

BCC 201 - Introdução à Programação

# Estruturas Homogêneas II (Matrizes)

Guillermo Cámara-Chávez  
UFOP

# Introdução I

- ▶ Imaginemos que queremos ler as notas de 4 provas para cada aluno e
- ▶ calcular a média do aluno e a média da classe.
- ▶ O tamanho máximo da turma é de 50 alunos
- ▶ Uma solução seria criar 4 vetores cada um com 50 posições.
- ▶ E então ler as respectivas informações

```
float nota1[50], nota2[50], nota3[50], nota4[50];
```

# Introdução II

	<b>Aluno</b>	<b>Nota1</b>	<b>Nota2</b>	<b>Nota3</b>	<b>Nota4</b>	<b>Média</b>
0	Pedro	5.6	6.0	7.3	5.6	6.1
1	Ana	10	4.0	5.0	7.3	6.6
2	Luiz	4.5	2.0	5.5	1.0	3.3
...	...	...	...	...	...	...
48	Matheus	7.2	6.6	8.1	8.8	7.7
49	Andre	6.0	9.0	7.3	4.5	6.6

# Introdução III

- ▶ Agora suponha que estamos trabalhando com no máximo 100 provas e 50 alunos
- ▶ Seria cansativo criar 100 vetores e atribuir 100 nomes diferentes

	Aluno		Nota1		Nota2		Nota3		Nota99		Nota100		Média
0	Pedro	0	5.6	0	6.0	0	7.3	0	7.3	0	5.6	0	6.1
1	Ana	1	10	1	4.0	1	5.0	1	5.0	1	7.3	1	6.6
2	Luiz	2	4.5	2	2.0	2	5.5	2	5.5	2	1.0	2	3.3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
48	Matheus	48	7.2	48	6.6	48	8.1	48	8.1	48	8.8	48	7.7
49	Andre	49	6.0	49	9.0	49	7.3	49	7.3	49	4.5	49	6.6

# Introdução IV

- ▶ Para resolver esse problema podemos utilizar matrizes

	<b>Aluno</b>		<b>Nota1</b>		<b>Nota2</b>		<b>Nota3</b>		<b>Nota99</b>		<b>Nota100</b>		<b>Média</b>
0	Pedro	0	5.6	0	6.0	0	7.3	0	7.3	0	5.6	0	6.1
1	Ana	1	10	1	4.0	1	5.0	1	5.0	1	7.3	1	6.6
2	Luiz	2	4.5	2	2.0	2	5.5	2	5.5	2	1.0	2	3.3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
48	Matheus	48	7.2	48	6.6	48	8.1	48	8.1	48	8.8	48	7.7
49	Andre	49	6.0	49	9.0	49	7.3	49	7.3	49	4.5	49	6.6

# Introdução V

- ▶ Para resolver esse problema podemos utilizar matrizes

	<b>Aluno</b>		<b>Nota1</b>	<b>Nota2</b>	<b>Nota3</b>		<b>Nota99</b>	<b>Nota100</b>		<b>Média</b>
0	Pedro	0	5.6	6.0	7.3	0	7.3	5.6	0	6.1
1	Ana	1	10	4.0	5.0	1	5.0	7.3	1	6.6
2	Luiz	2	4.5	2.0	5.5	2	5.5	1.0	2	3.3
...	...	...	...	...	...	...	...	...	...	...
48	Matheus	48	7.2	6.6	8.1	48	8.1	8.8	48	7.7
49	Andre	49	6.0	9.0	7.3	49	7.3	4.5	49	6.6

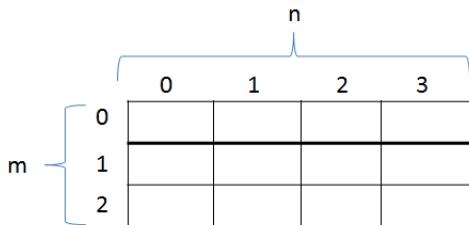
# Introdução VI

- ▶ Para resolver esse problema podemos utilizar matrizes

Aluno		0	1	2	...	98	99	Média		
0	Pedro	0	5.6	6.0	7.3	...	7.3	5.6	0	6.1
1	Ana	1	10	4.0	5.0	...	5.0	7.3	1	6.6
2	Luiz	2	4.5	2.0	5.5	...	5.5	1.0	2	3.3
...	...	...	...	...	...	...	...	...	...	...
48	Matheus	48	7.2	6.6	8.1	...	8.1	8.8	48	7.7
49	Andre	49	6.0	9.0	7.3	...	7.3	4.5	49	6.6

# Introdução VII

- ▶ Uma matriz é um vetor (conjunto de variáveis do mesmo tipo) que possui duas ou mais dimensões
- ▶ Matrizes podem ser considerados como “vetores de vetores” .
- ▶ Por exemplo, uma matriz bi-dimensional pode ser visto como uma tabela de  $m$  linhas (filas) e  $n$  colunas.





# Declarando uma matriz I

```
<tipo> nome_da_matriz [<linhas >][<colunas >]
```

- ▶ Uma matriz possui *linhas*  $\times$  *colunas* variáveis do tipo <tipo>
- ▶ As linhas são numeradas de 0 a *linhas* - 1
- ▶ As colunas são numeradas de 0 a *colunas* - 1
- ▶ Exemplo:

```
//matriz com 100 linhas e 50 colunas  
float notas[100][50];
```

# Acessando uma matriz I

- ▶ A forma de usar um elemento da matriz é:

```
nome_da_matriz[<linha >][<coluna >]
```

- ▶ Ex: acessar o elemento da linha 1 e coluna 10.

```
matriz [1][10]
```

- ▶ O compilador não verifica se foram utilizados valores válidos para a linha e para a coluna

## Exemplo 1 I

Criar uma matriz de  $3 \times 4$  (3 linhas e 4 colunas) elementos. Inserir valores e logo mostrar o conteúdo da matriz

## Exemplo 1 II

```
int main()
{
    int M[3][4];
    //Inserção de dados
    for (int i = 0; i < 3; i++) //para as linhas
        for (int j = 0; j < 4; j++) //para as colunas
        {
            cout << "Matriz[" <<i<< "]"[" <<j<<"] = ";
            cin >> M[i][j];
        }
    //mostrar o conteúdo da matriz
    for (int i = 0; i < 3; i++) //para as linhas
    {
        for (int j = 0; j < 4; j++) //para as colunas
            cout << M[i][j] <<" ";
        cout << endl;
    }
    return 0;
}
```

## Exemplo 2 I

- ▶ Escreva um programa que inicialize uma matriz  $5 \times 5$  com 0s em todas as posições.
- ▶ O usuário irá digitar o índice da linha e o índice da coluna e em seguida o valor das posições não nulas.
- ▶ A leitura será feita enquanto os índices forem não negativos.
- ▶ Após a leitura imprima a matriz na tela.

## Exemplo 2 II

M =

	0	1	2	3	4
0					
1					
2					
3					
4					

## Exemplo 2 III

M =

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

## Exemplo 2 IV





## Exemplo 2 V

M =

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	<b>4</b>	0	0	0
4	0	0	0	0	0

## Exemplo 2 VI

```
int main(){
    int matriz[5][5], linha = 0, coluna = 0, valor;
    //inicializa com zero
    for (int i = 0; i < 5; i++) //para as linhas
        for (int j = 0; j < 5; j++) //para as colunas
            matriz[i][j] = 0;
    // int matriz[5][5] = {0};
    while(1){
        cout << "\n Inserir posições: ";
        cin >> linha >> coluna;
        if (linha > -1 && coluna > -1){
            cout << "\n Inserir valor: ";
            cin >> matriz[linha][coluna];
        }
        else
            break;
    }
    . . .
}
```

## Exemplo 2 VII

```
int main(){
    .
    .
    .
    //mostrar o conteúdo da matriz
    for (int i = 0; i < 5; i++) //para as linhas
    {
        for (int j = 0; j < 5; j++) //para as colunas
            cout << matriz[i][j] <<" ";
        cout << endl;
    }
    return 0;
}
```

# Lembrando ... I

Passagem de um vetor como parâmetro:

## 1. Tamanho fixo

```
int Funcao (int Vet[10]){  
    ...  
}
```

## 2. Tamanho variável

```
int Funcao (int Vet[], int tamanho){  
    ...  
}
```

```
int Funcao (int* Vet, int tamanho){  
    ...  
}
```

# Passagem de uma matriz como parâmetro I

## 1. Tamanho fixo

```
int Funcao (int Mat[10][5]){  
    ...  
}
```

## 2. Tamanho variável

```
int Funcao (int Mat[][5], int lin){  
    ...  
}
```

## Exemplo 3 I

1. Escreva um programa que lê 2 matrizes  $5 \times 5$ , mostre-as na tela e mostre o maior valor entre elementos correspondentes.

	0	1	2	3	4
A = 0	1	13	6	8	7
1	7	10	16	6	8
2	15	18	1	16	9
3	3	2	4	9	17
4	5	8	5	11	12

	0	1	2	3	4
B = 0	8	6	19	9	3
1	5	11	6	7	10
2	11	20	19	11	18
3	7	1	6	2	15
4	7	10	15	19	20

	0	1	2	3	4
C = 0	8	13	19	9	7
1	7	11	16	7	10
2	15	20	19	16	18
3	7	2	6	9	17
4	7	10	15	19	20

## Exemplo 3 II

```
#define LIN 5
#define COL 5
void Insere(int A[LIN][COL]);
void MaiorElemento(int A[LIN][COL], int B[LIN][COL],
                  int C[LIN][COL]);
void Mostra(int A[LIN][COL]);
int main()
{
    int mat1[LIN][COL], mat2[LIN][COL], mat3[LIN][COL];
    cout << "Matriz 1 \n"; Insere(mat1);
    cout << "Matriz 2 \n"; Insere(mat2);
    MaiorElemento(mat1, mat2, mat3);
    cout << "Matriz 1: \n"; Mostra(mat1);
    cout << "Matriz 2: \n"; Mostra(mat2);
    cout << "Matriz 3: \n"; Mostra(mat3);
    return 0;
}
```

## Exemplo 3 III

```
void Inserir(int mat[LIN][COL])
{
    int i, j;
    for (i = 0; i < LIN; i++)
    {
        for (j = 0; j < COL; j++)
        {
            cout << "Matriz[" <<i<< "]" <<j<<"] = ";
            cin >> mat[i][j];
        }
    }
}
```



## Exemplo 3 IV

```
void MaiorElemento(int A[LIN][COL], int B[LIN][COL],
                  int C[LIN][COL])
{
    int i, j;
    for (i = 0; i < LIN; i++)
        for (j = 0; j < COL; j++)
            if (A[i][j] > B[i][j])
                C[i][j] = A[i][j];
            else
                C[i][j] = B[i][j];
}
```

## Exemplo 3 V

```
void Mostra(int mat[LIN][COL])
{
    int i, j;
    for (i = 0; i < LIN; i++)
    {
        for (j = 0; j < COL; j++)
            cout << mat[i][j] <<" ";
        cout << endl;
    }
}
```

## Exemplo 3 I

1. Criar uma matriz quadrada e mostrar os elementos da diagonal principal e da diagonal secundaria.

## Exemplo 3 II

Diagonal principal

M =

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

M =

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

Diagonal secundaria

## Exemplo 3 III

```
#define LIN 3
#define COL 3
int main()
{
    int mat[LIN][COL], i, j;
    for (i = 0; i < LIN; i++)
        for (j = 0; j < COL; j++)
            cin >> mat[LIN][COL];
    cout << "Mostrando a diagonal principal... \n";
    for (i = 0; i < LIN; i++)
        cout << mat[i][i] << " ";
    cout << "Mostrando a diagonal secundaria... \n";
    for (j = COL-1, i = 0; j >= 0; j--, i++)
        cout << mat[i][j] << " ";
    return 0;
}
```

## Exemplo 3 IV

Defina uma função para determinar os índices do maior valor dentre os  $n^2$  elementos de uma matriz de double

## Exemplo 3 V

0x10	?
0x11	?
0x12	?
0x13	?
0x14	?
0x15	?
0x16	?
0x17	?
0x18	?
0x19	?
0x20	?
0x21	?
0x22	?
0x23	?
0x24	?
0x25	?

```
int x, *px;
```

```
x = 23;
```

```
px = &x;
```

```
*px = 19;
```

## Exemplo 3 VI

0x10	?
0x11	?
0x12	
0x13	
0x14	
0x15	?
0x16	?
0x17	?
0x18	?
0x19	
0x20	
0x21	
0x22	?
0x23	?
0x24	?
0x25	?

**x**

**px**



```
int x, *px;
```

```
x = 23;
```

```
px = &x;
```

```
*px = 19;
```



## Exemplo 3 VII

0x10	?
0x11	23
0x12	
0x13	
0x14	
0x15	?
0x16	?
0x17	?
0x18	?
0x19	
0x20	
0x21	
0x22	?
0x23	?
0x24	?
0x25	?

**x**



**px**

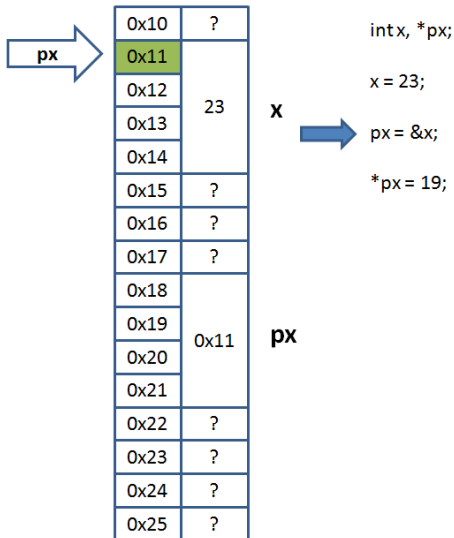
```
int x, *px;
```

```
x = 23;
```

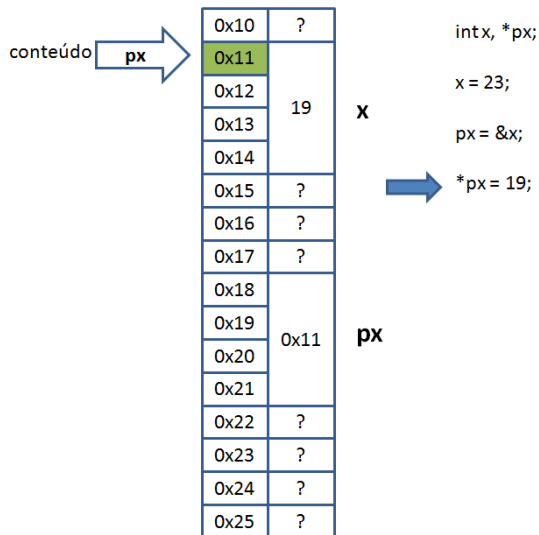
```
px = &x;
```

```
*px = 19;
```

## Exemplo 3 VIII



## Exemplo 3 IX



## Exemplo 3 X

```
#define N 5
void indiceMaior(int M[N][N], int n, int *plin ,
                int *pcol);
void Insere(int M[N][N], int n);
int main()
{
    int mat[N][N], pos_linha , pos_coluna;
    Insere(mat, N);
    indiceMaior(mat, N, &pos_lin , &pos_coluna);
    cout << "O maior elemento ocupa a posicao: "
          << pos_linha << "," << pos_coluna;
    return 0;
}
```

## Exemplo 3 XI

```
void indiceMaior(int M[N][N], int n, int *plin ,
                 int *pcol)
{
    int maior = M[0][0], i, j;
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
        {
            if (M[i][j] > maior){
                maior = M[i][j];
                *plin = i;
                *pcol = j;
            }
        }
}
```

## Exemplo 3 XII

```
void Inserir(int M[N][N], int n)
{
    int i, j;
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < n; j++)
        {
            cout << "Matriz[" <<i<< "][" <<j<<"] = ";
            cin >> mat[i][j];
        }
    }
}
```

## Exercícios propostos I

1. Defina duas funções, uma que calcula a média e outra que calcula o desvio padrão de uma amostra representada por uma matriz  $m \times n$  de `double`. Utilize as fórmulas e as declarações fornecidas. Observe que a função que calcula o desvio padrão amostral deverá fazer uma chamada à função que calcula a média amostral.

$$media = \frac{\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} X_{ij}}{m*n} \quad desviopadrao = \sqrt{\frac{\sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (X_{ij} - media)^2}{m*n-1}}$$

2. Ler uma matriz  $G$  30x30 e criar 2 vetores,  $SL$  e  $SC$ , de 30 elementos cada, contendo respectivamente as somas das linhas e das colunas de  $G$ . Escrever os vetores criados.

FIM