

Programação Estruturada

- Encontro 03 -

Sistemas de Informação
Prof.º Philippe Leal
philippeleal@yahoo.com.br



Definição de Matrizes

- A estrutura utilizada até o momento (vetor) possui apenas uma dimensão e, portanto, é tratada como uma lista de variáveis.
- Porém, há casos em que uma estrutura com mais de uma dimensão é mais útil, por exemplo, quando os dados são organizados em uma estrutura de linhas e colunas, como uma tabela. Para isso, utilizamos uma estrutura com duas dimensões, ou seja, uma matriz.
- Matriz é um conjunto bi-dimensional (linhas e colunas) de valores.
- Conjunto com dimensões maiores (3, 4, ...) também podem ser definidos em C.



Declaração de Matrizes

- A forma geral para declarar uma matriz é:

tipo nome_da_matriz [nº_de_linhas][nº_de_colunas];

Exemplos:

```
int mat[3][10]; //Matriz de inteiros mat com 3 linhas e 10 colunas
```

```
float valores[15][20]; //Matriz de reais valores com 15 linhas e 20 colunas
```

```
double resultado[100][100]; //Matriz de reais resultado com 100 linhas e 100 colunas
```

Obs.: O tamanho da matriz (número de linhas e colunas) deve ser uma constante inteira, isto é, não pode ser o valor de uma variável.



Utilizando Matrizes

- Após a declaração é reservado espaço na memória para todos os elementos da matriz.
`int mat[3][4]; //Espaço para 12 inteiros`
- Em C, assim com vetor, somente podemos acessar e processar elemento a elemento da matriz. Não tem como processar todos os elementos de uma só vez.
- Cada elemento é acessado usando:
nome_da_matriz[i][j], onde i é a linha e j a coluna, sendo que:
 - $0 \leq i \leq m-1$, onde m é o número de linhas;
 - $0 \leq j \leq n-1$, onde n é o número de colunas.

Ex.: `mat[0][1] = 2; //Armazena o valor 2 na primeira linha e segunda coluna de mat.`

`mat[2][3] = 6; //Armazena o valor 6 na terceira linha e quarta coluna de`



Utilizando Matrizes

- Podemos inicializar uma matriz utilizando um procedimento/função ou no momento da declaração, como nos exemplos a seguir:

```
int matriz[4][2] = {1, 2, 3, 4, 5, 6, 7, 8}; //Declara e inicializa uma matriz 4x2
```

```
int matriz[3][3] = {{1,2,3}, {4,5,6}, {7,8,9}}; //Declara e inicializa uma matriz 3x3
```

```
float matriz[][2] = {1.0, 2.0, 3.0, 4.0, 5.0, 6.0}; //Declara e inicializa uma matriz 3x2
```



Passando Matrizes por Parâmetro

- É análogo a passar um vetor por parâmetro:
 - Passa-se na verdade uma referência para a matriz;
 - Ao alterar um elemento da matriz, altera a matriz original.
- O parâmetro não precisa especificar o número de linhas, mas o número de colunas é obrigatório, devido ao endereçamento linear da memória, onde cada linha é armazenada uma após a outra.

Passando Matrizes por Parâmetro

```
1 #include <stdio.h>
2
3 void preencheMatriz(int mat[4][2], int linha, int coluna);
4 void imprimeMatriz(int mat[4][2], int linha, int coluna);
5
6 int main(){
7     int matriz[4][2];
8
9     preencheMatriz(matriz, 4, 2);
10
11     imprimeMatriz(matriz, 4, 2);
12
13     return 0;
14 }
15
16 void preencheMatriz(int mat[4][2], int linha, int coluna){
17     int i, j;
18     srand(time(NULL));
19     for(i = 0; i < linha; i++){
20         for(j = 0; j < coluna; j++){
21             mat[i][j] = rand() % 20;
22         }
23     }
24 }
25
26 void imprimeMatriz(int mat[4][2], int linha, int coluna){
27     int i, j;
28     printf("\nMatriz Gerada: \n");
29     for(i = 0; i < linha; i++){
30         for(j = 0; j < coluna; j++){
31             printf("%d ", mat[i][j]);
32         }
33         printf("\n");
34     }
35 }
```



Passando Matrizes por Parâmetro

```
1 #include <stdio.h>
2
3 void preencheMatriz(int mat[][2], int linha, int coluna);
4 void imprimeMatriz(int mat[][2], int linha, int coluna);
5
6 int main(){
7     int matriz[4][2];
8
9
10 |
11     preencheMatriz(matriz, 4, 2);
12
13     imprimeMatriz(matriz, 4, 2);
14
15
16     return 0;
17 }
18
19 void preencheMatriz(int mat[][2], int linha, int coluna){
20     int i, j;
21
22     srand(time(NULL));
23
24     for(i = 0; i < linha; i++){
25         for(j = 0; j < coluna; j++){
26             mat[i][j] = rand() % 20;
27         }
28     }
29 }
30
31 void imprimeMatriz(int mat[][2], int linha, int coluna){
32     int i, j;
33
34     printf("\nMatriz Gerada: \n");
35
36     for(i = 0; i < linha; i++){
37         for(j = 0; j < coluna; j++){
38             printf("%d ", mat[i][j]);
39         }
40         printf("\n");
41     }
42 }
```




Passando Matrizes por Parâmetro

```
1 #include <stdio.h>
2 |
3 #define LINHA 3
4 #define COLUNA 4
5
6 void preencheMatriz(int mat[][COLUNA]);
7 void imprimeMatriz(int mat[][COLUNA]);
8
9 int main(){
10
11     int matriz[LINHA][COLUNA];
12
13     preencheMatriz(matriz);
14
15     imprimeMatriz(matriz);
16
17     return 0;
18 }
19
20 void preencheMatriz(int mat[][COLUNA]){
21
22     int i, j;
23
24     srand(time(NULL));
25
26     for(i = 0; i < LINHA; i++){
27
28         for(j = 0; j < COLUNA; j++){
29
30             mat[i][j]= rand() % 20;
31
32         }
33     }
34
35 void imprimeMatriz(int mat[][COLUNA]){
36
37     int i, j;
38
39     printf("\nMatriz Gerada: \n");
40
41     for(i = 0; i < LINHA; i++){
42
43         for(j = 0; j < COLUNA; j++){
44
45             printf("%d ", mat[i][j]);
46
47         }
48     }
```