



Linguagem de Programação

Matrizes

Gil Eduardo de Andrade





Matrizes

Introdução

- Uma matriz pode ser definida, assim como um vetor, como um conjunto de elementos (variáveis) que apresentam um mesmo tipo e são referenciados por um nome comum;
- As matrizes em C, diferentemente dos vetores que possuem apenas uma dimensão, podem apresentar duas ou mais dimensões;





Matrizes

Matizes Bidimensionais

- Normalmente quando trabalhamos com matrizes em C estas são bidimensionais, ou seja, possuem duas dimensões, as quais denominamos *linhas* e *colunas*;
- Os elementos que compõem uma matriz bidimensional são acessados, necessariamente, através da especificação de dois índices inteiros, que correspondem respectivamente a *linha* e a *coluna* onde esse elemento se encontra dentro da matriz;





Matrizes

Declaração de Matrizes Bidimensionais

- A declaração de matrizes bidimensionais é efetuada, de forma geral, da seguinte maneira:
 - ***tipo nome_matriz[nr_linhas][nr_colunas];***
 - ***tipo***: indica qual é o tipo dos elementos (variáveis) que compõem a matriz – ex.: ***int, float, double***;
 - ***nome matriz***: indica o nome pelo qual a matriz é referenciada;
 - ***nr linhas***: indica o número total de linhas que a matriz possui;
 - ***nr colunas***: indica o número total de colunas que a matriz possui;





Matrizes

Matrizes Bidimensionais Inteiras:

- Considerando o slide anterior, poderíamos então declarar uma matriz do tipo *int* contendo **8 linhas** e **8 colunas**. Teríamos:
 - *int tabuleiro[8][8];*
- Ao declararmos a matriz *tabuleiro* garantimos que o espaço de memória necessário para armazenar todos os seus elementos seja reservado;





Matrizes

Matrizes Bidimensionais Inteiras:

- Contudo é importante observar que apenas declaramos a matriz, ou seja, ainda não especificamos os elementos (valores) que ele deve armazenar;
- Antes desta especificação, dizemos que a matriz possui armazenado *“lixo de memória”*, valores quaisquer gerados por outros programas que utilizaram o mesmo espaço de memória num momento anterior;





Matrizes

Armazenando Valores numa Matriz:

- Como mencionado, as matrizes bidimensionais possuem *linhas* e *colunas*, sendo assim ela pode armazenar um total de **“linhas x colunas”** elementos;
 - (Ex.: Uma matriz com 8 linhas e 8 colunas pode armazenar 64 elementos – 8 x 8);
- Para que seja possível especificar qual elemento da matriz inteira receberá um valor a ser armazenado, utilizamos dois índices (valores inteiros) que indicam o posição (linha, coluna) da matriz que receberá o dado;





Matrizes

Armazenando Valores numa Matriz Bidimensional:

- Os índices da matriz, tanto em linha como em coluna, variam de '0' até seu 'nr. de linha-1' ou 'nr. de coluna-1', ou seja, a matriz ***int tabuleiro[8][8]*** possui os índices: ***0, 1, 2, 3, 4, 5, 6 e 7***:
- Para armazenar dados nesta matriz utilizaríamos a seguinte sintaxe:
 - ***tabuleiro[0][0] = 1; tabuleiro[7][7] = -1;***





Matrizes

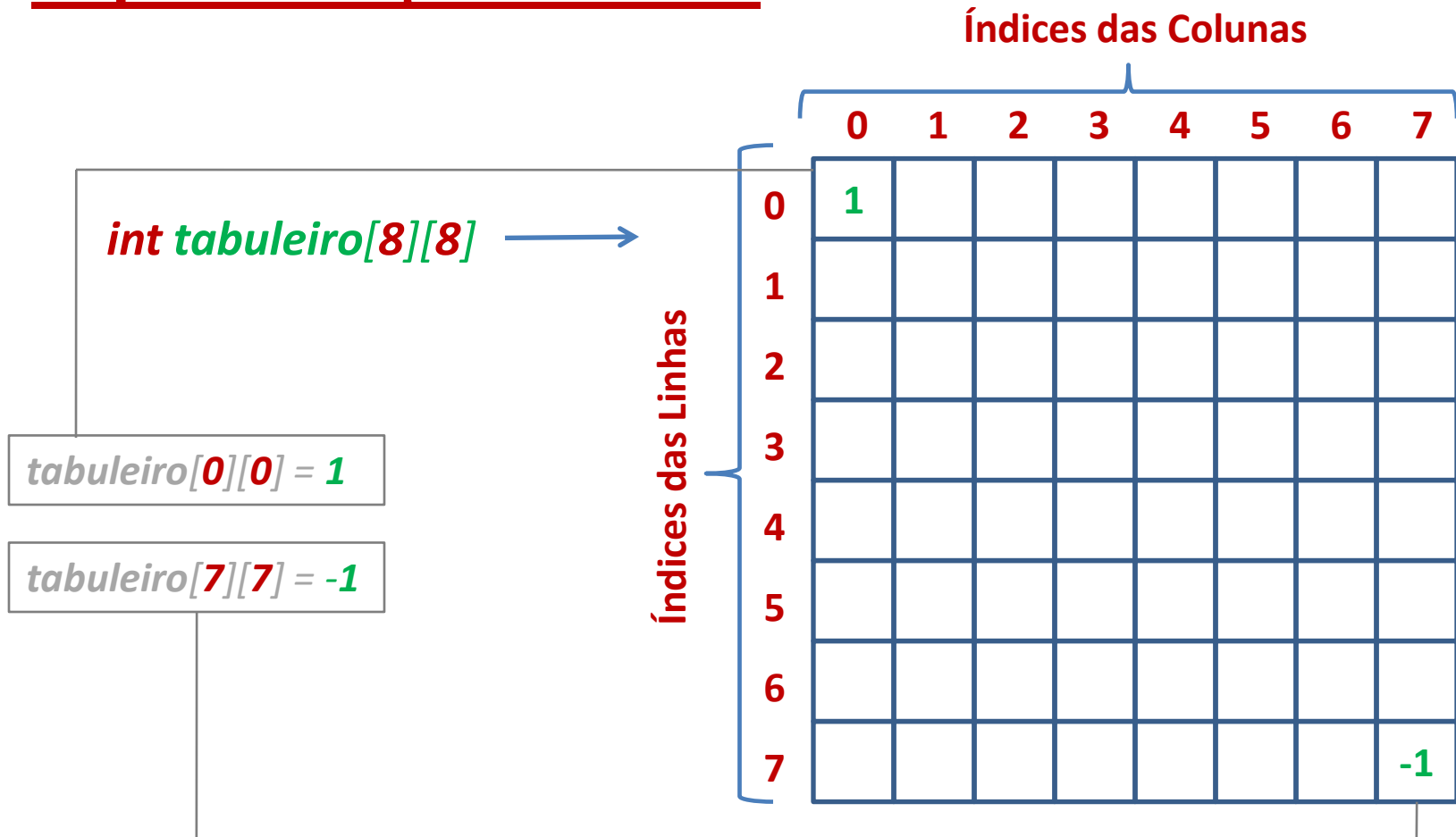
Representação Gráfica:

- Uma matriz, em programação de computadores, pode ser representada como uma planilha, já que esta contém linhas e colunas.
- Cada célula desta planilha é referenciada por dois índices, um indicando a linha e outro a coluna onde encontra-se o elemento que desejamos manipular;



Matrizes

Representação Gráfica:





Matrizes

Exemplos de Codificação (Atribuindo valores):

```
#include <stdio.h>

int main() {

    // declara a matriz tabuleiro do tipo 'int' com tamanho '8 x 8'
    int tabuleiro[8][8];

    // atribui o valor '1' a posição (0,0) da matriz
    tabuleiro[0][0] = 1;
    // atribui o valor '-1' a posição (7,7) da matriz
    tabuleiro[7][7] = -1;

    printf("\ntabuleiro[0][0] = %i", tabuleiro[0][0]);
    printf("\ntabuleiro[7][7] = %i", tabuleiro[7][7]);

    printf("\n");
    return 0;
}
```





Matrizes

Exemplos de Codificação (Atribuindo valores):

```
g1l3du4rd0@asus-ultrabook-g1l:/run/media/g1l3du4rd0/GEA/VirtualBox/Compartilhada/IFP... x
Arquivo Editar Ver Pesquisar Terminal Ajuda
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$ ./a.out
tabuleiro[0][0] = 1
tabuleiro[7][7] = -1
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$
```



Matrizes

Exemplos de Codificação (Declaração e Inicialização):

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

```
int main() {
```

```
    // declara a matriz tabuleiro do tipo 'int' com tamanho '8 x 8'  
    // e já o inicializa com valores pré-definidos
```

```
    int tabuleiro[8][8] = { {1, 3, 5, 7, 9, 11, 13, 15},  
                            {2, 4, 6, 8, 10, 12, 14, 16},  
                            {1, 3, 5, 7, 9, 11, 13, 15},  
                            {2, 4, 6, 8, 10, 12, 14, 16},  
                            {1, 3, 5, 7, 9, 11, 13, 15},  
                            {2, 4, 6, 8, 10, 12, 14, 16},  
                            {1, 3, 5, 7, 9, 11, 13, 15},  
                            {2, 4, 6, 8, 10, 12, 14, 16}  
    };
```

```
    printf("\ntabuleiro[2][5] = %i", tabuleiro[2][5]);  
    printf("\ntabuleiro[7][1] = %i", tabuleiro[7][1]);
```

```
    printf("\n");  
    return 0;
```

```
}
```



Matrizes

Exemplos de Codificação (Declaração e Inicialização):

```
g1l3du4rd0@asus-ultrabook-g1l:/run/media/g1l3du4rd0/GEA/VirtualBox/Compartilhada/IF... x
Arquivo Editar Ver Pesquisar Terminal Ajuda
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$ ./a.out
tabuleiro[2][5] = 11
tabuleiro[7][1] = 4
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$
```





Matrizes

Exemplos de Codificação (Percorrendo um Vetor):

```
#include <stdio.h>

int main() {

    // declara a matriz tabuleiro do tipo 'int' com tamanho '8 x 8'
    // e já o inicializa com valores pré-definidos
    int tabuleiro[8][8] = { {1, 2, 3, 4, 5, 6, 7, 8},
                           {9, 8, 7, 6, 5, 4, 3, 2},
                           {1, 2, 3, 4, 5, 6, 7, 8},
                           {9, 8, 7, 6, 5, 4, 3, 2},
                           {1, 2, 3, 4, 5, 6, 7, 8},
                           {9, 8, 7, 6, 5, 4, 3, 2},
                           {1, 2, 3, 4, 5, 6, 7, 8},
                           {9, 8, 7, 6, 5, 4, 3, 2},
    };

    int linha, coluna;

    for(linha=0; linha<8; linha++) {
        for(coluna=0; coluna<8; coluna++) {
            printf("(%i,%i)=%i ", linha, coluna, tabuleiro[linha][coluna]);
        }
        printf("\n");
    }

    printf("\n");
    return 0;
}
```

Matrizes

Exemplos de Codificação (Percorrendo um Vetor):

```
g1l3du4rd0@asus-ultrabook-g1l:/run/media/g1l3du4rd0/GEA/VirtualBox/Compartilhada/IFPR/2015/PC/aula10/ex... x
Arquivo Editar Ver Pesquisar Terminal Ajuda
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$ ./a.out
(0,0)=1 (0,1)=2 (0,2)=3 (0,3)=4 (0,4)=5 (0,5)=6 (0,6)=7 (0,7)=8
(1,0)=9 (1,1)=8 (1,2)=7 (1,3)=6 (1,4)=5 (1,5)=4 (1,6)=3 (1,7)=2
(2,0)=1 (2,1)=2 (2,2)=3 (2,3)=4 (2,4)=5 (2,5)=6 (2,6)=7 (2,7)=8
(3,0)=9 (3,1)=8 (3,2)=7 (3,3)=6 (3,4)=5 (3,5)=4 (3,6)=3 (3,7)=2
(4,0)=1 (4,1)=2 (4,2)=3 (4,3)=4 (4,4)=5 (4,5)=6 (4,6)=7 (4,7)=8
(5,0)=9 (5,1)=8 (5,2)=7 (5,3)=6 (5,4)=5 (5,5)=4 (5,6)=3 (5,7)=2
(6,0)=1 (6,1)=2 (6,2)=3 (6,3)=4 (6,4)=5 (6,5)=6 (6,6)=7 (6,7)=8
(7,0)=9 (7,1)=8 (7,2)=7 (7,3)=6 (7,4)=5 (7,5)=4 (7,6)=3 (7,7)=2

[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$
```





Matrizes do tipo 'char'

Matrizes x Strings:

- Assim como visto na aula de vetores (String) é possível trabalharmos com matrizes de caracteres (*char*);
- Sendo assim, podemos considerar que uma matriz de caracteres permite armazenar mais de uma String que nada mais é que um conjunto de caracteres (elementos do tipo *char*);





Matrizes do tipo 'char'

Capturando e Armazenado Strings:

- Torna-se possível trabalharmos com o método *gets()* para armazenar strings em linhas individuais de uma matriz do tipo *char*;
- Sendo assim ao solicitarmos que o usuário digite uma string (ex.: nome) utilizamos a função *gets()* passando a esta a matriz e especificando apenas a linha dela que receberá a string digitada;
 - *Ex.: gets(matriz[0]);*





Matrizes do tipo 'char'

Capturando e Armazenado Strings:

- Torna-se possível trabalharmos com o método *gets()* para armazenar strings em linhas individuais de uma matriz do tipo *char*;
- Sendo assim ao solicitarmos que o usuário digite uma string (ex.: nome) utilizamos a função *gets()* passando a esta a matriz e especificando apenas a linha dela que receberá a string digitada;
 - *Ex.: gets(matriz[0]);*



Matrizes do tipo 'char'

Exemplos de Codificação (Capturando String):

```
#include <stdio.h>

int main() {

    // declara a matriz 'nomes' do tipo 'char' com tamanho '3 x 50'
    int nomes[3][50];
    int linha;

    // Solicita ao usuário que digite 3 nomes e armazena cada um
    // deles numa linha da matriz 'nomes'
    for(linha=0; linha<3; linha++) {
        printf("Nome: ");
        __fpurge(stdin);
        gets(nomes[linha]);
    }
    printf("\nNOMES ARMAZENADOS\n-----");

    // Mostra os nomes que foram armazenados na matriz 'nomes'
    for(linha=0; linha<3; linha++) {
        printf("\nNome %i: %s", linha, nomes[linha]);
    }

    printf("\n");
    return 0;
}
```

Matrizes do tipo 'char'

Exemplos de Codificação (Capturando String):

```
g1l3du4rd0@asus-ultrabook-g1l/run/media/g1l3du4rd0/GEA/VirtualBox/Compartilhada/IFPR/20... x
Arquivo Editar Ver Pesquisar Terminal Ajuda
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$ ./a.out
Nome: Gil Eduardo
Nome: Maria Chiquinha
Nome: Maicon Asask

NOMES ARMAZENADOS
-----
Nome 0: Gil Eduardo
Nome 1: Maria Chiquinha
Nome 2: Maicon Asask
[g1l3du4rd0@asus-ultrabook-g1l pc_exdoc10]$
```



Matrizes

Exemplos Utilizados no Documento

http://www.gileduardo.com.br/ifpr/pci/downloads/pc_exdoc10.zip

Mais Exemplos sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/pci/downloads/pc_ex10.zip

Exercícios sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/lp/downloads/pc_pratica10.pdf

