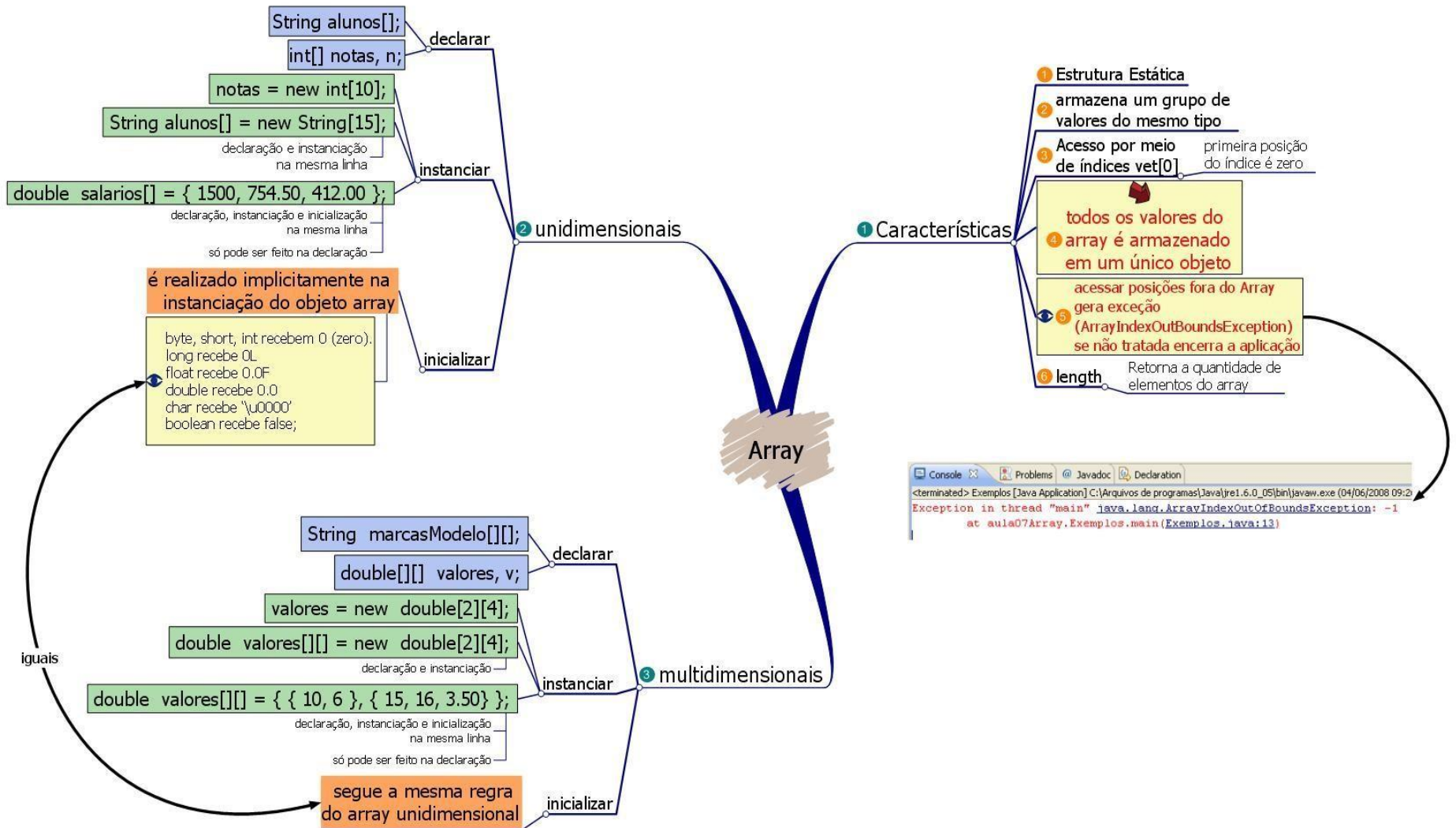
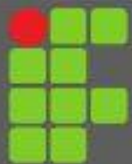


# **Programação Orientada a Objetos**

**Professor Vinícius Bortolini**

# Mapa conceitual da aula

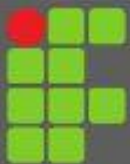




# Objetivos

Ao final desta aula, o aluno será capaz de:

1. declarar e instanciar Arrays;
2. popular e percorrer Arrays; e
3. implementar programas com Arrays unidimensionais e multidimensionais



# Arrays

```
f(i){ aux=1; aux<1000; aux++){  
    System.out.println("Please, God!!");  
}
```

- Um Array é uma **estrutura de dados homogênea**
  - um **grupo de valores do mesmo tipo** armazenados sequencialmente na memória.
- Para acessar um destes valores armazenados é necessário **indicar a posição** dentro do Array (**índice**).
- Java armazena os valores de um Array dentro de um **único Objeto**.
- Um Array é uma **estrutura de dados estática**.
  - Depois que foi definido o seu tamanho ele não pode mais ser alterado (**tamanho imutável**).

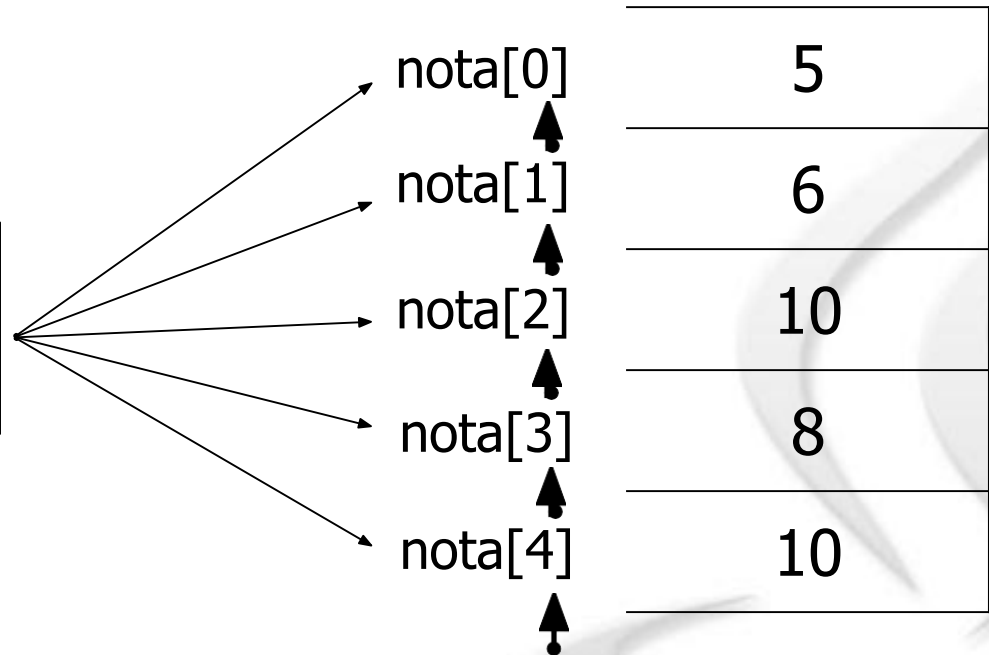


# Arrays

Exemplo:

**Nome do Array**

Todas as posições têm o mesmo nome...

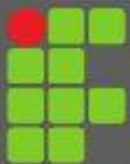


... mas diferem pela ordem do elemento no Array (**índice**)

**Atenção:** o índice obrigatoriamente deve ser um **número inteiro**.

- a **primeira posição** do Array é acessada pelo **índice zero**;
- a **última posição** tem valor igual ao **tamanho do Array menos 1**.

Acessar **posições fora do Array** gera exceção (ArrayIndexOutOfBoundsException) que, caso não seja tratada, **encerra a execução da aplicação**.



# Arrays

```
f (int aux=1; aux<1000; aux++){  
    System.out.println("Please, God!!");  
}
```

Para se utilizar um Array:

1. **declarar** a variável do tipo Array;
2. **instanciar** o objeto Array (alocar memória);
3. **inicializar** os valores do objeto Array.



# Declaração de um Array

- Pode-se construir um Array a partir de um tipo primitivo ou por meio de uma variável de instância para referenciar objetos de uma classe.
  - **char, int, float, boolean, ...**
  - **String, Cliente, Produto, ...**

Array do  
tipo int

Array do tipo double

Variável normal do  
tipo double

```
int notas[];  
double precos[], salarios;  
String[] professores, alunos;
```

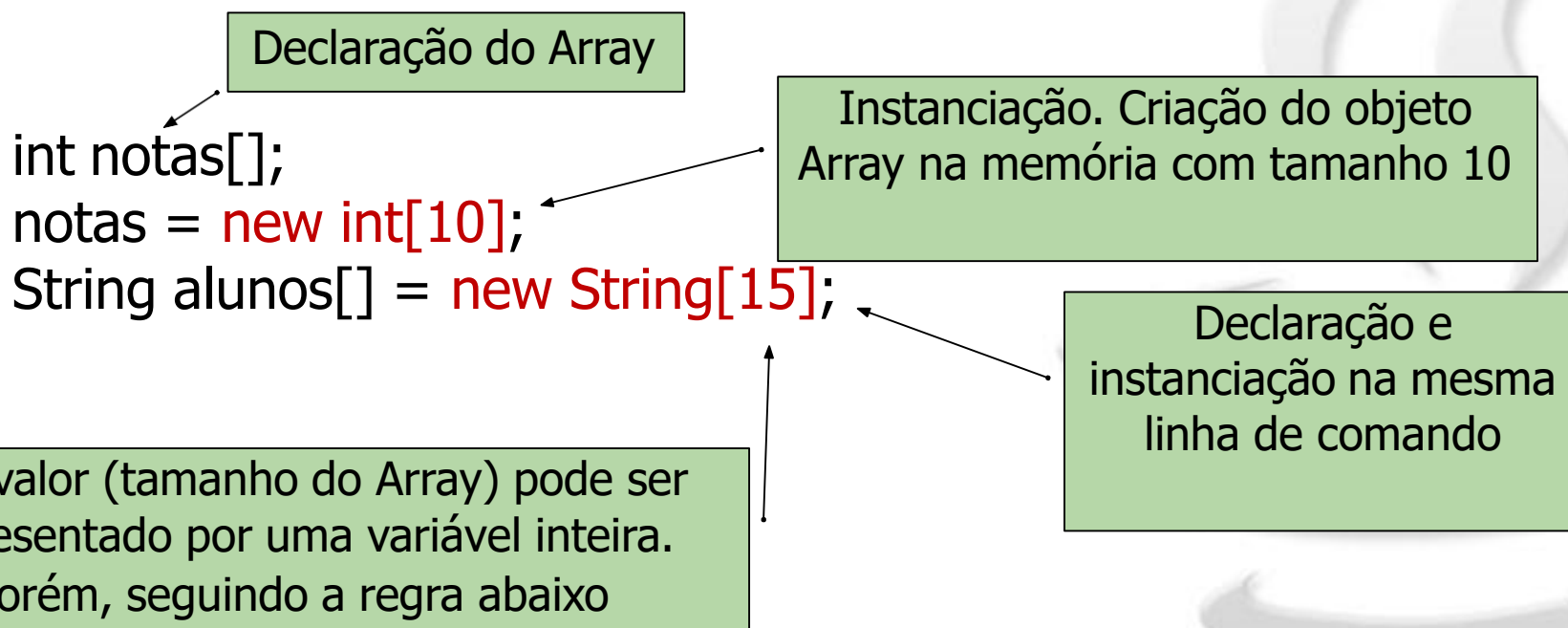
: Arrays do tipo String

**Atenção:** Quando o [] está junto ao tipo de dado durante uma declaração, todas as variáveis definidas nesta instrução são Arrays.



# Instanciação de um Array

- A instanciação do novo Array é realizada através do operador **new**, como acontece com todos os objetos em Java.



**Atenção:** depois de alocado, um Array não pode mais mudar de tamanho (imutável);

- gera um erro em tempo de compilação declarar o tamanho do Array igual em linguagem C:  
`int notas[10];`





# Inicialização dos valores de um Array

- Após a alocação de um Array todas as suas posições recebem **implicitamente** um valor padrão **nulo**.
  - Tipos primitivos
    - **byte**, **short**, **int** recebem 0 (zero).
    - **long** recebe 0L
    - **float** recebe 0.0F
    - **double** recebe 0.0
    - **char** recebe '\u0000'
    - **boolean** recebe **false**;
  - Tipos Construídos
    - as variáveis de referência a objetos recebem **null**.

**Atenção:** arrays de tipos primitivos guardam valores, arrays de objetos guardam referências.



# Inicialização dos valores de um Array

- Após a instanciação de um objeto Array, é possível **atribuir valores** às suas posições utilizando o **índice de acesso** a posição.

```
int notas[] = new int[100];  
notas[0] = 50;  
notas[1] = 70;
```

Após a instanciação do objeto  
todas as posições recebem o  
valor padrão **zero**.

```
String ruas[];  
ruas = new String[100];  
ruas[18] = "Higienópolis";  
ruas[19] = "Tietê";
```

Após a instanciação do objeto  
todas as posições recebem o  
valor padrão **null**.



# Inicialização dos valores de um Array

- Declaração, instanciação e inicialização de um Array **em uma única operação**.

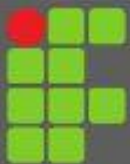
→ **String** nomes[] = { "Alexandre", "Maria", "Jose" };

**double** salarios[] = { 1500, 754.50, 412.00 };

**float** vet[] = {2.5f, 3.4f, 7.8f, 10, 9.1f};

Exemplificando: declarou-se um Array de String chamado nomes, alocado com 3 posições, e por último inicializou-se cada uma de suas posições com o respectivo valor da lista passada.

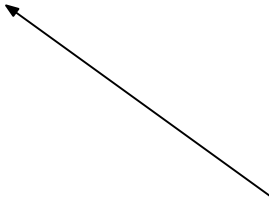
**Atenção:** este procedimento de passar uma lista de parâmetros para iniciar um Array só pode ser feito na declaração da variável.



# Propriedade (tamanho) de um Array

- Todo Array possui uma propriedade chamada **length** do tipo **int**, que representa a quantidade de elementos do Array.

```
double valores[] = new double[1000];  
for(int x= 0; x < valores.length; x++)  
{  
    valores[x] = x* 2;  
}
```



O valor retornado por este atributo é 1000.

**Atenção:** sempre faça o controle de laços que manipulam Arrays usando a propriedade **length** do próprio Array a ser manipulado.

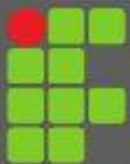


# Arrays

## Exemplo 1

```
public class ExemploArray1 {  
    public static void main( String[] args ){  
        int notas[] = { 100, 97, 88, 46, 79 };  
        float soma = 0;  
        for( int indice = 0; indice < notas.length; indice++ ) {  
            System.out.print( "[" + notas[indice] + " ] " );  
            soma += notas[indice];  
        }  
        System.out.println( "\nA media eh " + ( soma/notas.length ) );  
    }  
}
```

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.exe'. The command prompt shows the command 'C:\Fontes\Aulas>java ExemploArray1' being executed. The output is '[100] [97] [88] [46] [79]' on the first line and 'A media eh 82.0' on the second line. The window has standard Windows controls (minimize, maximize, close) and a scrollbar.

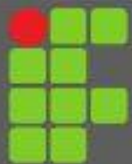


# Arrays

## Exemplo 2

```
public class ExemploArray2 {  
    public static void main( String[] args ) {  
        String nomes[] = { "Joao", "Jose", "da Silva", "Xavier" };  
        System.out.println( "Tamanho criado do vetor = " + nomes.length );  
        for( int indice = 0; indice < nomes.length; indice++ ){  
            System.out.println( "Valor da posicao [" + indice + "] eh: " + nomes[indice]  
        );  
        }  
    }  
}
```

The screenshot shows a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The command prompt displays the output of the Java program: "C:\Fontes\Aulas>java ExemploArray2", followed by "Tamanho criado do vetor = 4", and then four lines of output: "Valor da posicao [0] eh: Joao", "Valor da posicao [1] eh: Jose", "Valor da posicao [2] eh: da Silva", and "Valor da posicao [3] eh: Xavier". The window has standard Windows window controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.



# Malhação Cerebral

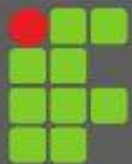
**MC5\_01.** Escreva uma aplicação Java que receba 15 valores, do usuário, armazene em um vetor e depois apresente na tela a quantidade de valores pares.

**MC5\_02.** Escreva uma aplicação Java que receba e armazene em um vetor o salário de 10 pessoas. Ao final mostre todos os salários maiores que a média salarial desse grupo.

**MC5\_03.** Escreva uma aplicação Java para armazenar a idade de 12 pessoas. Apresente a quantidade de pessoas maiores de idade e também uma lista de todas as idades em ordem crescente.







# Arrays

```
f (int aux=1; aux<1000; aux++){  
    System.out.println("Please, God!!");  
}
```

## Arrays multidimensionais

- É possível em Java criar vetores com mais de uma dimensão
  - Arrays com **uma dimensão** são simplesmente chamados de **vetores**.
  - Arrays com **duas dimensões**, ou Arrays bidimensionais, normalmente são chamados de **matrizes** e representam uma tabela de valores.
  - Arrays com **N dimensões**. São necessários N índices para localizar um elemento, necessitando especificar um índice para cada dimensão definida.





# Arrays

```
f (int aux=1; aux<1000; aux++){  
    System.out.println("Please, God!!");  
}
```

Para os Arrays multidimensionais são válidas as mesmas regras dos Arrays unidimensionais

- 3 passos (declaração / instanciação / inicialização)
- quando instanciado, todas suas posições recebem valores nulos
- depois de instanciado é imutável (não altera tamanho)



# Arrays

```
f (int aux=1; aux<1000; aux++){  
    System.out.println("Please, God!!");  
}
```

## Exemplos

- Declaração:  
    double[][] valores;  
    String marcasModelo[][];
- Instanciação:  
    valores = new double[2][4];  
    marcaModelo = new String[3][2];
- Declaração e instanciação (tudo junto)  
    double valores[][] = new double[2][4];  
    String marcaModelo[][] = new String[3][2];



# Arrays

```
f (int aux=1; aux<1000; aux++) {  
    System.out.println("Please, God!!");  
}
```

## Exemplos

- Inicialização:

```
valores[1][2] = 3.50;
```

```
valores[0][3] = 7.28;
```

```
marcaModelo[1][0] = "ford";
```

```
marcaModelo[1][1] = "ecosport";
```

- Declaração, instanciação e inicialização:

```
double valores[][] = { { 10, 6, 32, 7.28 }, { 15, 16, 3.50,  
44 } };
```

```
String marcaModelo[][] = { { "gm", "vectra" },  
    { "ford", "ecosport" },  
    { "fiat", "stilo" } };
```



# Arrays

Exemplo de Array bidimensional:

```
int notas[][] = new int[3][5];
```

Nome do Array. Todas as posições têm o mesmo nome

	Coluna 0	Coluna 1	Coluna 2	Coluna 3	Coluna 4
Linha 0	notas[0][0]	notas[0][1]	notas[0][2]	notas[0][3]	notas[0][4]
Linha 1	notas[1][0]	notas[1][1]	notas[1][2]	notas[1][3]	notas[1][4]
Linha 2	notas[2][0]	notas[2][1]	notas[2][2]	notas[2][3]	notas[2][4]

Índice de acesso  
a linha

Índice de acesso  
a coluna



# Arrays

## Propriedades de um Array multidimensional

- Todo Array multidimensional também possui uma propriedade chamada **length** do tipo **int**, para cada uma das dimensões que possui, representando a **quantidade máxima de elementos** que podem ser armazenados.

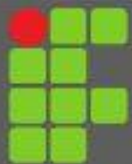
### **Exemplo:**

```
double notas[][] = new double[100][4];
```

```
for( int linha = 0; linha < notas.length; linha++ ){  
    for( int coluna = 0; coluna < notas[linha].length; coluna++ ){  
        notas[linha][coluna] = 10;  
    }  
}
```

Quantidade  
de Linhas

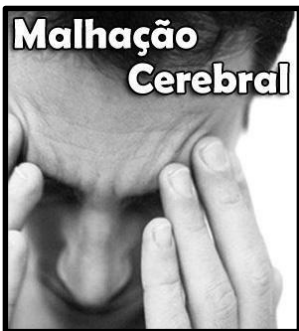
Quantidade de  
Colunas por Linha



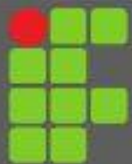
# Malhação Cerebral

**MC5\_04.** Faça uma aplicação em Java que crie e alimente uma matriz 6 x 6, simulando um tabuleiro. Receba somente valores 0s e 1s. Procure e aponte qual a linha com mais valores nulos (0s).

**MC5\_05.** Crie uma aplicação que pergunte ao usuário quantos alunos existem na turma. Depois crie uma matriz de inteiros que suporte armazenar as notas de dois bimestres para todos os alunos da turma. Faça a leitura com o usuário de todas as notas de cada aluno. Depois imprima as notas, a média final de cada aluno ( $\text{Nota 1} + \text{Nota 2} / 2$ ), e sua situação: Aprovado para ( $\text{média} \geq 60$ ) ou reprovou caso contrário. Os valores devem aparecer alinhados como na “tabela”.



Nota 1	Nota 2	Média	Situação
100	50	75	Aprovado
80	60	70	Aprovado
50	40	45	Reprovado



# Malhação Cerebral

**MC5\_06.** Na teoria de sistemas, define-se como elemento minimax de uma matriz o menor elemento da linha a qual encontra o maior elemento da matriz. Construa uma classe Java para encontrar o minimax numa matriz de 5 linhas e 5 colunas. Os dados serão informados pelo usuário.

**MC5\_07.** Considere uma matriz da seguinte forma (esta matriz já deve ser inicializada com esses valores). Os valores iguais a zero correspondem a diagonal principal. Para calcular corretamente esses valores, multiplicar todos os outros elementos de uma linha e dividir o resultado pelo menor elemento desta respectiva linha (excluindo o zero). Ex: Para calcular o elemento  $[0][0] = (12 * 17 * 9 * 21) / 9...$

$$[3][3] = (9 * 15 * 8 * 13) / 8$$

	0	1	2	3	4
0	0	12	17	9	21
1	15	0	21	17	13
2	9	21	0	6	11
3	9	15	8	0	13
4	11	30	7	10	0





# Referências

- SIERRA, Kathy & BATES, Bert. Use a cabeça Java. 2 ed. Rio de Janeiro: Alta Books, 2005.
- DEITEL, H. M.; DEITEL, P. J. Java: como programar. 4 ed. Porto Alegre: Bookman, 2003.
- HORSTMANN, Cay. Big Java. Porto Alegre: Bookman, 2004.
- Material elaborado pela iniciativa JEDI (Java Education and Development Initiative). <http://www.dfjug.org/DFJUG/jedi/index.jsp>
- Material elaborado pelo Professor Alexandre Rômolo Moreira Feitosa da UTFPR de Cornélio Procópio
- Material elaborado pelo Professor Gabriel Canhadas Genvigir da UTFPR de Cornélio Procópio
- Material elaborado pelo Professor Davi Bernardo do IFSC de Gaspar.
- Outros livros / revistas / [WWW](http://www).



# **Programação Orientada a Objetos**

**Professor Vinícius Bortolini**