

# Matrizes

*Algoritmos e Programação II*

*2º semestre de 2023*

Profa. Ana Grasielle

Prof. Bruno da Silva Rodrigues

Prof. Ivan Carlos Alcântara de Oliveira

Prof. Rogério de Oliveira

Prof. Tomaz Mikio Sasaki



# Matrizes

---

- Tipo de dado usado para representar uma coleção de variáveis de um mesmo tipo.
- Estrutura de dados bidimensional ou tridimensional.
- Uma matriz bidimensional contém matrizes unidimensionais.
- Sintaxe: *tipo variavel[linhas][colunas];*



# Exemplo 1

```
2  /*imprime 50 combinações de jogos para a mega sena*/
3  #include <stdio.h>
4  #include <stdlib.h>
5  #define linha 50
6  #define coluna 6
7
8  int main(){
9      int matriz[linha][coluna], i, j;
10
11     for (i=0; i<linha; i++){
12         for (j=0; j<coluna; j++){
13             matriz[i][j]=rand()%60+1; // números aleatórios de 1 a 60
14         }
15     }
16
17     for (i=0; i<linha; i++){
18         printf("Combinacao %2d: ", i+1);
19         for (j=0; j<coluna; j++){
20             printf("%2d ", matriz[i][j]);
21         }
22         printf("\n");
23     }
24     system("PAUSE");
25     return 0;
26 }
```

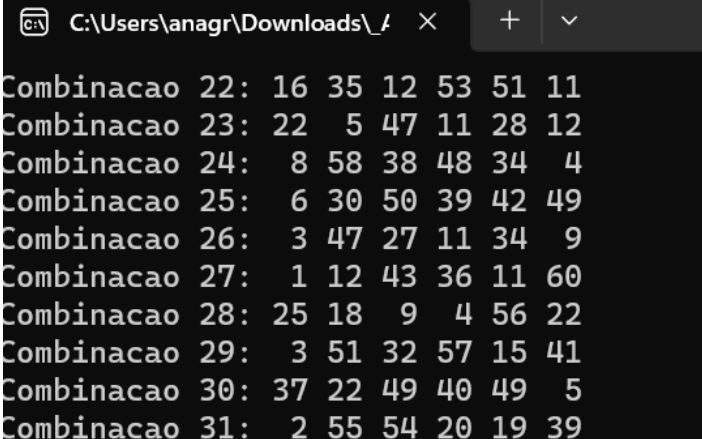
← Tamanho máximo da matriz 50x6

← Declaração da matriz

← Laço i para percorrer as linhas

← Laço j para percorrer as colunas

← Imprime elemento por elemento da matriz onde i → linha e j → coluna



```
C:\Users\anagr\Downloads\ / × + v
Combinacao 22: 16 35 12 53 51 11
Combinacao 23: 22 5 47 11 28 12
Combinacao 24: 8 58 38 48 34 4
Combinacao 25: 6 30 50 39 42 49
Combinacao 26: 3 47 27 11 34 9
Combinacao 27: 1 12 43 36 11 60
Combinacao 28: 25 18 9 4 56 22
Combinacao 29: 3 51 32 57 15 41
Combinacao 30: 37 22 49 40 49 5
Combinacao 31: 2 55 54 20 19 39
```

# Matrizes

---

Podemos fazer a atribuição da matriz manualmente:

```
#define linha 2
```

```
#define coluna 2
```

← Define a dimensão da matriz

```
int main() {
```

```
    int matrizA[linha][coluna];
```

← Cria matriz 2x2

```
    int matrizB[linha][coluna] = {{6,3}, {1,4}};
```

← Cria e inicializa matriz 2x2

```
    matrizA[0][0]=6;
```

← Atribuição de elementos

```
    matrizA[0][1]=3;
```

```
    matrizA[1][0]=1;
```

```
    matrizA[1][1]=4;
```

```
    return 0;
```

```
}
```



# Matrizes

---

Podemos fazer a atribuição da matriz usando laço for:

```
#define linha 3  
#define coluna 3
```

```
int main() {  
    int matriz[linha][coluna];  
    for (i=0; i<linha; i++){  
        for (j=0; j<coluna; j++){  
            scanf("%d", &matriz[i][j]);  
        }  
    }  
    return 0;  
}
```



Valor é atribuído por referência (&)



## Exemplo 2

```
2  #include <stdlib.h>
3  #define linha 3
4  #define coluna 3
5
6  int main(){
7      int matriz[linha][coluna], i, j;
8
9      for (i=0; i<linha; i++){
10         for (j=0; j<coluna; j++){
11             printf("Elemento: ");
12             scanf("%d", &matriz[i][j]);
13         }
14     }
15
16     for (i=0; i<linha; i++){
17         for (j=0; j<coluna; j++){
18             printf("%2d ", matriz[i][j]);
19         }
20         printf("\n");
21     }
22     system("PAUSE");
23     return 0;
24 }
```

Define a dimensão da matriz

Cria a matriz 3x3

Entrada de dados via teclado na matriz

Imprime elemento por elemento da matriz onde i → linha e j → coluna

# Matriz como argumento de função

---

Passar uma matriz como argumento de função exige atenção!

Ao declarar o parâmetro da função, ou (a) declaramos todas as dimensões ou (b) declaramos todas as dimensões menos a primeira.



## Exemplo 3

```
3  #define linha 10
4  #define coluna 10
5
6  void chamaMatriz(int matriz[][coluna]){
7      int i, j;
8
9      for (i=0; i<linha; i++){
10         for (j=0; j<coluna; j++){
11             printf("%2d ", matriz[i][j]);
12         }
13         printf("\n");
14     }
15 }
16
17 int main(){
18     int matriz[linha][coluna];
19     int i, j;
20
21     for (i=0; i<linha; i++){
22         for (j=0; j<coluna; j++){
23             matriz[i][j]=rand()%60+1;
24         }
25     }
26
27     chamaMatriz(matriz);
28
29     system("PAUSE");
30     return 0;
31 }
```

Matriz como argumento da função

Chamada da função



# Matriz como argumento de função (C99)

Se o seu compilador for compatível com C99 ou superior, você pode passar as dimensões como parâmetros, **antes** da matriz:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  - void sortear(int linhas, int colunas, int m[linhas][colunas]) {
6  -     for (int i=0; i<linhas; i++) {
7  -         for (int j=0; j<colunas; j++) {
8  -             m[i][j] = rand() % 10;
9  -         }
10     }
11 }
12
13 - void mostrar(int linhas, int colunas, int m[linhas][colunas]) {
14 -     for (int i=0; i<linhas; i++) {
15 -         for (int j=0; j<colunas; j++) {
16 -             printf("%d # ", m[i][j]);
17 -         }
18         printf("\n");
19     }
20 }
21
22 - int main() {
23     srand(time(NULL));
24     int matriz[4][3];
25     sortear(4, 3, matriz);
26     mostrar(4, 3, matriz);
27     return 0;
28 }
29
```



# Exercícios

---

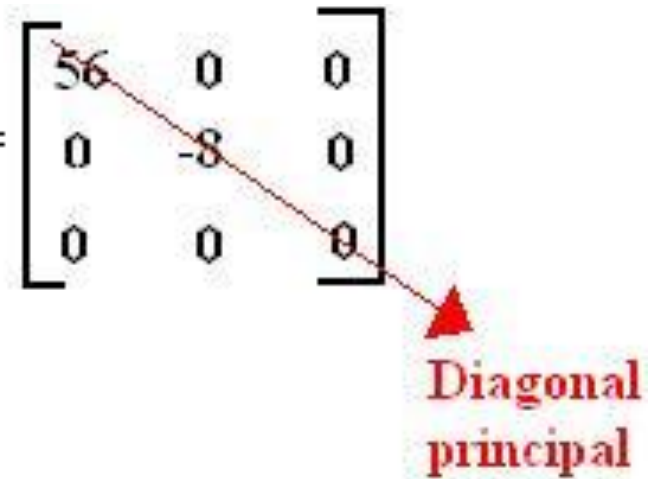
- 1) Escreva programa em C que implemente:
  - A. Uma função **verificaMaior(linhas, colunas, matriz)** que recebe como parâmetro uma matriz de números inteiros e devolve o maior elemento encontrado na matriz.
  - B. Uma função **criaMatriz (linhas, colunas, matriz)** que preenche a matriz com números inteiros aleatórios (1 a 100).
  - C. **main()**: a função pede ao usuário que digite os valores para definir as dimensões da matriz, ou seja, linhas e colunas da matriz. Na sequência, chama as funções **criaMatriz** e **verificaMaior** e, por último imprime a matriz gerada.



# Exercícios

---

- 2) O traço de uma matriz é a soma dos elementos de sua diagonal principal. Implemente uma função que recebe uma matriz quadrada, ou seja, número de linhas igual ao número de colunas, e devolva o seu traço.

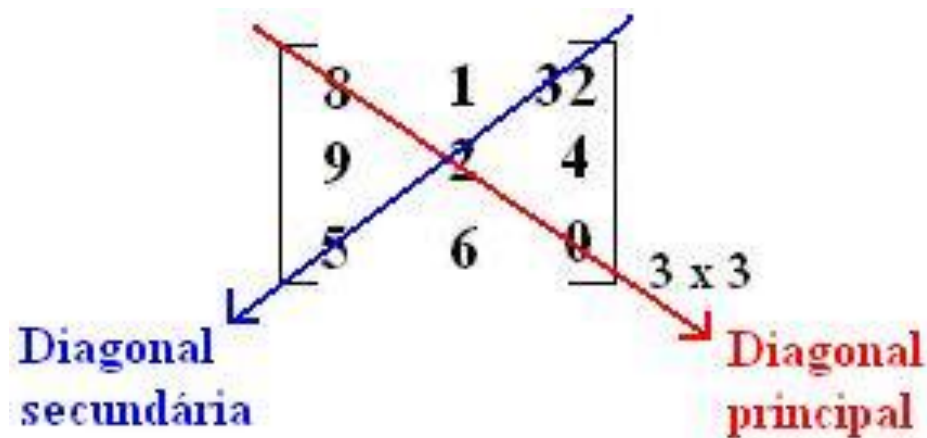
$$A = \begin{bmatrix} 56 & 0 & 0 \\ 0 & -8 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$


The diagram shows a 3x3 matrix A. The elements on the main diagonal (top-left to bottom-right) are 56, -8, and 0. A red line with an arrow points from the top-left element to the bottom-right element, and the text "Diagonal principal" is written below the arrow.

# Exercícios

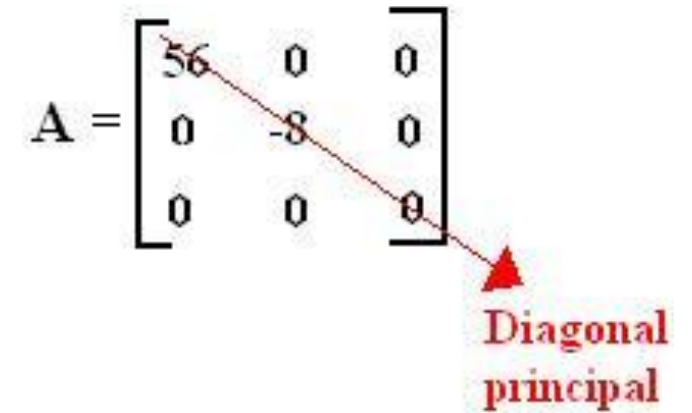
---

3) Quando a matriz é quadrada nela podemos perceber a presença de uma diagonal secundária e uma diagonal principal. Crie uma função que recebe uma matriz quadrada e devolve a soma dos valores da diagonal secundária.



# Exercícios

---

$$A = \begin{bmatrix} 56 & 0 & 0 \\ 0 & -8 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$


Diagonal principal

4) Para que uma matriz tenha diagonal ela deverá ser uma matriz quadrada, ou seja,  $N=M$ .

Uma **matriz diagonal** é uma matriz quadrada onde os elementos que não pertencem à diagonal principal são obrigatoriamente iguais a zero. Obs: *isso não impede que os elementos que pertencem à diagonal principal sejam iguais a zero, ou seja, uma matriz onde todos os seus elementos são iguais a zero é uma matriz diagonal.*

Crie uma função em C que receba uma matriz e verifica se a matriz é uma **matriz diagonal**. Obs: *sua função deve retornar True caso seja uma matriz diagonal e False caso contrário.*



# Exercícios

5) Dada uma matriz A, a transposta desta matriz é uma matriz que tem as colunas da matriz A como linhas.

Exemplo, se  $A = \begin{bmatrix} 0 & 6 \\ -1 & 2 \\ 5 & 0 \end{bmatrix}$

a matriz transposta  $A^t$  será  $\begin{bmatrix} 0 & -1 & 5 \\ 6 & 2 & 0 \end{bmatrix}$

Desenvolva uma função com a assinatura

```
void transposta(int lin, int col, int A[lin][col], int At[col][lin])
```

que devolva em **At** a transposta de **A**.



# Exercícios

---

6) Escreva uma função que recebe como parâmetros três matrizes: **A**, **B** e **C**, com **n** linhas e **m** colunas.

Sua função deve calcular a soma de **A + B** e armazenar o resultado na matriz **C**.

7) Escreva uma função que recebe como parâmetros:

- a matriz **A**, com dimensões **nxm**;
- a matriz **B**, com dimensões **mxp**;
- e a matriz **C**, com dimensões **nxp**.

Sua função deve calcular a multiplicação de **A** por **B** e armazenar o resultado na matriz **C**.



# Referências

MENOTTI, D.; OLIVEIRA, L. **CI-1002: Programação 2**. Disponível em: <<https://wiki.inf.ufpr.br/maziero/doku.php?id=prog2:start>>. Acesso em: 03 de janeiro de 2023.

DEITEL, P.; DEITEL, H. **C: Como programar**. 6ª edição. Editora Pearson, 2011. (*disponível na Biblioteca Virtual Pearson*)







Universidade Presbiteriana  
**Mackenzie**



1952 – 2022



Faculdade de  
**Computação e Informática**