**Aula XIV - Desenvolvimento Web**

**Array (vetores)**

**Array List**

**Matrizes**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Fontes:***

[*https://distancia.qi.edu.br/mod/book/view.php?id=18698*](https://distancia.qi.edu.br/mod/book/view.php?id=18698)

[*https://www.devmedia.com.br/explorando-a-classe-arraylist-no-java/24298*](https://www.devmedia.com.br/explorando-a-classe-arraylist-no-java/24298)

[*https://www.w3schools.com/java/java\_arraylist.asp*](https://www.w3schools.com/java/java_arraylist.asp)

*Livro de Lógica de Programação - Curso Técnico em Informática - QI Faculdade & Escola Técnica*

[*https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html*](https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html)

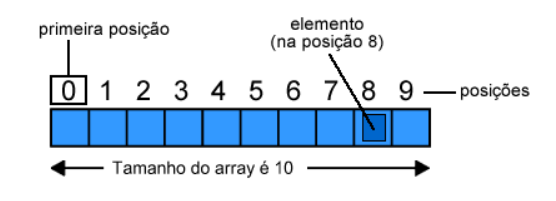
[*http://www.javaprogressivo.net*](http://www.javaprogressivo.net/)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Array**

Um array é um objeto capaz de armazenar um número fixo de valores de um único tipo. A quantidade de elementos que armazena é estabelecida quando ele é criado e após a criação seu comprimento de mantém fixo (ORACLE, 2012).

O uso de arrays é indicado quando temos uma certa quantidade de elementos que serão manipulados/armazenados. Por exemplo: as notas de um aluno, as questões de uma prova, as alternativas das questões, um conjunto de números, e assim por diante.

Array de 10 elementos

|  |
| --- |
| **Para que serve um Array?**  **Um array é uma sequência não ordenada de dados. Os elementos residem em um lugar separado na memória, e seu acesso é feito por meio de um índice na primeira posição de cada elemento** |

**Criando um Array**

A declaração de um array utiliza um conjunto de colchetes vazio ao lado do tipo. O tipo pode ser tanto primitivo quanto de referência.

* **Tipos primitivos: são os tipos de dados básicos da linguagem, com eles podemos criar outros tipos. Exemplos: int, char, byte, boolean, etc.**
* **Tipos de referência: são as classes do sistema, tanto as do Java quanto as criadas pelo programador. Exemplos: String, Pessoa, etc.**

**Declarações do Array**

**1ª forma: Declaração e instância em linhas separadas**

O exemplo abaixo primeiro declara o **Array, utilizando o tipo desejado seguido de colchetes vazios e o nome que se deseja atribuir ao array**. Em outra linha, o comando **new é responsável pela criação do objeto na memória.** No exemplo, seria criado um array com capacidade para 5 números inteiros, organizados em índices de 0 até 4. Cria-se um array vazio, pronto para receber elementos.



**2ª forma: Declaração e instância na mesma linha**

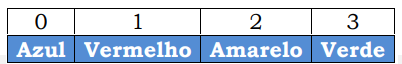
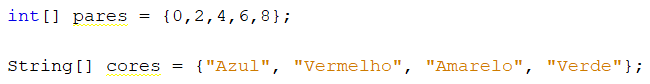
O exemplo abaixo é uma forma abreviada de declarar e instanciar o array também vazio.



**3ª forma: Declaração com atribuição de valores**

O exemplo abaixo demonstra como criar um array já com elementos armazenados.

O primeiro exemplo criaria um array com 5 elementos, conforme a imagem abaixo. O segundo exemplo criaria um array chamado “cores”



**Armazenando valores no array**

Após a declaração e instância do array, ele está pronto para armazenar valores. Para isso, é necessário indicar em qual posição queremos armazenar o valor. Esta posição pode ser indicada por um número inteiro definido ou por uma variável do tipo inteiro que contenha um valor de índice válido.

**Vamos utilizar como exemplo o seguinte array:**



Vamos armazenar um número qualquer na primeira posição do array:



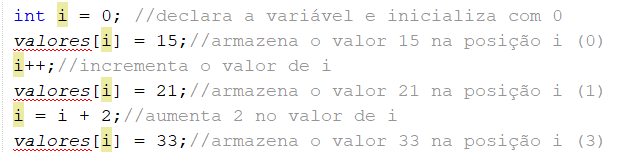
**Ocupando a posição 0**, as demais ficam à disposição para armazenar outros valores. Caso você utilize novamente a posição 0 para armazenar um valor, ele substituirá o valor antigo pelo novo. **Observe o exemplo:**



**Preenchendo outras posições:**

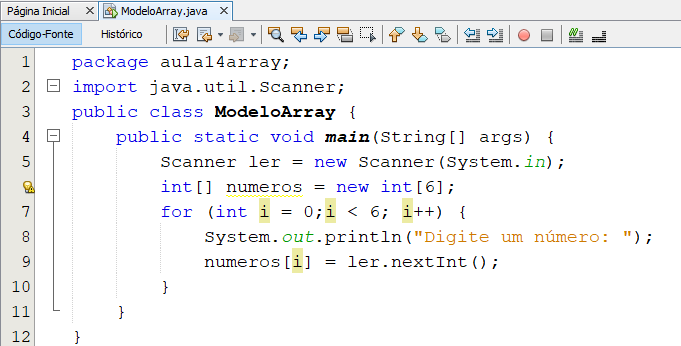


A posição pode ser indicada por uma variável inteira. Observe o exemplo (considere que o array está vazio):

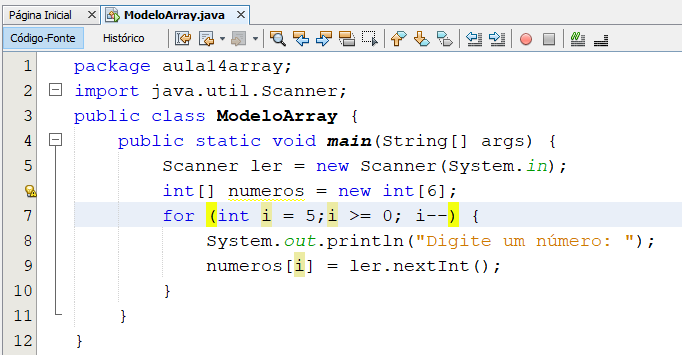


**Preenchendo um array com valores informados pelo usuário:**

Para preencher um array com valores informados pelo usuário podemos utilizar um laço de repetição. Observe o exemplo, que preencherá um array criado na classe Main com valores digitados no terminal:



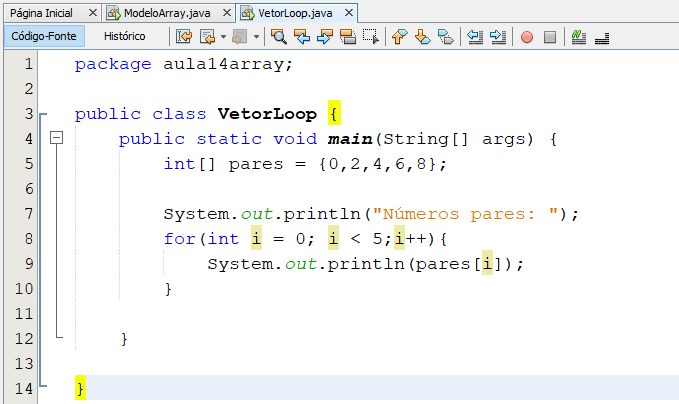
**Observe que na linha 9 foi utilizada a variável “i”** para indicar a posição que receberá o elemento digitado pelo usuário. **Assim, a cada volta do loop, o número digitado pelo usuário é armazenado numa posição diferente, partindo da primeira (0). Também poderíamos preencher o array de trás para frente, invertendo o for:**



**Exibindo valores armazenados no array**

Para exibir os valores de um array podemos utilizar um laço de repetição, imprimindo uma posição de cada vez no terminal, ou utilizar o método toString da classe Arrays.

**Utilizando um loop para exibir o conteúdo de um array**

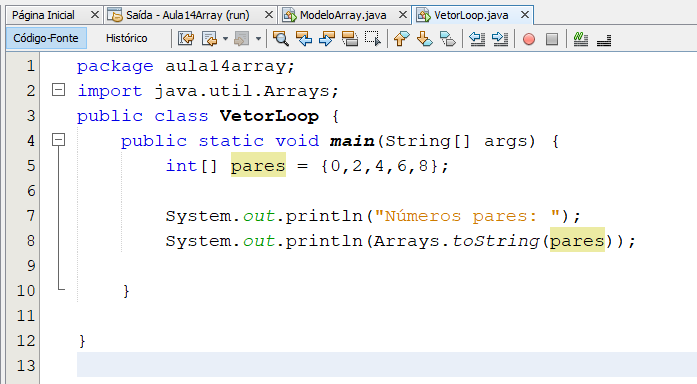


**Utilizando a classe Arrays para exibir o conteúdo de um array**

A classe Arrays deve ser importada no início da classe. **O método toString** já possui um formato padrão de exibição, que apresenta os elementos entre colchetes separados por ponto e vírgula.

|  |
| --- |
| **O que é o método toString?**  **Classe Object – método toString()**  **O método toString() retorna uma String que representa o objeto que está chamando o método. Isso é útil para o propósito de log e debug. Vamos descrever o que esse método está fazendo**. |

**Observe o exemplo:**



**O atributo lenght**

**O atributo lenght armazena a quantidade de elementos que um array pode armazenar** (considera inclusive as posições vazias). Observe seu uso na linha 10 do exemplo abaixo.

