# Programação de Computadores I – BCC701

## Caderno de Exercícios

## Módulo 7 (parte 2) – Matriz

2020/1

Elaborado pela Comissão de Unificação da Disciplina BCC701,
com a colaboração de professores e estagiários docentes
*http://www.decom.ufop.br/bcc701/*

DECOM – Departamento de Computação
ICEB – Instituto de Ciências Exatas e Biológicas
UFOP – Universidade Federal de Ouro Preto

# Sumário

MÓDULO

# Estruturas Homogêneas

**Atenção:** Para a resolução de alguns dos exercícios você poderá utilizar as funções apresentadas nas Seção 7.3, para a manipulação de vetores e matrizes.

### Questão 7.1 (2013-2)

A transposta de uma matriz **A**, de dimensão $m \times n$, é uma matriz **AT**, de dimensão $n \times m$, onde cada linha de **AT** é a coluna de número correspondente de **A**, e vice versa. Por exemplo:

$$A = \begin{bmatrix} 3 & 8 & 1 \\ 0 & 4 & 9 \\ 5 & 6 & 0 \end{bmatrix}, \quad AT = \begin{bmatrix} 3 & 0 & 5 \\ 8 & 4 & 6 \\ 1 & 9 & 0 \end{bmatrix}$$

Implemente um programa para ler uma matriz e calcular e imprimir a transposta dessa matriz, tal como mostrado no exemplo de execução a seguir. O programa deve calcular a matriz transposta usando comandos de repetição aninhados.

*Exemplo 1:*

```
Digite a matriz: 1, 2, 3; 4, 5, 6
Matriz transposta:
1    4
2    5
3    6
```

### Questão 7.2 (2014-1)

O traço de uma matriz quadrada **A**, de dimensão $n \times n$, é a soma dos elementos de sua diagonal principal, ou seja:

$$\text{traço}(A) = \sum_{i=1}^{n} a_{i,i}$$

Implemente um programa que defina uma matriz quadrada **A**, de valores reais, através de uma entrada do usuário.

Complete o programa para:

1. criar e imprimir um vetor constituído pelos elementos da diagonal principal de A;
2. calcular e imprimir o traço da matriz (com precisão de 2 casas decimais);
3. determinar o número de elementos fora da diagonal principal que são nulos.

*Exemplo 1:*

```
Digite a matriz A: 2.5, 0.23, 0; 0, 3.82, 4; 0, 18, 0
Vetor da diagonal: [ 2.5, 3.82, 0.0 ]
Traço(A) = 6.32
Elementos nulos fora da diagonal principal: 3
```

### Questão 7.3 (2014-2)

Uma matriz triangular superior é uma matriz quadrada onde todos os elementos abaixo da diagonal principal são nulos (com valor zero), conforme ilustrado a seguir:

$$\begin{bmatrix} 2 & 3 & 8 & 9 \\ 0 & 1 & 4 & 6 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 8 \end{bmatrix}$$

Implemente um programa que leia uma matriz quadrada (de números inteiros), tal como mostrado no exemplo a seguir, e indique se ela é uma matriz triangular superior ou não. O programa deve também imprimir a dimensão da matriz lida. Não é necessário verificar se a matriz lida é uma matriz quadrada.

*Exemplo 1:*

```
Digite a matriz: 2, 3, 8, 9; 0, 1, 4, 6; 0, 0, 2, 1; 0, 0, 0, 8
Dimensão da matriz: 4x4
A matriz é triangular superior
```

*Exemplo 2:*

```
Digite a matriz: 1, 1, 1; 1, 0, 1; 0, 0, 1
Dimensão da matriz: 3x3
A matriz não é triangular superior
```

**Questão 7.4 (2016-1)**

A tabela a seguir mostra os resultados dos últimos $n$ jogos entre Cruzeiro e Atlético.

| Atlético | Cruzeiro |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 1 | 0 |
| 3 | 1 |
| 3 | 1 |
| 1 | 1 |
| 2 | 3 |
| 1 | 3 |
| 3 | 2 |
| 1 | 5 |

Implemente um programa que leia uma matriz com $n$ linhas e 2 colunas representando a tabela de placares dos jogos entre os dois times, onde a primeira coluna representa o Atlético e a segunda coluna representa o Cruzeiro. O programa deve calcular a quantidade de vitórias e de empates de cada time. Também, o programa imprime qual é o melhor time, de acordo com o número de vitórias, ou se os times empataram neste quesito. A leitura da matriz deve ser realizada com a entrada de todos os elementos de uma única vez. As impressões seguem os exemplos de execução apresentados.

*Exemplo 1:*

```
Matriz de jogos: 0, 0; 1, 0; 0, 1; 1, 3; 3, 1; 1, 1; 2, 3; 1, 3; 3,2;
   1, 5
Vitórias do Atlético: 3
Vitórias do Cruzeiro: 5
Empates  nos  jogos : 2
O Cruzeiro é o melhor time
```

*Exemplo 2:*

```
Matriz de jogos: 0, 0; 1, 0; 0, 1; 1, 0; 3, 1; 1, 1; 2, 0; 1, 3; 3, 2;
    1, 5
Vitórias do Atlético: 5
Vitórias do Cruzeiro: 3
Empates  nos  jogos : 2
O Atlético é o melhor time
```

*Exemplo 3:*

```
Matriz de jogos: 0, 0; 1, 1; 0, 1; 1, 0; 1, 1; 1, 1; 2, 0; 3, 3; 2, 2;
    1, 5
Vitórias do Atlético: 2
Vitórias do Cruzeiro: 2
Empates  nos  jogos : 6
Os times tiveram a mesma quantidade de vitórias
```

## Questão 7.5 (2013-2)

Saber da existência de caminho entre duas cidades é uma tarefa fácil para seres humanos, podendo ser realizada com uma simples consulta visual a um mapa. Porém, quando levamos este problema para o mundo computacional precisamos representá-lo de outra forma. Uma forma de fazer tal representação é: utilizando uma matriz, consideramos que os índices da matriz representam as cidades e cada elemento de uma entrada (i, j) da matriz representa se há caminho (valor 1) ou não há caminho (valor 0) entre as cidades de números i e j.

| i \ j | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 |
| 2 | 1 | 0 | 1 | 0 |
| 3 | 1 | 1 | 0 | 1 |
| 4 | 0 | 0 | 1 | 0 |

Implemente um programa que faça a leitura da matriz, que contém as informações sobre as conexões entre as cidades, tal como no exemplo anterior e, para cada cidade, imprima o número de cidades às quais ela está conectada e quais são as cidades conectadas a ela. Para isto, você deverá criar uma função chamada `ProcessaCidade`, que recebe um vetor com as informações da cidade como argumento de entrada (a linha da matriz que representa a cidade), e gera a saída para esta cidade no terminal. Por exemplo, o texto de saída da cidade 1 ("A cidade 1 está conectada a outras 2 cidades: 2, 3") será gerado pela chamada da função quando $i = 0$: `ProcessaCidade(matriz[i])`, a da cidade 2, quando $i = 1$, e assim por diante. A função imprime a mensagem no terminal e não retorna valor. Note que o número de cidades depende da dimensão da matriz.

*Exemplo 1:*

```
Matriz: 0, 1, 1, 0; 1, 0, 1, 0; 1, 1, 0, 1; 0, 0, 1, 0
A cidade 1 está conectada a outras 2 cidades: 2, 3
A cidade 2 está conectada a outras 2 cidades: 1, 3
A cidade 3 está conectada a outras 3 cidades: 1, 2, 4
A cidade 4 está conectada a outras 1 cidades: 3
```

## Questão 7.6 (2013-2)

Implemente um programa para determinar e imprimir os ganhadores de um concurso da Mega Sena. Cada aposta da Mega Sena consiste em 6 números inteiros distintos, no intervalo de 1 a 60. As apostas do concurso são definidas em matriz fornecida como entrada pelo usuários (em uma única *string*), onde cada linha representa uma aposta, e as suas 6 colunas representam os números da aposta. O resultado do sorteio é definido em um vetor fornecido como entrada pelo usuário (também em uma única *string*), onde os 6 elementos constituem os números sorteados. Considere que as entradas das apostas e dos resultados contém os números em ordem crescente.

Seja a matriz `apostas` a seguir:

$$\begin{bmatrix} 19 & 24 & 25 & 28 & 42 & 58 \\ 19 & 20 & 25 & 28 & 30 & 58 \\ 20 & 21 & 22 & 30 & 40 & 51 \\ 19 & 24 & 25 & 28 & 42 & 58 \\ 10 & 24 & 25 & 28 & 42 & 47 \end{bmatrix}$$

E considerando também o vetor `sorteados` a seguir:

$$\begin{bmatrix}19 & 24 & 25 & 28 & 42 & 58\end{bmatrix}$$

Observa-se que as apostas 1 e 4 foram as ganhadoras, pois comparando cada número sorteado com o seu correspondentes em cada aposta, apenas para estas duas apostas todos eles são iguais entre si.

Seu programa deve determinar os ganhadores de um concurso da Mega Sena. Caso não exista nenhum ganhador, o programa deve imprimir uma mensagem informando que não houve ganhador e que o prêmio será acumulado. Caso existam ganhadores, o programa deve imprimir os números das apostas desses ganhadores.

Para identificar se uma aposta ganhou ou não, implemente uma função chamada `AvaliaAposta`, que recebe como argumentos de entrada o vetor da aposta (uma linha da matriz) e o vetor de números sorteados, avalia se todos os valores de índices correspondentes são iguais e, neste caso retorna **True**, caso contrário (se pelo menos um valor for diferente), retorna **False**. Ou seja, para saber se uma aposta ganhou ou não, você deve chamar a função, ou seja, para cada aposta (um laço), controlado pela variável contadora **a**, identifica-se se ela é ganhadora chamando: `AvaliaAposta(apostas[a], sorteados)`. Note que o número de apostas depende do número de linhas da matriz `aposta`.

*Exemplo 1:*

```
Apostas: 19, 24, 25, 28, 42, 58; 19, 20, 25, 28, 30, 58; 20, 21, 22,
   30, 40, 51; 19, 24, 25, 28, 42, 58; 10, 24, 25, 28, 42, 47
Números sorteados: 19, 24, 25, 28, 42, 58
Apostas ganhadoras: 1, 4
```

*Exemplo 2:*

```
Apostas: 19, 24, 25, 28, 42, 58; 19, 20, 25, 28, 30, 58; 20, 21, 22,
   30, 40, 51; 19, 24, 25, 28, 42, 58; 10, 24, 25, 28, 42, 47
Números sorteados: 20, 24, 25, 28, 42, 60
Não houve ganhador, prêmio acumulado
```

## Questão 7.7 (2016-1)

O Sr. Apu Nahasapeemapetilon é proprietário de uma vendedora de acessórios para computador, a Kwik E' Mart. Esta empresa possui várias lojas e o Sr. Apu mantém um estoque mínimo de produtos em todas elas. Estas informações são representadas por vetores. Por exemplo:

$$\text{produtos} = \begin{bmatrix}\text{“}HD\text{”}, & \text{“}Impressora\text{”}, & \text{“}Monitor\text{”}, & \text{“}Notebook\text{”}, & \text{“}Tablet\text{”}\end{bmatrix}$$

$$\text{minimo} = \begin{bmatrix}100, & 40, & 50, & 30, & 80\end{bmatrix}$$

E os estoques em uma matriz:

$$\text{estoque} = \begin{bmatrix} 42, & 48, & 32, & 32, & 81 \\ 102, & 38, & 50, & 15, & 85 \\ 100, & 40, & 50, & 30, & 80 \\ 100, & 40, & 50, & 35, & 78 \end{bmatrix}$$

Considere um programa que leia os vetores `produto` e `minimo`, assim como a matriz `estoque`. Em seguida verifica o estoque de cada produto em cada uma das lojas, identificando a ocorrência de uma das seguintes possíveis situações em cada loja:

1. todos os produtos estão de acordo com o estoque mínimo. Neste caso, é impressa uma mensagem informando tal situação; ou

2. são impressos os produtos cujo estoque está abaixo do estoque mínimo e a quantidade que falta para atingir esse estoque.

A saída para cada loja, por exemplo, para a loja 1 corresponde às mensagens: "- Produto: HD - faltam 59 unidades" e "- Produto: Monitor - faltam 18 unidades", deve ser gerada em uma função chamada `ProcessaLoja`, que recebe como argumentos de entrada o vetor de estoque de uma loja (uma linha da matriz `estoque`), e os vetores de nomes de produtos e estoque mínimo, imprime as mensagens necessárias no terminal e não retorna valores. Por exemplo, considerando uma variável contadora para representar a loja (no programa principal), chamada `l`, a função será chamada assim: `ProcessaLoja(estoque[l], produtos, minimo)`.

Abaixo, um exemplo de execução do programa com os valores anteriores. Observe que, para este exemplo, as saídas "Loja 1:", "Loja 2:", "Loja 3:", "Loja 4:", são geradas no programa principal, e as saídas com informações de produtos de cada loja são gerados pelas respectivas chamadas à função. Note que o número de lojas depende da quantidade de linhas da matriz.

*Exemplo 1:*

```
Produtos: "HD", "Impressora", "Monitor", "Notebook", "Tablet"
Estoque mínimo: 100, 40, 50, 30, 80
Estoque: 42, 48, 32, 32, 81; 102, 38, 50, 15, 85; 100, 40, 50, 30, 80;
    100, 40, 50, 35, 78
Loja 1:
- Produto: HD - faltam 59 unidades
- Produto: Monitor - faltam 18 unidades
Loja 2:
- Produto: Impressora - faltam 2 unidades
- Produto: Notebook - faltam 15 unidades
Loja 3:
- Todos os produtos possuem o estoque mínimo
Loja 4:
- Produto: Tablet - faltam 2 unidades
```