Leitura complementar

Variáveis indexadas e Laços encadeados

- ✓ Vetores e matrizes;
- ✓ Laços encadeados.

Vetores e matrizes

Neste capítulo aprenderemos a utilizar as variáveis indexadas em programação.
Variáveis indexadas são um conjunto de variáveis que apresentam o mesmo nome, são do mesmo tipo, mas são diferentes no valor de seu índice.

As variáveis indexadas podem ter várias dimensões:

- **Vetores**: uma dimensão;
- **Matrizes**: n dimensões.

VETOR

Y

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

MATRIZ

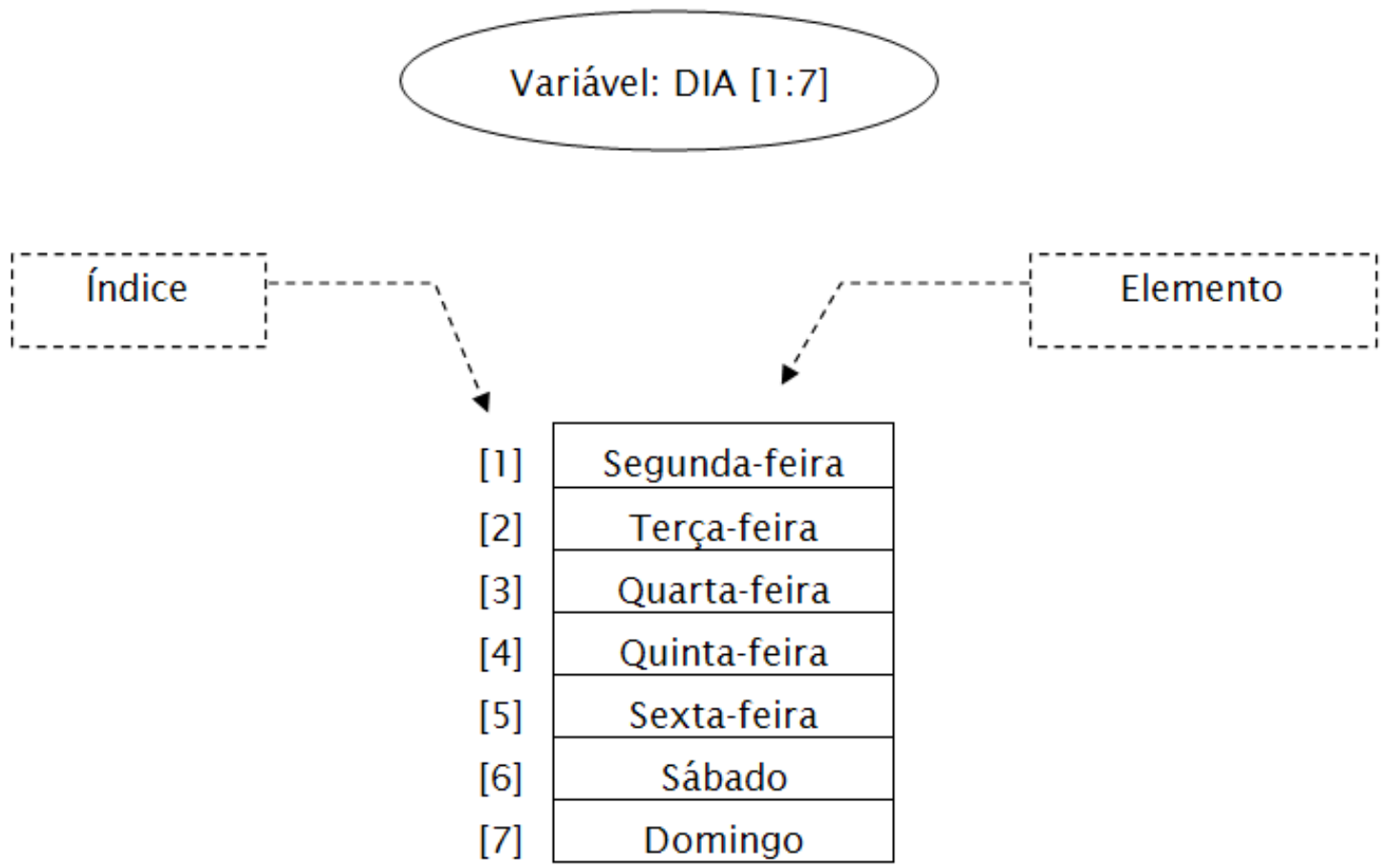
Z

	1	2	3	4
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				

Dados dois números inteiros positivos **m** e **n**, chama-se matriz **m** x **n** a tabela formada por **m.n** números reais, dispostos em **m** linhas (horizontais) e **n** colunas (verticais).

Importante: Cada elemento é indicado por $\mathbf{a}_{ij}$, em que **i** indica a linha e **j**, a coluna, às quais $\mathbf{a}_{ij}$ pertence.

Um elemento de uma tabela pode ser referenciado de duas formas: **Implícita** e **Explícita**. Vejamos a seguinte tabela:



- **Referência implícita**: Usamos o índice para nos referenciar a um certo elemento da tabela. A seguir, temos um exemplo de referência implícita, considerando a tabela de dias da semana mostrada anteriormente:

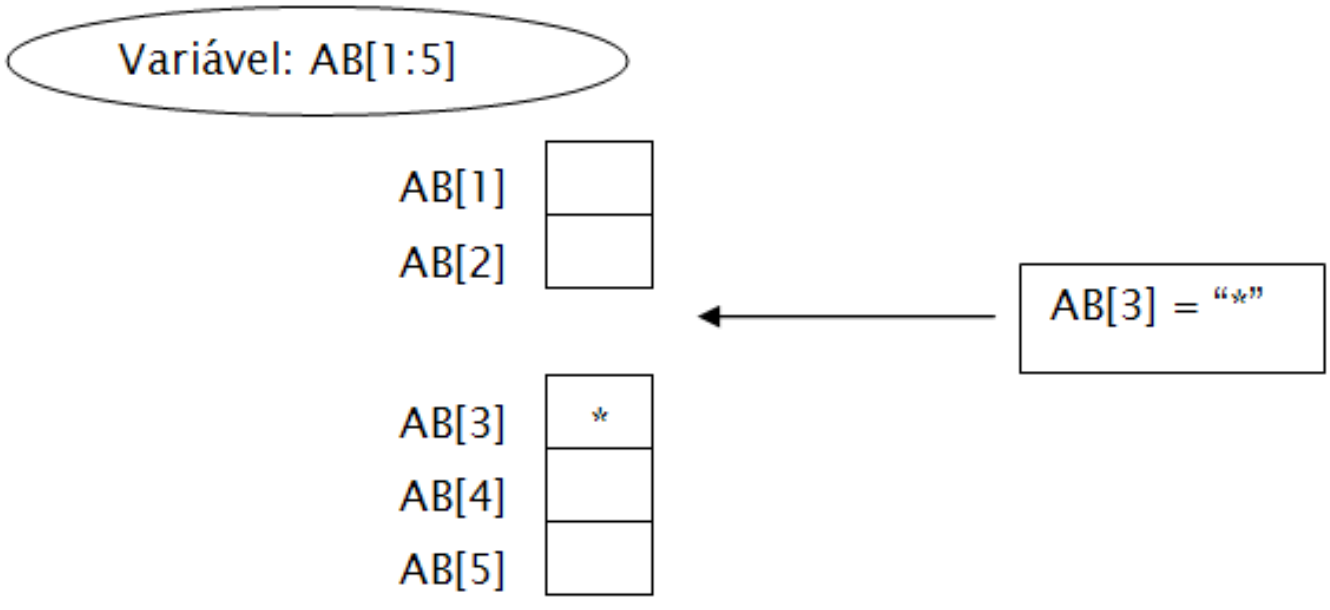
A = 2 ➔ DIA[A] = Terça-feira

- **Referência explícita**: Referenciamo-nos diretamente ao elemento desejado. A seguir, temos um exemplo de referência explícita, considerando nossa tabela de dias da semana:

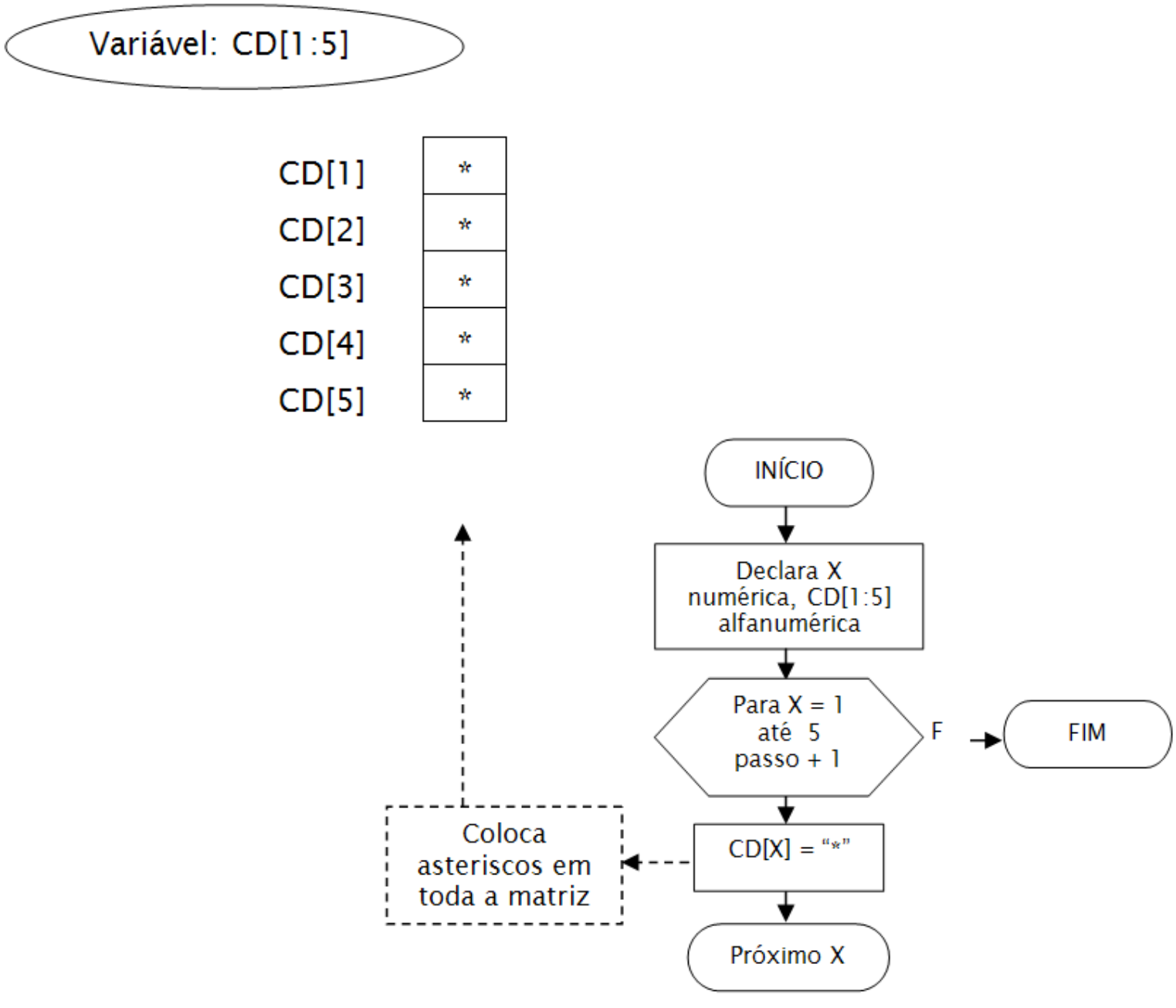
DIA[2] = Terça-feira

Vejamos estes outros exemplos:

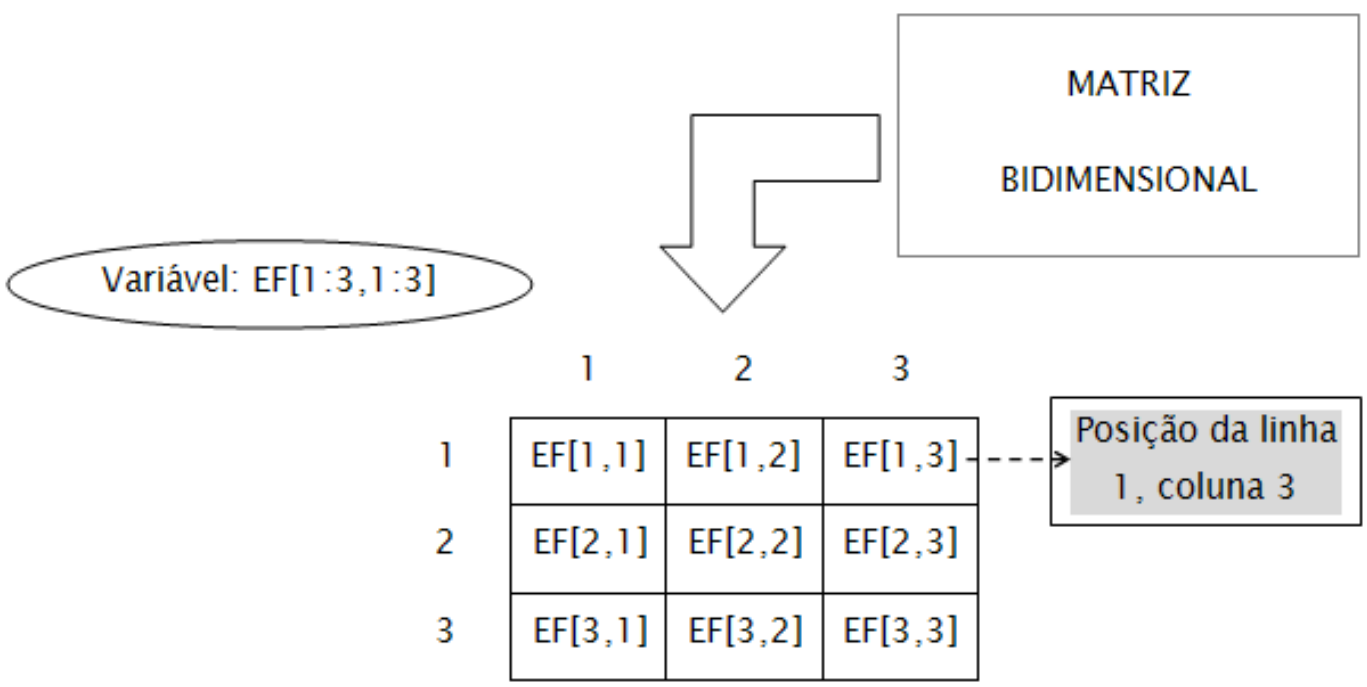
- **Exemplo 1**

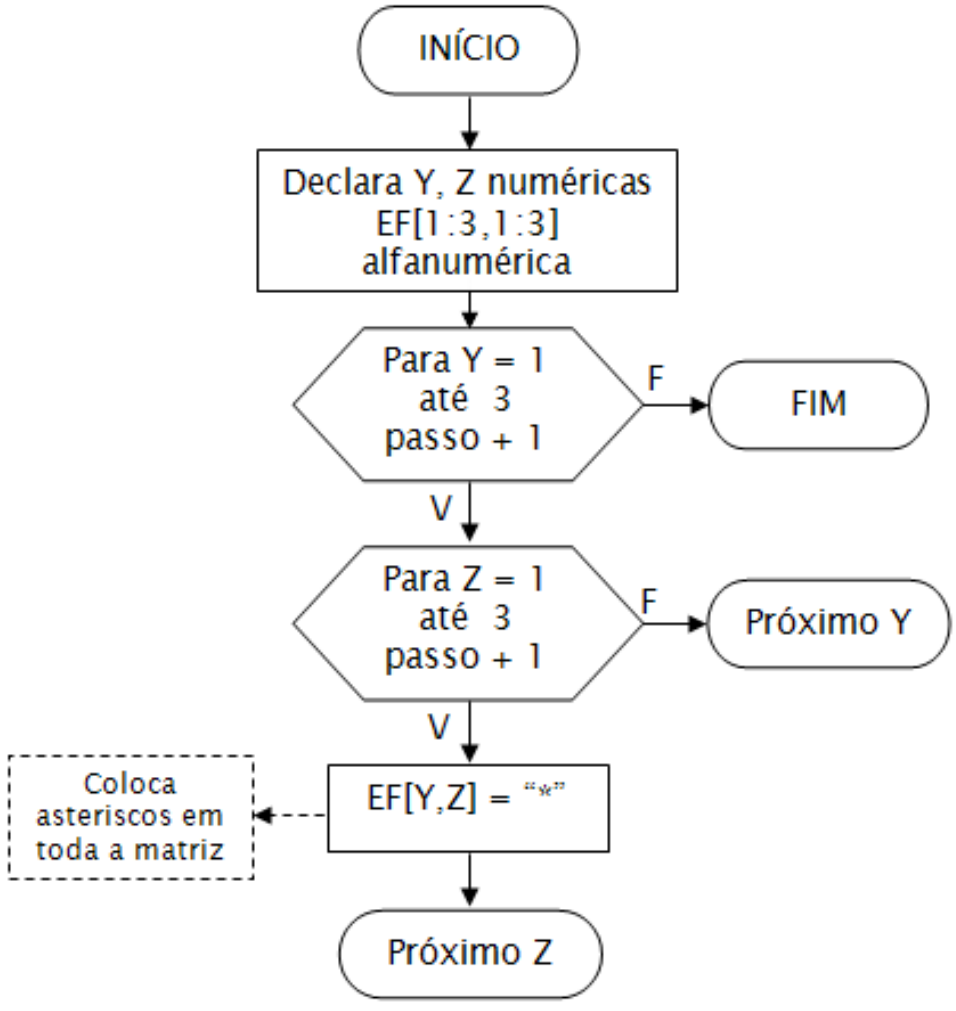


- **Exemplo 2**

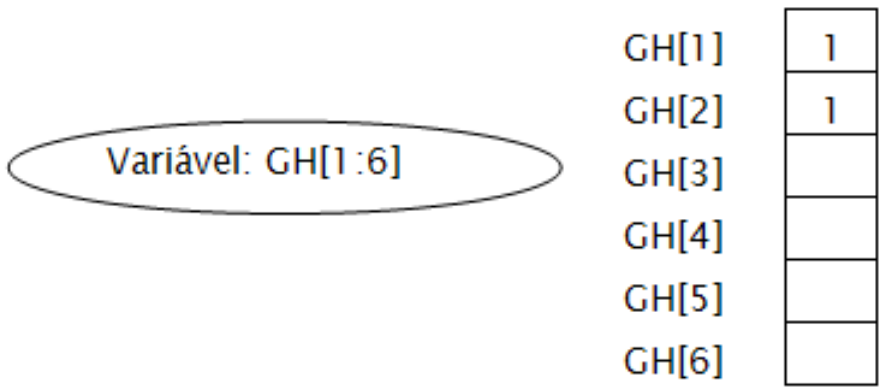


- **Exemplo 3**


MATRIZ
BIDIMENSIONAL
Variável: EF[1:3,1:3]
1
2
3
1
2
3
EF[1,1]
EF[1,2]
EF[1,3]
EF[2,1]
EF[2,2]
EF[2,3]
EF[3,1]
EF[3,2]
EF[3,3]
Posição da linha 1, coluna 3


INÍCIO
Declara Y, Z numéricas
EF[1:3,1:3]
alfanumérica
Para Y = 1
até 3
passo + 1
F
FIM
V
Para Z = 1
até 3
passo + 1
F
Próximo Y
V
EF[Y,Z] = "*"
Coloca asteriscos em toda a matriz
Próximo Z

- **Exemplo 4**



A seguir, temos o algoritmo e o teste de mesa do exemplo 4:

- **Algoritmo**

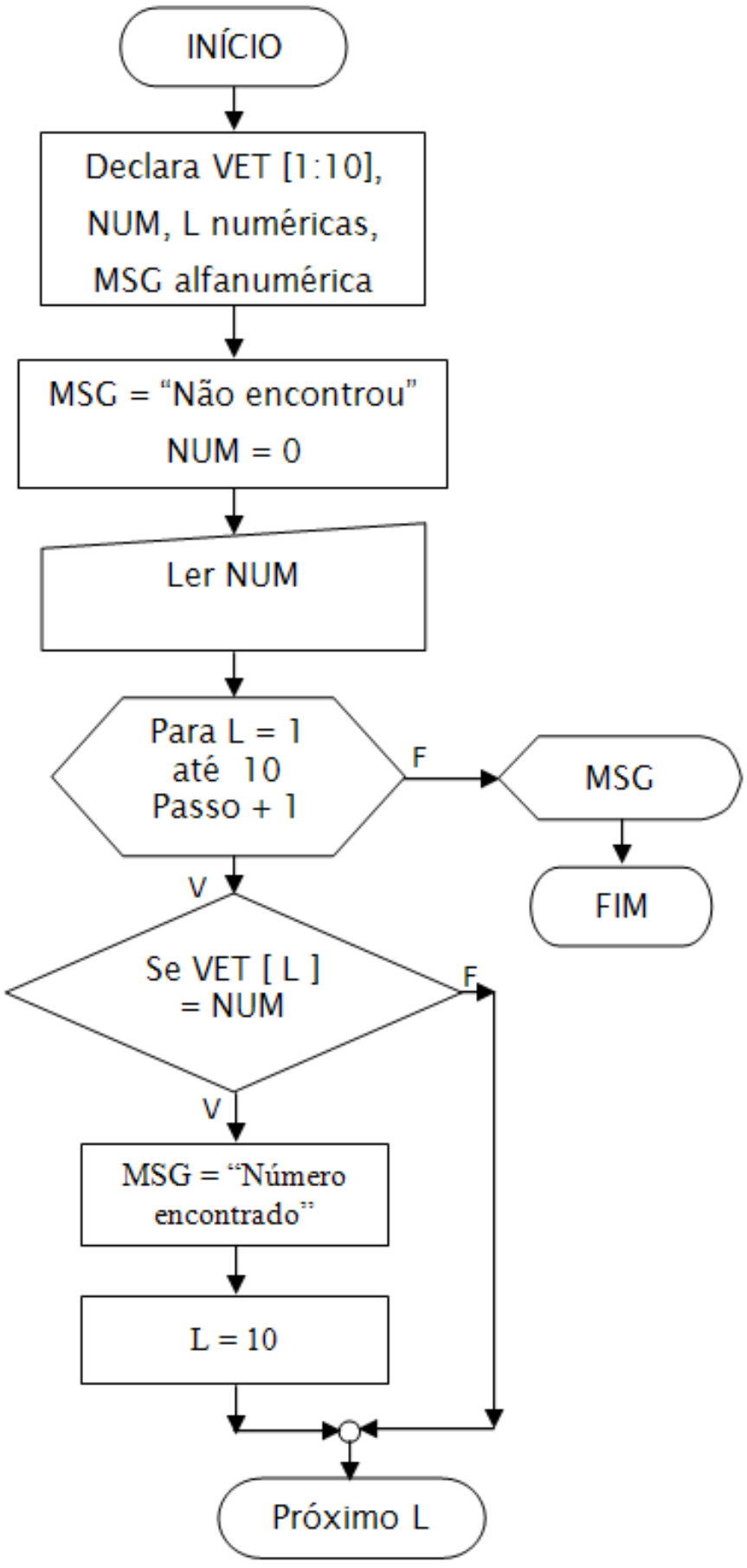
```
INÍCIO
  Declara GH[1:6], J numéricas
  GH[1] = 1
  GH[2] = 1
  Para J = 3 até 6 passo + 1
      GH[J] = GH[J – 1] + GH[J – 2]
  Próximo J
FIM
```

- **Teste de Mesa**

```
GH[ 3 ] = GH[ 3 – 1 ] + GH[ 3 – 2 ]
          GH[ 2 ] + GH[ 1 ]
          2

GH[ 4 ] = GH[ 4 – 1 ] + GH[ 4 – 2 ]
          GH[ 3 ] + GH[ 2 ]
          3

GH[ 5 ] = GH[ 5 – 1 ] + GH[ 5 – 2 ]
          GH[ 4 ] + GH[ 3 ]
          5

GH[ 6 ] = GH[ 6 – 1 ] + GH[ 6 – 2 ]
          GH[ 5 ] + GH[ 4 ]
          8
```

A seguir, temos um algoritmo e o fluxograma correspondente que verifica se um número digitado é encontrado num vetor. Será exibida uma mensagem informando se o número foi encontrado ou não. Como vetor, vamos considerar **VET [1:10]**.

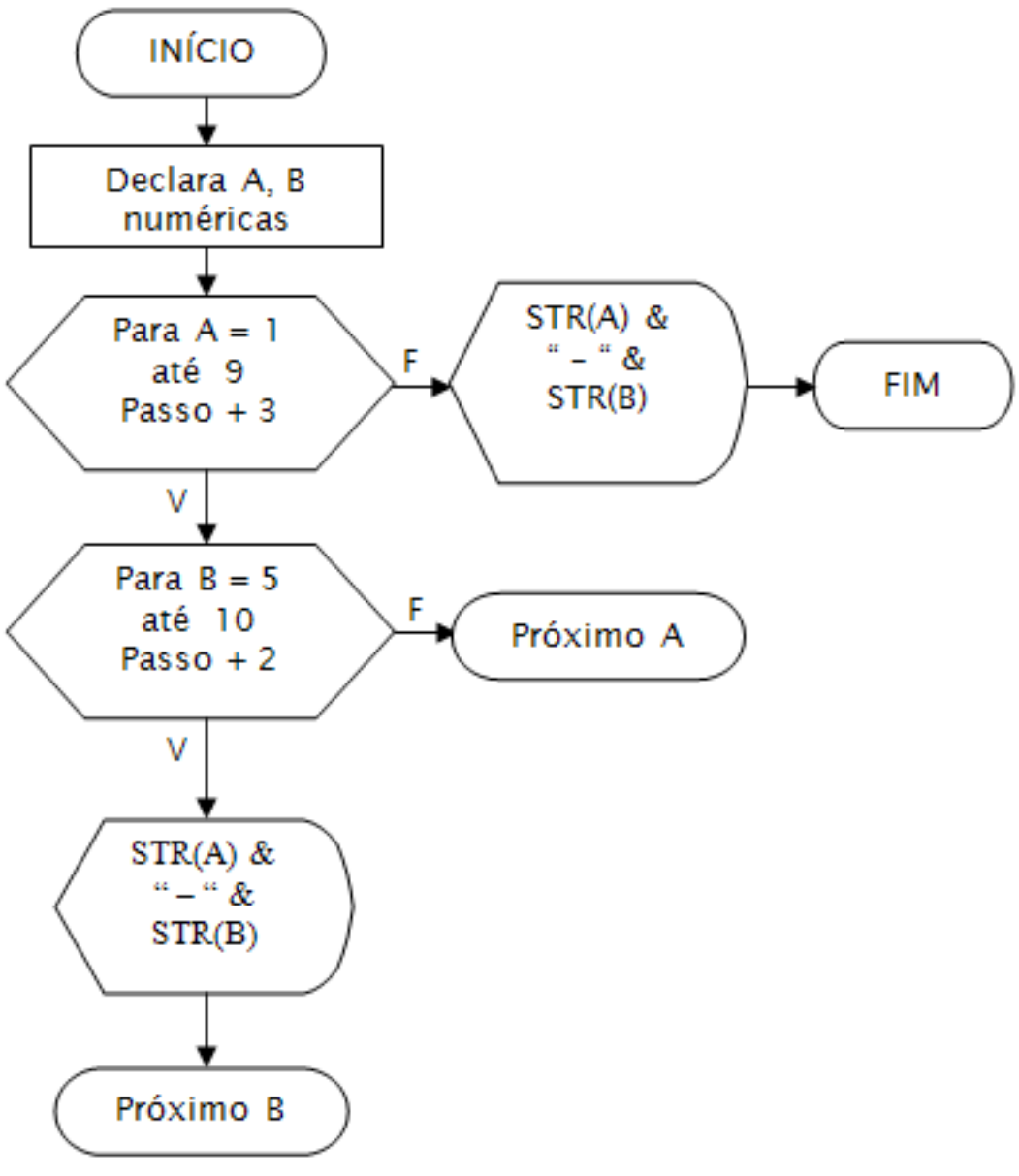
```
INÍCIO
  Declara VET [ 1:10 ], NUM, L numéricas, MSG alfanumérica
  MSG = “Não encontrou”
  NUM = 0
  Ler NUM
  Para L = 1 até 10 passo + 1
     Se VET [ L ] = NUM
           Então MSG =        “Número encontrado”
                 L = 10
     Fim Se
  Próximo L
  Exibir MSG
FIM
```


INÍCIO
Declara VET [1:10], NUM, L numéricas, MSG alfanumérica
MSG = "Não encontrou"
NUM = 0
Ler NUM
Para L = 1 até 10 Passo + 1
F
MSG
FIM
V
Se VET [L] = NUM
F
V
MSG = "Número encontrado"
L = 10
Próximo L

Laços encadeados

Laços ou Loops encadeados são laços executados dentro de outros laços. No caso do comando **PARA**, o primeiro a ser criado é o último a ser fechado. O comando **PRÓXIMO** fecha o laço. O último comando **PARA** aberto é o primeiro a ser fechado com o comando **PRÓXIMO**, ou seja, o último loop criado é o primeiro a ser fechado. Vejamos um exemplo de laço encadeado a seguir:

```
INÍCIO
   Declara A, B numéricas
   Para A = 1 até 9 passo + 3
    Para B = 5 até 10 passo + 2
          Exibir A, " - ", B
    Próximo B
   Próximo A
   Exibir A , " - ", B
FIM
```



Teste de mesa		
A	-	B
1	-	5
1	-	7
1	-	9
4	-	5
4	-	7
4	-	9
7	-	5
7	-	7
7	-	9
10	-	11

Neste outro exemplo, temos um algoritmo que preenche a matriz conforme a figura utilizando apenas um laço (loop):

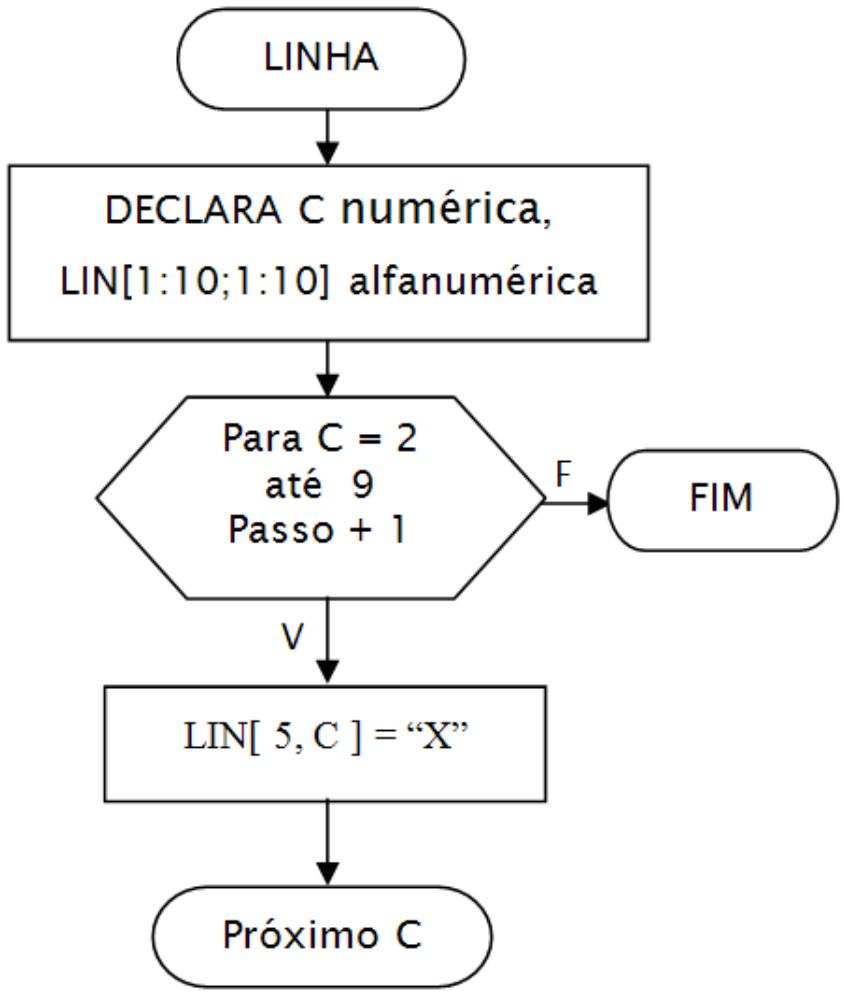
Matriz

LIN

	1	2	3	4	5	6	7	8	9	0
1										
2										
3										
4										
5		X	X	X	X	X	X	X	X	
6										
7										
8										
9										
0										

```
LINHA
Declara C numérica, LIN[ 1:10,1:10 ] alfanumérica
Para C = 2 até 9 passo + 1
    LIN[ 5, C ] = “X”
Próximo C
FIM
```

Note que foram preenchidas as posições, na linha 5, da coluna 2 até a 9.



A seguir, temos um algoritmo que preenche a matriz conforme a figura utilizando laço (loop) encadeado:

Matriz Q

	1	2	3	4	5	6	7	8	9	0
1										
2		X	X	X	X	X	X	X		
3		X	X	X	X	X	X	X		
4		X	X	X	X	X	X	X		
5		X	X	X	X	X	X	X		
6		X	X	X	X	X	X	X		
7		X	X	X	X	X	X	X		
8		X	X	X	X	X	X	X		
9										
0										

```
QUADRADO
Declara L, C numéricas, Q[ 1:10,1:10 ] alfanumérica
Para L = 2 até 8 passo + 1
Para C = 2 até 8 passo + 1
        Q[ L, C ] = “X”
    Próximo C
Próximo L
FIM
```

Vejamos o teste de mesa do último exemplo. Considerando que a variável **L** está sendo usada para a posição da linha e a variável **C** está sendo usada para a posição da coluna, veja, na tabela a seguir, que foi preenchida toda a matriz, desde a posição **[2,2]** até a posição **[8,8]**.

L	C		L	C		L	C		L	C		L	C		L	C		L	C
2	2		3	2		4	2		5	2		6	2		7	2		8	2
2	3		3	3		4	3		5	3		6	3		7	3		8	3
2	4		3	4		4	4		5	4		6	4		7	4		8	4
2	5		3	5		4	5		5	5		6	5		7	5		8	5
2	6		3	6		4	6		5	6		6	6		7	6		8	6
2	7		3	7		4	7		5	7		6	7		7	7		8	7
2	8		3	8		4	8		5	8		6	8		7	8		8	8

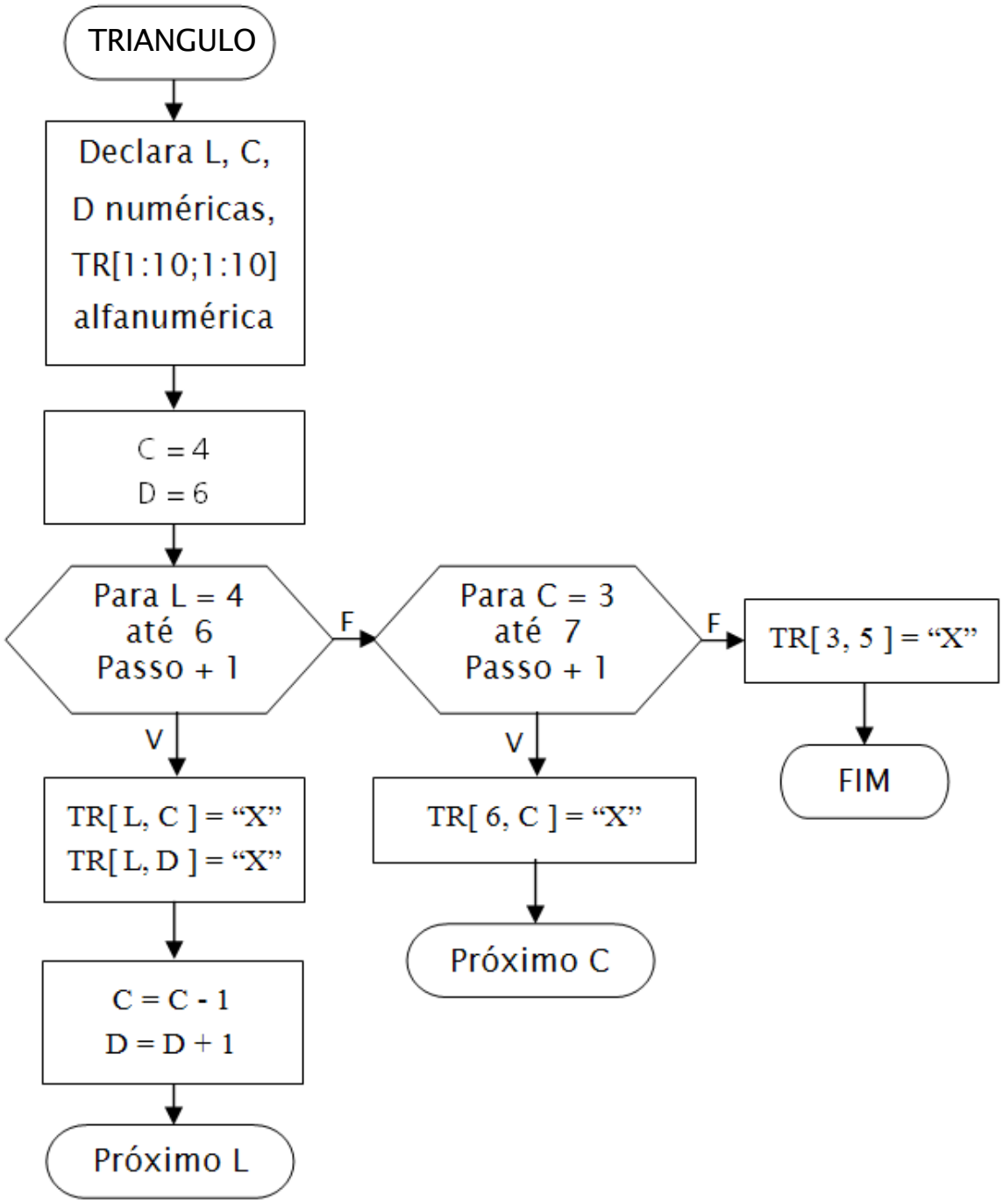
A seguir, temos um algoritmo que preenche a matriz conforme a figura utilizando dois laços:

Matriz

TR

	1	2	3	4	5	6	7	8	9	0
1										
2										
3					X					
4				X		X				
5			X				X			
6		X	X	X	X	X	X	X		
7										
8										
9										
0										

```
TRIANGULO
Declara L, C, D numéricas, TR[ 1:10,1:10 ] alfanumérica
C = 4
D = 6
Para L = 4 até 6  passo + 1
    TR[ L, C ] = “X”
    TR[ L, D ] = “X”
    C = C – 1
    D = D + 1
Próximo L
Para C = 3 até 7  passo + 1
    TR[ 6, C ] = “X”
Próximo C
TR[ 3, 5 ] = “X”
FIM
```



Considerando o exemplo do triângulo na matriz **TR**, observe, a seguir, as linhas e colunas preenchidas pelos valores das variáveis **L**, **C** e **D**:

LINHA (L)	COLUNA (C)
4	4
5	3
6	2

LINHA (L)	COLUNA (D)
4	6
5	7
6	8

Note que, se somarmos os valores das variáveis **L** e **C,** o resultado é sempre **8** (oito), portanto **C = 8 – L**. Após encontrarmos essa solução matemática, não será mais preciso utilizar a variável **C** no primeiro loop.

Perceba, também, que, se subtrairmos dos valores da variável **D** os valores da variável **L**, o resultado é sempre **2** (dois), portanto **D = 2 + L**. Após encontrarmos essa solução matemática, não será mais preciso utilizar a variável **D**.

L	+	C	=	8
4	+	4	=	8
5	+	3	=	8
6	+	2	=	8

D	-	L	=	2
6	-	4	=	2
7	-	5	=	2
8	-	6	=	2

Veja, a seguir, como ficou o algoritmo que preenche a matriz **TR**:

```
TRIANGULO
Declara L, C  numéricas, TR[ 1:10,1:10 ] alfanumérica
Para L = 4 até 6  passo + 1
    TR[ L, 8 - L ] = “X”
    TR[ L, 2 + L ] = “X”
Próximo L
Para C = 3 até 7  passo + 1
    TR[ 6, C ] = “X”
Próximo C
TR[ 3, 5 ] = “X”
FIM
```