de *mat1* é adicionada na posição [0][0] de *mat2*. A média da linha 1 é adicionada na posição [1][1] de *mat2* etc..

e) *maior\_e\_menor* (*float mat1[4][4]*, *float mat2[4][4]*): as linhas pares da segunda matriz são preenchidas com o maior elemento da primeira, da matriz ao passo que as linhas ímpares recebem o menor elemento de *mat1*.

f) *acima\_da\_media* (*float mat1[4][4]*, *float mat2[4][4]*): *mat2* recebe apenas os elementos de *mat1* cujo valor está acima da média dos valores da primeira matriz. As demais posições não preenchidas de *mat2*, se existirem, devem receber 0.

g) *ordenacao* (*float mat1[4][4]*, *float mat2[4][4]*): a segunda matriz recebe, posição a posição, os elementos da primeira matriz ordenados crescentemente.

**03.** Crie um tipo de dado denominado `Vetor` que representa uma estrutura chamada `est_Vetor` para representar um vetor de dimensões (coordenadas)  $x$ ,  $y$  e  $z$  no espaço tridimensional:  $V(x, y, z)$ . Desenvolva uma função para calcular e retornar a soma de dois vetores  $A(x_a, y_a, z_a)$  e  $B(x_b, y_b, z_b)$  e uma função para calcular e retornar a multiplicação de  $A$  pela coordenada de maior valor de  $B$ . Faça um programa principal que leia do usuário as coordenadas dos dois vetores e que chame as duas funções citadas para imprimir os resultados retornados por elas.

**04.** Desenvolva uma função que receba como parâmetro duas variáveis da estrutura `est_Data`, representada abaixo (tipo de dado `Data`):

```
typedef struct est_Data
{
    int dia;
    int mes;
    int ano;
} Data;
```

Essa estrutura representa uma data válida e é composta por três valores inteiros: dia, mês e ano. A função deve retornar o número de dias que separa uma data da outra.

**05.** Elabore uma estrutura composta por três matrizes quadradas de dimensões iguais a 4 e dois vetores de tamanho igual a 16. No programa principal, faça a leitura de uma matriz 4x4 e armazene-a na primeira matriz da estrutura. A seguir, desenvolva funções realizar as seguintes tarefas:

- a) armazenar na segunda matriz da estrutura, a transposta da matriz lida;
- b) armazenar na terceira matriz da estrutura, a matriz lida multiplicada pelo seu elemento de maior valor;
- c) armazenar no primeiro vetor da estrutura, de forma sequencial (linha por linha), a matriz lida;
- d) armazenar no segundo vetor da estrutura, o resultado do produto da matriz lida por sua transposta.

**06.** Preocupada com o aumento dos valores da conta de energia, uma família deseja reduzir o gasto energético que tem com os seus eletrodomésticos. Assim, a família contratou os serviços de um programador para criar um sistema de controle do consumo de energia da casa. Assuma a função do programador e desenvolva um programa para representar e ler os 7 eletrodomésticos da casa, cada um deles possuindo os seguintes dados: nome, potência (em kilowatts) e o tempo diário de funcionamento (em horas). O programa deve ler um intervalo de tempo (em dias) para o qual o consumo total da casa e o consumo relativo a cada eletrodoméstico (% consumo/consumo total) devem ser calculados e impressos.

**07.** O sistema de uma biblioteca online deve ser capaz de procurar um dado livro pelo o seu título ou por parte dele. Dessa forma, desenvolva um programa para registrar 6 livros no sistema. O programa deve solicitar ao usuário o título (ou parte dele) do livro a ser buscado e imprimir o resultado da busca com todos os dados do livro em questão. Cada livro é armazenado com o seu título (máximo de 50 caracteres), autor (máximo de 30 caracteres), código e preço.

**08.** Um *quadrado mágico* é uma matriz quadrada, de valores inteiros, na qual a soma dos elementos de cada linha, a soma dos elementos de cada coluna e a soma dos elementos da diagonal primária e da diagonal secundária são iguais. A matriz quadrada de ordem 4 abaixo é um exemplo de quadrado mágico, no qual todas as somas são iguais a 34:

4	14	15	1
9	7	6	12
5	11	10	8
16	2	3	13

Desenvolva um programa que leia do usuário uma matriz quadrada de ordem  $n$  e imprima se a matriz é um quadrado mágico ou não.

**09.** A matriz triangular abaixo é conhecida como triângulo de Pascal:

1	1	1	1	1
1	2	3	4	
1	3	6		
1	4			
1				

Figura 1

Note que cada elemento dessa matriz é composto pela soma do elemento antecessor com o elemento imediatamente acima. Outra forma de representação do triângulo de Pascal encontra-se abaixo:

1				
1	1			
1	2	1		
1	3	3	1	
1	4	6	4	1

Figura 2

Observe que cada elemento dessa matriz é composto pela soma do elemento imediatamente acima com o antecessor do elemento imediatamente acima.

Crie duas funções para impressão do triângulo de Pascal, sendo uma função para imprimi-lo de acordo com a Figura 1 e outra função para imprimi-lo de acordo com a Figura 2. Essas funções devem utilizar matrizes para cálculo dos elementos do triângulo de Pascal e devem receber o número de linhas da matriz por parâmetro. Note que a matriz em questão é uma matriz quadrada. Crie um método principal para receber do usuário o número de linhas da matriz. Esse método deve chamar as duas funções de impressão de triângulo de Pascal criadas anteriormente.

**10.** Um número real pode ser representado, por exemplo, pela estrutura abaixo, na qual *esq* e *dir* representam os dígitos à esquerda e à direita do ponto decimal, respectivamente. Se *esq* for um inteiro negativo, o número real representado será negativo.

```
struct numeroReal
{
    int esq;
    int dir;
};
```

- a) Escreva uma função que receba como parâmetro a estrutura acima e retorne o número real representado por ela.
- b) Elabore as funções *soma*, *subtracao* e *multiplicacao* que tenham como parâmetros duas variáveis do tipo da estrutura acima e que armazenem o resultado da soma, subtração e multiplicação, respectivamente, dos dois parâmetros de entrada, em uma terceira estrutura. O número real representado pela última estrutura deve ser impresso por cada uma das funções.

**11.** Uma empresa deseja desenvolver um sistema que contém os dados de seus funcionários. O cadastro de cada um deles deve conter os seguintes campos: nome, endereço, telefone, e-mail, data de aniversário (dia, mês e ano), ano de chegada à empresa e um campo para possíveis observações. Desenvolva um sistema para a agenda proposta com um menu para executar cada uma das seguintes ações:

- a) Ler a quantidade de funcionários;
- b) Cadastrar os funcionários;
- c) Imprimir uma lista com os dias de nascimento dos funcionários nascidos em um mês desejado;
- d) Imprimir o nome e o telefone dos funcionários com uma quantidade específica de anos de empresa;
- e) Imprimir o funcionário que possui o campo de observações com preenchimento mais extenso;
- f) Verificar se o e-mail de um funcionário possui o caractere '@' e a string ".com";
- g) Imprimir os dados de todos os funcionários.

**12.** Alguns produtos são elaborados utilizando-se para isso uma receita com quantidades fixas de cada ingrediente. Para esse exercício, pede-se:

- a) Crie a estrutura *Produto*. Essa estrutura é composta por um *Código* (número inteiro somente para controle do sistema), um *Nome* e uma *Unidade*. Exemplo de valores para cada campo dessa estrutura: 1, "Farinha de Trigo", "gramas".
- b) Crie a estrutura *Estoque*. Essa estrutura é composta por um *Produto* e uma *Quantidade* (número do tipo real).
- c) Crie um vetor de *Estoque* para representar o estoque total de sua cozinha. Cada elemento *Estoque* deve ser inserido na posição do vetor que corresponde ao código de seu *Produto*;
- d) Crie a estrutura *Ingrediente*. Um ingrediente é composto por um *Produto* e uma *Quantidade* (número do tipo real).
- e) Crie uma estrutura para representar uma receita. Uma receita é composta por um *Nome* e um vetor de ingredientes.
- f) Crie um vetor de receitas para representar todas as receitas existentes em sua cozinha.
- g) Crie uma função que receba uma receita e o estoque total da cozinha. Essa função deve retornar 1 se existir estoque suficiente para a receita ou 0 se o estoque for insuficiente.

h) Crie uma função que receba uma receita e o estoque total da cozinha. Essa função deve subtrair do estoque da cozinha as quantidades dos respectivos itens da receita.

i) Crie uma função principal que permita ao usuário digitar o índice da receita desejada. Se existir estoque suficiente para que a receita seja feita (função criada em (e)), o sistema deve imprimir "Receita liberada!" e deve subtrair o estoque de cada ingrediente do estoque da cozinha (função criada em (f)). Se não houver estoque suficiente para a receita, o sistema deve apresentar a mensagem "Estoque insuficiente!". O sistema deve continuar solicitando o índice da receita e tratando o estoque até que receba um índice inválido.

**13 – DESAFIO.** O quebra-cabeça abaixo foi popular entre crianças nascidas na década de 80. O objetivo desse quebra-cabeça é posicionar as letras em ordem alfabética.

T	R	G	S	J
X	D	O	K	I
M		V	L	N
W	P	A	B	E
U	Q	H	C	F

Note que existe um único espaço que não possui letra alguma. Uma letra pode ser movida para o espaço somente se estiver imediatamente à direita, esquerda, acima ou abaixo do mesmo.

a) Faça uma função que leia do teclado cinco strings passadas por parâmetro pelo usuário. Cada string passada deve:

- Representar uma linha da matriz de nosso quebra-cabeça;
- Possuir exatamente cinco caracteres;
- Possuir o caractere espaço ou caracteres entre A e X;
- Possuir somente caracteres que ainda não se encontram na matriz.

b) Faça uma função que receba um inteiro como parâmetro. O tratamento do parâmetro passado deve ser:

- Se o inteiro passado for 8, a função deve trocar o elemento espaço com o elemento imediatamente acima do espaço.
- Se o inteiro passado for 2, a função deve trocar o elemento espaço com o elemento imediatamente abaixo do espaço.
- Se o inteiro passado for 4, a função deve trocar o elemento espaço com o elemento imediatamente à esquerda do espaço.
- Se o inteiro passado for 6, a função deve trocar o elemento espaço com o elemento imediatamente à direita do espaço.

c) Faça uma função para imprimir o tabuleiro.

d) Faça uma função que retorne 1 se o tabuleiro estiver em ordem alfabética ou 0 se o tabuleiro não estiver em ordem alfabética. Para checar se o tabuleiro está em ordem alfabética, utilize o comando *for*.

e) Crie a função principal. Essa função deve primeiramente chamar a função criada em (a) para leitura do tabuleiro. Após isso, a função deve solicitar ao usuário uma jogada (8, 2, 4 ou 6) e esse valor deve ser passado como parâmetro para a função criada em (b). Após cada jogada, o tabuleiro deve ser impresso, utilizando para isso a função criada em (c). O programa deve continuar solicitando jogadas até que a função criada em (d) retorne 1.