

# Matrizes

Em algoritmos uma matriz é uma variável composta por linhas e colunas alocadas sequencialmente na memória.

# Matrizes

O que distingue cada célula de uma matriz são dois índices que referenciam sua localização dentro da estrutura.

Matriz[linha][ coluna]

# Matrizes

## Criação e inicialização de uma matriz:

- Considere uma matriz  $M[L][C]$ , onde  $L$  é o número de linhas e  $C$  é o número de colunas.
- A criação e inicialização de  $M$  são feitas conforme segue:

`int M[L][C] = {0};`

- Todos as células recebem o mesmo valor 0.

# Matrizes

Neste exemplo, inicializa quatro células com *zero*:

```
int matriz[2][2] = {0};
```

0 0

0 0



# Matrizes

Neste exemplo, inicializa todas as células com “UEM”:

```
char m[3][3] = {"UEM","UEM","UEM"};
```

# Matrizes

```
char m[3][3] = {"UEM","UEM","UEM"};
for(int i=0; i<3; i++){
    for(int k=0; k<3; k++)
        printf("%c ", m[i][k]);
    printf("\n");
}
```

U	E	M
U	E	M
U	E	M

# Exemplo de Matriz

```
#define linhas 2
#define colunas 6
int main(void){
    int m[linhas][colunas] = {0};
    for(int i=0; i<linhas; i++){
        for(int k=0; k<colunas; k++){
            printf("%i ", m[i][k]);
            printf("\n");
        }
    }
```

0	0	0	0	0	0
0	0	0	0	0	0

# Matrizes

**Atribuindo valores à Matriz:**

```
int m[linhas][colunas] = {0};
```

```
for(int k=0; k<colunas; k++)  
    m[0][k] = 1;
```

1	1	1	1	1
0	0	0	0	0



# Carregando uma Matriz

```
int m[linhas][colunas] = {0};  
for(int i=0; i<linhas; i++){  
    for(int k=0; k<colunas; k++){  
        printf("m[%i][%d] = ", i,k);  
        scanf("%i", &m[i][k]);  
    }  
    printf("\n");  
}
```

```
M[0][0] = 10  
M[0][1] = 20  
M[0][2] = 30  
M[0][3] = 40  
M[0][4] = 50  
M[1][0] = 60  
...
```

10	20	30	40	50
60	0	0	0	0

# Mostrando a Matriz

10	20	30	40	50
60	70	80	90	100

```
for(int i=0; i<linhas; i++){  
    for(int k=0; k<colunas; k++){  
        printf("%4i ", m[i][k]);  
        printf("\n");  
    }  
}
```

10	20	30	40	50
60	70	80	90	100

# EXERCÍCIO 1

Faça um programa que carregue uma matriz  $2 \times 2$ , calcule e mostre uma matriz resultante que será a matriz digitada multiplicada pelo maior elemento da matriz.

# EXERCÍCIO 1

```
#define linhas 2
```

```
#define colunas 2
```

```
int main(void){
```

```
    int i,k, m[linhas][colunas] = {0};
```

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

```
        for(k=0; k<colunas; k++){
```

```
            printf("\nDigite um numero: ");    scanf("%i", &m[i][k]);
```

```
        }
```

```
//Mostrar a matriz inicial
```

```
...
```

## EXERCÍCIO 1

```
int m[linhas][colunas] = {0};
for(i=0; i<linhas; i++)
    for(k=0; k<colunas; k++){
        printf("\nDigite um numero: ");  scanf("%i", &m[i][k]);
    }
printf("\nMATRIZ INICIAL:\n");
for(i=0; i<linhas; i++){
    for(k=0; k<colunas; k++)
        printf("%4i ", m[i][k]);
    printf("\n");
}

//Encontrar o maior
....
```

## EXERCÍCIO 1

```
...
printf("\nMATRIZ INICIAL:\n");
for(i=0; i<linhas; i++){
    for(k=0; k<colunas; k++)
        printf("%4i ", m[i][k]);
    printf("\n");
}
//Encontrar o maior
int maior = m[0][0];
for(i=0; i<linhas; i++)
    for(k=0; k<colunas; k++)
        if(m[i][k] > maior) maior = m[i][k];
printf("\nMaior = %i\n", maior);

//Mostrar a matriz resultante
...
```

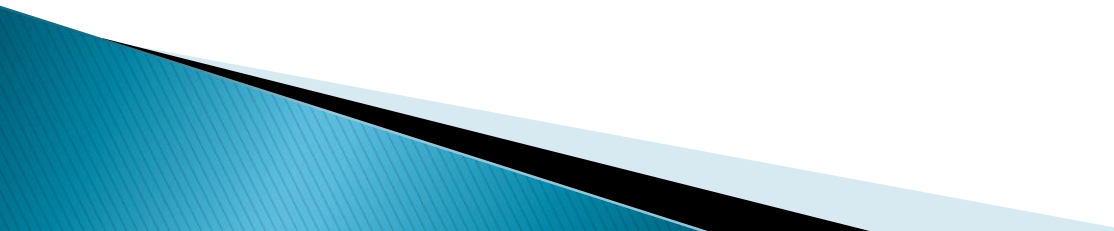
## EXERCÍCIO 1

```
...
//Encontrar o maior
int maior = m[0][0];
for(i=0; i<linhas; i++)
    for(k=0; k<colunas; k++)
        if(m[i][k] > maior) maior = m[i][k];
printf("\nMaior = %i\n", maior);

int m2[linhas][colunas];
printf("\nMATRIZ RESULTANTE:\n");
for(i=0; i<linhas; i++){
    for(k=0; k<colunas; k++){
        m2[i][k] = m[i][k]*maior;
        printf("%4i ", m2[i][k]);
    }
    printf("\n");
}
}
```

# EXERCÍCIO 2

Faça um programa que carregue uma matriz 6 x 4 com números inteiros. Calcule e mostre quantos elementos dessa matriz são maiores que 30 e, em seguida, monte uma segunda matriz com os elementos diferentes de 30. No lugar do número 30 da segunda matriz coloque o número zero.





## EXERCÍCIO 2

```
#define linhas 3  
#define colunas 2
```

```
int main(void){  
    int i,k, nMaior30=0, m[linhas][colunas] = {0};  
    int m2[linhas][colunas];  
    for( i=0; i<linhas; i++)  
        for( k=0; k<colunas; k++){  
            printf("\nDigite um numero: ");    scanf("%i", &m[i][k]);  
        }  
}
```

## EXERCÍCIO 2

```
#define linhas 3
```

```
#define colunas 2
```

```
int main(void){
```

```
    int i,k, nMaior30=0, m[linhas][colunas] = {0};
```

```
    int m2[linhas][colunas];
```

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

```
        for( k=0; k<colunas; k++){
```

```
            printf("\nDigite um numero: ");    scanf("%i", &m[i][k]);
```

```
        }
```

```
    printf("\nMATRIZ INICIAL:\n");
```

```
    for( i=0; i<linhas; i++){
```

```
        for( k=0; k<colunas; k++){
```

```
            printf("%4i ", m[i][k]);
```

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

```
    }
```

```
    //Localizar elementos maiores ou iguais a 30
```

## EXERCÍCIO 2

```
...
printf("\nMATRIZ INICIAL:\n");
for( i=0; i<linhas; i++){
    for( k=0; k<colunas; k++)
        printf("%4i ", m[i][k]);
    printf("\n");
}
//Localizar elementos maiores ou iguais a 30
for( i=0; i<linhas; i++)
    for( k=0; k<colunas; k++){
        if (m[i][k] > 30)
            nMaior30 += 1;
        if (m[i][k] == 30)
            m2[i][k] = 0;
        else
            m2[i][k] = m[i][k];
    }
printf("\nElementos maiores do que 30 = %i\n", nMaior30);
...
```

## EXERCÍCIO 2

```
...
//Localizar elementos maiores ou iguais a 30
for( i=0; i<linhas; i++)
    for( k=0; k<colunas; k++){
        if (m[i][k] > 30)
            nMaior30 += 1;
        if (m[i][k] == 30)
            m2[i][k] = 0;
        else
            m2[i][k] = m[i][k];
    }
printf("\nElementos maiores do que 30 = %i\n", nMaior30);
printf("\nMATRIZ RESULTANTE:\n");
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++){
        printf("%4i ", m2[i][k]);
    }
    printf("\n");
}
}
```

# EXERCÍCIO 3

Faça um programa que carregue uma matriz  $10 \times 20$  com números inteiros e some cada uma das linhas, armazenando o resultado das somas em um vetor. A seguir, multiplique cada elemento da matriz pela soma da linha e mostre a matriz resultante.

## EXERCÍCIO 3

```
#define linhas 2  
#define colunas 4
```

```
int main(void){  
    int i,k, m[linhas][colunas], m2[linhas][colunas];  
    for(i=0; i<linhas; i++)  
        for( k=0; k<colunas; k++){  
            printf("\nDigite um numero: ");  scanf("%i", &m[i][k]);  
        }  
    ...  
}
```

## EXERCÍCIO 3

```
int main(void){
    int i,k, m[linhas][colunas], m2[linhas][colunas];
    for(i=0; i<linhas; i++)
        for( k=0; k<colunas; k++){
            printf("\nDigite um numero: ");   scanf("%i", &m[i][k]);
        }
    printf("\nMATRIZ INICIAL:\n");
    for(i=0; i<linhas; i++){
        for( k=0; k<colunas; k++)
            printf("%4i ", m[i][k]);
        printf("\n");
    }
    int somaLinha[linhas];
    for(i=0; i<linhas; i++){
        somaLinha[i] = 0;
        for( k=0; k<colunas; k++)
            somaLinha[i] = somaLinha[i] + m[i][k];
    }
    ...
}
```

## EXERCÍCIO 3

```
...
int somaLinha[linhas];
for(i=0; i<linhas; i++){
    somaLinha[i] = 0;
    for( k=0; k<colunas; k++)
        somaLinha[i] = somaLinha[i] + m[i][k];
}
for(i=0; i<linhas; i++){
    for( k=0; k<colunas; k++)
        printf("%i ",m[i][k]);
    printf(" --> Soma da linha %d = %d\n", i, somaLinha[i]);
}
...
```



## EXERCÍCIO 3

```
...
for(i=0; i<linhas; i++){
    for( k=0; k<colunas; k++)
        printf("%i ",m[i][k]);
    printf(" --> Soma da linha %d = %d\n", i, somaLinha[i]);
}
for(i=0; i<linhas; i++)
    for(k=0; k<colunas; k++)
        m2[i][k] = m[i][k] * somaLinha[i];
printf("\nMATRIZ RESULTANTE:\n");
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++)
        printf("%4i ", m2[i][k]);
    printf("\n");
}
}
```

# Exercício 4

Criar um algoritmo que leia os elementos de uma **matriz** inteira 5 x 5. Em seguida substitua por um valor, fornecido pelo usuário, somente os elementos acima da diagonal principal.

# Exercício 4

```
int valor, m[linhas][colunas];
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++){
        printf("m[%i][%d] = ", i,k);
        scanf("%i", &m[i][k]);
    }
    printf("\n");
}
printf("\n\nDigite um valor: ");    scanf("%i", &valor);

//Substituir o valores acima da diagonal principal
```

# Exercício 4

```
int valor, m[linhas][colunas];
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++){
        printf("m[%i][%d] = ", i,k);
        scanf("%i", &m[i][k]);
    }
    printf("\n");
}
//Substituir o valores acima da diagonal principal
printf("\n\nDigite um valor: ");  scanf("%i", &valor);
for(int i=0; i<linhas-1; i++)
    for(int k=i+1; k<colunas; k++)
        m[i][k] = valor;

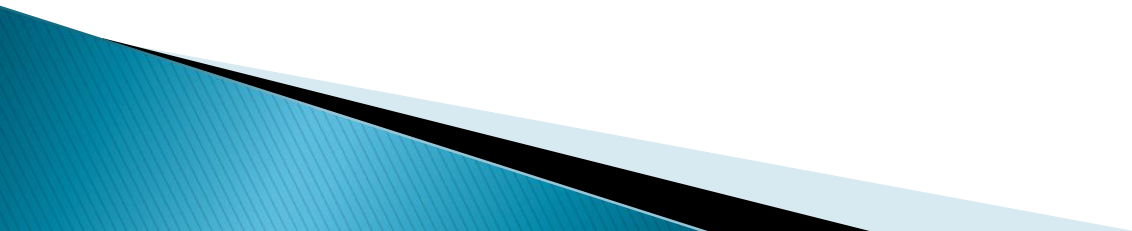
printf("\nMATRIZ RESULTANTE:\n");
...
```

# Exercício 4

```
...  
//Substituir o valores acima da diagonal principal  
printf("\n\nDigite um valor: "); scanf("%i", &valor);  
for(int i=0; i<linhas-1; i++)  
    for(int k=i+1; k<colunas; k++)  
        m[i][k] = valor;  
  
printf("\nMATRIZ RESULTANTE:\n");  
for(int i=0; i<linhas; i++){  
    for(int k=0; k<colunas; k++)  
        printf("%4i ", m[i][k]);  
    printf("\n");  
}
```

# Exercício 5

Entrar com valores para uma matriz  $A$   $3 \times 4$ .  
Gerar e imprimir uma matriz  $B$  que é o triplo dos elementos ímpares e o dobro dos elementos pares da matriz  $A$ .



# Exercício 5

```
int a[linhas][colunas], b[linhas][colunas];
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++){
        printf("m[%i][%d] = ", i,k);        scanf("%i", &a[i][k]);
    }
    printf("\n");
}
printf("\nMATRIZ INICIAL:\n");
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++)
        printf("%4i ", a[i][k]);
    printf("\n");
}
//Gerar a matriz resultante
...
```

# Exercício 5

```
...  
//Gerar a matriz resultante  
for(int i=0; i<linhas; i++)  
    for(int k=0; k<colunas; k++)  
        if(a[i][k]%2==1)  
            b[i][k] = a[i][k]*3;  
        else  
            b[i][k] = a[i][k]*2;  
  
printf("\nMATRIZ RESULTANTE:\n");  
for(int i=0; i<linhas; i++){  
    for(int k=0; k<colunas; k++)  
        printf("%4i ", b[i][k]);  
    printf("\n");  
}
```



# Exercício 6

Criar um algoritmo que armazene dados inteiros em uma matriz de ordem 4.

Em seguida imprima a raiz quadrada da soma dos quadrados dos números localizados abaixo da diagonal secundária.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

# Exercício 6

```
int soma=0, m[linhas][colunas];
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++){
        printf("m[%i][%d] = ", i,k);          scanf("%i", &m[i][k]);
    }
    printf("\n");
}
printf("\nMATRIZ INICIAL:\n");
for(int i=0; i<linhas; i++){
    for(int k=0; k<colunas; k++)
        printf("%4i ", m[i][k]);
    printf("\n");
}
//Localizar elementos abaixo da diagonal secundaria
```

...

# Exercício 6

```
...
printf("\nElementos abaixo da diagonal secundaria:");
for(int k=1; k<colunas; k++){
    for(int i=linhas-1; i>(linhas - k-1); i--){
        soma += m[i][k]*m[i][k]; //Soma dos Quadrados
        printf("\n%i", m[i][k]);
    }
}
//Mostrar resultado
...
```

# Exercício 6

```
...
printf("\nElementos abaixo da diagonal secundaria:");
for(int k=1; k<colunas; k++){
    for(int i=linhas-1; i>(linhas - k-1); i--){
        soma += m[i][k]*m[i][k];
        printf("\ni", m[i][k]);
    }
}
float r = sqrt(soma);
printf("\n\nResultado = %5.2f\n", r);
```

Resultado = 31.72

# Exercício 7

Faça uma função que receba como parâmetro um inteiro no intervalo fechado de 1 a 9 e mostre a seguinte tabela de multiplicação:

Exemplo,  $N = 5$ :

[1,	0,	0,	0,	0]
[2,	4,	0,	0,	0]
[3,	6,	9,	0,	0]
[4,	8,	12,	16,	0]
[5,	10,	15,	20,	25]

# Exercício 7

Faça uma função que receba como parâmetro um inteiro no intervalo fechado de 1 a 9 e mostre a seguinte tabela de multiplicação:

Exemplo,  $N = 5$ :

```
[1, 0, 0, 0, 0]
[2, 4, 0, 0, 0]
[3, 6, 9, 0, 0]
[4, 8, 12, 16, 0]
[5, 10, 15, 20, 25]
```

```
int main(void){
    int num;
    printf("Digite um numero no intervalo de 1 a 9: ");
    scanf("%i", &num);
    while (num < 1 || num > 9){
        printf("Digite um numero no intervalo de 1 a 9: ");
        scanf("%i", &num);
    }
    printf("\n");
    multiplic(num); //chamada da função
}
```

# Exercício 7

```
void multiplic(int n){
    int i,k, m[max][max]={0};
    for(i=0; i<n; i++)
        for( k=0; k<i+1; k++)
            m[i][k] = (i+1)*(k+1);
    for(i=0; i<n; i++){
        for( k=0; k<n; k++)
            printf("%4i", m[i][k]);
        printf("\n");
    }
}
```

```
int main(void){
    int num;
    printf("Digite um numero no intervalo de 1 a 9: ");
    scanf("%i", &num);
    while (num < 1 || num > 9){
        printf("Digite um numero no intervalo de 1 a 9: ");
        scanf("%i", &num);
    }
    printf("\n");
    multiplic(num); //chamada da função
}
```

# EXERCÍCIO 8

Faça um programa que carregue uma matriz 10 x 3 com as notas de dez alunos em três provas.

Mostre um relatório com o número do aluno (número da linha) e a prova em que cada aluno obteve menor nota.

Ao final do relatório, mostre quantos alunos tiveram menor nota na prova 1, quantos alunos tiveram menor nota na prova 2 e quantos alunos tiveram menor nota na prova 3.

