Matrizes

Objetivos

- Conceituação de Vetores Bidimensionais
- Manipulação de Vetores Bidimensionais
- Entender como manipular entrada, saída e índices de vetores bidimensionais
- Resolver problemas que requeiram o uso de matrizes bidimensionais

Relembrando

Vetores (No Portugol Studio)

- Permite a declaração de **variáveis do tipo CONJUNTO**
- Uma variável do tipo CONJUNTO pode armazenar **mais de um valor**.
- No ato da declaração da variável deve-se informar o seu **tamanho.**
- Para o problema de armazenar **10 notas**, pode-se definir uma variável de **tamanho 10** do tipo **real**.
  - **Declaração**:

real nota[10]

nota

conteúdo	5.5	6.5	8.0	3.0	7.5	2.5	7.5	6.0	4.5	10.0
índice	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**

Vetores (No Portugol Studio)

- **Declaração:**

real nota[10]

- A declaração acima cria em memória uma variável chamada NOTA com 10 posições do tipo REAL
- As 10 posições são numeradas de 0 a 9 (índice)

nota

conteúdo	5.5	6.5	8.0	3.0	7.5	2.5	7.5	6.0	4.5	10.0
índice	0	1	2	3	4	5	6	7	8	9

- **Acesso a variável:**

Para acessar cada posição deve-se usar o nome da variável e a sua posição ou índice (entre colchetes)

A instrução abaixo imprime O TERCEIRO ELEMENTO da variável NOTA.

• escreva(" A TERCEIRA NOTA = " , nota [2])

O índice pode ser uma variável:
Ex. nota [x]

Mostrar as notas acima da média da turma

```
funcao inicio()
{
    real nota[10], soma=0, media
    inteiro ind=0, qacima = 0

    enquanto (ind <10) {
        escreva("Digite a nota : " )
        leia (nota[ind])
        soma = soma + nota[ind]
        ind = ind+1
    }
    media = soma/10
    escreva ("\nMedia = ", media)
```

Lê as notas via teclado e guarda no vetor

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

Vetor no Portugol Studio

```
    ind = 0
    enquanto (ind<10) {
        se (nota[ind] > media) {
            qacima = qacima + 1
        }
        ind = ind + 1
    }
    escreva ("\nExistem ", qacima , " notas acima da media ")
}
```

Verifica se cada nota armazenada está acima da média

nota

conteúdo	5.5	6.5	8.0	3.0	7.5	2.5	7.5	6.0	4.5	10.0
índice	0	1	2	3	4	5	6	7	8	9

media

6.1

PROBLEMA

Como representar na memória do computador dados com mais de uma dimensão?

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	3	0	0	3	0	0	4	4	4	4	0	0	5	5	5	5	0	0	7	7	7	7	0	0	9	9	9	9	0
0	3	0	0	3	0	0	4	0	0	0	0	0	5	0	0	5	0	0	7	0	0	7	0	0	9	0	0	0	0
0	3	0	0	3	0	0	4	4	4	0	0	0	5	5	5	5	0	0	7	7	7	0	0	0	9	0	0	0	0
0	3	0	0	3	0	0	4	0	0	0	0	0	5	0	0	5	0	0	7	0	0	7	0	0	9	0	0	0	0
0	3	3	3	3	0	0	4	0	0	0	0	0	5	0	0	5	0	0	7	7	7	7	0	0	9	9	9	9	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IMAGEM

Problema

- **Escreva um programa para armazenar as notas de 20 alunos em 5 disciplinas.**

Escreva um programa para armazenar as notas de 20 alunos em 5 disciplinas.

ALUNOS	DISC01	DISC02	DISC03	DISC04	DISC05
1	5,5	8,3	6,5	10,0	9,5
2	8,0	7,5	8,5	6,5	10,0
3	...	...	...	...	...
4	...	...	...	...	...
5	...	...	...	...	...
6	...	...	...	...	...
7	7,5	8,0	9,0	8,0	8,5
...	6,5	3,5	6,5	4,5	9,0
18	...	...	...	...	...
19	...	...	...	...	...
20	8,0	9,0	8,0	10,0	9,0

Escreva um programa para armazenar as notas de 20 alunos em 5 disciplinas.

ALUNOS	DISC01	DISC02	DISC03	DISC04	DISC05
1	5,5	8,3	6,5	10,0	9,5
2	8,0	7,5	8,5	6,5	10,0
3	...	...	...	...	...
4	...	...	...	...	...
5	...	...	...	...	...
6	...	...	...	...	...
7	7,5	8,0	9,0	8,0	8,5
...	6,5	3,5	6,5	4,5	9,0
18	...	...	...	...	...
19	...	...	...	...	...
20	8,0	9,0	8,0	10,0	9,0

Poderíamos enxergar essa estrutura como:

- 5 vetores de 20 posições
- 20 vetores de 5 posições

Ou

1 matriz de 20 x 5

MATRIZES

Matrizes

- Podem ser definidas como vários vetores agrupados em uma só variável
- Uma matriz é muito semelhante a uma tabela contendo linhas e colunas
- As matrizes, como os vetores, são capazes de guardar várias informações de um mesmo tipo de dado
- Para manipular os dados de uma matriz será necessário usar **dois laços** ao invés de um
  - Um laço para caminhar pelas linhas da matriz e
  - Outro para caminhar pelas colunas da matriz

MATRIZES

Matrizes são estruturas MULTIDIMENSIONAIS (mais de uma dimensão) capazes de armazenar dados

A figura abaixo representa uma matriz BIDIMENSIONAL de números inteiros

10	5	33	41
53	20	-10	0
29	17	30	8

Declaração de uma variável matriz

- **tipo** **identificador** [linhas] [colunas];
- Exemplo:
  - inteiro A [3] [4] // declara a matriz A : 3 linhas e 4 colunas

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

- **Acesso a elementos da matriz**

A

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

- escreva (**A [2] [1])** // imprime o valor 17

- inteiro i = 0, j = 2
- escreva (A [i] [j]) // imprime o valor **33**
- escreva (A [j] [i]) // imprime o valor **29**

- **Para acessar TODOS os elementos da matriz é necessário que os índices assuma todas as combinações possíveis**

- (i, j)
- (0,0, (0,1) (0,2), (0, 3) ← i = 0 e j = 0, 1, 2, 3
- (1,0, (1,1) (1,2), (1, 3) ← i = 1 e j = 0, 1, 2, 3
- (2,0, (2,1) (2,2), (2, 3) ← i = 2 e j = 0, 1, 2, 3

A

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

Entrada de Dados

Matrizes

- **Declarando uma Matriz de 3 linhas e 3 colunas (vetor de duas dimensões)**

```
1.    Algoritmo "Matriz"
2.    VAR
3.    VALORES : VETOR [1..3,1..3] DE REAL
```

- **Lê uma Matriz de duas dimensões**

```
Para i de 1 ate 3 faca
   Para j de 1 ate 3 faca
      Escreva("Digite um valor para a matriz")
      Leia (VALORES[i,j])
   fimpara
fimpara
```

- **Imprime uma Matriz de duas dimensões**

```
Para i de 1 ate 3 faca
   para j de 1 ate 3 faca
      escreval (VALORES[i,j])
   fimpara
fimpara
```

Problema

- **Escreva um programa para LER (via teclado) uma matriz de 3 x 4**

Entrada de Dados

```
const inteiro LIN = 3, COL = 4  // declara duas constantes
inteiro A[LIN][COL]      // cria a matriz A

inteiro l, c    // l para percorrer linha e c para coluna

l = 0
enquanto (l < LIN) {
      c = 0
      enquanto(c < COL) {
            escreva ("Digite o elemento ", l, " ,", c, " : ")
            leia (A[l][c])
            c++
      }
      l++
}
```

Entrada de Dados – para

```
para(l=0; l < LIN; l++) {
     para (c=0; c< COL; c++) {
          escreva ("Digite o elemento ", l, " ,", c, " : ")
          leia (A[l][c])
     }
}
```

```
l = 0
enquanto (l < LIN) {
     c = 0
     enquanto(c < COL) {
          escreva ("Digite o elemento ", l, " ,", c, " : ")
          leia (A[l][c])
          c++
     }
     l++
}
```

Saída de Dados

Problema

- **Escreva um programa para EXIBIR na tela uma matriz de 3 x 4. Considere que a matriz já está na memória.**

A

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

Saída de Dados

```
//  SAÍDA DE DADOS

l = 0
enquanto (l < LIN) {
      c = 0
      enquanto(c < COL) {
            escreva (A[l][c], " ;")
            c++
      }
      escreva ("\n")
      l++
}
```

A

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

- **Saída de Dados: Para - Até**

```
//  SAÍDA DE DADOS

para(l=0; l < LIN; l++) {
      para (c=0; c< COL; c++) {
            escreva (A[l][c], " ;")
      }
      escreva ("\n")
}
```

A

	0	1	2	3
0	10	5	33	41
1	53	20	-10	0
2	29	17	30	8

```
//  SAÍDA DE DADOS

l = 0
enquanto (l < LIN) {
      c = 0
      enquanto(c < COL) {
            escreva (A[l][c], " ;")
            c++
      }
      escreva ("\n")
      l++
}
```

Coisas para não esquecer:

- **MATRIZ permite declarar vetores BIDIMENSIONAIS (multidimensionais)**

- **Sintaxe para declaração de matriz (bidimensional)**:
  tipo identificador [quantidade linhas] [quantidade colunas]

- **Sintaxe para acesso ao vetor**:
  identificador [indice_linha] [indice_coluna]

Matrizes em Java

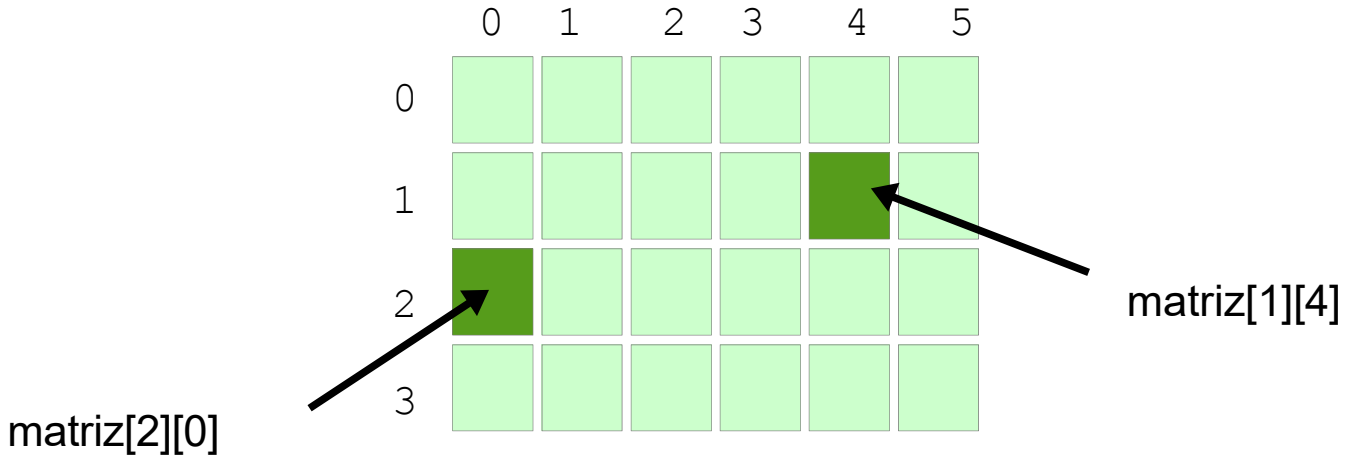
Vetor bidimensional em Java

Declaração de uma variável que representa uma **matriz de inteiros 10 linhas, 20 colunas**

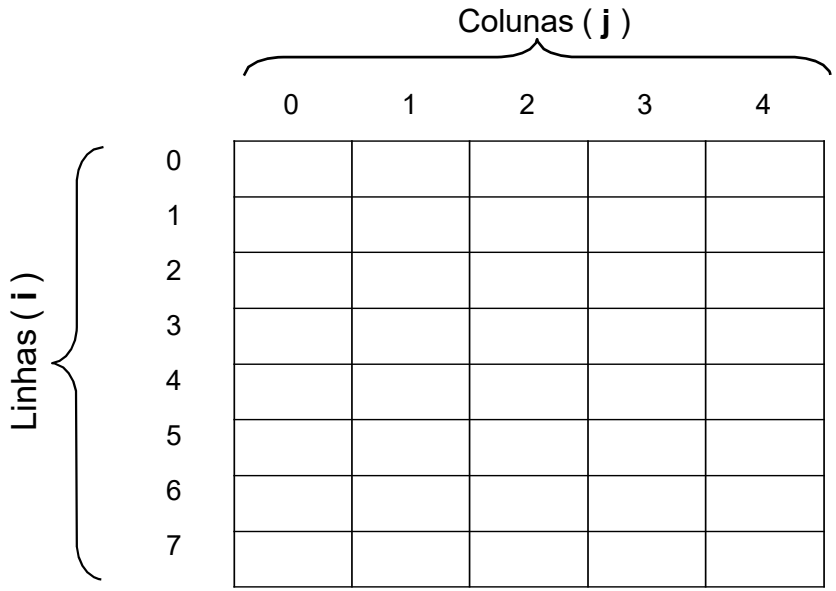
```
int matriz[][];
matriz = new int[10][20];
```

```
int matriz[][] = new int[10][20];
```

Matriz bidimensional em Java



Matrizes



No exemplo ao lado temos uma matriz de

8 linhas por 5 colunas

- **Sintaxe**:

```
tipo nome[][] = new tipo[linhas][colunas];
```

Alguns padrões para manipulação de Matriz (java)

Matrizes

- Lendo valores para uma Matriz

```
int mat[][] = new int [2][2];
   int soma = 0;
int i, j;

for(i=0;i<2;i++)
   for(j=0;j<2;j++) {
mat[i][j] = in.nextInt();
}
```

- Imprimindo valores

```
int mat[][] = new int [2][2]; int soma
   = 0;
int i, j;

for(i=0;i<2;i++) for(j=0;j<2;j++) {
System.out.println("Elemento[" + i
+ "]["+ j + "]: “+        mat[i][j]);
}
```

Matrizes - Manipulação

- Modificando valores

```
int mat[][] = new int [2][2];
   int soma = 0;
int i, j;

for(i=0;i<2;i++)
   for(j=0;j<2;j++) {
mat[i][j] = i*3;
}}
```

Matrizes

```
int mat[][] = new int [2][2]; int soma = 0;
int i, j;

for(i=0;i<2;i++) for(j=0;j<2;j++) {
System.out.println("Elemento[" + i + "]["+ j + "]: "); mat[i][j] =
in.nextInt();
soma = soma + mat[i][j];
}}
System.out.println("Soma dos elementos da Matriz: "+ soma);
```

Matrizes

- Carregando valores

```
int mat[][] = new int [2][2];
int soma = 0;
int i, j;

for(i=0;i<2;i++)
  for(j=0;j<2;j++) {

System.out.println("Elemento
  [" + i + "]["+ j + "]: ");
mat[i][j] = in.nextInt();
}
```

- Imprimindo valores

```
int mat[][] = new int [2][2];
int soma = 0;
int i, j;

for(i=0;i<2;i++)
  for(j=0;j<2;j++) {

System.out.println("Elemento
  [" + i + "]["+ j + "]: "+
mat[i][j]);
}
```

Entradas = 10, 20, 4, 5, 7, 10, 11, 8, 50, 60, 50, 23, 89, 100, 1, 0, 34,6, 20,24

```
for (int i  = 0; i < 5; i++)
    for(int j = 0; j < 4; j++)
        A[i][j] =    in.nextInt();
```

	0	1	2	3
0	10	20	4	5
1	7	10	11	8
2	50	60	50	23
3	89	100	1	0
4	34	6	20	24

Passo	i	j	A[i,j]
1	0	0	A[0,0] = 10
2	0	1	A[0,1] = 20
3	0	2	A[0,2] = 4
4	0	3	A[0,3] = 5
5	1	0	A[1,0] = 7
6	1	1	A[1,1] = 10
7	1	2	A[1,2] = 11
8	1	3	A[1,3] = 8
9	2	0	A[2,0] = 50
...			
19	4	2	A[4,2] = 20
20	4	3	A[4,3] = 24

Matrizes - Manipulação

- Modificando valores

```
int mat[][] = new int [2][2];
int soma = 0;
 int i, j;

  for(i=0;i<2;i++) for(j=0;j<2;j++) {
mat[i][j] = i*3;
}
```

```
int soma = 0;
int i, j;


for(i=0;i<2;i++) for(j=0;j<2;j++) {
System.out.println("Elemento[" +
  mat[i][j] = in.nextInt();          i + "]["+ j + "]: ");
soma = soma + mat[i][j];

  }
  System.out.println("Soma dos elementos da Matriz:
  "+ soma);
```

Atribuir valores para uma matriz

```
int i, j;
int M[][] = new int[5][10];

for (i=0; i<5; i=i+1) {
    for (j=0; j<10; j=j+1) {
        if (i%4==0 || j%9==0 )
            M[i][j] = 1;
        else
            M[i][j] = 0;
    }
}
```

```
1 1 1 1 1 1 1 1 1 1

1 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 0 1
1 1 1 1 1 1 1 1 1 1
```

Exercício: Triângulo de Pascal

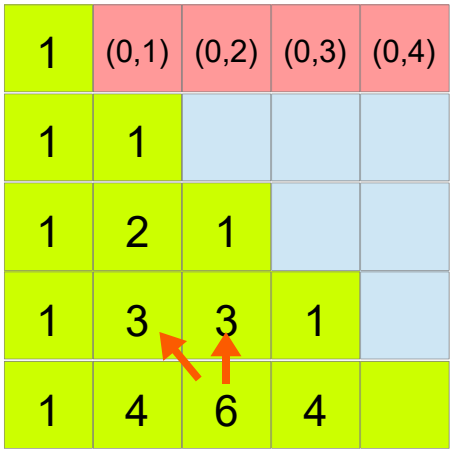
Crie um programa que permita calcular os valores para as **10** primeiras linhas do triângulo de Pascal.

```
int i, j;
int M[][] = new int[10][10];
```

```
 1   0   0    0     0    0    0    0   0   0
 1   1   0    0     0    0    0    0   0   0
 1   2   1    0     0    0    0    0   0   0
 1   3   3    1     0    0    0    0   0   0
 1   4   6    4     1    0    0    0   0   0
 1   5   10   10    5    1    0    0   0   0
 1   6   15   20   15    6    1    0   0   0
 1   7   21   35   35   21    7    1   0   0
 1   8   28   56   70   56   28    8   1   0
 1   9   36   84  126  126   84   36   9   1
```

Exercício: Triângulo de Pascal

```
int i, j;
int M[][] = new int[10][10];

for (i=0; i<10; i=i+1) {
    for (j=0; j<=i; j=j+1) {
        if (j==0 || i==j )
            M[i][j] = 1;
        else
            M[i][j] = M[i-1][j] + M[i-1][j-1];
    }
}
```



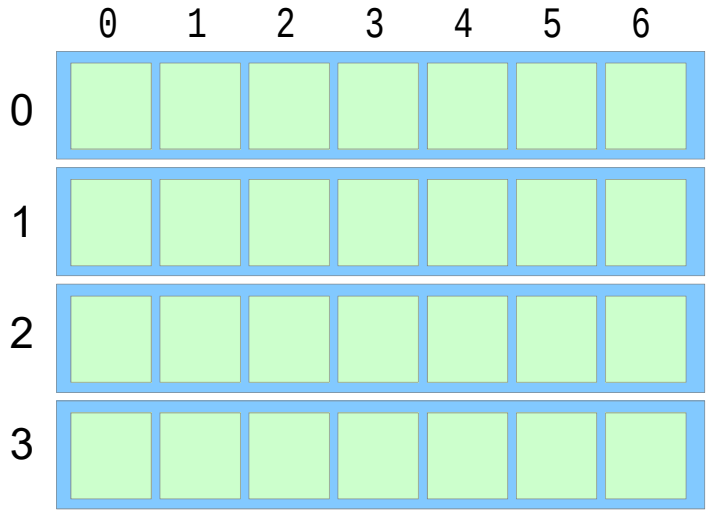
Tamanho dinâmico

```
int i, j;

Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
int m = sc.nextInt();

int M[][] = new int[n][m];

for (i=0; i<n; i=i+1) {
    for (j=0; j<m; j=j+1) {
        M[i][j] = 10;
    }
}


for (i=0; i<n; i=i+1) {
    for (j=0; j<m; j=j+1) {
        System.out.print(M[i][j]+" ");
    }
    System.out.print("\n");
}
```

```
3
7
10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10
```

Número de linhas e colunas

M:=




```
M.length       → 4
M[0].length    → 7
M[1].length    → 7
M[2].length    → 7
M[3].length    → 7
M[4].length    → erro
```

Matriz como parâmetro em função

```
static void imprimirMatriz(int M[][]) {
    int i, j;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            System.out.print(M[i][j]+" ");
        }
        System.out.print("\n");
    }
}
```

```
10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10
```

Matrizes

```java
static void imprimirMatriz(int M[][]) {
    int i, j;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            System.out.print(M[i][j]+" ");
        }
        System.out.print("\n");
    }
}
```

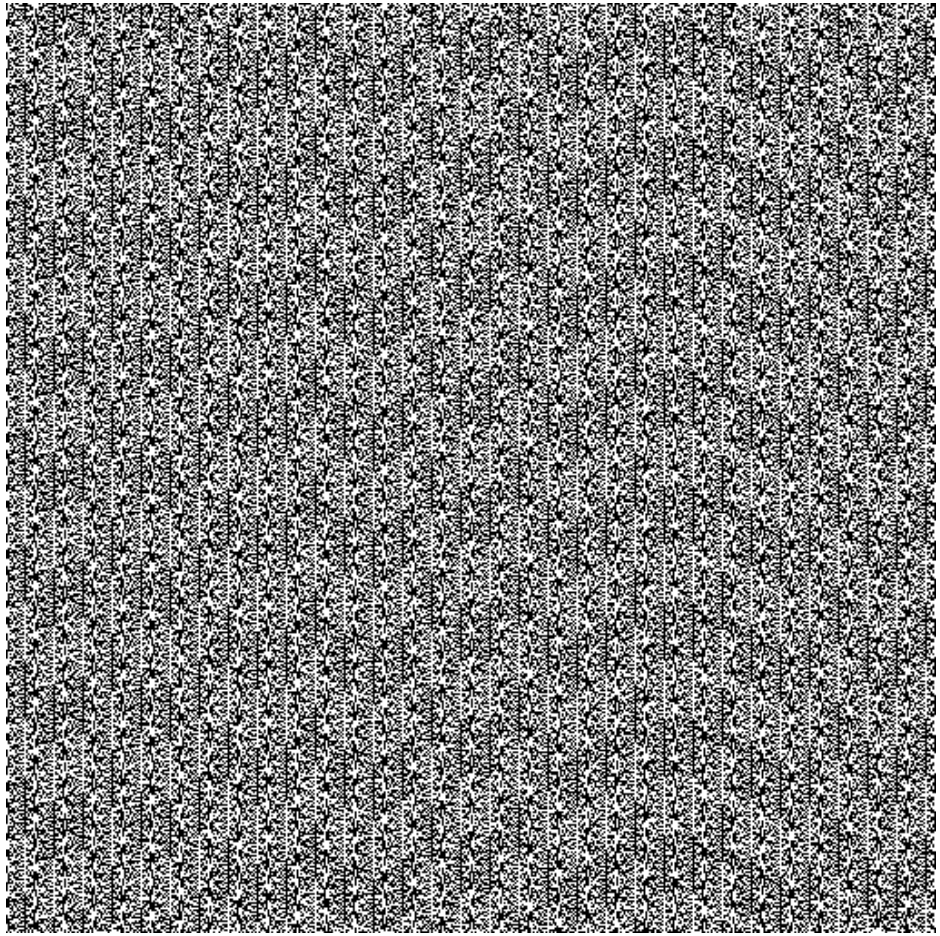
```java
public static void main(String []args) {
    int i, j;

    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    int m = sc.nextInt();

    int M[][] = new int[n][m];

    for (i=0; i<n; i=i+1) {
        for (j=0; j<m; j=j+1) {
            M[i][j] = 10;
        }
    }

    imprimirMatriz(M);
}
```

Números aleatórios

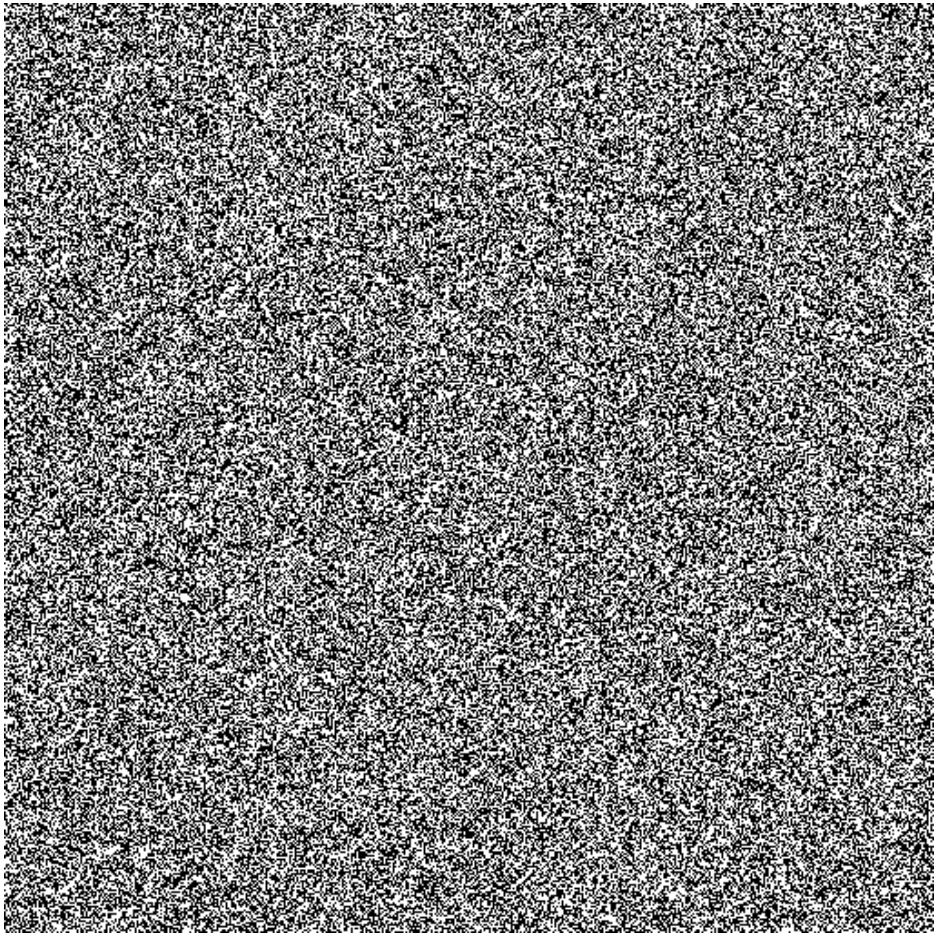
```
double x;
int y;

x = Math.random();       // gera um numero real aleatorio [0, 1)

x = Math.random()*30;    // gera um numero real aleatorio [0, 30)

y = (int) (Math.random()*30); // gera um numero inteiro aleatorio [0, 30)

y = (int) (Math.random()*30)+10; // gera um numero inteiro aleatorio [10, 40)

y = (int) (Math.random()*60)-30; // gera um numero inteiro aleatorio [-30, 30)
```

Números pseudo-aleatórios

Pseudo-random



True-random



Matriz com números aleatórios

```
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
int m = sc.nextInt();

int M[][] = new int[n][m];

for (i=0; i<n; i=i+1) {
    for (j=0; j<m; j=j+1) {
        M[i][j] = (int) (Math.random()*5)+6;
    }
}
```

```
3
7
6 8 8 10 10 8 7
10 7 7 8 10 9 6
10 6 6 6 6 7 10
```

Exemplo: Primeiro menor

```
static int primeiroMenor (int M[][]) {
    int i, j;
    int menor = M[0][0];

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if (menor>M[i][j]) {
                menor = M[i][j];
            }
        }
    }
    return menor;
}
```

Qual o custo Computacional?

```
 6  6  7  7  7  9  9

 8 10  7  7  9  7  8

 9  9 10 10  7 10  9
```

Exemplo: Primeiro menor

```
static int primeiroMenor (int M[][]) {
    int i, j;
    int menor = M[0][0];

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if (menor>M[i][j]) {
                menor = M[i][j];
            }
        }
    }
    return menor;
}
```

Qual o custo Computacional?

Algo relacionado com o número de elementos: n*m (#linhas * #colunas)

T(n,m) = O(nm)

```
 6  6  7  7  7  9  9

 8 10  7  7  9  7  8

 9  9 10 10  7 10  9
```

AGORA É PRATICAR!!!

Atividade 01: Matriz quadrada

Dada uma matriz M, crie uma função que permita verificar se a matriz é quadrada.

Assinatura: `static boolean matrizQuadrada(int M[][])`

A função devolve 'true' se for quadrada, caso contrário devolve 'false'.

Atividade 01: Matriz quadrada

```
static boolean matrizQuadrada(int M[][]) {

    if (M.length==M[0].length)
        return true;
    else
        return false;
}
```

Atividade 02: Índices

O que faz a seguinte função?

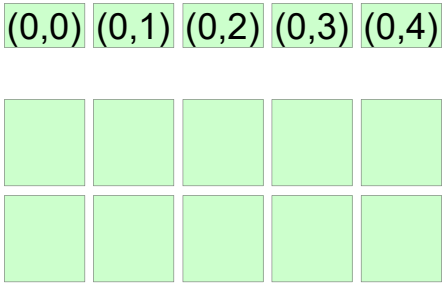
Considere M uma matriz de dimensão (3,5)

```
static void indices(int M[][]) {
    int i, j;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            System.out.println(i+" "+j);
        }
    }
}
```

Atividade 02: Índices

Imprime a seguinte sequência de pares:

```
0 0
0 1
0 2
0 3
0 4
1 0
1 1
1 2
1 3
1 4
2 0
2 1
2 2
2 3
2 4
```



Atividade 03: Índices

O que faz a seguinte função?

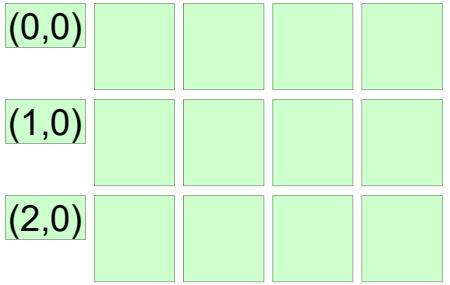
Considere M uma matriz de dimensão (3,5)

```
static void indices2(int M[][]) {
    int i, j;

    for (j=0; j<M[0].length; j=j+1) {
        for (i=0; i<M.length; i=i+1) {
            System.out.println(i+" "+j);
        }
    }
}
```

Atividade 03: Índices

Imprime a seguinte sequência de pares:

```
0 0
1 0
2 0
0 1
1 1
2 1
0 2
1 2
2 2
0 3
1 3
2 3
0 4
1 4
2 4
```



Quantidade de elementos não nulos

Dada uma matriz M, crie uma função que permita devolta a quantidade de elementos não nulos contidos na matriz.

Assinatura: `static int contaNaoNulos(int M[][])`

$$\begin{bmatrix} 6 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 \\ 0 & 0 & 8 & 0 \end{bmatrix}$$

Número de não nulos = 3

Atividade 04:

Matrizes

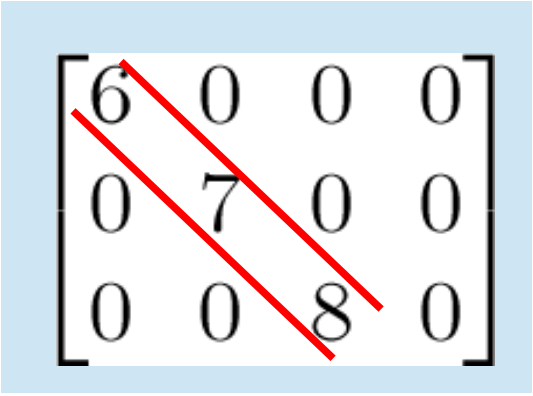
```
static int contaNaoNulos(int M[][]) {
    int i, j, cont=0;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if (M[i][j]!=0)
                cont = cont+1;
        }
    }

    return cont;
}
```

Atividade 05:

Diagonal principal

- Dada uma matriz M, crie uma função que determine a somatória de todos os números presentes na sua diagonal principal.

- Assinatura: `static int somaDiagonal(int M[][])`



Somatória da diagonal = 21

```
static int somaDiagonal(int M[][]) {
    int i, j, soma=0;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if (i==j)
                soma = soma+M[i][j];
        }
    }

    return soma;
}
```

Diagonal principal

Solução mais eficiente:

```
static int somaDiagonal2(int M[][]) {
    int i, soma=0;

    for (i=0; i<M.length && i<M[0].length; i=i+1) {
        soma = soma + M[i][i];
    }

    return soma;
}
```

Matriz identidade

Dada uma matriz quadrada M, crie uma função que permita verificar se a matriz é identidade.

Assinatura: `static boolean matrizIdentidade (int M[][])`

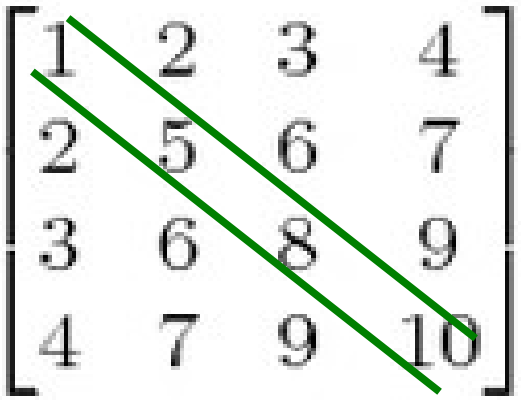
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```java
static boolean matrizIdentidade (int M[][]) {
    int i, j;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if ( (i==j && M[i][j]!=1) || (i!=j && M[i][j]!=0) )
                return false;
        }
    }

    return true;
}
```

Matriz simétrica

Dada uma matriz quadrada M, crie uma função que permita verificar se a matriz é simétrica.

Assinatura: `static boolean matrizSimetrica (int M[][])`



```java
static boolean matrizSimetrica (int M[][]) {
    int i, j;

    for (i=0; i<M.length; i=i+1) {
        for (j=0; j<M[0].length; j=j+1) {
            if ( M[i][j]!=M[j][i] )
                return false;
        }
    }

    return true;
}
```

Exercícios

Exercícios

1 – Ler duas matrizes A e B 5x3. Construir uma matriz C de mesma dimensão, sendo que C é formada pela soma dos elementos da matriz A com os elementos da matriz B. Apresentar os elementos da matriz C.

2 – Ler dois vetores A e B com 12 elementos inteiros. Construir uma matriz C 12x2, sendo que a primeira coluna da matriz C deve ser formada pelos elementos do vetor A multiplicados por 2, e a segunda coluna deve ser formada pelos elementos do vetor B subtraídos de 5. Apresentar os elementos da matriz C