



# String e Arrays

# M

## matriz 2D e matriz multidimensional

Uma matriz é um grupo de variáveis digitadas semelhantes que são referidas por um nome comum. Matrizes de qualquer tipo podem ser criadas e podem ter uma ou mais dimensões. Um elemento específico em uma matriz é acessado por seu índice. A matriz é um tipo simples de estrutura de dados que pode armazenar objetos ou variáveis primitivas. Por exemplo, imagine se você tivesse que armazenar o resultado de seis assuntos, podemos fazê-lo usando uma matriz. Para criar um valor de matriz em Java, use a nova palavra-chave, assim como você cria um objeto.


## Definindo e construindo uma matriz unidimensional

```
int resultadoArray[] = new int[6];
```

Tipo

Nome do Array

Tamanho do Array

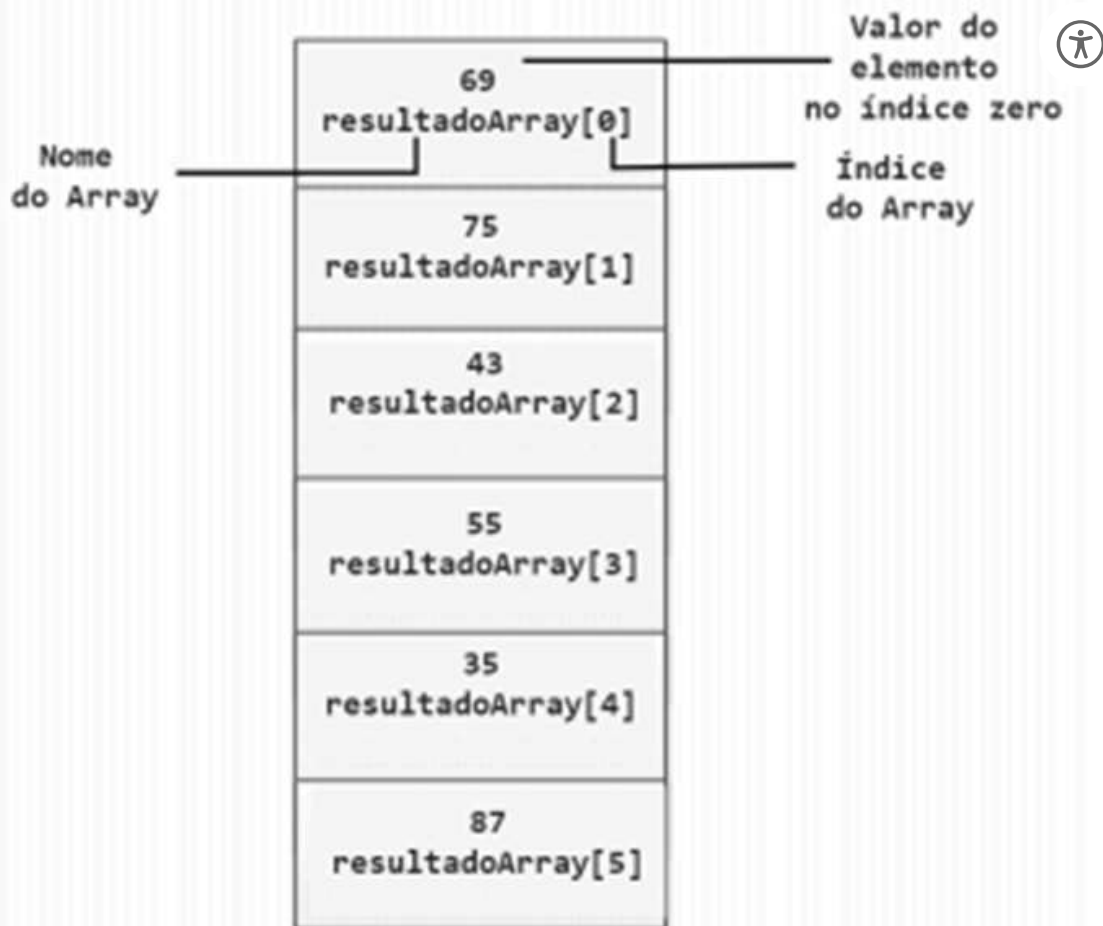
Aqui, `type` especifica o tipo de variáveis (`int`, `boolean`, `char`, `float` etc.) que estão sendo armazenadas, `size` especifica o número de elementos na matriz e `arrayna`  é o nome da variável que é a referência à matriz (array). O tamanho da matriz deve ser especificado ao criar uma matriz. Se você estiver criando um `int []`, por exemplo, deverá especificar quantos valores `int` deseja que ele mantenha (na instrução acima `resultadoArray []` está tendo tamanho 6 `int`). Depois que uma matriz é criada, ela nunca pode crescer ou encolher.

## Inicializando matriz

Você pode inicializar um elemento específico na matriz especificando seu índice entre colchetes. Todos os índices da matriz começam em zero.

```
resultadoArray [0] = 69;
```

Isso inicializa o primeiro elemento (índice zero) de `resultadoArray []` com o valor inteiro 69. Os elementos da matriz podem ser inicializados / acessados em qualquer ordem. Na memória, ele criará uma estrutura semelhante à figura abaixo.



## Literais de matriz

O literal nulo usado para representar a ausência de um objeto também pode ser usado para representar a ausência de uma matriz. Por exemplo:

```
String [] nome = null;
```

Além do literal nulo, Java também define uma sintaxe especial que permite especificar valores de matriz literalmente em seus programas. Essa sintaxe pode ser usada apenas ao declarar uma variável do tipo de matriz. Combina a criação do objeto de matriz com a inicialização dos elementos da matriz:

```
String [] diasDaSemana = {"Domingo", "Segunda-feira", "Terça-feira",
```

“Quarta-feira”, “Quinta-feira”, “Sexta-feira”, “Sábado”};



Isso cria uma matriz que contém os sete elementos da sequência

que representam os dias da semana entre os chaves. Observe que não usamos a nova palavra-chave ou especificamos o tipo da matriz nessa sintaxe literal da matriz. O tipo está implícito na declaração da variável da qual o inicializador faz parte. Além disso, o comprimento da matriz não é especificado explicitamente com esta sintaxe; é determinado implicitamente contando o número de elementos listados entre as chaves.

Vamos ver um exemplo de programa Java para entender melhor esse conceito. Este programa ajudará a entender a inicialização e o acesso a elementos específicos da matriz.



```
package br.com.java.aula;
import java.util.Arrays;
public class ResultListDemo {
    public static void main(String[] args) {
        //Array Declaration
        int resultadoArray[] = new int[6];
        // Inicialização de matriz
        resultadoArray[0] = 69;
        resultadoArray[1] = 75;
        resultadoArray[2] = 43;
        resultadoArray[3] = 55;
        resultadoArray[4] = 35;
        resultadoArray[5] = 87;
        //Acesso a elementos de matriz
        System.out.println("Marcas do Primeiro Sujeito -
        "+resultadoArray[0]);

        System.out.println("Marcas do Segundo Sujeito -
        "+resultadoArray[1]);

        System.out.println("Marcas do Terceiro sujeito -
        "+resultadoArray[2]);

        System.out.println("Marcas do Quarto sujeito -
        "+resultadoArray[3]);

        System.out.println("Marcas do Quinto sujeito -
        "+resultadoArray[4]);

        System.out.println("Marcas do Sexto sujeito -
        "+resultadoArray[5]);
    }
}
```



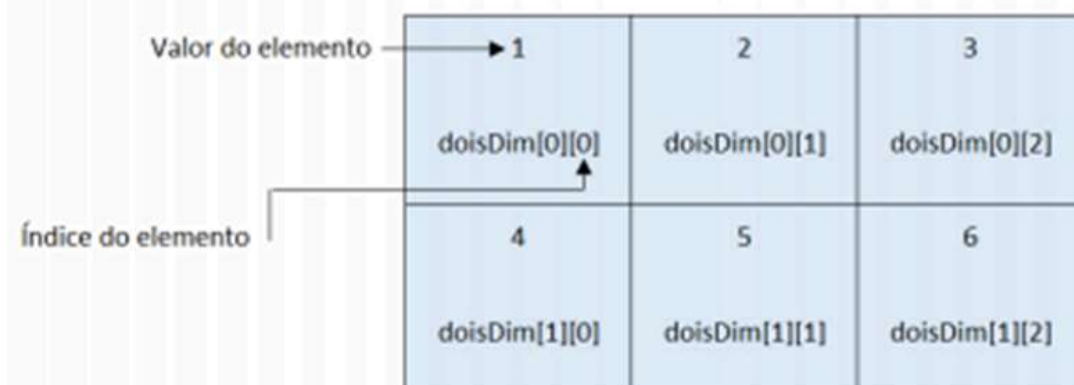
```
<terminated> ResultListDemo [Java Application] C:\Program Files
Marcas do Primeiro Sujeito- 69
Marcas do Segundo Sujeito - 75
Marcas do Terceiro Sujeito - 43
Marcas do Quarto Sujeito - 55
Marcas do Quinta Sujeito - 35
Marcas do Sexto Sujeito - 87
```

Sintaxe alternativa para declaração, inicialização da matriz na mesma instrução `int [] resultadoArray = {69,75,43,55,35,87};`

## Matrizes multidimensionais

Em Java, matrizes multidimensionais são na verdade matrizes de matrizes. Essas, como você pode esperar, parecem e agem como matrizes multidimensionais regulares. No entanto, como você verá, existem algumas diferenças sutis. Para declarar uma variável de matriz multidimensional, especifique cada índice adicional usando outro conjunto de colchetes. Por exemplo, o seguinte exemplo declara uma variável de matriz bidimensional chamada `doisDim`. Isso criará uma matriz do tamanho 2x3 na memória.

```
int doisDim [] [] = new int [2] [3];
```

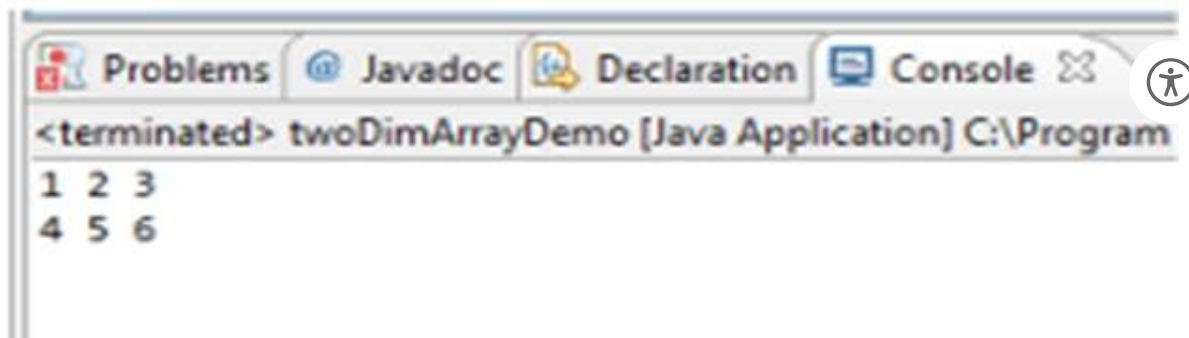


Vamos dar uma olhada no programa abaixo para entender a matriz bidimensional

```
package br.com.java.aula;
public class doisDimArrayDemo {
    public static void main(String[] args) {
        intdoisDim[][] = newint[2][3];
        doisDim[0][0] = 1;
        doisDim[0][1] = 2;
        doisDim[0][2] = 3;
        doisDim[1][0] = 4;
        doisDim[1][1] = 5;
        doisDim[1][2] = 6;

        System.out.println(doisDim[0][0] + " " + doisDim[0][1] + "
        "+doisDim[0][2]);
        System.out.println(doisDim[1][0] + " " + doisDim[1][1] + " " +
            doisDim[1][2]);
    }
}
```

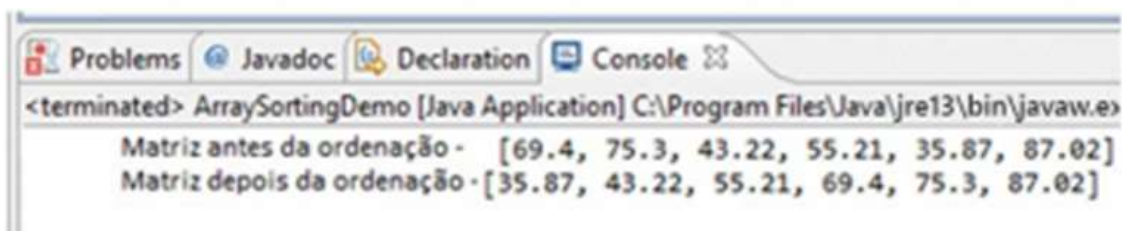
**Resultado:**




## Classe auxiliar incorporada (java.util.Arrays) para manipulação de matrizes

Java fornece classe auxiliar muito importante (java.util.Arrays) para manipulação de array. Essa classe possui muitos métodos utilitários, como classificação de matriz, impressão de valores de todos os elementos da matriz, pesquisa de um elemento, cópia de uma matriz em outra matriz, etc. Vamos ver um exemplo de programa para entender essa classe para melhor programação. No programa abaixo, a matriz flutuante foi declarada. Estamos imprimindo os elementos da matriz antes da classificação e após a classificação.

```
package br.com.java.aula;
import java.util.Arrays;
public class ArrayOrdenacaoDemo {
    public static void main(String[] args) {
        //Declarando matriz de elementos flutuantes
        float[] resultadoArray = {
            69.4 f,
            75.3 f,
            43.22 f,
            55.21 f,
            35.87 f,
            87.02 f
        };
        System.out.println("Matriz antes da ordenação - "
            +Arrays.toString(resultadoArray));
        //a linha abaixo ordenará a matriz em ordem crescente
        Arrays.sort(resultadoArray); System.out.println("Matriz depois da ordenação - "
            +Arrays.toString(resultadoArray));
    }
}
```





Semelhante ao “java.util.Arrays”, a classe System também possui uma funcionalidade de copiar dados de maneira eficiente de uma matriz para outra  Sintaxe como abaixo:

```
public static void arraycopiarDados(Object src, int srcPos, Object dest, int destPos, int length)
```

Os dois argumentos do objeto especificam a matriz da qual copiar e a matriz para a qual copiar. Os três argumentos int especificam a posição inicial na matriz de origem, a posição inicial na matriz de destino e o número de elementos da matriz a serem copiados.

### **Pontos importantes**

- Você receberá “ArrayIndexOutOfBoundsException” se tentar acessar uma matriz com um índice ilegal, ou seja, com um número negativo ou com um número maior ou igual ao seu tamanho.
- Matrizes são amplamente usadas com loops (para loops, enquanto loops).

### **Atividade Extra**

Vídeo: Aprendendo sobre Matriz. Link para o vídeo: <https://www.youtube.com/watch?v=gTJNb1hSY8o>



## Referências Bibliográficas

BARNES, D. J. KOLLING, M. **Programação orientada a objetos com java: uma introdução prática usando o bluej**. 4.ed. Pearson: 2009.

FELIX, R. (Org.). **Programação orientada a objetos**. Pearson: 2017.

MEDEIROS, L. F. de. **Banco de dados: princípios e prática**. Intersaberes: 2013;

ORACLE. Java Documentation, 2021. **Documentação oficial da plataforma Java**. Disponível em: < <https://docs.oracle.com/en/java/> >.

**Ir para exercício**