

The background of the slide features a series of concentric, light gray circles that are centered on the left side and extend towards the right, creating a subtle, abstract pattern.

Matrizes

Prof: M.Sc Mário Angel Praia Garcia

1. Matrizes

- Conceito
 - Vetor de duas dimensões, também conhecido por matrizes, é um conjunto de elementos, todos de mesmo tipo, que podem ser acessados individualmente a partir de único nome.
 - Exemplos:
 - Matriz de valores inteiros

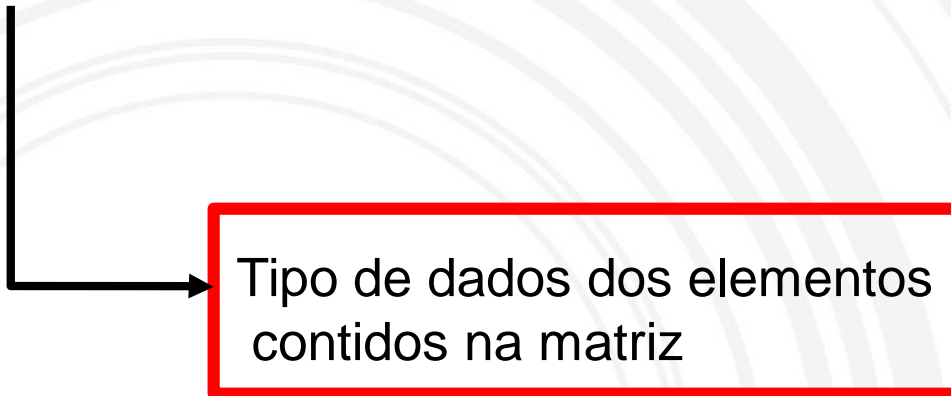
10	5
6	99

OBS: Não existe qualquer limite para o número de dimensões que um vetor pode conter.

1. Matrizes

- Declaração de matrizes:

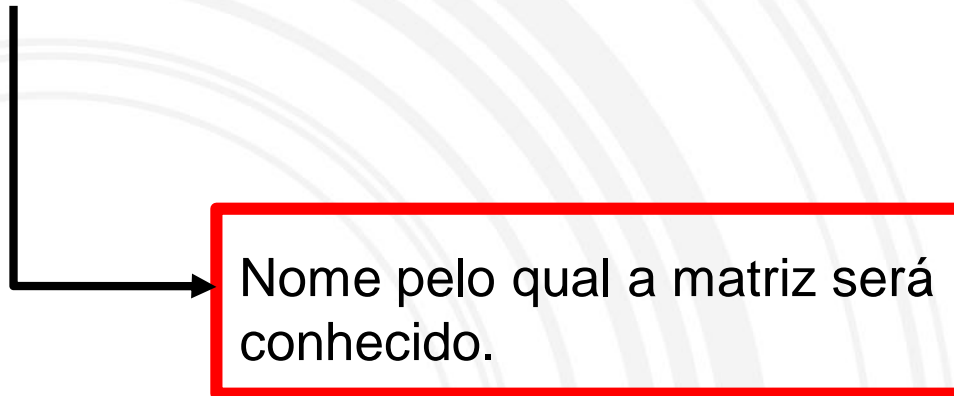
tipo **nome_da_matriz** [linha][coluna];



1. Matrizes

- Declaração de matrizes:

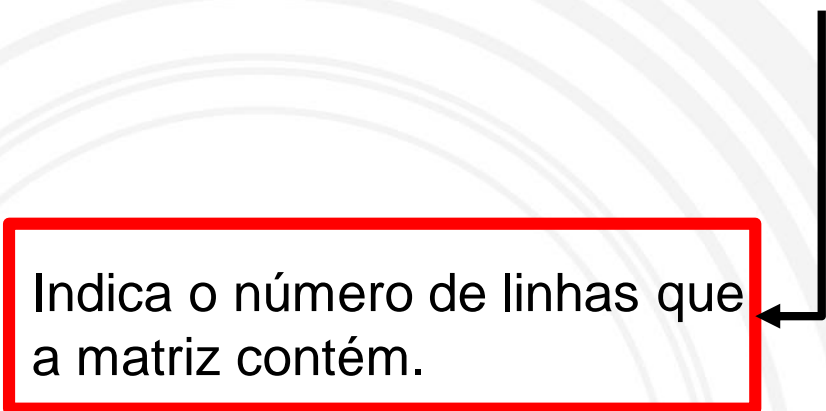
tipo **nome_da_matriz** [linha][coluna];



1. Matrizes

- Declaração de matrizes:

tipo **nome_da_matriz** [linha][coluna];




Indica o número de linhas que a matriz contém.

1. Matrizes

- Declaração de matrizes:

tipo **nome_da_matriz** [linha][coluna];



Indica o número de colunas
que a matriz contém.

1. Matrizes

- Exemplos de declaração de matrizes:
 - **int** g[3][3];
 - Matriz de números inteiros, chamado **g**, que possui 3 linhas e 3 colunas.
 - **float** renda[2][3];
 - Matriz de números reais, chamado **renda**, que possui 2 linhas e 3 colunas.
 - **int** mat[3][2];
 - Matriz de números inteiros, chamado **mat**, que possui 3 linhas e 2 colunas.

1. Matrizes

- Como uma matriz é exemplificada pelos seus índices:
 - **int** mat[2][2];

mat[0][0]	mat[0][1]
mat[1][0]	mat[1][1]

OBS: Em C os índices de uma matriz com n elementos variam SEMPRE entre 0 e n – 1.

1. Matrizes

- Como uma matriz é exemplificada pelos seus índices:
 - `int mat[2][2];`

<code>mat[0][0]</code>	<code>mat[0][1]</code>
<code>mat[1][0]</code>	<code>mat[1][1]</code>

OBS: Em C os índices de uma matriz com n elementos variam **SEMPRE** entre 0 e $n - 1$.

OBS: As posições de uma matriz **SEMPRE** podem ser acessadas através de índice colocado entre os colchetes[].

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[0][0] = 123;

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[0][0] = 123;

123	

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[0][1] = mat[0][0] * 2;

123	

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
- **int** mat[2][2];

mat[0][1] = mat[0][0] * 2;

123	246

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
- **int** mat[2][2];

mat[1][0] = mat[0][0] + mat[0][1];

123	246

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[1][0] = mat[0][0] + mat[0][1];

123	246
369	

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[1][1] = mat[0][0];

123	246
369	

1. Matrizes

- Inserindo valores em uma matriz utilizando seus índices:
 - **int** mat[2][2];

mat[1][1] = mat[0][0];

123	246
369	123

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
 - **int** mat[2][2];

```
for ( i = 0; i < 2; i++)  
    for ( j = 0; j < 2; j++)  
        mat[i][j] = i;
```


1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```


1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
 - `int mat[2][2];`

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```


1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

```
for ( i = 0; i < 2; i++)  
    for ( j = 0; j < 2; j++)  
        mat[i][j] = i;
```

0	

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

```
for ( i = 0; i < 2; i++)  
    for ( j = 0; j < 2; j++)  
        mat[i][j] = i;
```

0	1

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	1

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
 - `int mat[2][2];`

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	1

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

```
for ( i = 0; i < 2; i++)  
    for ( j = 0; j < 2; j++)  
        mat[i][j] = i;
```

0	1
0	

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
 - **int** mat[2][2];

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	1
0	

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	1
0	1

1. Matrizes

- Inserindo valores em uma matriz utilizando o **for**:
 - **int** mat[2][2];

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        mat[i][j] = i;
```

0	1
0	1

OBS: Outros laços de repetição(while, do..while) também podem ser usados para inserir valores em vetores. O scanf também é ser utilizado para ler valores para uma vetor

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

0	1
0	1

```
for ( i = 0; i < 2; i++)  
    for ( j = 0; j < 2; j++)  
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa:

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa:

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: mat[0][0] = 0

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: `mat[0][0] = 0`

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: mat[0][0] = 0
mat[0][1] = 1

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: `mat[0][0] = 0`
`mat[0][1] = 1`

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: mat[0][0] = 0

mat[0][1] = 1

mat[1][0] = 0

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- `int mat[2][2];`

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: `mat[0][0] = 0`

`mat[0][1] = 1`

`mat[1][0] = 0`

1. Matrizes

- Imprimindo valores em uma matriz utilizando o **for**:
- **int** mat[2][2];

0	1
0	1

```
for ( i = 0; i < 2; i++)
```

```
    for ( j = 0; j < 2; j++)
```

```
        printf("mat[%d][%d] = %d", i, j, mat[i][j]);
```

Saída do programa: mat[0][0] = 0

mat[0][1] = 1

mat[1][0] = 0

mat[1][1] = 1

1. Matrizes

- Inicializando uma matriz:
 - Tal como as variáveis, as matrizes quando são criados contêm valores aleatórios (LIXO) em cada uma das suas posições.

`tipo var[n][m] = { valor1, valor2, ..., valorn};`

- Exemplos:
 - `int v[3][3] = {1,2,3,4,5,6};`
 - `float matriz[2][2] = {1.0, 2.0, 3.0, 4.0};`


1. Matrizes

- Inicializando uma matriz:
- Se uma matriz com n elementos, inicialmente, receber apenas alguns elementos, as posições restantes serão iniciadas com o valor 0 (ZERO).

tipo var[n][m] = { valor₁, valor₂, ..., valor_n};


- Exemplos:

- `int a[2][2] = {1,2};`



1	2
0	0

- `float a[3][2] = {1.0, 2.0, 3.0};`



1.0	2.0	3.0
0.0	0.0	0.0

1. Matrizes

- Exercícios:

1. Crie um programa que leia uma matriz de inteiros 3x3 e imprima os valores lidos.
2. Leia uma matriz 4x4, conte e escreva quantos valores menor que 5 ela possui.
3. Crie uma matriz 10x10, imprima a matriz e a localização (linha e coluna) do maior valor.
4. Crie um programa que multiplique duas matrizes. Vale lembrar, que o n° de linhas da primeira matriz deve ser igual ao n° de colunas da segunda matriz.

1. Matrizes

- Matrizes em funções – 1º Forma:

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

```
void inic (int s[2][2])
```

```
{
```

```
    int i,j;
```

```
    for(i=0;i<6; i++)
```

```
        for(i=0;i<6; i++)
```

```
        {
```

```
            s[i][j] = i;
```

```
            printf("%d\n", s[i][j]);
```

```
        }
```

```
    }
```

```
int main ()
```

```
{
```

```
    int mat[2][2];
```

```
    inic(mat);
```

```
}
```

1. Matrizes

- Matrizes em funções – 2º Forma:

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

```
void inic (int s[][[]], int n, int b)
```

```
{
```

```
    int i,j;
```

```
    for(i=0;i<n; i++)
```

```
        for(j=0;j<b; j++)
```

```
        {
```

```
            s[i][j] = i;
```

```
            printf("%d\n", s[i][j]);
```

```
        }
```

```
}
```

```
int main ()
```

```
{
```

```
    int mat[2][2];
```

```
    inic(mat,2, 2);
```

```
}
```

1. Vetores

- Exercícios:

1. Crie um programa que leia duas matrizes de inteiros de tamanho 2×2 e uma função que imprima a soma dessas duas matrizes.
2. Faça uma função que recebe, por parâmetro, uma matriz $A[4][4]$ e retorna a soma dos seus elementos.
3. Faça uma função que recebe, por parâmetro, uma matriz $A[7][6]$ e uma coluna N e retorna a soma de todos os elementos dessa coluna.