

INSTITUTO FEDERAL DE  
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
RIO GRANDE DO NORTE

# AULA 11

## MATRIZES

Disciplina: Algoritmos

Professora: Alba Lopes

[alba.lopes@ifrn.edu.br](mailto:alba.lopes@ifrn.edu.br)

<http://docente.ifrn.edu.br/albalopes>

# MATRIZES

- O que é uma matriz?
  - Uma estrutura de dados que contém várias variáveis do mesmo tipo
- Qual a diferença de **vetores** para **matrizes**?
  - Vetores são, na verdade, matrizes de uma única dimensão:

## *Vetores*

1	3	4	6
---	---	---	---

a	maria	jota
---	-------	------

## *Matrizes*

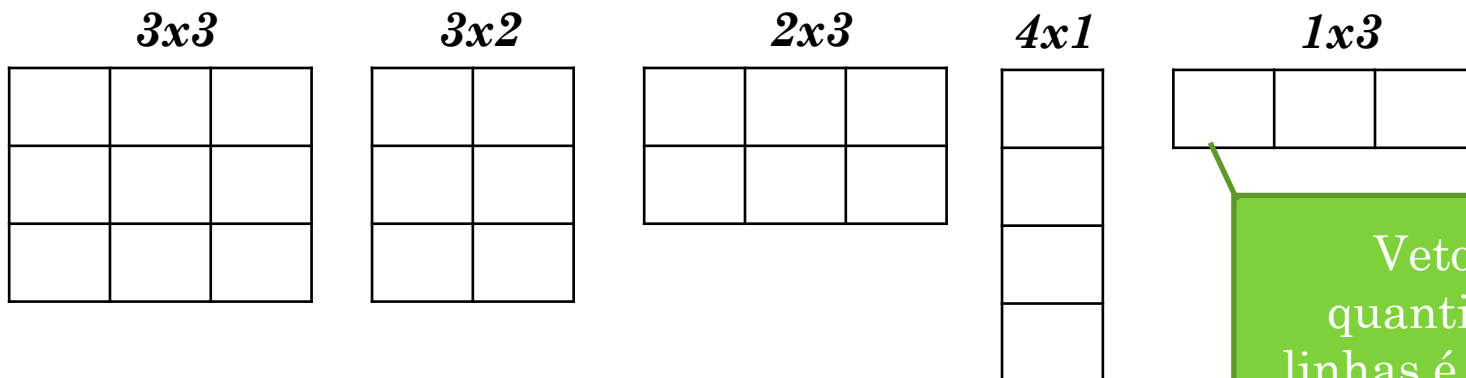
1	3
40	4
6	12

M	J	K
G	A	C
L	Z	H

1.1	7.5	9.2	8.8
9.0	1.3	5.5	7.9

# MATRIZES

- As matrizes são, comumente referenciadas através de suas dimensões (quantidade de linhas e colunas)
- A notação comum é:  $M \times N$ , onde
  - $M$  é a dimensão vertical (**quantidade de linhas**)
  - $N$  é dimensão horizontal (**quantidade de colunas**)
- Exemplo:

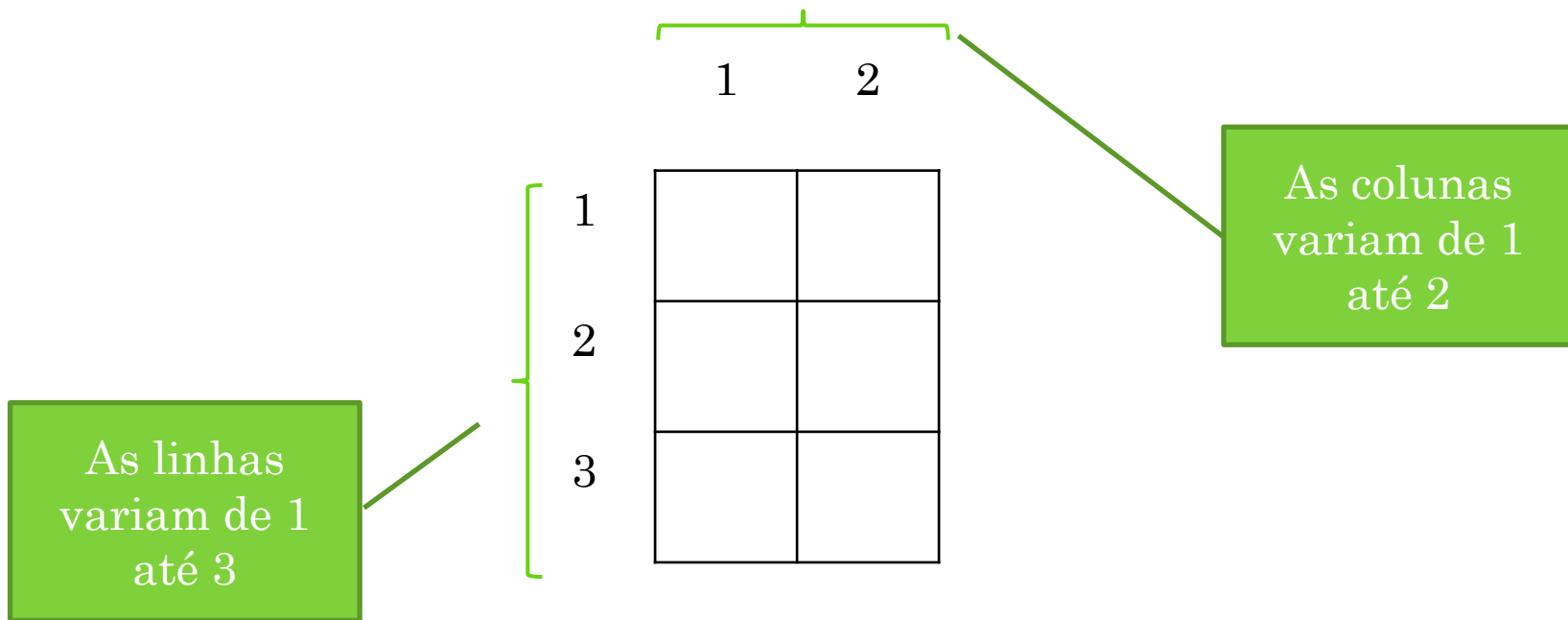


Vetores: a quantidade de linhas é sempre 1!

# MATRIZES

## ○ Notação

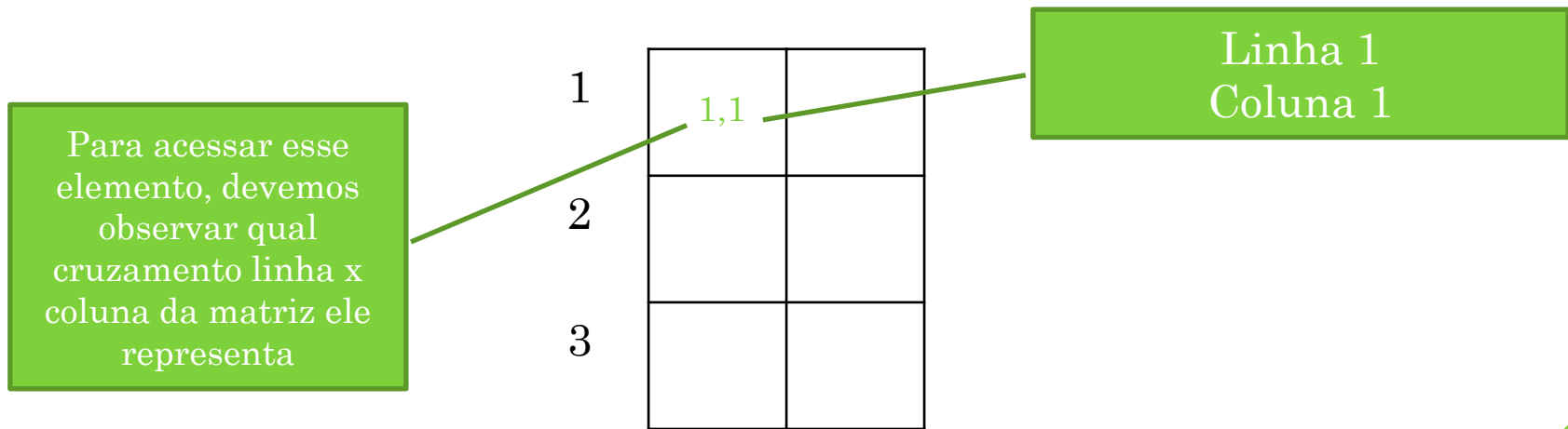
- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)



# MATRIZES

## ○ Notação

- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)



# MATRIZES

## ○ Notação

- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)

	1	2
1	1,1	1,2
2		
3		

Linha 1  
Coluna 2



# MATRIZES

## ○ Notação

- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)

	1	2
1	1,1	1,2
2	2,1	
3		

Linha 2  
Coluna 1



# MATRIZES

## ○ Notação

- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)

	1	2
1	1,1	1,2
2	2,1	2,2
3		

A diagram illustrating a 3x2 matrix. The matrix is represented as a grid with three rows and two columns. The rows are labeled 1, 2, and 3 on the left. The columns are labeled 1 and 2 at the top. The elements in the first two rows are labeled as follows: row 1 contains '1,1' and '1,2'; row 2 contains '2,1' and '2,2'. A green box with a pointer indicates the element at row 2, column 2, labeled 'Linha 2 Coluna 2'. The bottom row (row 3) is empty.

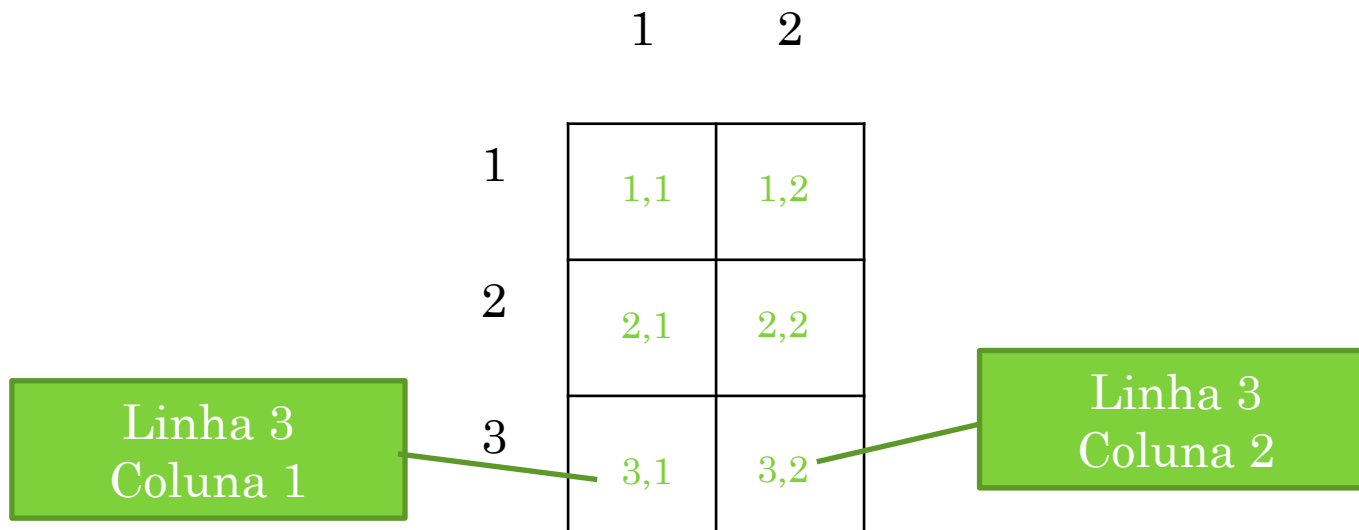




# MATRIZES

## ○ Notação

- Como referenciar um elemento específico da matriz?
- Exemplo: Matriz 3x2 (*três linhas e duas colunas*)



# SINTAXE NO VISUALG

## ○ Declaração:

```
<nome_variavel>: vetor [li..lf, ci..cf] de <tipo>
```

## ○ Onde:

- li e lf representam, respectivamente o índice inicial e final das **linhas** e
- ci e cf representam, respectivamente o índice inicial e final das **colunas**



# SINTAXE NO VISUALG

## ○ Exemplo:

- Para declarar uma matriz 3x2 de inteiro

```
algoritmo "exemplo_matriz"  
var  
    exMatriz: vetor [1..3, 1..2] de inteiro  
inicio  
    ...
```

Linhas: o índice das linhas varia de 1 até 3

Colunas: o índice das colunas varia de 1 até 2



# SINTAXE NO VISUALG

- **Preenchendo e acessando uma matriz**
  - As posições das matrizes são identificados pelos índices das linhas e colunas
- **Atribuição**

```
<nome_variavel> [<linha>, <coluna>] ← <valor>  
<nome_variavel> [<linha>, <coluna>] := <valor>  
leia(<nome_variavel> [<linha>, <coluna>])
```



# SINTAXE NO VISUALG

## o Exemplo:

```
algoritmo "exemplo_matriz"
```

```
var
```

```
exMatriz: vetor [1..3, 1..2] de inteiro
```

```
inicio
```

```
exMatriz[1,1] ← 10
```

```
leia(exMatriz[1,2])
```

```
exMatriz[3,1] := 4
```

```
fimalgoritmo
```

	1	2
1		
2		
3		

*exMatriz*



# SINTAXE NO VISUALG

## o Exemplo:

```
algoritmo "exemplo_matriz"  
var  
    exMatriz: vetor [1..3, 1..2] de inteiro  
inicio  
    exMatriz[1,1] ← 10  
    leia(exMatriz[1,2])  
    exMatriz[3,1] := 4  
finalgoritmo
```

	1	2
1	10	
2		
3		

*exMatriz*



# SINTAXE NO VISUALG

## o Exemplo:

```
algoritmo "exemplo_matriz"  
var  
    exMatriz: vetor [1..3, 1..2] de inteiro  
inicio  
    exMatriz[1,1] ← 10  
    leia(exMatriz[1,2])  
    exMatriz[3,1] := 4  
finalgoritmo
```

	1	2
1	10	7
2		
3		

*exMatriz*



# SINTAXE NO VISUALG

## o Exemplo:

```
algoritmo "exemplo_matriz"  
var  
    exMatriz: vetor [1..3, 1..2] de inteiro  
inicio  
    exMatriz[1,1] ← 10  
    leia(exMatriz[1,2])  
    exMatriz[3,1] := 4  
finalgoritmo
```

	1	2
1	10	7
2		
3	4	

*exMatriz*





# SINTAXE NO VISUALG

## ○ Preenchendo uma matriz

- Se quisermos atribuir valores a todas as posições da matriz, podemos fazer:

```
algoritmo "preencher"  
var  
    numeros: vetor[1..3, 1..2] de inteiro  
    i, j: inteiro  
inicio  
    escreva("Digite um valor para a posição [1,1]")  
    leia(numeros[1,1])  
    escreva("Digite um valor para a posição [1,2]")  
    leia(numeros[1,2])  
    escreva("Digite um valor para a posição [2,1]")  
    leia(numeros[2,1])  
    escreva("Digite um valor para a posição [2,2]")  
    leia(numeros[2,2])  
    escreva("Digite um valor para a posição [3,1]")  
    leia(numeros[3,1])  
    escreva("Digite um valor para a posição [3,2]")  
    leia(numeros[3,2])  
fimalgoritmo
```



# SINTAXE NO VISUALG

## ○ **Preenchendo uma matriz**

- Entretanto, à medida que a quantidade de elementos da matriz aumenta, fica complicado fazermos manualmente para todas as posições.
- O melhor caminho é utilizar laços de repetição!



# SINTAXE NO VISUALG

## ○ Preenchendo uma matriz

- Podemos criar um laço de repetição para variar pelas linhas, por exemplo:

```
algoritmo "preencher"  
var  
    numeros: vetor[1..3, 1..2] de inteiro  
    i, j: inteiro  
inicio  
    para i de 1 ate 3 faça  
        escreva("Digite um valor para a posição [", i, ",1")  
        leia(numeros[i,1])  
        escreva("Digite um valor para a posição [", i, ",2")  
        leia(numeros[i,2])  
    fimpara  
fimalgoritmo
```



# SINTAXE NO VISUALG

## ○ Preenchendo uma matriz

- E podemos ainda incluir um laço de repetição para variar pelas colunas também, por exemplo:

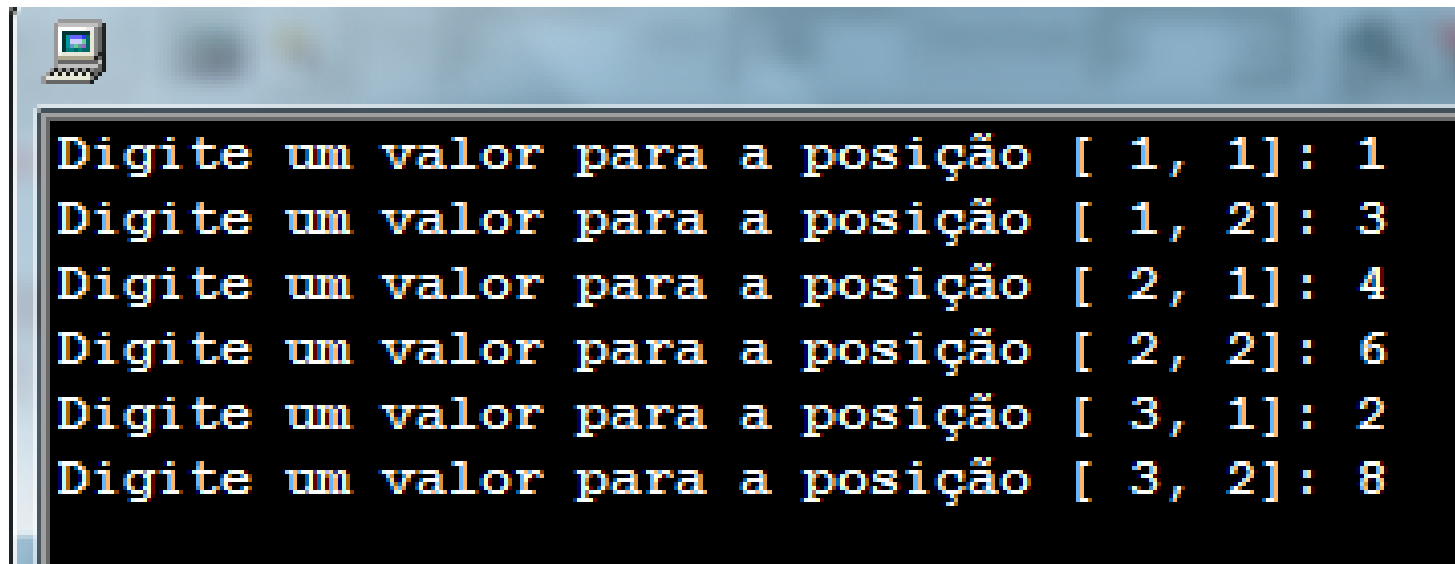
```
algoritmo "preencher"  
var  
    numeros: vetor[1..3, 1..2] de inteiro  
    i, j: inteiro  
inicio  
    para i de 1 ate 3 faca  
        para j de 1 ate 2 faca  
            escreva("Digite um valor para a posição [", i, ", ", j, "]: ")  
            leia(numeros[i,j])  
        fimpara  
    fimpara  
fimalgoritmo
```



# SINTAXE NO VISUALG

## ○ Preenchendo uma matriz

- Saída:



```
Digite um valor para a posição [ 1, 1]: 1
Digite um valor para a posição [ 1, 2]: 3
Digite um valor para a posição [ 2, 1]: 4
Digite um valor para a posição [ 2, 2]: 6
Digite um valor para a posição [ 3, 1]: 2
Digite um valor para a posição [ 3, 2]: 8
```



# SINTAXE NO VISUALG

## ○ Exibindo o conteúdo de uma matriz:

```
...  
    escreva("O valor que está na posição [1,1] é: ", numeros[1,1])  
    escreva("O valor que está na posição [1,2] é: ", numeros[1,2])  
    escreva("O valor que está na posição [2,1] é: ", numeros[2,1])  
    escreva("O valor que está na posição [2,2] é: ", numeros[2,2])  
    escreva("O valor que está na posição [3,1] é: ", numeros[3,1])  
    escreva("O valor que está na posição [3,2] é: ", numeros[3,2])  
fimalgoritmo
```



# SINTAXE NO VISUALG

## ○ Exibindo o conteúdo de uma matriz

- Ou podemos utilizar um laço de repetição para facilitar a exibição dos valores de uma matriz
- Criando um laço para percorrer as linhas:

## ○ Exemplo:

```
para i de 1 ate 3 faca
    escreval("O valor que está na posição [", i, ", 1] é: ", numeros[i, 1])
    escreval("O valor que está na posição [", i, ", 2] é: ", numeros[i, 2])
fimpara
```



# SINTAXE NO VISUALG

## ○ Exibindo o conteúdo de uma matriz

- E podemos ainda incluir um laço de repetição para variar pelas colunas também, por exemplo:

```
para i de 1 ate 3 faca
  para j de 1 ate 2 faca
    escreval("O valor que está na posição [", i,",", " j,"] é: ", numeros[i, j])
  fimpara
fimpara
```





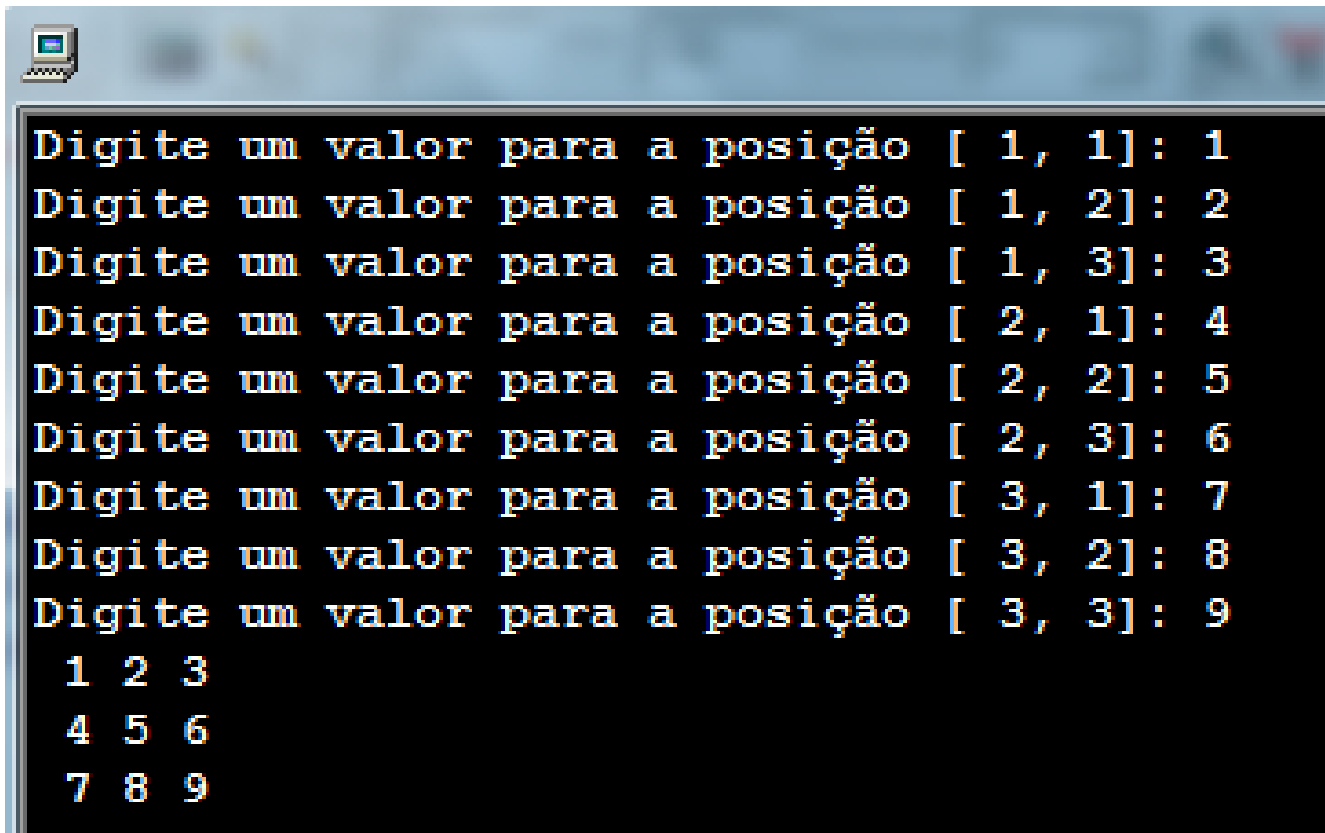
# EXEMPLO 1

- Criar um algoritmo que leia uma matriz 3x3 e exiba a matriz preenchida:

```
algoritmo "exemplo01"  
var  
  numeros: vetor[1..3, 1..3] de inteiro  
  i, j: inteiro  
inicio  
  para i de 1 ate 3 faca  
    para j de 1 ate 3 faca  
      escreva("Digite um valor para a posição [", i, ",", j, "]: ")  
      leia(numeros[i,j])  
    fimpara  
  fimpara  
  
  para i de 1 ate 3 faca  
    para j de 1 ate 3 faca  
      escreva(numeros[i, j])  
    fimpara  
  escreval  
fimpara  
finalgoritmo
```

# EXEMPLO 1

- Saída:



```
Digite um valor para a posição [ 1, 1]: 1
Digite um valor para a posição [ 1, 2]: 2
Digite um valor para a posição [ 1, 3]: 3
Digite um valor para a posição [ 2, 1]: 4
Digite um valor para a posição [ 2, 2]: 5
Digite um valor para a posição [ 2, 3]: 6
Digite um valor para a posição [ 3, 1]: 7
Digite um valor para a posição [ 3, 2]: 8
Digite um valor para a posição [ 3, 3]: 9

 1 2 3
 4 5 6
 7 8 9
```



## EXEMPLO 2

- Criar um algoritmo que leia uma matrizes 3x3. Em seguida, exiba a soma dos elementos de cada uma das linhas. Ex:

1	2	2
3	2	3
4	1	1

Soma Linha 1 = 5

Soma Linha 2 = 8

Soma Linha 3 = 6



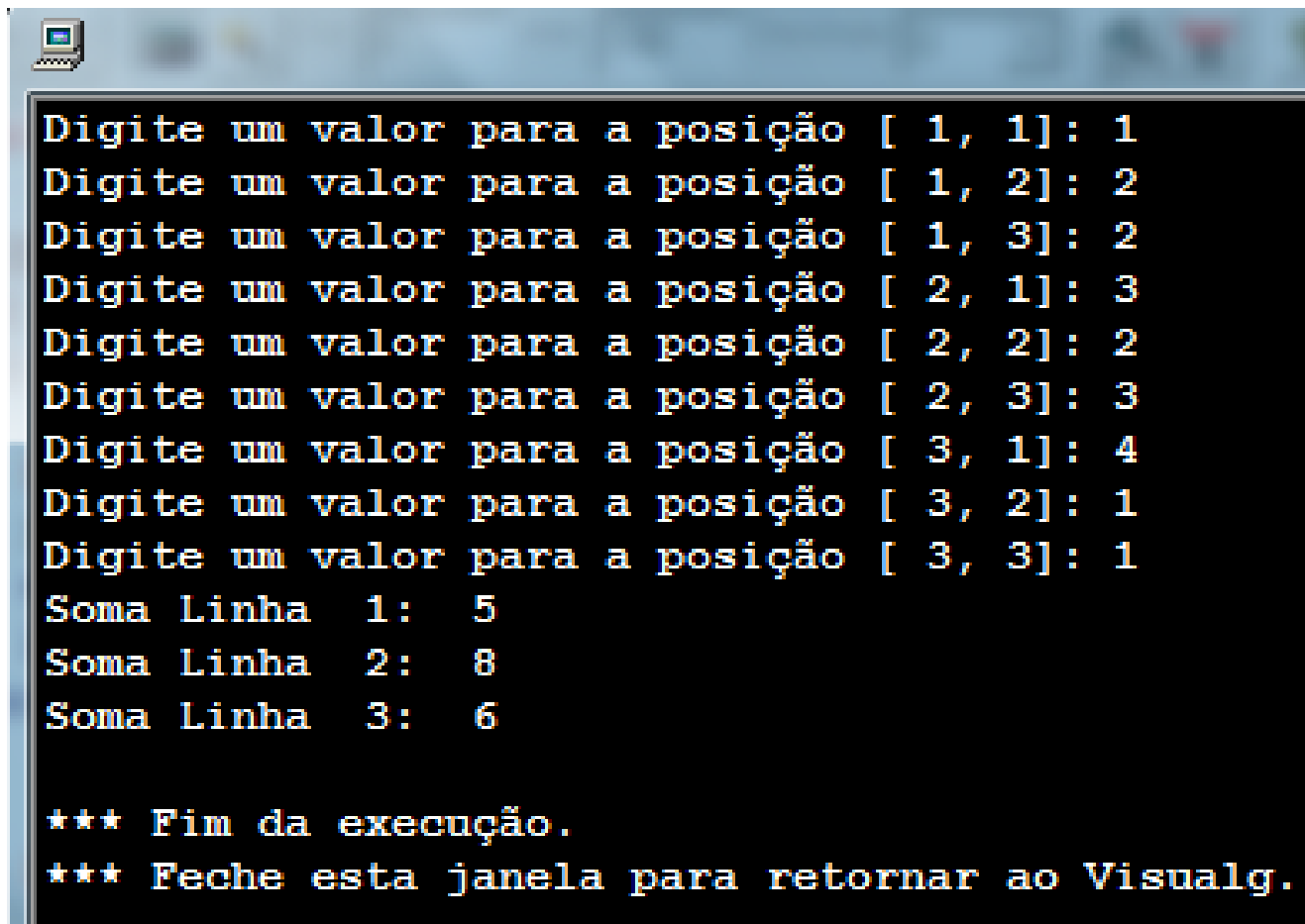
# EXEMPLO 2

## ○ Resolução:

```
algoritmo "exemplo01"  
var  
  numeros: vetor[1..3, 1..3] de inteiro  
  i, j: inteiro  
  soma: inteiro  
inicio  
  para i de 1 ate 3 faca  
    para j de 1 ate 3 faca  
      escreva("Digite um valor para a posição [", i, ",", j, "]: ")  
      leia(numeros[i,j])  
    fimpara  
  fimpara  
  
  para i de 1 ate 3 faca  
    soma <- 0  
    para j de 1 ate 3 faca  
      soma <- soma + numeros[i, j]  
    fimpara  
    escreval ("Soma Linha ", i, ": ", soma)  
  fimpara  
fimalgoritmo
```

## EXEMPLO 2

- Saída:



```
Digite um valor para a posição [ 1, 1]: 1
Digite um valor para a posição [ 1, 2]: 2
Digite um valor para a posição [ 1, 3]: 2
Digite um valor para a posição [ 2, 1]: 3
Digite um valor para a posição [ 2, 2]: 2
Digite um valor para a posição [ 2, 3]: 3
Digite um valor para a posição [ 3, 1]: 4
Digite um valor para a posição [ 3, 2]: 1
Digite um valor para a posição [ 3, 3]: 1
Soma Linha  1:  5
Soma Linha  2:  8
Soma Linha  3:  6

*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.
```



## EXEMPLO 3

- Escreva um algoritmo que leia uma matriz 4x3. Em seguida, receba um novo valor do usuário e verifique se este valor se encontra na matriz. Caso o valor se encontre na matriz, escreva a mensagem “O valor se encontra na matriz”. Caso contrário, escreva a mensagem “O valor NÃO se encontra na matriz”.



# EXEMPLO 3

algoritmo "exemplo03"

var

numeros: vetor[1..4, 1..3] de inteiro

i, j, buscar: inteiro

achou: logico

inicio

para i de 1 ate 4 faca

para j de 1 ate 3 faca

escreva("Digite um valor para a posição [", i, ",", j, "]: ")

leia(numeros[i,j])

fimpara

fimpara

escreva("Digite um valor para ser buscado na matriz: ")

leia(buscar)

achou <- falso

para i de 1 ate 4 faca

para j de 1 ate 3 faca

se (numeros[i,j] = buscar) entao

achou <- verdadeiro

fimse

fimpara

fimpara

se achou=verdadeiro entao

escreva("O número se encontra na matriz.")

senao

escreva("O número NÃO se encontra na matriz.")

fimse

fimalgoritmo



# EXEMPLO 3

- Saída:

```
Digite um valor para a posição [ 1, 1]: 1
Digite um valor para a posição [ 1, 2]: 2
Digite um valor para a posição [ 1, 3]: 3
Digite um valor para a posição [ 2, 1]: 4
Digite um valor para a posição [ 2, 2]: 5
Digite um valor para a posição [ 2, 3]: 6
Digite um valor para a posição [ 3, 1]: 7
Digite um valor para a posição [ 3, 2]: 8
Digite um valor para a posição [ 3, 3]: 9
Digite um valor para a posição [ 4, 1]: 10
Digite um valor para a posição [ 4, 2]: 11
Digite um valor para a posição [ 4, 3]: 12
Digite um valor para ser buscado na matriz: 7
O número se encontra na matriz.
*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.
```

```
Digite um valor para a posição [ 1, 1]: 1
Digite um valor para a posição [ 1, 2]: 2
Digite um valor para a posição [ 1, 3]: 3
Digite um valor para a posição [ 2, 1]: 4
Digite um valor para a posição [ 2, 2]: 5
Digite um valor para a posição [ 2, 3]: 6
Digite um valor para a posição [ 3, 1]: 7
Digite um valor para a posição [ 3, 2]: 8
Digite um valor para a posição [ 3, 3]: 9
Digite um valor para a posição [ 4, 1]: 10
Digite um valor para a posição [ 4, 2]: 11
Digite um valor para a posição [ 4, 3]: 12
Digite um valor para ser buscado na matriz: 15
O número NÃO se encontra na matriz.
*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.
```





# EXERCÍCIOS

1. Crie um algoritmo que leia uma matriz 5x5. Em seguida, conte quantos números pares existem na matriz.
2. Crie um algoritmo que leia uma matriz 3x3 e calcule a soma dos valores das colunas da matriz. Ex:

1	2	2
3	2	3
4	1	1

Soma Coluna 1 = 8

Soma Coluna 2 = 5

Soma Coluna 3 = 6



# EXERCÍCIOS

3. Crie um algoritmo que calcule a média dos elementos de uma matriz 5x2.
4. Crie um algoritmo informe qual o maior e qual o menor elemento existente em uma matriz 6x3.
5. Crie um algoritmo que leia uma matriz 3x3 e crie uma nova matriz que seja a matriz transposta da primeira (troque as linhas por colunas) Ex:

Matriz

1	2	3
4	5	6
7	8	9

Matriz Transposta

1	4	7
2	5	8
3	6	9



# EXERCÍCIOS

7. Crie um algoritmo que leia duas matrizes 2x5 e crie uma terceira matriz também 2x5 com o valor da soma dos elementos de mesmo índice. Ex:

$$\text{Matriz1} + \text{Matriz2} = \text{Matriz3}$$

1	2
3	2
4	1
5	5
1	2

2	4
5	3
7	7
4	4
1	9

3	6
8	5
11	8
9	9
2	11

