Algoritmos II
INF0002

Prof. Dr. Josenalde Barbosa de Oliveira

josenalde.oliveira@ufrn.br

Aulas: 35T12 – 4 CRDS – 60h

Variáveis indexadas (arrays bidimensionais e multi)

- No caso bidimensional, pode-se imaginar vetores linha e vetores coluna integrados, constituindo uma matriz com n linhas e m colunas.

```
int X[4][4], y;
X[2][2] = 20;
/* atribuição do valor 20 ao índice [2,2] da matriz X */
y = X[2][2]; // acesso ao índice [2,2] de X
```

Índices – matriz com nXm elementos = 4x4=16

X:

	0	1	2	3
0				
1				
2			20	
3				

Qual o espaço alocado na memória em bytes para armazenar esta estrutura de dados?

Variáveis indexadas (arrays bidimensionais e multi)

- Uma matriz pode ser declarada e inicializado com uma lista de valores

```
float X[3][2] = {{1,4},{5,3},{10,20}};
```

- Pode inclusive armazenar caracteres, a denominada matriz de strings

```
char S[3][30] = {{“fundamentos”}, {“desenvolvimento web”},
                  {“estruturas de dados”}};
onde 30 é o tamanho máximo do texto. Equivale a 30 colunas de
caracteres.

cout << “disciplinas do tads: ”;
cout << S[0] << S[1] << S[2];
```

- Ou, usando STL e os tipos Vector e String:

```
vector<string> tadsComponentes = {"fundamentos", "algoritmos", "matematicaI"};
std::cout << tadsComponentes[2]; // ou .front() para primeiro e .back() para
último
```

Variáveis indexadas (arrays bidimensionais e multi)

- No caso bidimensional, pode-se imaginar vetores linha e vetores coluna integrados, constituindo uma matriz com n linhas e m colunas.

```
int i2[5][7];
```

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

Você pode declarar matrizes multidimensionais que têm uma lista de inicializadores. Nessas declarações, a expressão constante que especifica os limites para a primeira dimensão pode ser omitida. Por exemplo:

```
const int cMarkets = 4;
double TransportCosts[][cMarkets] = {
   { 32.19, 47.29, 31.99, 19.11 },
   { 11.29, 22.49, 33.47, 17.29 },
   { 41.97, 22.09,  9.76, 22.55 }
};
```

- Lendo as dimensões da matriz e inserindo valores

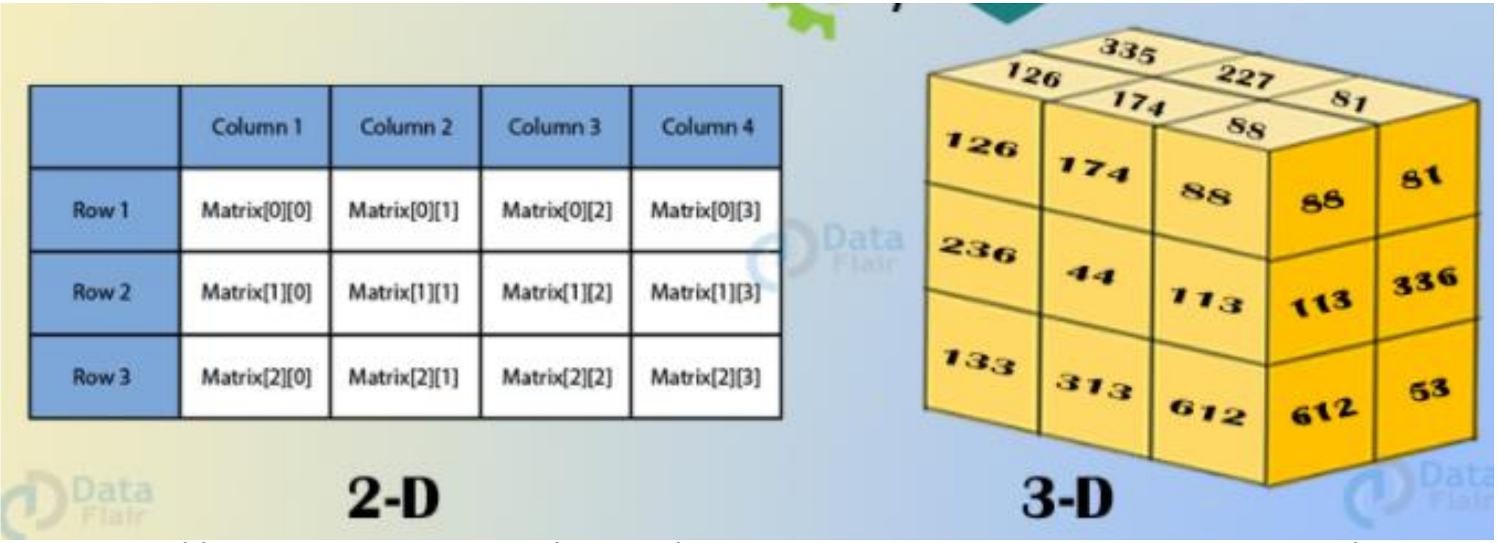
```
int L, C;
cin >> L >> C;
int v[L][C];
for (int i=0; i<L; i++)
   for (int j=0; j<C; j++)
       cin >> v[i][j];
```

-

```
int L, C;
cin >> L >> C;
int v[L][C];
for (int i=0; i<L; i++)
   for (int j=0; j<C; j++)
       cin >> v[i][j];
maior = v[0][0]; // assume o primeiro como maior
for (int i=0; i<L; i++)
   for (int j=0; j<C; j++)
       if (v[i][j] > maior) maior = v[i][j];
```

Variáveis indexadas (arrays bidimensionais e multi)

```
double multi[4][4][3]; // Declara matriz 3D. Para percorrer,
usar laço encadeado triplo (i,j,k)
```



https://data-flair.training/blogs/multi-dimensional-arrays-in-c-cpp/

Variáveis indexadas (arrays bidimensionais e multi)

```
int i, j, k, amostra[3][2][3], tamanho;

tamanho=3*2*3;

for(i = 0; i < 3; ++i) {
   for (j = 0; j < 2; ++j) {
       for(k = 0; k < 3; ++k ) {
          cin >> amostra[i][j][k]);
       }
   }
}

for(i = 0; i < 3; i++) {
   for (j = 0; j < 2; j++) {
       for(k = 0; k < 3; k++) {
cout << “amostra ” << i <<“,” << j << “,” << k << “:” << amostra[i][j][k]);
       }
   }
}
```

Matrizes especiais e algoritmos

MATRIZ QUADRADA – número de linhas igual ao número de colunas, definida por uma dimensão N (ou seja, N x N)

Diagonal principal

Diagonal secundária

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Seja i (linha) e j (coluna), então

DP: **i == j**

DS: **i+j = N + 1**, onde N é a dimensão

$$A_{3x3} = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 3 \\ 0 & 0 & 0 \end{pmatrix} \rightarrow \textit{No matriz triangular.}$$

$$B_{3x3} = \begin{pmatrix} -1 & 1 & 1 \\ 0 & 1 & 2 \\ 0 & 0 & 3 \end{pmatrix} \rightarrow \textit{Matriz triangular superior.}$$

$$C_{3x3} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 2 \\ 3 & -2 & 1 \end{pmatrix} \rightarrow \textit{No matriz triangular.}$$

$$D_{3x3} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \rightarrow \textit{Matriz triangular inferior.}$$

$$E_{3x3} = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix} \rightarrow \textit{Matriz triangular superior.}$$

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ 0 & A_{22} & A_{23} & \dots & A_{2n} \\ 0 & 0 & A_{33} & \dots & A_{3n} \\ 0 & 0 & 0 & \dots & A_{4n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & A_{nn} \end{bmatrix}_{n \times n} \Rightarrow A^t = \begin{bmatrix} A_{11} & 0 & 0 & \dots & 0 \\ A_{12} & A_{22} & 0 & \dots & 0 \\ A_{13} & A_{23} & A_{33} & \dots & 0 \\ A_{14} & A_{24} & A_{34} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_{1n} & A_{2n} & A_{3n} & \dots & A_{nn} \end{bmatrix}_{n \times n}$$

TRIANGULARES

Matrizes especiais e algoritmos

MATRIZ IDENTIDADE, DIAGONAL e NULA – diagonal principal com 1

→ Matriz diagonal

Apenas os elementos da diagonal principal são diferentes de zero

$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 7 \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

→ Matriz identidade

A identidade é uma matriz diagonal cujo elementos da diagonal principal são todos iguais a um.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Chamamos a matriz acima de I_3 (identidade de ordem 3)

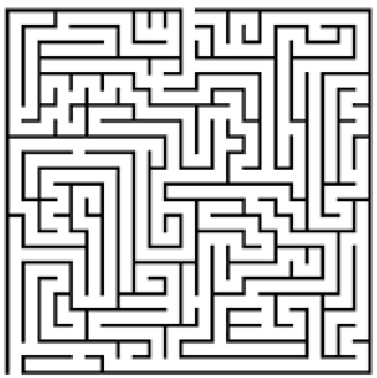
No geral, I_n onde n é a ordem da matriz.

$$I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

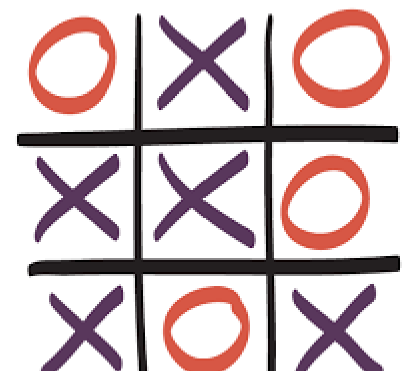
Operações com matrizes

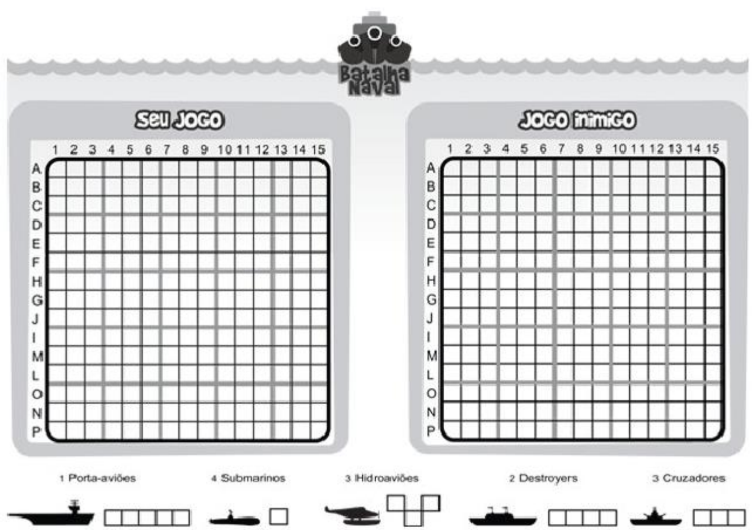
Portanto, ao desenvolvedor de sistemas é importante conhecer tais conceitos e implementar algoritmos para percorrer as matrizes em sua totalidade ou em partes (como as diagonais), com operações de soma, subtração, média, seleção de valores, ordenação, etc.

Exemplos de aplicações



1	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	0	1	1
1	1	1	0	1	1	0	1	0	1
0	0	0	0	1	0	0	1	0	0
1	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	0
1	0	0	0	1	0	0	1	0	1
1	0	1	1	1	1	0	0	1	1
1	1	0	0	1	0	0	0	0	1
1	0	1	1	1	1	0	1	0	0





Exemplos de aplicações: robô (obi 2013)

https://olimpiada.ic.unicamp.br/pratique/p1/2013/f1/robo/

Exemplos de aplicações: quadrado mágico (obi 2011)

https://olimpiada.ic.unicamp.br/pratique/p2/2011/f2/magico/

Exemplos de aplicações: chuva (obi 2019)

https://olimpiada.ic.unicamp.br/pratique/p2/2019/f1/chuva/

Exemplos de aplicações: matriz super legal (obi 2019)

https://noic.com.br/materiais-informatica/comentario/2019-fase2-p1/

https://olimpiada.ic.unicamp.br/pratique/p2/2019/f2/matriz/