



UNIVERSIDADE  
FEDERAL DO CEARÁ

TÓPICO  
**07**



# PROGRAMAÇÃO ORIENTADA A OBJETOS

Prof. Marcos Vinicius de Andrade Lima  
E-mail: [marcos.vinicius@ufc.br](mailto:marcos.vinicius@ufc.br)



## Olá!

### Sou Marcos Vinicius

No tópico passado nós aprendemos como controlar o fluxo de execução...

Neste tópico aprenderemos como trabalhar com **arrays**!

“

**Cada sonho que você deixa para trás,  
é um pedaço do seu futuro que deixa  
de existir** (Steve Jobs)



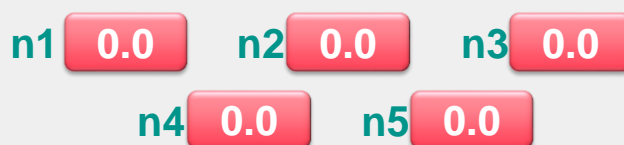
## **Arrays Unidimensionais**

## INTRODUÇÃO

- Imagine que queremos trabalhar com as notas dos alunos. Supondo que cada aluno deva ter cinco notas, podemos criar cinco variáveis do tipo `float` para armazenar as notas de um único aluno.

```
float n1, n2, n3, n4, n5;
```

- Representação na memória:



Prof. Marcos Vinicius – UFC/Russas - POO

5/30

## VAMOS PENSAR UM POUCO...

Quais problemas podem ocorrer utilizando a abordagem anterior para notas?



- Principais problemas:**
- É mais complicado trabalhar com um número grande de variáveis;
- Passar as notas para um método exige a utilização de muitos parâmetros;
- A solução:** utilização de *arrays*!

Prof. Marcos Vinicius – UFC/Russas - POO

6/30

## ARRAYS

- Um **array** é estrutura de dados que define uma **coleção ordenada** de um **número fixo de elementos** de dados **homogêneos** (mesmo tipo).
- Em Java, **arrays são objetos**, portanto, precisamos criá-los antes de começarmos a usá-los.



Prof. Marcos Vinicius – UFC/Russas - POO

7/30

## ARRAYS: SINTAXE

- **Sintaxe para declaração de array unidimensional:**  
`tipo[] nomeDoArray;      tipo nomeDoArray[];`  
  
Onde o tipo pode ser qualquer tipo primitivo (int, char, ...), classe ou interface.
- **Sintaxe para criação de arrays:**  
`nomeDoArray = new tipo[ tamanho ];`

Prof. Marcos Vinicius – UFC/Russas - POO

8/30

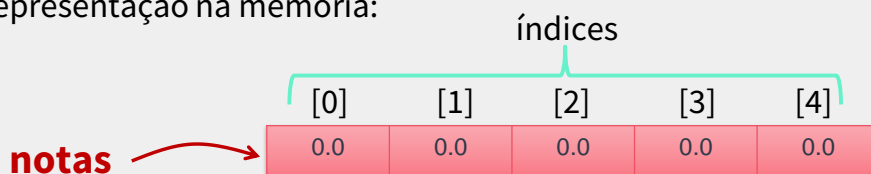


# CUIDADO!!!

Em Java, o tamanho do *array* **não** é fornecido na declaração, somente na criação!

## MUITA CALMA NESTA HORA...

- Para declarar um array de notas, fazemos:  
`float notas[];` ou `float[] notas;`
- Para criar o array de notas, fazemos:  
`notas = new float[5];`
- Também podemos fazer os passos de uma vez só:  
`float notas[] = new float[5];`
- Representação na memória:





# CUIDADO!!!

Em Java, uma vez criado um *array*, seu tamanho **nunca** muda!

## SE LIGA NAS IDEIAS...

- Quando o *array* é criado, todos os elementos são inicializados para o valor **default** do tipo de dado:

```
// 10 inteiros com valor 0
```

```
int a[] = new int[10];
```

```
// 5 booleanos com false
```

```
boolean b[] = new boolean[5];
```

```
// 50 strings com null
```

```
String c[] = new String[50];
```

## OLHA SÓ QUE LEGAL...

- Para saber o tamanho do array, usa-se o atributo **length** do array:

```
// retorna o tamanho do array notas
int tam = notas.length;
```

- Para acessar um elemento do array, fazemos:  
**nomeDoArray[ índice ]**;

## ARRAYS EM AÇÃO...

```
...
float[] notas = new float[5];

notas[0] = 10;

notas[ notas.length-1 ] = 8;

System.out.println("1a nota=" + notas[0]);

System.out.println("última nota=" +
                    notas[notas.length-1]);

...
```

## CRIANDO E INICIALIZANDO ARRAYS

- Podemos inicializar o array já na declaração fazendo:

```
// cria o array com o tamanho equivalente  
// a quantidade de valores  
tipo[] nomeDoArray = {valor1,valor2,... };
```

Prof. Marcos Vinicius – UFC/Russas - POO

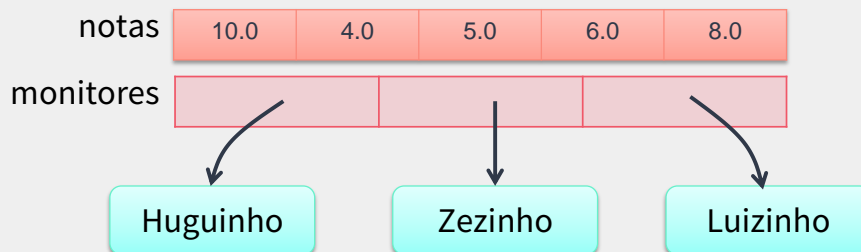
15/30

## CRIANDO E INICIALIZANDO ARRAYS

- Exemplo:**

```
float notas [] = {10, 4, 5, 6, 8 };  
String[] monitores={ "Huguinho","Zezinho","Luizinho" };
```

- Representação na memória:**



Prof. Marcos Vinicius – UFC/Russas - POO

16/30



## IMPORTANTE!

- Pode-se usar os elementos do *array* separadamente:

```
for (int i = 0; i < notas.length; i++ ) {  
    System.out.println("nota=" + notas[i]);  
}
```

- Ou pode-se utilizar o *array* todo, como acontece em operações de atribuição e na passagem de parâmetro:

```
calcularMedia ( notas );  
float[] novasNotas = notas;
```

Prof. Marcos Vinicius – UFC/Russas - POO

17/30

Se houver uma tentativa de **acesso indevido aos índices** do *array*, uma exceção (voadora) será gerada:



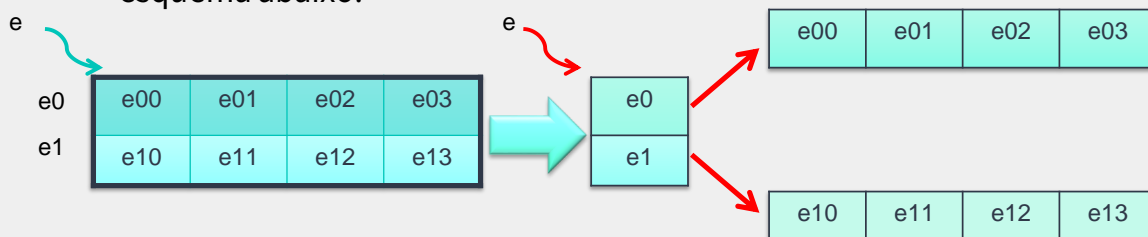
**ArrayIndexOutOfBoundsException**



# Arrays Multidimensionais

## INTRODUÇÃO

- *Arrays* podem ter múltiplas dimensões (*arrays de arrays*).
- **Por exemplo:** sendo “e” um *array* de duas dimensões com dois elementos na primeira e quatro na segunda (2x4), a implementação desse *array* em Java utilizará três objetos do tipo *Array*, conforme o esquema abaixo:



## SINTAXE PARA MULTIDIMENSIONAIS

- Podemos definir *arrays* multidimensionais conforme a sintaxe:

```
tipo [][][]...[] nomeDoArray;
```

ou

```
tipo nomeDoArray [][][]...[] ;
```

ou

```
tipo[][]...[] nomeDoArray[][][]...[];
```

Prof. Marcos Vinicius – UFC/Russas - POO

21/30

## ARRAYS MULTIDIMENSIONAIS EM AÇÃO

```
// Array de duas dimensões
int[][] mXnArray;
```

```
// Array de duas dimensões
int [] umArray[];
```

```
// Criando array 4x5 de inteiros
umArray = new int[4][5];
```

```
// combinando declaração e criação
int [][] outroArray = new int[4][5];
```

Prof. Marcos Vinicius – UFC/Russas - POO

22/30

## INICIALIZANDO ARRAYS MULTIDIMENSIONAIS

- Para inicializar *arrays* multidimensionais:

```
double taxas[][] = {{ 1,2,3 }, {4,5,6}};
```

- Para acessar um elemento do *array* bidimensional, fazemos:

```
nomeDoArray[indice1][indice2]...[indiceN]
```

Prof. Marcos Vinicius – UFC/Russas - POO

23/30

## PAY ATTENTION!

```
int [][] arr = new int [3][4];
arr[1][2] = 5;
arr[2][3] = 7;
System.out.println ( a[1][2] );
System.out.println ( a[2][3] );
```



	0	1	2	3
0	0	0	0	0
1	0	0	5	0
2	0	0	0	7

arr[1][2] arr[2][3]

Prof. Marcos Vinicius – UFC/Russas - POO

24/30

## VEJA BEM...

- Num *array* multidimensional, para saber o tamanho da primeira dimensão, usa-se o atributo `length`:  
`int [][] meuArray = new int [4][5];`  
`meuArray.length;` → retorna o tamanho da 1ª Dimensão = 4

- Para varrer um *array* bidimensional, precisamos utilizar dois laços de repetição:

```
int arr[][] = new int[4][5];
for (int i = 0; i < arr.length; i++ )
    for (int j = 0; j < arr[i].length; j++ )
        System.out.println( arr[i][j] );
```

tamanho da 1ª dimensão

tamanho da 2ª dimensão

Prof. Marcos Vinicius – UFC/Russas - POO

25/30

## IMPORTANTE!

- Na criação de *arrays* multidimensionais com o operador `new`, a profundidade dos *arrays* mais internos pode ser omitida:

```
double matrix[][] = new double[3][];
```

- A inicialização poderia ser assim:

```
for (int i = 0; i < matrix.length; ++i)
    matrix[ i ] = new double[ i+1 ];
```

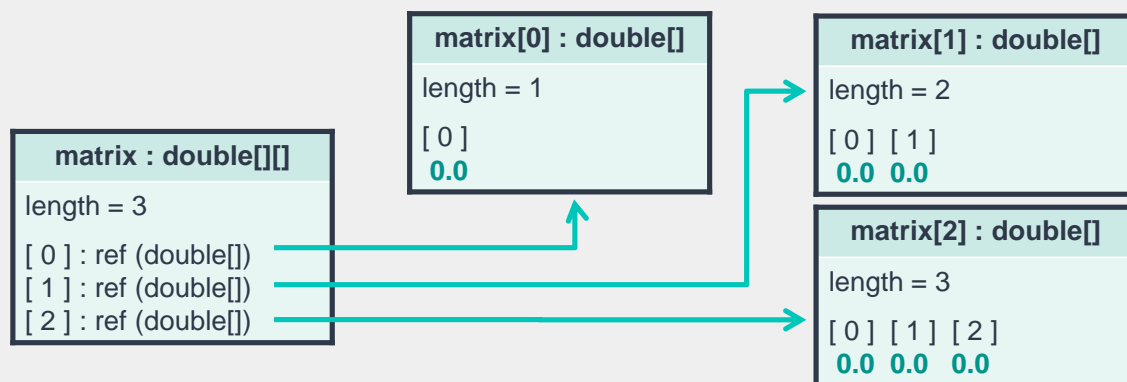
É muita  
informação!



Prof. Marcos Vinicius – UFC/Russas - POO

26/30

## REPRESENTAÇÃO NA MEMÓRIA




Prof. Marcos Vinicius – UFC/Russas - POO

27/30

Vamos **brincar** um  
pouco?





Crie uma **matriz 10x10** que represente **Robopólis**.  
Em **cada elemento** dessa matriz é possível adicionar  
uma “carga” ou “bomba”.

Ao todo temos **20 cargas e 5 bombas** espalhadas  
**aleatoriamente** por Robópolis.

Dica: pesquisa sobre **java.util.Random**



**Obrigado!**  
**Mais alguma dúvida?**

Acesse o **AME** para mais informações e  
treinamento do **NERDS!**

<http://ame2.russas.ufc.br>

