

The logo for Oracle Academy is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

# ORACLE

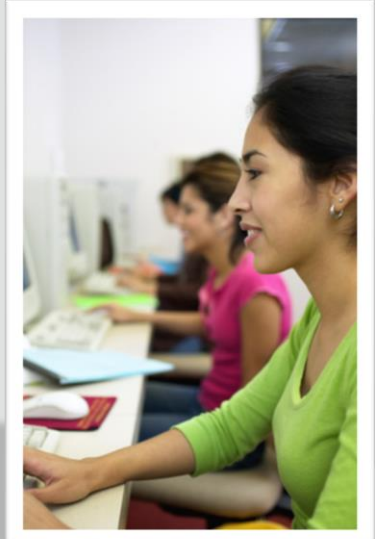
## Academy

# Java Foundations

**8-1**

**Matrizes Unidimensionais**

**ORACLE**  
Academy



Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

# Objetivos

- Esta lição abrange os seguintes objetivos:
  - Criar e inicializar matrizes unidimensionais
  - Modificar um elemento de matriz
  - Percorrer uma matriz unidimensional usando um loop for
  - Identificar a causa de uma `ArrayIndexOutOfBoundsException`



**ORACLE**  
Academy

JFo 8-1  
Expressões Booleanas e Construções  
if/else

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

3

# Uma Variável Pode Conter Mais de Um Valor?

- Até o momento, usamos muitos tipos de variáveis, mas cada uma armazena um único valor por vez:
  - int, String ou double
- Veja um exemplo de uma variável String, rockBand, que pode conter qualquer valor: Joe, Paul, Ed, Rob
  - Como só existem quatro valores possíveis, não é tão difícil alterar o valor da variável manualmente

```
String rockBand = "Joe";  
String rockBand = "Paul";  
String rockBand = "Ed";  
String rockBand = "Rob";
```

## Número de Variáveis Obrigatórias

- Mas terá vezes em que você precisará reter mais de um valor em uma variável
- E se você quisesse reservar uma variável para cada uma das músicas RockBand? (Haveria 300 variáveis para cada música!)
- No entanto, pode ser demorado e monótono criar centenas de variáveis

```
String rockBandSong1 = "Rainy day";  
String rockBandSong2 = "Forever";  
String rockBandSong3 = "Something about you";  
String rockBandSong4 = "Love you always";
```

.....

# As Matrizes Podem Ser uma Solução

- Em Java, uma matriz é um recipiente indexado que contém um conjunto de valores de um único tipo
- As matrizes permitem que você crie um único identificador para organizar muitos itens do mesmo tipo de dados

0	1	2	3	4	5	6
27	12	82	70	54	1	30

Índices

Elementos

# As Matrizes Podem Ser uma Solução

- Cada item em uma matriz é chamado de elemento
- As matrizes fazem com que seja simples e fácil armazenar e acessar um grande número de valores

0	1	2	3	4	5	6
27	12	82	70	54	1	30

Índices

Elementos

## As Matrizes São Acessadas pelo Respetivo Índice

- Você pode acessar cada elemento em uma matriz por seu índice numérico
- O índice do primeiro elemento é 0
- Uma matriz com 10 elementos tem de 0 a 9 índices



A matriz é um recipiente que armazena um conjunto de valores de `String`, um conjunto de valores `int`, um conjunto de valores `double` e assim por diante.

Os elementos (itens) da matriz são acessados por meio de um índice numérico. Usando esse índice, você pode definir ou obter um valor de um elemento específico.



# Tipos de Dados das Matrizes

- As matrizes podem ter qualquer tipo de dados, mas todos os elementos compartilham o mesmo tipo de dados, como:

- Primitivo:

- Exemplo: matriz de tipos int

27	12	82	70	54	1	30
----	----	----	----	----	---	----

- Objetos predefinidos:

- Exemplo: matriz de Strings

Dom.	Seg.	Ter.	Qua.	Qui.	Sex.	Sáb.
------	------	------	------	------	------	------

## Tipos de Dados das Matrizes

- As matrizes podem ter qualquer tipo de dados, mas todos os elementos compartilham o mesmo tipo de dados, como:
  - Objetos definidos pelo programador:
    - (como instâncias de uma classe que você cria)
    - Exemplo: matriz de objetos da classe Student

Student1	Student2	Student3	Student4	Student5
----------	----------	----------	----------	----------

## Declarando uma Matriz

- As matrizes, como todas as variáveis, devem ser declaradas antes de serem usadas
- Você pode declarar uma matriz usando esta sintaxe:

```
tipo[] identificador da matriz;
```

- Observe a notação de colchete [] depois do tipo de dados

## Declarando uma Matriz de Valores de Temperatura

- Suponha que você queira armazenar leituras diferentes de temperatura em uma matriz
- Você pode declarar uma matriz da seguinte maneira:

```
double[] temperature;
```

Tipo de dados baseado nos itens que você deseja armazenar na matriz

Subscrito

Nome da matriz

## Declarando uma Matriz: dois métodos

- Você pode declarar uma matriz de duas maneiras:

```
1. int[] prime;  
2. int prime[];
```

- As duas sintaxes são equivalentes
- O primeiro formato geralmente é o mais legível e o que deve ser usado

# É Suficiente Declarar uma Matriz?

- Declarar uma matriz não é suficiente para começar a usá-la em seu programa
- Antes de usar uma matriz, você precisa informar ao Java para criar espaço na memória para os elementos que ela armazenará

# É Suficiente Declarar uma Matriz?

- Use a seguinte sintaxe:

```
tipo_de_dados[] nome_da_variável = new tipo_de_dados[tamanho];  
nome_da_variável[índice] = valor; //repita para cada elemento
```

- O valor do tamanho determina o número de itens que uma matriz pode armazenar
- As matrizes não podem crescer além desse tamanho

## Criando uma Matriz

- Por exemplo, se quiser criar uma matriz para conter 100 números inteiros, você pode fazer o seguinte:

```
int[] myIntArray;  
myIntArray = new int[100];
```

- Se preferir, pode executar estas duas linhas em uma única etapa:

```
int[] myIntArray = new int[100];
```



# O que os Trechos de Código Fazem?

```
int[] ages = new int[3];  
ages[0] = 19;  
ages[1] = 42;  
ages[2] = 92;
```

1

```
String[] names = new String[3];  
names[0] = "Mary";  
names[1] = "Bob";  
names[2] = "Carlos";
```

2

Nome da Variável

Índice

Valor

**ORACLE**  
Academy

JFo 8-1  
Expressões Booleanas e Construções  
if/else

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

17

O primeiro trecho de código cria uma matriz de valores inteiros de `idades` com três elementos.

O segundo trecho de código cria uma matriz `String` de `nomes` com três elementos.

## E Sobre a Declaração e a Inicialização de uma Matriz em uma Única Etapa?

- Você também pode declarar e inicializar a matriz em uma única etapa com valores conhecidos:

```
tipo[] identificador da matriz = {lista de valores  
separados por vírgulas};
```

- Por exemplo, declare matrizes dos tipos String e int:

```
String[] names = {"Mary", "Bob", "Carlos"}; }  
int[] ages = {25, 27, 48};
```

Declaração  
e  
inicialização  
em uma  
etapa

## E Sobre a Declaração e a Inicialização de uma Matriz em uma Única Etapa?

- Observe que esse método não especifica o tamanho
- É atribuído um tamanho com base no número de elementos entre as chaves ( { } )

```
String[] names = {"Mary", "Bob", "Carlos"}; }  
int[] ages = {25, 27, 48};
```

Declaração  
e  
inicialização  
em uma  
etapa

## Acessando Elementos da Matriz

- As matrizes são estruturas sequenciais, o que significa que os itens são armazenados um após o outro em uma matriz
- Você pode acessar um elemento individual de uma matriz usando uma notação de chaves
- Por exemplo, veja a seguir como você obtém valores da matriz ages:

```
int[] ages = {25, 27, 48};  
int myAge = ages[0];  
int yourAge = ages[1];  
System.out.println("Minha idade é " + ages[0]);
```

## Como Você Define o Valor de um Elemento de uma Matriz?

- Você pode definir valores para os elementos da matriz da seguinte forma:

```
String[] names = {"Mary", "Bob", "Carlos"};  
names[0] = "Gary";  
names[1] = "Rob";
```

- Depois que você definir os valores para os elementos nos índices 0 e 1, a matriz names terá esta aparência:

0	1	2
Gary	Rob	Carlos
names[0]	names[1]	names[2]

## Exercício 1

- Você consegue identificar os três componentes da declaração de uma matriz para cada uma dessas matrizes de tipos de dados primitivos?
  - Tipo de Dados
  - Nome
  - Tamanho

```
int[] myArray;  
  
myArray = new int[20];  
  
char[] sentence = new char[100];  
  
double[] teamPoints = new double[5];
```

# Inicialização Padrão de Matrizes

- Quando as matrizes são declaradas sem terem sido inicializadas, é atribuído aos elementos o valor padrão associado ao tipo de dados
- Veja um exemplo a seguir:

```
int[] myArray = new int[5];
```

Valores padrão dos  
elementos dessa matriz

Índice:	0	1	2	3	4
Valor:	0	0	0	0	0

Veja a seguir as inicializações padrão das matrizes de tipos de dados diferentes:

- Valores numéricos: `int`: 0, `double`: 0.0)
- Valores booleanos para `false`
- valores `char` para `'\u0000'` (unicode para caractere em branco)
- Tipos de objetos como `String` para `null`

## Como Você Acessa o Comprimento de uma Matriz?

- Até o momento, você criou uma matriz com determinado número de elementos
- Depois da criação, você pode alterar o comprimento de uma matriz. Elas não podem crescer além desse tamanho
- Você pode acessar o tamanho de qualquer matriz usando a propriedade `length` dela

```
int primes[] = {2, 3, 5, 7, 11, 13, 17};  
System.out.println("Array length: " + primes.length);  
  
//imprime 7
```



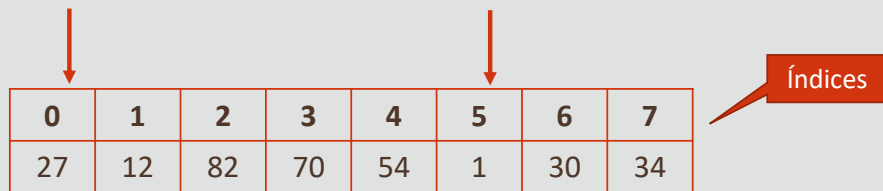
# Índices e Comprimento de uma Matriz

- Por exemplo, o trecho de código a seguir exibe o tamanho da matriz ages:

```
int ages[] = {27, 12, 82, 70, 54, 1, 30, 34};  
System.out.println(ages.length); //imprime 8
```

Primeiro índice

Elemento no índice 5



0	1	2	3	4	5	6	7
27	12	82	70	54	1	30	34

← O comprimento da matriz é 8 →  
(ages.length)

## Exercício 2

- Crie um novo projeto e adicione o arquivo `ArrayEx1.java` a ele
- Examine `ArrayEx1.java`
- Modifique o programa para implementação...
  - Declare uma matriz unidimensional denominada `score` do tipo `int` que possa conter 9 valores
  - Declare e inicialize uma matriz de bytes unidimensional denominada `values` de tamanho 10 para que todas as entradas contenham 1
  - Remova a marca de comentário das duas linhas que estão assinaladas para serem ignoradas e, em seguida, resolva os erros de sintaxe

## Percorrendo uma Matriz

- Iterar ou percorrer uma matriz significa processar cada elemento da matriz pelo número do índice
- Você pode acessar cada elemento de uma matriz para...
  - Imprimir os elementos
  - Procurar um elemento
  - Inicializar os elementos de uma matriz com o mesmo valor

## Usando um Loop for para Percorrer Matrizes

- Você pode usar um loop for para percorrer matrizes
- O loop for permite que você itere facilmente em um intervalo desconhecido
- Você pode visitar cada elemento da matriz usando a propriedade length da matriz na condição de iteração

```
int[] array = { -20, 19, 1, 5, -1, 27, 19, 5 } ;  
int min = array[0]; // initialize the current minimum  
for (int index=0; index < array.length; index++ )  
    if (array[index] < min)  
        min = array[index] ;  
System.out.println("The minimum of this array is: " + min);
```

## Como Você Imprime os Valores de uma Matriz names?

- Considere a matriz de Strings names:

```
String names[] = {"Tom", "David", "Mike"};
```

- Percorra a matriz names usando o loop for:

```
for (int idx = 0; idx < names.length; idx++){  
    System.out.println(names[idx]);  
} //fim for
```

expressão booleana

Contador usado como índice da matriz

## Usando um Loop for-each para Percorrer uma Matriz

- Você pode usar um loop for-each, uma alternativa ao uso do loop for, para iteração em uma matriz
- O loop for-each...
  - Funciona da mesma maneira que o loop for, mas é implementado de maneira mais simples
  - Também é denominado loop for aprimorado

# Usando um Loop for-each para Percorrer uma Matriz

- Sintaxe:

```
for (<tipo> <variável de iteração> : <nome da matriz>) {  
    <bloco_de_código a ser executado para cada elemento da  
    matriz>  
} fim for
```

## Como Você Imprime os Valores de uma Matriz names Usando um Loop for-each?

- Veja a seguir um exemplo de como percorrer a matriz names usando um loop for-each:

```
for(String name: names){  
    System.out.println(name);  
} //fim for
```



## Como Você Imprime os Valores de uma Matriz names Usando um Loop for-each?

- Para cada iteração do loop, o próximo elemento na matriz é recuperado e armazenado em uma variável de iteração
- O tipo deve ser o mesmo que o dos elementos armazenados na coleção

# Loop for-each x Loop for

- Loop for-each

```
for(String name: names){  
    System.out.println(name);  
}//fim for
```

- Loop for

```
for (int idx = 0; idx < names.length; idx++){  
    System.out.println(names[idx]);  
}//fim for
```

- A saída dos dois loops é a mesma

# Processando uma Matriz de Strings

O loop acessa  
cada elemento,  
um após o outro

A matriz names de tipos  
String

George

Jill

Xinyi

Ravi

```
for(String name : names ) {  
    System.out.println("O nome é " + name);  
} //fim for
```

Cada iteração  
retorna o elemento  
seguinte da matriz

## • Saída:

O nome é George  
O nome é Jill  
O nome é Xinyi  
O nome é Ravi

**ORACLE**  
Academy

JFo 8-1  
Expressões Booleanas e Construções  
if/else

Copyright © 2022, Oracle e/ou suas empresas afiliadas. Oracle, Java e MySQL são marcas comerciais registradas da Oracle Corporation e/ou de suas empresas afiliadas. Outros nomes podem ser marcas comerciais de seus respectivos proprietários.

35

O exemplo do slide mostra quatro elementos na matriz names. O bloco de código é executado quatro vezes, e a variável name contém um elemento de matriz distinto em cada vez.

# Colocando Tudo Junto

- Vamos dar uma olhada em um exemplo em que você precisa...
  - Inserir as pontuações de 10 estudantes usando um objeto Scanner
  - Exibir as pontuações que inseriu
  - Calcular a média das pontuações que inseriu

# Vamos Calcular a Pontuação Média

```
public class StudentScores {  
    public static void main(String args[]) {  
        double scores[] = new double[10];  
        double sum = 0.0, avg = 0.0;  
        Scanner keyboard = new Scanner(System.in);  
  
        System.out.println("Insira as pontuações de 10 estudantes");  
        for(int i = 0; i < scores.length; i++) {  
            scores[i] = keyboard.nextInt();  
        }  
        System.out.println("Exiba as pontuações de 10 estudantes");  
        for(int i = 0; i < scores.length; i++) {  
            System.out.println(scores[i]);  
        }  
        for(int i = 0; i < scores.length; i++) {  
            sum = sum + scores[i];  
            avg = sum / scores.length;  
        }  
        System.out.println("A pontuação média da classe: " + avg);  
    }  
}
```

No exemplo do slide, o programa solicita que você insira uma pontuação para 10 estudantes de uma classe usando a classe `Scanner`. As pontuações são armazenadas em uma matriz denominada `scores`. Em seguida, as pontuações armazenadas na matriz são impressas no console percorrendo a matriz com um loop `for`. Usando outro loop `for`, a soma das 10 pontuações é calculada. A pontuação média é obtida dividindo a soma por 10 (ou seja, o número total de pontuações). Por fim, a pontuação média da classe é exibida.

## Exercício 3

- Adicione o arquivo `ComputeAvg.java` ao projeto que você criou para o exercício 2
- Examine `ComputeAvg.java`
- Modifique o programa para implementação...
  - Em determinada classe, existem cinco testes, cada um valendo 100 pontos
  - Insira cinco pontuações de teste do console
  - Armazene as pontuações de teste em uma matriz
  - Calcule as pontuações médias do estudante

## O que É uma ArrayIndexOutOfBoundsException?

- Como você já sabe, uma matriz tem um tamanho fixo
- O índice deve ser um intervalo  $[0, n-1]$ , em que  $n$  seja o tamanho da matriz
- Se um índice for negativo ou maior ou igual ao tamanho da matriz, o índice da matriz estará fora dos limites
- Se um índice estiver fora dos limites, a JVM lançará uma `ArrayIndexOutOfBoundsException`
- Isso denomina-se verificação automática dos limites

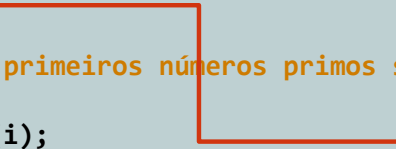
## O Que Acontece Quando Essa Exceção Ocorre?

- `ArrayIndexOutOfBoundsException` só é lançada no run-time
- O compilador Java não verifica essa exceção quando um programa é compilado
- O programa será terminado se essa exceção for tratada



# Como Você Identifica ArrayIndexOutOfBoundsException?

```
public static void main(String[] args) {  
    int primes[] = {2, 3, 5, 7, 11, 13, 17};  
    System.out.println("Comprimento da matriz: " + primes.length);  
    primes[10] = 20; //  
  
    System.out.println("Os primeiros números primos são:");  
    for (int i : primes) {  
        System.out.println(i);  
    } //fim for  
} //fim do método main
```



O índice da matriz é  
0-6

Ele está tentando  
acessar um elemento no  
índice 10

- Saída:

```
Array length: 7  
Exception in thread "main"  
java.lang.ArrayIndexOutOfBoundsException: 10  
    at arraysdemo.ArraysDemo.main(ArraysDemo.java:21)  
Java Result: 1
```

## Exercício 4

- Adicione o arquivo `ArrayEx2.java` ao projeto que você criou para o exercício 2
- Examine `ArrayEx2.java`
- Faça o seguinte:
  - Execute o programa e observe o erro
  - Modifique o programa e resolva o erro
  - Usando um loop for-each, exiba todos os navegadores que estão armazenados na matriz

# Resumo

- Nesta lição, você deverá ter aprendido a:
  - Criar e inicializar matrizes unidimensionais
  - Modificar um elemento de matriz
  - Percorrer uma matriz unidimensional usando um loop for
  - Identificar a causa de uma `ArrayIndexOutOfBoundsException`



The Oracle Academy logo is centered on a light gray background. It features the word "ORACLE" in a bold, orange, sans-serif font. Below it, the word "Academy" is written in a smaller, dark gray, sans-serif font. The entire logo is framed by a thin black border, with dark gray horizontal bars at the top and bottom.

# ORACLE

## Academy