Instituto Luterano de Ensino Superior de Ji-Paraná
Curso Bacharelado em Informática
Estrutura de Dados I
Prof.: José Luiz A. Duizith

LINGUAGEM ALGORITMICA

- Variáveis : → Toda em maiúscula : **CONSTANTE**
  → Iniciando em Maiúscula : **Comum**
  → Toda em minúscula : **apontador**

- Atribuição : ←

- Condição : **Se** (condição)
  Então [comando
  Senão [comando *"[-indica que a condição terá vários comandos"*

- Repetição : - **Enquanto** (condição)
  faça comandos
  - **Repita**
  comandos
  até (condição)
  - **Para** variável_contadora ← valor inicial **até** valor final
  faça comando

- Subprogramas :

  - **Procedimento** nome (parâmetros)
  - **Função** nome (parâmetros) : retorno

- **OBS.:** Qualquer variável utilizada num subprograma que não for parâmetro deve ser considerada como **variável local.**

MATRIZES

- Matrizes :

$$M = \begin{bmatrix} -4.3 & 8.7 \\ 2.1 & 3.9 \end{bmatrix}$$

Pascal :

```
Type
    Matriz = Array [1..2, 1..2] of Real;
Var
  M : Matriz ;
```

Algorítmo :

M (I, J) , I = 1..2, J = 1..2

MATRIZ UNIDIMENSIONAL (VETOR)

Ex.: Matriz unidimensional de 6 elemetos reais com índices entre -2 e 3. M(I), I = -2..3

M	7.2	3.5	0.4	-1.0	2.3	-9.2	**elementos**
I	-2	-1	0	1	2	3	**índices**

- Operações :

Alteração : dada uma matriz M, um índice I e um valor V, o valor de V é armazenado na posição M [I].

Consulta : dada uma matriz M, e um índice I, retorna o valor V, armazenado na posição M [I].

Operações Básicas

Procedimento Alteração (M, Ind_Min, Ind_Max, I, V)
{ **M :** Matriz unidimensional c/ índices entre Ind_Min e Ind_Max
I : índice da posição a ser alterada
V : valor a ser armazenado }
Se (Ind_Min <= I <= Ind_Max)
Entao M [I] ← V
Senao ERRO

Função Consulta (M, Ind_Min, Ind_Max, I) : V
Se (Ind_Min <= I <= Ind_Max)
Entao V ← M [I]
Senao ERRO

Tratamento de Erro

1) Ignorar : em Pascal não teria a condição senão;
2) Terminar a Execução : em Pascal, não há continuação do programa enquanto o erro não for solucionado;
3) Mensagens de Erro
4) Rotinas de Tratamento de Erro

Pascal

```
Var
  M = Array {-2,+3] of real;
  x : real
Begin
  M [1] := 7;   alteração
  x := M [2];   consulta, porém a matriz fica do lado direito
```

Algoritmo (aplicação)

Parâmetros do procedimento alteração

M (I) , I = -2..3
Alteração (M, -2, 3, 1, 7) { M [1] := 7 }
x ← consulta (M, -2, 3, 2) { x := M [2] }

Parâmetros da função consulta

Representação Natural

M (I) , I = a1..a2
M : Matriz (representação natural)
I : Índice (representação natural)
a1 : Índice inicial
a2 : Índice final

Ex.:

M	7.2	+3.5	0.4	-1.0	2.3	-9.2
I	-2	-1	0	1	2	3
	a1					**a2**

Representação Linear

L (J)
L : Matriz (repres. Linear)
J : Índice (repres. Linear)
b : Índice inicial (base)

Ex.:

L	7.2	3.5	0.4	-1.0	2.3	-9.2
J	b	b+1	b+2	b+3	b+4	b+5

<u>Linearização</u>

Transformação da representação natural em linear.

Matriz Unidimensional : L [J] = M [I], onde $\boxed{J = b + I - a1}$

Ex.:

0	
b	
1000	7.2
1001	3.5
1002	0.4
1003	-1.0
1004	2.3
1005	-9.4
640 K	

Matriz

J

-2, -1, 0, 1, **2**, 3

I = 2

I = b + I -a1

I = 100 + 2 - (-2)

$\boxed{I = 1004}$

<u>MATRIZES BIDIMENSIONAIS</u>

→ Representação Natural

M (I_1 , I_2) , $I_1 = a_1..a_2$, $I_2 = b_1..b_2$

M : Matriz (repres. Natural)
I_1 : índice da 1º dimensão (linha)
I_2 : índice da 2º dimensão (coluna)
x_1 : valor inicial
x_2 : valor final

Ex.:

$$M = \begin{array}{ll} \mathbf{a_1}\ 0 & \\ \phantom{\mathbf{a_1}}\ 1 & \\ \mathbf{a_2}\ 2 & \end{array} \begin{bmatrix} 7 & 3 & 4 & 0 \\ 5 & 2 & 3 & -1 \\ 2 & 9 & 8 & 6 \end{bmatrix}$$

$$\begin{array}{cccc} -2 & -1 & 0 & 1 \\ \mathbf{b_1} & & & \mathbf{b_2} \end{array}$$

T_1 $= b_2 - b_1 + 1$ (nº de elementos por linha)
1 + 2 + 1

T_2 $= T_1 (a_2 - a_1 + 1)$ (nº de elementos da matriz)
2 - 0 + 1
nº de linha

→ Representação Linear

$L(J), J = b..b + T_2 - 1$

L : matriz (repres. Linear)
J : índice (repres. Linear)
b : índice inicial (base)

$L(J) = M(I_1, I_2)$ onde:

$J = b + (I_1 - a_1) * T_1 + (I_2 - b_1)$
$J = b + (I_1 - a_1) * (b_2 - b_1 + 1) + (I_2 - b_1)$

Ex.:

Pascal:

M = Array [0..2, -2..1] of integer { M [x,y] }
ou
M = Array [0..2] of Array [-2..1] of integer { M [x] [y] }
[[7 3 4 0] [5 2 3 -1] [2 9 8 6]]

Como ficaria armazenado na memória

Ex.: Arquivo (est. Linear) → matriz linear é um arquivo

b →	1	7
	2	3
	3	4
	4	0
	5	5
	6	2
	7	3
	8	-1
	9	2
	10	9
J →	11	8
	12	6

$$\begin{array}{c|cccc|} 0 & - & - & - & - \\ 1 & - & - & - & - \\ 2 & - & - & \mathbf{8} & - \\ \hline & -2 & -1 & 0 & +1 \end{array}$$

Quero obter este nº no arquivo. Qual a posição?

$I_1 = 2$ (3º linha)
$I_2 = 0$ (3º coluna)

$J = b + (I_1 - a_1) * (b_2 - b_1 + 1) + (I_2 - b_1)$
$J = 1 + (2 - 0) * (1 - (-2) + 1) + (0 - -(2))$
$J = 1 + (2) * (4) + 2$
$J = 1 + 8 + 2$
$\boxed{J = 11}$

⇒ Tem que calcular o índice e recuperar a informação no arquivo. Porque os elementos estão armazenados em 1 arquivo. Após descoberto o elemento no arquivo, volta a representação natural.

→ Função Consulta ($M, I_1, I_2, a_1, a_2, b_1, b_2, L, b$) : V

Limites **Base**

{ esta função é executada na matriz de representação linear }

Se (a_1 <= I_1 <= a_2) e (b_1 <= I_2 <= b_2)
Entao J ← b + (I_1 - a_1) * (b_2 - b_1 + 1) + (I_2 - b_1)
Posiciona no registro J
V ← (J)
Senao ERRO { índice inválido }

→ **Procedimento Alteração** (M, I_1, I_2 , a_1, a_2, b_1, b_2, L, b, V)
Se (a_1 <= I_1 <= a_2) e (b_1 <= I_2 <= b_2)
Entao J ← b + (I_1 - a_1) * (b_2 - b_1 + 1) + (I_2 - b_1)
L (J) ← V
Senao ERRO { índice inválido }

MATRIZ TRIDIMENSIONAL (3 DIMENSÕES)

M (I_1, I_2, I_3) , I_1 = $a_1..a_2$, I_2 = $b_1..b_2$, I_3 = $c_1..c_2$

Coluna
Linha
Profundidade

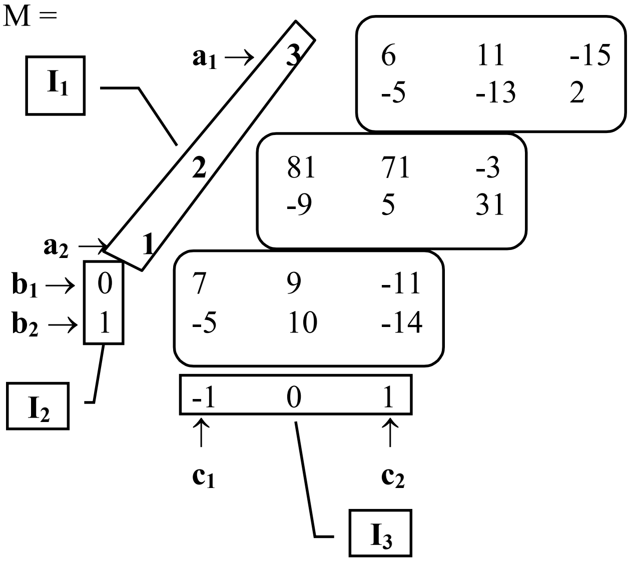
L (J) = M (I_1, I_2, I_3) , onde :

$$J = b + (I_1 - a_1) * \underbrace{(b_2 - b_1 + 1)}_{n_2} * \underbrace{(c_2 - c_1 + 1)}_{n_3} + (I_2 - b_1) * \underbrace{(c_2 - c_1 + 1)}_{n_3} + (I_3 - c_1)$$

$$J = b + (I_1 - a_1) * n_2 * n_3 + (I_2 - b_1) * n_3 + (I_3 - c_1)$$

Ex.: Pascal
M : Array [1..3, 0..1, -1..1] of integer

M =



Ex.: Por a matriz tridimensional em uma memória

	0	
	1	
	999	
	1000	7
	1001	9
	2	-11
	3	-5
L	4	10
	5	-14
	6	81
	7	71
J	8	-3
	9	-9
	10	5
	1011	31
	1012	....

$I_1 = 2 \quad I_2 = 0 \quad I_3 = 1$

$J = b + (I_1 - a_1) * (b_2 - b_1+1) * (c_2-c_1+1) + (I_2-b_1) * (c_2-c_1+1) + (I_3-c_1)$

$J = 1000 + (2-1) * (1-0+1) * (1 -(-1)+1) + (0-0*(1-(-1)+1)+(1-(-1))$

$J = 1000 + 1 * (2) * (3) + 2$

$J = 1000 + 6 + 2$

$\boxed{J = 1008}$

<u>Matriz de 'n' dimensões</u>

M (I_1, I_2, I_3 , ... , I_k) , $I_1 = a_1..a_2$, $I_2 = b_1..b_2$, $I_3 = c_1..c_2$, $I_x = z_1..z_2$

L (J) = M (I_1, I_2, I_3 , ... , I_k) , onde

$$\boxed{J = b+(I_1-a_1) * n_2 * n_3 + ... + n_k + (I_2-b_1) * n_3 * n_4 * ... * n_k ... + (I_k - z_1)}$$

MATRIZES ESPECIAIS

1) <u>Matriz Diagonal</u>

Supondo M (L,C) seja uma matriz diagonal então qualquer elemento de coordenadas L <> C é nulo.

Ex.:

$$M = \begin{array}{c} \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \\ \end{array} \begin{array}{c} \left[\begin{array}{cccc} -3.5 & 0 & 0 & 0 \\ 0 & 7.4 & 0 & 0 \\ 0 & 0 & 8.3 & 0 \end{array}\right] \\ \begin{array}{cccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} \end{array} \end{array}$$

Uma matriz diagonal pode ser armazenada em um vetor N (P), onde P é o menor valor entre L e C de M.

Ex.:

M (45,10) $\rightarrow$ P = 10, N (10)

M (100,1000) $\rightarrow$ P = 100, N (100)
M (3,4) $\rightarrow$ P = 3, N (3)

N = [	-3.5	7.4	8.3]
	1	**2**	**3**

Vetor N onde a matriz está armazenada

$\rightarrow$ **Função Consulta** (M, I_1, I_2 , a_1, a_2, b_1, b_2, N, nulo) : Valor
Se ($a_1 <= I_1 <= a_2$) e ($b_1 <= I_2 <= b_2$)
Entao Se ($I_1 = I_2$) *{diagonal principal}*
Entao V $\leftarrow$ N [I_1] *{pode ser I_2, porque eles são iguais}*
Senao V $\leftarrow$ Nulo
Senao ERRO *{índice inválido}*

$\rightarrow$ **Procedimento Alteração** (M, I_1, I_2 , a_1, a_2, b_1, b_2, N, V)
Se ($a_1 <= I_1 <= a_2$) e ($b_1 <= I_2 <= b_2$)
Entao Se ($I_1 = I_2$)
Entao N [I_1] $\leftarrow$ V *{diagonal principal}*
Senao ERRO

MATRIZES ESPARSAS

Uma matriz esparsa pode ser definida sumariamente como sendo aquela que possui a maioria dos valores nulos (em torno de 70%).

Outra forma de caracterizar uma matriz esparsa é:

Seja M (L,C), **L** linha e **C** coluna, e **K** elemento não nulo, se 3 . K < L . C então M é considerada uma matriz esparsa; justificando métodos de armazenamento mais sofisticados que ofereçam economia de espaço de armazenamento.

a) Caso Unidimensional:

Ex.: M (I) , I = 4..11

M = [	2.0	0.0	0.7	1.0	0.0	0.0	3.5	0.0]
	4	**5**	**6**	**7**	**8**	**9**	**10**	**11**

Valores $\rightarrow$	2.0	0.0	0.7	1.0	0.0	0.0	3.5	0.0
Índices $\rightarrow$	4	5	6	7	8	9	10	11

2 vetores de 8 células $\rightarrow$ 16 bytes

Para poupar espaço, armazena-se apenas as informações dos valores não nulos:

Valores $\rightarrow$	2.0	0.7	1.0	3.5		

Índices →	4	6	7	10	-1	-1

Posição de folga (usa-se índices <> da matriz normal). Criou-se p/ poder incluir outros valores neste vetor.

Para permitir que a representação de uma matriz esparsa (em 2 vetores) seja flexível, adicionam-se posições de folga que são identificados por índices fora do *intervalo* normal.

b) **Caso Bidimensional**

Ex:

M =

1	10	7	10	10	10	10
2	1	10	-2	10	10	10
3	10	3	10	10	10	10
4	12	10	10	10	10	10
5	10	10	10	10	10	10
	1	**2**	**3**	**4**	**5**	**6**

5x6 = 30 células

1 célula = 1 byte = 30 bytes

Valor →	7	1	-2	3	12	-	
Índice (linha) →	1	2	2	3	4	-1	-1
Índice (coluna)→	2	1	3	2	1	-1	-1

15 bytes = 15 células

Posição de folga

MÉTODO DA MATRIZ DE BITS

- Matriz de bits
- Vetor de valores não nulos

Matriz de bits → Tem as mesmas dimensões da matriz original, sendo que a posição de 1 valor não nulo na matriz original corresponde a um bit ligado (1) na matriz de bits e um valor nulo (na original) corresponde a um bit desligado (0) na de bits.

O vetor armazena os valores não nulos da matriz original na ordem em que os mesmos são localizados se a matriz esparsa fosse lida linha a linha.

Ex:

M =

1	10	7	10	10	10	10
2	1	10	-2	10	10	10
3	10	3	10	10	10	10

4	12	10	10	10	10	10
	1	**2**	**3**	**4**	**5**	**6**

Matriz de bits

	1	2	3	4	5	6
1	0	1	0	0	0	0
2	1	0	1	0	0	0
3	0	1	0	0	0	0
4	1	0	0	0	0	0

→ 24 posições (bits)
↓
3 bytes

\+

Vetor de valores ñ nulos →

7	1	-2	3	12

5 bytes

---

8 bytes

MATRIZ DE 4 LINHAS

- Matriz de 4 linhas x K colunas (onde K é o nº de elementos não nulos)
- Vetor VL e VC

Matriz de 4 linhas (Q) é uma matriz de 4 linhas por K colunas, onde K é o nº de elementos não nulos onde as linhas são:

1º linha : O valor do elemento não nulo (os elementos devem entrar na ordem de linha por linha da matriz esparsa).

2º linha : Coordenada de linha do elemento não nulo

3º linha : Coordenada de coluna do elemento não nulo

4º linha : referência (elo) a coluna da matriz de 4 linhas que contém a descrição do elemento seguinte na mesma coluna do elemento não-nulo.

VL : Vetor utilizado para localizar o início da descrição de uma linha.

VC : Vetor utilizado para localizar o início da descrição de uma coluna.

1º Matriz

	1	**2**	**3**	**4**	**5**
1º	7	1	-2	3	12
2º	1	2	2	3	4
3º	2	1	3	2	1
4º	**4**	5	-	-	-

Q(4,k) onde K é o nº de elem. Ñ nulos. Esta matriz contém 5 elementos ñ nulos Q(4,5)

4 - na segunda matriz existe 1 outro elemento ñ nulo logo abaixo do elemento **7 ?** Sim, o elemento 3. Agora na 1º matriz (1º linha) em que coluna está o elemento **3**? Está na 4º coluna, então coloca o nº 4 na 4º linha e 1º coluna da 1º matriz.

2º Matriz

1º		7			
2º	1		-2		
3º		3			

4º 12

1 2 3 4 5

VL (4) →

1	2	4	5
1	**2**	**3**	**4**

procura na 2º matriz o elemento não nulo e procura a linha em que ele se encontra (se existe na mesma linha vários elementos, pega só o 1º)

VL (6) →

2	1	3	-	-	-
1	2	3	4	5	6

2º coluna

procura na 2º matriz o 1º elemento não nulo, (coluna) após procura na 1º matriz a coluna em que ele está.