

Matrizes [][]

PROFESSORA: FERNANDA DOS
SANTOS CUNHA

ESTAGIÁRIO: VINICIUS DE
AQUINO PIAI



Matrizes

- Duas ou mais dimensões;
- Mesmo tipo de dados;
- Iterado a partir de dois ou mais laços de repetição

	5 notas					média
	1	2	3	4	5	6
aluno1						
aluno2						
aluno3						
aluno4						
aluno5						
aluno6				7,5		
aluno7						

aluno6[4] é a 4ª nota do 6º aluno

Declaração - Visualg

<nome_variável> : vetor [..

<nome_variável>: nome da variável declarada

: início da linha, sempre 1

<lf>: final da linha

<ci>: início da coluna, sempre 1

<cf>: final da coluna

<tipo de dado> : tipo de dado da variável(inteiro, real, ...)

Exemplo - Visualg

Var

```
numeros : vetor [1..5, 1..5] de inteiro  
nomes : vetor [1..4, 1..3] de caractere  
cadeiras : vetor [1..4, 1..2] de caracter
```

Inicio

Fimalgoritmo

Declaração - C++

- ♦ `<tipo_de_dado> <nome_variável>[<l>][<c>];`
 <tipo_de_dado>: tipo do dado da matriz (int, float, string, ...)
 <nome_variável>: nome da variável declarada
 <l>: quantidade de linhas da matriz
 <c> : quantidade de colunas da matriz

Exemplo – C++

```
#include <iostream>
using namespace std;

int main()
{
    int matrizInteiro[5][5];
    string matrizString[2][4];
    char matrizChar[3][2];
    bool matrizBool[10][1000];
    float matrizFloat[2][15];
}
```

Exemplo – C++

```
#include <iostream>
using namespace std;
const int TMAX = 2;

int main()
{
    int numeros[TMAX][TMAX] = {{0}};
    string nomes[TMAX][TMAX] = {{ "José" }};

    int numeros[TMAX][TMAX] = {{1, 2}, {3, 4}};
    string nomes[TMAX][TMAX] = {{ "José", "Camila", "Juan", "Beatriz" }};

    int numeros[TMAX][TMAX] = {{1, 2}};
    string nomes[TMAX][TMAX] = {{ "José", "Camila" }};

    return 0;
}
```

Para o enunciado a seguir

- ♦ Desenvolva um código utilizando matrizes, aonde uma sala tem 5 alunos e cada um tem 3 notas e uma média. Leia as notas de todos os alunos e calcule a nota de cada um, colocando-as na matriz.

Visualg

Var

// Seção de Declarações das variáveis

notas : vetor [1..5, 1..4] de real

soma : real

i, j: inteiro

Inicio

para **i** de 1 ate 5 faca

soma <- 0

 para **j** de 1 ate 4 faca

 se **j** < 4 entao

notas[**i**, **j**] <- randi(10)

soma <- **soma** + **notas**[**i**, **j**]

 senao

notas[**i**, **j**] <- **soma** / 3

 fimse

 fimpara

fimpara

para **i** de 1 ate 5 faca

 para **j** de 1 ate 4 faca

 escreva(**notas**[**i**, **j**], " ")

 fimpara

 escreval()

fimpara

Fimalgoritmo



C++

```
#include <iostream>
#include <iomanip>
using namespace std;

const int TLINHAMAX = 5;
const int TCOLUNAMAX = 4;

int main()
{
    srand(time(NULL));

    float notas[TLINHAMAX][TCOLUNAMAX], soma;

    for(int i = 0; i < TLINHAMAX; i++){
        soma = 0;
        for(int j = 0; j < TCOLUNAMAX; j++){
            if(j < TCOLUNAMAX - 1){
                notas[i][j] = rand() % 10;
                soma += notas[i][j];
            }else{
                notas[i][j] = soma / (TCOLUNAMAX - 1);
            }
        }
    }

    for(int i = 0; i < TLINHAMAX; i++){
        for(int j = 0; j < TCOLUNAMAX; j++){
            cout << notas[i][j] << "\t";
        }
        cout << endl;
    }

    return 0;
}
```

Matrizes – Diagonal Principal

- ♦ Consiste em encontrar ou preencher a diagonal principal da matriz
- ♦ Se caracteriza por sempre estar nas posições aonde o indexador i é igual ao indexador j

Matrizes – Diagonal Principal

Var

```
numeros : vetor [1..5, 1..5] de inteiro  
i, j : inteiro
```

Inicio

```
para i de 1 ate 5 faca  
  para j de 1 ate 5 faca  
    se i = j entao  
      numeros[i, j] <- 1  
    fimse  
  fimpara  
fimpara  
  
para i de 1 ate 5 faca  
  para j de 1 ate 5 faca  
    escreva(numeros[i, j], " ")  
  fimpara  
  escreval()  
fimpara
```

Fimalgoritmo

```
#include <iostream>  
using namespace std;  
  
const int TMAX = 5;  
  
int main()  
{  
  
  int numeros[TMAX][TMAX] = {{0}};  
  
  for(int i = 0; i < TMAX; i++){  
    for(int j = 0; j < TMAX; j++){  
      if(i == j){  
        numeros[i][j] = 1;  
      }  
    }  
  }  
  
  for(int i = 0; i < TMAX; i++){  
    for(int j = 0; j < TMAX; j++){  
      cout << numeros[i][j] << " ";  
    }  
    cout << endl;  
  }  
  
  return 0;  
}
```

Multiplicação de matrizes

- ♦ Só entre matrizes numéricas
- ♦ Feito com 3 laços de repetição
- ♦ Segue a regra onde:

A quantidade de colunas a matriz A deve ser igual a quantidade de linhas da matriz B

Visualg

Var

```
A, B : vetor [1..10, 1..10] de inteiro  
res : vetor [1..10, 1..10] de inteiro  
i, j, k, C1, C2, R1, R2 : inteiro
```

Inicio

```
C1 <- 2  
C2 <- 2  
R1 <- 2  
R2 <- 2
```

```
A[1, 1] <- 1  
A[1, 2] <- 1  
A[2, 1] <- 2  
A[2, 2] <- 2
```

```
B[1, 1] <- 1  
B[1, 2] <- 1  
B[2, 1] <- 2  
B[2, 2] <- 2
```

```
para i de 1 ate R1 faca  
  para j de 1 ate C2 faca  
    res[i, j] <- 0  
    para k de 1 ate R2 faca  
      res[i, j] <- res + A[i, k] * B[k, j]  
    fimpara  
  escreva(res[i, j], " ")  
  fimpara  
escreval()  
fimpara
```

Fimalgoritmo



C++

```
#include <iostream>
using namespace std;

const int R1 = 2;
const int C1 = 2;
const int R2 = 2;
const int C2 = 2;

int main()
{

    int matrizA[R1][C1] = {{ 1, 1},
                           {2, 2}},
        matrizB[R2][C1] = {{ 1, 1},
                           {2, 2}};

    int resultado[R1][C2];
    if(C1 != R2){
        cout << "A quantidade de colunas da primeira matriz deve ser";
        cout << "igual a quantidade de linhas da segunda matriz";
    }else{
        for (int i = 0; i < R1; i++) {
            for (int j = 0; j < C2; j++) {
                resultado[i][j] = 0;
                for (int k = 0; k < R2; k++) {
                    resultado[i][j] += matrizA[i][k] * matrizB[k][j];
                }
                cout << resultado[i][j] << "\t";
            }
            cout << endl;
        }
    }

    return 0;
}
```

```
#include <iostream>
using namespace std;

const int R1 = 2;
const int C1 = 2;
const int R2 = 2;
const int C2 = 2;

int main()
{

    int matrizA[R1][C1] = {{ 1, 1},
                           {2, 2}},
        matrizB[R2][C1] = {{ 1, 1},
                           {2, 2}};

    int resultado[R1][C2];
    if(C1 != R2){
        cout << "A quantidade de colunas da primeira matriz deve ser";
        cout << "igual a quantidade de linhas da segunda matriz";
    }else{
        for (int i = 0; i < R1; i++) {
            for (int j = 0; j < C2; j++) {
                resultado[i][j] = 0;
                for (int k = 0; k < R2; k++) {
                    resultado[i][j] += matrizA[i][k] * matrizB[k][j];
                }
                cout << resultado[i][j] << "\t";
            }
            cout << endl;
        }

        return 0;
    }
}
```

	B	0	1
0		1	1
1		2	2

⋮

A	0	1
0	1	1
1	2	2

Res	0	1
0	3	3
1	6	6

i	j	k	Res[0][0]	Res[0][1]	Res[1][0]	Res[1][1]
0	0	0	1*1	0	0	0
0	0	1	1*1 + 1*2	0	0	0
0	1	0	2	1*1	0	0
0	1	1	2	1*1 + 1*2	0	0
1	0	0	2	3	2*1	0
1	0	1	2	3	2*1 + 2*2	0
1	1	0	2	3	6	2*1
1	1	1	2	3	6	2*1 + 2*2