BCC701 – Programação de Computadores I
Universidade Federal de Ouro Preto
Departamento de Ciência da Computação

www.decom.ufop.br/gustavo

Aula Teórica 14
Matrizes

Material Didático Proposto

Agenda

- Introdução;
- Declaração de Matrizes;
- Algumas operações com matrizes;
- Algumas funções aplicadas a matrizes;
- Exercícios.

Introdução;
Declaração de matrizes;
Algumas operações com matrizes;
Algumas funções aplicadas a matrizes;
Exercícios.

INTRODUÇÃO

Conjunto de variáveis

- Ao estudar **vetores** observamos que, em determinadas situações, é necessário utilizar muitas variáveis com um propósito comum. Relembrando exemplos:

  - Para armazenar três notas de um mesmo aluno:
    - Nota(1) = input('Digite a nota 1: ');
    - Nota(2) = input('Digite a nota 2: ');
    - Nota(3) = input('Digite a nota 3: ');

  - Ler um vetor com cinco números:
    - for i = 1 : 5

      Numero(i) = input('Digite um número: ');

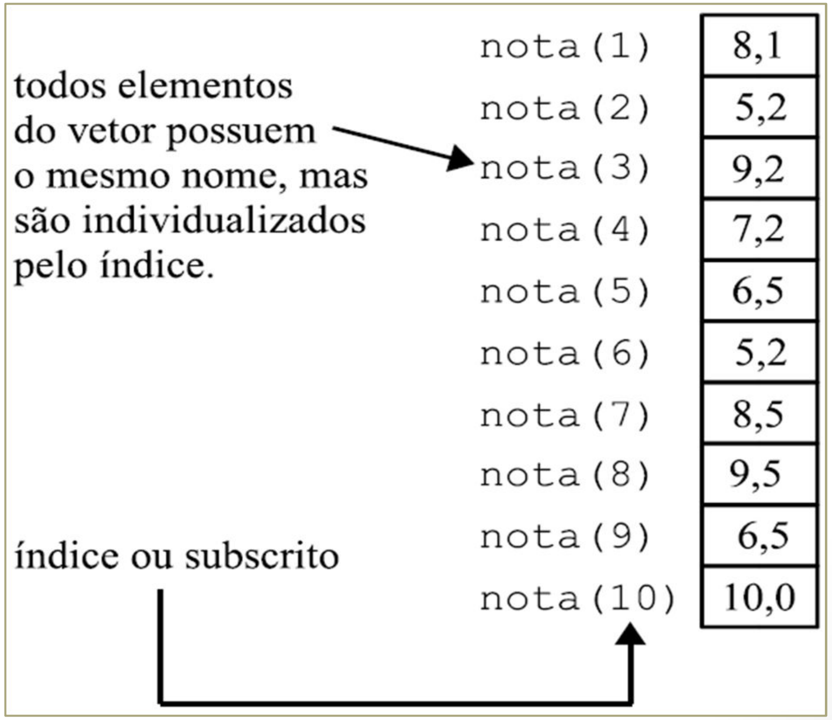
      end

Relembrando **Vetor**

- Nestes casos, todas as variáveis representam um conjunto de valores, possuem um objetivo em comum e são do mesmo tipo de dados;

- Um vetor representa conjuntos ordenados de valores homogêneos (do mesmo tipo), que podem ser números, *strings* e booleanos;
  - A palavra ordenado é empregada no sentido dos valores estarem localizados em posições ordenadas de memória, e não no sentido de estarem respeitando uma relação (<, <=, >, ou >=).

- Os valores contidos em um vetor são chamados de **elementos**;
- A posição do elemento no vetor é chamado de **índice** (subscrito), e é usado para individualizar um elemento do vetor;
- O vetor nota = [8.1 5.2 9.2 7.2 6.5 5.2 8.5 9.5 6.5 10.0], pode ser representado na memória como uma sequência de variáveis distintas, com o mesmo nome, mas diferenciadas pelo índice:



O tipo de dados **Matriz**

- Agora imagine a seguinte situação:
  - Desejo armazenar **3 notas** para **5 alunos**;
  - Para isto eu preciso de **3 vetores** ou de **5 vetores**?

O tipo de dados **Matriz**

- Agora imagine a seguinte situação:
  - Desejo armazenar **3 notas** para **5 alunos**;
  - Para isto eu preciso de **3 vetores** ou de **5 vetores**?
  - **Nenhum dos dois: podemos utilizar uma matriz em que cada linha representa um aluno e cada coluna representa uma nota:**

	Nota 1	Nota 2	Nota 3
Aluno 1	8.1	9.2	6.0
	5.2	6.8	9.5
⋮	6.0	6.1	6.2
	3.5	5.2	8.3
Aluno 5	2.4	1.5	5.3

O tipo de dados **Matriz**

- Matrizes são variáveis que contêm uma quantidade potencialmente grande de valores do mesmo tipo;
- Assim como nos vetores, elementos da matriz são acessados através de índices;
- Uma matriz bidimensional ***A***, tem dimensão ***m*** *x* ***n*** , ou seja, de ***m*** linhas e ***n*** colunas:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

- **OBS:** Um vetor corresponde a uma matriz ***m x 1*** (no caso de um **vetor coluna**), ou uma matriz ***1 x n*** (no caso de um **vetor linha**).

O tipo de dados **Matriz**

- Além das matrizes serem muito úteis para o armazenamento e manipulação de um grande volume de dados, elas também são muito utilizadas em diversas áreas:
  - Para se resolver sistemas de equações lineares;
  - Translação, rotação, escala de objetos em computação gráfica;
  - Para resolver problemas de circuitos elétricos e linhas de transmissão de energia elétrica;
  - Algoritmos para determinar rotas entre dois pontos;
  - E muito mais;

- É no tratamento de matrizes que o Scilab mostra grande superioridade sobre linguagens como C, Fortran ou Java;

Exemplos de uso de Matriz

- Na resolução de sistemas de equações lineares:
  - Dado um sistema linear do tipo: **A * X = B**;
  - A solução é obtida resolvendo: **X = A^{-1} * B**;
  - Exemplo:

$$\begin{aligned} 3x + y + 2z &= 13 \\ x + y - 8z &= -1 \\ -x + 2y + 5z &= 13 \end{aligned} \Longrightarrow \underbrace{\begin{bmatrix} 3 & 1 & 2 \\ 1 & 1 & -8 \\ -1 & 2 & 5 \end{bmatrix}}_{A_{33}} \underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{X_{31}} = \underbrace{\begin{bmatrix} 13 \\ -1 \\ 13 \end{bmatrix}}_{B_{31}}$$

Exemplos de uso de Matriz

- Na resolução de sistemas de equações lineares:
  - Exemplo:

$$\begin{aligned} 3x + y + 2z &= 13 \\ x + y - 8z &= -1 \\ -x + 2y + 5z &= 13 \end{aligned} \Longrightarrow \begin{bmatrix} 3 & 1 & 2 \\ 1 & 1 & -8 \\ -1 & 2 & 5 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 13 \\ -1 \\ 13 \end{bmatrix}$$

A_{33} X_{31} B_{31}

```
--> A = [3, 1, 2; 1, 1, -8; -1, 2, 5];
--> B = [13; -1; 13];
--> X = inv(A) * B
X =   2.
      5.
      1.
```

Assim, chega-se à solução:

x = 2, y = 5, z = 1.

Introdução;

Declaração de matrizes;

Algumas operações com matrizes;

Algumas funções aplicadas a matrizes;

Exercícios.

DECLARAÇÃO DE MATRIZES

- Em muitos casos temos a necessidade de usar um laço dentro do corpo de um outro laço.
- Situações como esta aparecem sempre que estamos trabalhando com uma tabela, ou situações parecidas.
- Para percorrer uma tabela temos que percorrer as linhas e para cada linha, percorrer as suas colunas.

	j = 1	j = 2	j = 3	j = 4
i = 1	**a11**	**a12**	**a13**	**a14**
	a21	**a22**	**a23**	**a24**
	a31	**a32**	**a33**	**a34**

- Em muitos casos temos a necessidade de usar um laço dentro do corpo de um outro laço.
- Situações como esta aparecem sempre que estamos trabalhando com uma tabela, entre outros.
- Para percorrer uma tabela temos que percorrer as linhas e para cada linha, percorrer as suas colunas.

	j = 1	j = 2	j = 3	j = 4
	a11	a12	a13	a14
i = 2	a21	a22	a23	a24
	a31	a32	a33	a34

- Em muitos casos temos a necessidade de usar um laço dentro do corpo de um outro laço.
- Situações como esta aparecem sempre que estamos trabalhando com uma tabela, entre outros.
- Para percorrer uma tabela temos que percorrer as linhas e para cada linha, percorrer as suas colunas.

	j = 1	j = 2	j = 3	j = 4
	a11	a12	a13	a14
	a21	a22	a23	a24
i = 3	a31	a32	a33	a34

Percorrendo a tabela por linha

a11	a12	a13	a14
a21	a22	a23	a24
a31	a32	a33	a34

```
for linha = 1 : 3
      printf("\nlinha %g", linha)
end
```

Saída:
linha 1
linha 2
linha 3

Percorrendo a tabela por linha

a11	a12	a13	a14
a21	a22	a23	a24
a31	a32	a33	a34

```
for linha = 1 : 3
      printf("\nlinha %g: ", linha)
      for coluna = 1 : 4
            printf("col. %g", coluna)
      end
      printf(" ::  fim da linha %g\n", linha)
end
```

Saída:
linha 1 : col. 1 col. 2 col. 3 col. 4 :: fim da linha 1
linha 2 : col. 1 col. 2 col. 3 col. 4 :: fim da linha 2
linha 3 : col. 1 col. 2 col. 3 col. 4 :: fim da linha 3

Percorrendo a tabela por coluna

a11	a12	a13	a14
a21	a22	a23	a24
a31	a32	a33	a34

```
for coluna = 1 : 4
      printf("\ncoluna %g", coluna)
end
```

Saída:
coluna 1
coluna 2
coluna 3
coluna 4

Percorrendo a tabela por coluna

a11	a12	a13	a14
a21	a22	a23	a24
a31	a32	a33	a34

```
for coluna = 1 : 4
      printf("\ncoluna %g : ", coluna)
      for linha = 1 : 3
            printf("linha %g", linha)
      end
      printf(" ::  fim da coluna %g\n", coluna)
end
```

Saída:

coluna 1 : linha 1 linha 2 linha 3 :: fim da coluna 1
coluna 2 : linha 1 linha 2 linha 3 :: fim da coluna 2
coluna 3 : linha 1 linha 2 linha 3 :: fim da coluna 3
coluna 4 : linha 1 linha 2 linha 3 :: fim da coluna 4

Matrizes × Vetores

$$V = (v_1, v_2, v_3, \ldots, v_n)$$

Os vetores têm um único índice, portanto basta um for para percorrê-lo.

```
for i = 1: n
     v(i) é cada um dos n elementos do vetor v
end
```

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

As matrizes têm dois índices, portanto são necessários dois fors para percorrê-las

```
for i = 1: tot_linhas
     for j = 1: tot_colunas
          A(i, j) é cada um dos elementos da matriz A
     end
end
```

<tópicos>

Leitura de Matriz elemento a elemento

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

Percorrendo a matriz por linha

```
for i = 1: tot_linhas       // para cada linha
     for j = 1: tot_colunas // percorre as colunas
          A(i, j) = input("matriz linha i, coluna j")
     end
end
"A(i, j) contém cada elemento da matriz".
```

Percorrendo a matriz por coluna

```
for j = 1: tot_colunas     // para cada coluna
     for i = 1: tot_linhas // percorre as linhas
          A(i, j) = input("matriz linha i, coluna j")
     end
end
```

Definindo todos os elementos

- Utiliza-se colchetes para delimitar todos os elementos;
- Cada elemento de uma linha é separado por espaço ou vírgula;
- Cada linha é separada por um ponto-e-vírgula;
- Exemplo:

```
--> M = [1, 2, 3; 4, 5, 6; 7, 8, 9]
M  =
   1.   2.   3.
   4.   5.   6.
   7.   8.   9.
-->
```

A partir de matrizes

- A definição pode ser feita a partir de matrizes já existentes;
- Exemplos:

```
--> A = [1 2; 3 4]
 A  =          1.   2.
               3.   4.
 --> B = [5 6; 7 8]
 B  =          5.   6.
               7.   8.
 --> C = [A B]
 C  =          1.   2.   5.   6.
               3.   4.   7.   8.
 -->D = [A; B]
 D  =          1.   2.
               3.   4.
               5.   6.
               7.   8.
-->
```

Matriz de 1's

- Todos os elementos assumirão valor inicial **1**:

Matriz = ones(<linhas>, <colunas>)

  - **Matriz**: nome da variável do tipo matriz;
  - **ones**: função que retorna uma **matriz** com valores 1;
  - **<linhas>**: número de linhas;
  - **<colunas>**: número de colunas.

Matriz de 1's

- Exemplos:
  - --> M1 = ones(2, 5)

    M1 = 1. 1. 1. 1. 1.

    1. 1. 1. 1. 1.

    -->
  - --> M2 = ones(5, 2)

    M1 = 1. 1.

    1. 1.

    1. 1.

    1. 1.

    1. 1.

    -->

Matriz de 0’s

- Todos os elementos assumirão valor inicial **0**:

Matriz = zeros(<linhas>, <colunas>)

  - **Matriz**: nome da variável do tipo matriz;
  - **zeros**: função que retorna uma **matriz** com valores 0;
  - **<linhas>**: número de linhas;
  - **<colunas>**: número de colunas.

Matriz de 0's

- Exemplos:
  - --> M1 = zeros(2, 5)

    M1 = 0. 0. 0. 0. 0.
    0. 0. 0. 0. 0.

    -->
  - --> M2 = zeros (5, 2)

    M1 = 0. 0.
    0. 0.
    0. 0.
    0. 0.
    0. 0.

    -->

Acesso aos elementos

- **Para acessar um elemento específico**:

Matriz(<índice de linha>, <índice de coluna>)

- Exemplo:
  --> M = [1, 2, 3; 4, 5, 6];

Matematicamente temos: M = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$

```
--> E1 = M(2, 3)
 E1 = 6.
--> E2 = M(1, 2)
 E2 = 2.
-->
```

- Pode ser usado para modificar o valor: **M(1, 3) = 300**, modifica o valor da **linha 1 e coluna 3** de **3** para **300**.
  - OBS.: Utilizando este recurso é possível definir uma matriz definindo o valor de cada um dos seus elementos individualmente.

Transposição de matrizes

- **Operador apóstrofo (’)**: Matriz’
  - Transforma linhas em colunas e colunas em linhas;
  - Exemplo:

```
x  =    23.   30.   29.   50.   91.   28.   68.
        23.   93.   56.   43.   4.    12.   15.
        21.   21.   48.   26.   48.   77.   69.
        88.   31.   33.   63.   26.   21.   84.
        65.   36.   59.   40.   41.   11.   40.
--> y = x’
y =     23.   23.   21.   88.   65.
        30.   93.   21.   31.   36.
        29.   56.   48.   33.   59.
        50.   43.   26.   63.   40.
        91.   4.    48.   26.   41.
        28.   12.   77.   21.   11.
        68.   15.   69.   84.   40.
```

<tópicos>

Dimensões

```
-->M
 M  =

    1.   1.   1.   1.
    2.   2.   2.   2.

[a , b] = size(M)     // IMPORTANTÍSSIVO, USAREMOS MUUUIIITO!
a  =
    2.  // total de linhas da matriz

 b  =
    4. // total de colunas da matriz
```

Adição e subtração de matrizes

```
--> A = [1 2 3; 4 5 6]
 A  =
    1.   2.   3.
    4.   5.   6.

-->B = [1 2 3; 4 5 6]
 B  =

    1.   2.   3.
    4.   5.   6.
```

```
-->C = A + B
 C  =

    2.   4.   6.
    8.   10.   12.

-->C = A - B
 C  =

    0.   0.   0.
    0.   0.   0.
```

Multiplicação por um escalar

- Uma matriz pode ser multiplicada por um valor escalar;

```
-->A
 A  =

    1.    2.    3.
    4.    5.    6.

-->C = 2*A
 C  =

    2.    4.     6.
    8.    10.    12.
```

Multiplicação entre matrizes

```
--> A = [1 2 3; 4 5 6]
 A  =
   1.    2.    3.
   4.    5.    6.

-->B = [1 2 3; 4 5 6]
 B  =

   1.    2.    3.
   4.    5.    6.
```

```
-->C = A*B
        !--error 10
Multiplicação incoerente.
 -->C = A'*B
 C  =

   17.    22.    27.
   22.    29.    36.
   27.    36.    45.

-->C = A*B'
 C  =

   14.    32.
   32.    77.
```

Exercícios

1. Escreva um programa que leia as matrizes A e B via teclado **<u>em lote</u>** e posteriormente some seus elementos gerando C = A + B, somando elemento a elemento. Mostrar o resultado.

A soma só pode ser realizada se a dimensão das duas matrizes for igual. Caso contrário emitir uma mensagem de erro

2. Leia uma matriz A em lote e posteriormente somar os elementos de cada linha da matriz. Informar esta soma

Exemplo : A = $\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 6 & -2 \end{bmatrix}$, então V = $\begin{bmatrix} 3 \\ 7 \\ 4 \end{bmatrix}$