



MATRIZES

Introdução à Programação – 2022.1

Prof^a. Giorgia Mattos – giorgiamattos@gmail.com

Linguagem C - Matrizes

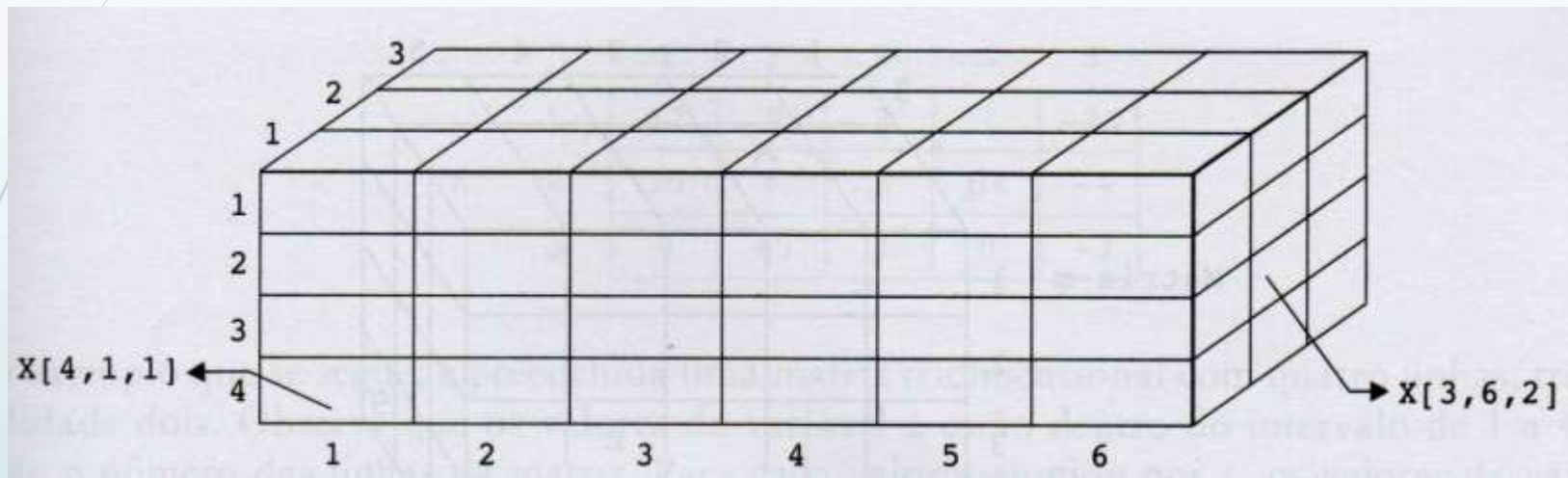
- **Definição** – Uma matriz é uma variável composta homogênea multidimensional. Ela é formada por uma sequência de variáveis, do mesmo tipo, com o mesmo identificador (nome), e alocadas sequencialmente na memória
 - Uma vez que as variáveis tem o mesmo nome, o que as distingue são os índices que referenciam sua localização dentro da estrutura
 - Uma variável do tipo matriz precisa de um índice para cada uma de suas dimensões.

M	1	2	3	4	5
1					
2					
3					

Diagram illustrating a 3x5 matrix **M**. The rows are indexed 1 to 3, and the columns are indexed 1 to 5. Arrows point to specific elements: $M[1,1]$ (row 1, column 1) and $M[3,4]$ (row 3, column 4).

Matriz **M** bidimensional, onde o tamanho da 1ª dimensão (linha) é 3 e o tamanho da 2ª dimensão (coluna) é 5.

Linguagem C - Matrizes



Matriz **X** tridimensional, onde o tamanho da 1ª dimensão (linha) é 4; o da 2ª dimensão (coluna) é 6; e o tamanho da 3ª dimensão (profundidade) é 3.

Linguagem C - Matrizes

- A linguagem C permite a declaração de matrizes unidimensionais (conhecidas como vetores), bidimensionais e multidimensionais
- O padrão ANSI prevê 12 dimensões, entretanto o limite de dimensões fica por conta da quantidade de recursos computacionais disponíveis
- As matrizes mais utilizadas possuem duas dimensões. Para cada dimensão deve ser utilizado um índice (linha e coluna)
- Os índices utilizados em C, para identificar as posições de uma matriz, começam sempre em zero e vão até o tamanho da dimensão menos uma unidade
- Os índices devem sempre ser representados pelo tipo inteiro

Linguagem C - Matrizes

► Declaração

tipo nome_matriz [dimensão1] [dimensão2]...[dimensãoN];

Onde:

- tipo: é o tipo dos dados que serão armazenados na matriz;
- nome_matriz: é o nome dado à variável;
- [dimensão1]: representa o tamanho da 1ª dimensão;
- [dimensão2]: representa o tamanho da 2ª dimensão;
- [dimensãoN]: representa o tamanho da n-ésima dimensão da matriz;

** Em C, o tamanho das dimensões de uma matriz deve ser feito por um valor inteiro fixo

Linguagem C - Matrizes

➡ Exemplos

Exemplo 1: `float X[2][6];`

Variável matriz **X** contendo 2 linhas (de 0 a 1) com 6 colunas cada (de 0 a 5), capaz de armazenar números reais

X	0	1	2	3	4	5
0						
1						

Exemplo 2: `int M[4][3];`

Variável matriz **M** contendo 4 linhas (de 0 a 3) com 3 colunas cada (de 0 a 2), capaz de armazenar números inteiros

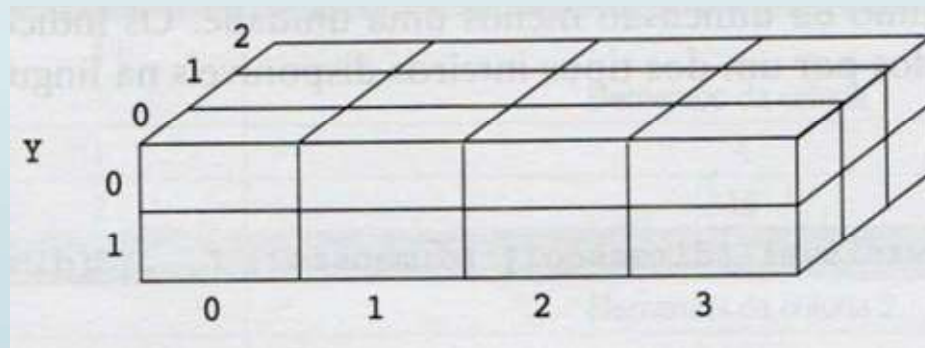
M	0	1	2
0			
1			
2			
3			

Linguagem C - Matrizes

➤ Exemplos

Exemplo 3: `float Y[2][4][3];`

Variável matriz **Y** contendo 2 linhas (de 0 a 1) com 4 colunas cada (de 0 a 3) e profundidade três (de 0 a 2), capaz de armazenar números reais



Linguagem C - Matrizes

► Atribuindo valores a uma matriz

- Significa armazenar informação em seus elementos, identificados de forma única por meio dos seus índices

X[1][4] = 5.3; → Atribui o valor 5.3 à posição identificada pelos índices 1 (2ª linha) e 4 (5ª coluna)

X	0	1	2	3	4	5
0						
1					5.3	

Linguagem C - Matrizes

► Atribuindo valores a uma matriz

- Significa armazenar informação em seus elementos, identificados de forma única por meio dos seus índices

$M[3][2] = 4;$ → Atribui o valor 4 à posição identificada pelos índices 3 (4ª linha) e 2 (3ª coluna)

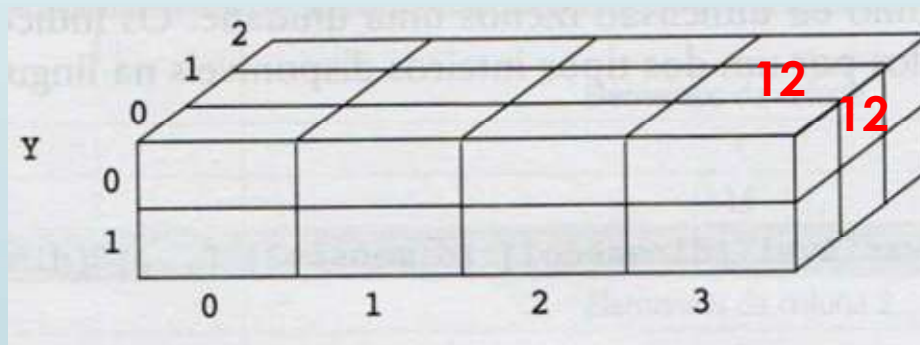
M	0	1	2
0			
1			
2			
3			4

Linguagem C - Matrizes

► Atribuindo valores a uma matriz

- Significa armazenar informação em seus elementos, identificados de forma única por meio dos seus índices

Y[0][3][1] = 12; → Atribui o valor 12 à posição identificada pelos índices 0 (1ª linha), 3 (4ª coluna) e 1 (2ª profundidade)



Linguagem C - Matrizes

► Preenchendo uma matriz

- Significa percorrer todos os seus elementos, atribuindo-lhes um valor. Esse valor pode ser recebido do usuário, por meio do teclado, ou pode ser gerado pelo programa. Considerar `int M[7][3]`;

```
for(i=0; i<7; i++)  
    for(j=0; j<3; j++)  
        scanf("%d", &M[i][j]);
```

Todos os elementos da matriz M são percorridos, atribuindo-lhes valores digitados pelo usuário.

Como M possui 7 linhas e 3 colunas, o exemplo apresenta duas estruturas for para garantir que a variável i assuma todos os valores para linha (0 a 6) e a variável j assuma todos os valores possíveis para coluna (0 a 2) da matriz M. Assim, para cada execução das estruturas de repetição, uma posição diferente da matriz é preenchida por um valor digitado pelo usuário.

Linguagem C - Matrizes

► Mostrando os elementos de uma matriz

- Significa percorrer todos os elementos de uma matriz acessando seu conteúdo. Considerar float X[10][6];

```
for (i=0; i<10; i++)  
    for (j=0; j<6; j++)  
        printf("%f", X[i][j]);
```

Todos os elementos da matriz X são percorridos, apresentando os seus valores através do *printf*.

Como X possui 10 linhas e 6 colunas, o exemplo apresenta duas estruturas for para garantir que a variável *i* assuma todos os valores para linha (0 a 9) e a variável *j* assuma todos os valores possíveis para coluna (0 a 5) da matriz X. Assim, para cada execução das estruturas de repetição, uma posição diferente da matriz é acessada e seu conteúdo mostrado por meio do comando *printf*.

Linguagem C - Matrizes

► Percorrendo uma matriz

- Para preencher uma matriz e mostrá-la foi necessário passar por todas as suas posições, ou seja, foi preciso percorrer a matriz
- Uma das formas mais simples de percorrer uma matriz pode ser por meio do uso de uma estrutura de repetição para cada dimensão
- A disposição de tais estruturas de repetição define a forma como a matriz será percorrida

X	0	1	2	3
0	4	5	1	10
1	16	11	76	8
2	9	54	32	89

```
for (i=0; i<3; i++) {  
    printf ("Elementos da linha %d: ", i);  
    for (j=0; j<4; j++)  
        printf ("%d\n", X[i][j]);  
}
```

Linguagem C - Matrizes

Memória		Tela
i	j	
0		Elementos da linha 0
0	0	4
0	1	5
0	2	1
0	3	10
1		Elementos da linha 1
1	0	16
1	1	11
1	2	76
1	3	8
2		Elementos da linha 2
2	0	9
2	1	54
2	2	32
2	3	89

Linguagem C - Matrizes

- Outra visão da forma utilizada para percorrer a matriz
 - A direção das setas indica a mudança no valor das variáveis i e j e o caminho utilizado para percorrer a matriz

X	0	1	2	3
0	4	5	1	10
1	16	11	76	8
2	9	54	32	89

Linguagem C - Matrizes

- Percorrer a matriz de tal forma a mostrar todos os elementos gravados em cada coluna
 - Seguir o mesmo procedimento modificando a ordem dos índices
 - A direção das setas indica a mudança no valor das variáveis i e j e o caminho utilizado para percorrer a matriz

X	0	1	2	3
0	4	5	1	10
1	16	11	76	8
2	9	54	32	89

Linguagem C - Matrizes

- As matrizes podem ser inicializadas no momento da sua declaração

```
int M [3][3] = {4, -1, 5, 10, 3, 6, 21, -5, 1};
```

```
int M[][3] = {4, -1, 5, 10, 3, 6, 21, -5, 1};
```

```
int M[][] = {4, -1, 5, 10, 3, 6, 21, -5, 1}; ERRADO!!
```

Linguagem C - Matrizes

- Lendo uma matriz do teclado

```
/* Leitura */
```

```
for (i=0; i<4; i++)  
    for (j=0; j<4; j++) {  
        printf ("Matriz [%d][%d]: ", i, j);  
        scanf ("%d", &matriz[i][j]);  
    }
```

Linguagem C - Matrizes

- Escrevendo uma matriz na tela

```
/* Escrita*/
```

```
for (i=0; i<4; i++) {  
    for (j=0; j<4; j++)  
        printf ("%3d", matriz[i] [j]);  
  
    printf ("\n");  
}
```

Linguagem C - Matrizes

- **Exemplo 1:** Mostra na tela a matriz 2x5 inicializada na sua declaração.

```
int main() {  
    int i, j;  
    float m[2][5] = {{1, 2, 3, 4, 5}, {6, 7, 8, 9, 10}};  
  
    for (i=0; i<2; i++) {  
        for (j=0; j<5; j++)  
            printf("%7.2f", m[i][j]);  
  
        printf ("\n");  
    }  
    return 0;  
}
```

Linguagem C - Matrizes

- **Exemplo 2:** Calcular a soma dos elementos de uma matriz 3x4.

```
int main () {  
    int m[3][4], i, j, s=0;  
  
    for (i=0; i<3; i++)  
        for (j=0; j<4; j++) {  
            scanf ("%d", &m[i][j]);  
            s = s + m[i][j];  
        }  
    printf ("Soma dos elementos: %d",s);  
    return 0;  
}
```

Linguagem C - Matrizes

- **Exemplo 3:** Dada uma matriz 5x5 contar quantos elementos são negativos.

```
int main () {  
    int m[5][5], i, j, n=0;  
    ... //supor que a matriz já foi preenchida  
    for (i=0; i<=4; i++)  
        for (j=0; j<=4; j++)  
            if (m[i][j] < 0)  
                n++;  
    printf ("Quantidade de elementos negativos: %d",n);  
    return 0;  
}
```

Linguagem C - Matrizes

- **Exemplo 4:** Fazer um programa que leia uma matriz 4x4, multiplique os elementos da sua diagonal principal por uma constante k e mostre a matriz após a multiplicação.

```
#include <stdio.h>
```

```
#define k 5 //definição da constante
```

```
int main () {  
    int m[4][4], i, j;
```

Linguagem C - Matrizes

```
for (i=0; i<4; i++)           // Lê a matriz
    for (j=0; j<4; j++) {
        printf ("M[%d][%d] = ",i,j);
        scanf ("%d",&m[i][j]);
    }
```

```
for (i=0; i<4; i++)           //Multiplica a diagonal por k=5
    m[i][i] = m[i][i] * k;
```

```
for (i=0; i<4; i++) {         //Mostra a matriz multiplicada
    for (j=0; j<4; j++)
        printf ("%5d",m[i][j]);
    printf ("\n");
}
```

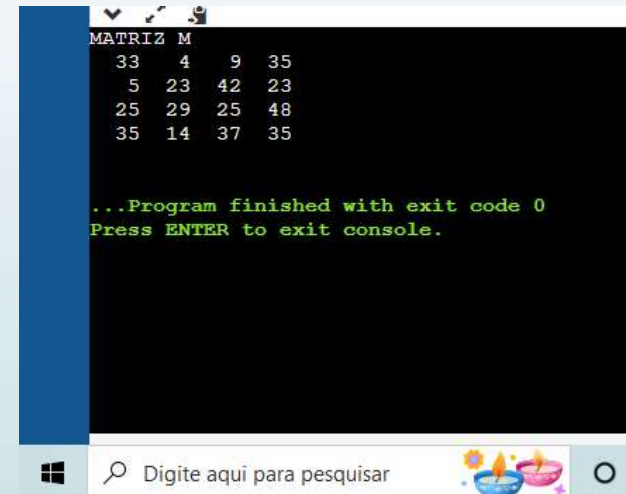
```
return 0;
```

```
}
```


Linguagem C - Matrizes

- Matrizes e funções → idem aos vetores

```
main.c
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  void gera_matriz (int m[][4]){
6      int L, C;
7      for (L=0; L<4; L++){
8          for (C=0; C<4; C++){
9              m[L][C] = rand()%50;
10         }
11     }
12
13     void mostra_matriz (int m[][4]){
14         int L, C;
15         for (L=0; L<4; L++){
16             for (C=0; C<4; C++){
17                 printf ("%4d",m[L][C]);
18                 printf ("\n");
19             }
20         }
21     }
22
23     int main()
24     {
25         int M[4][4], lin, col;
26         srand (time(NULL));
27         gera_matriz (M);
28         printf ("MATRIZ M\n");
29         mostra_matriz (M);
30         return 0;
31     }
```



```
MATRIZ M
33  4  9 35
 5 23 42 23
25 29 25 48
35 14 37 35

...Program finished with exit code 0
Press ENTER to exit console.
```

Linguagem C – Matrizes

Exercícios práticos

1. Seja R uma matriz 4x5. Determine o maior elemento de R e a sua posição.
2. Seja A uma matriz 3x3. Fazer um programa que:
 - a) Determine a soma dos elementos da diagonal principal.
 - b) Armazene os elementos da diagonal principal de A em um vetor D.
3. Idem ao exercício anterior para diagonal secundária.
4. Seja A uma matriz 3x3. Determine a matriz T transposta de A. (obs.: $T[l][c] = A[c][l]$). Mostre a matriz T.
5. Dada uma matriz B, determine a linha de B que possui a maior soma de seus elementos.

Linguagem C – Matrizes

Exercícios práticos

6. Faça um programa que preencha uma matriz $M(3 \times 3)$, calcule e mostre a matriz R , resultante da multiplicação dos elementos de M pelo seu menor elemento.
7. Escreva um programa que preencha uma matriz $M(6 \times 4)$ com números inteiros, calcule e mostre quantos elementos dessa matriz são maiores que 30 e, em seguida, monte uma segunda matriz com os elementos diferentes de 30. No lugar do número 30, da segunda matriz, coloque o número zero.
8. Faça um programa que preencha uma matriz $M(8 \times 8)$ com números inteiros e some cada uma das linhas, armazenando o resultado das somas em um vetor. A seguir, o programa deverá multiplicar cada elemento da matriz pela soma da linha correspondente e mostrar a matriz resultante.
9. Escreva um programa que preencha uma matriz $M(10 \times 10)$ com números inteiros, execute as trocas especificadas a seguir e mostre a matriz resultante: a linha 2 com a linha 8; a coluna 4 com a coluna 10.

Linguagem C – Matrizes

Exercícios práticos

10. Seja a declaração: `int N[LINHA][COLUNA];` Escrever um programa capaz de:
- a) ler os elementos da matriz.
 - b) identificar o número de elementos iguais a zero em cada uma das linhas.
 - c) identificar o número de elementos iguais a zero em cada uma das colunas.
 - d) identificar o número de elementos pares em determinada linha (a linha será fornecida pelo usuário).
 - e) identificar o número de elementos pares em determinada coluna (a coluna será fornecida pelo usuário).
 - f) calcular a média aritmética dos elementos de cada uma das linhas, armazenando esses valores em um vetor.
 - g) identificar a linha que tem a maior média de seus elementos.