

Vetores (*arrays*) – Parte 2

Prof. Bruno Nogueira

Inicialização de vetores

- ▶ É possível inicializar os valores que serão armazenados em um vetor

```
double [] valores = new double [3];  
valores[0] = 3.1;  
valores[1] = 3.2;  
valores[2] = 3.3;  
for(int i = 0; i < valores.length; i++)  
    System.out.println(valores[i]);
```



```
double [] valores = {3.1, 3.2, 3.3};  
for(int i = 0; i < valores.length; i++)  
    System.out.println(valores[i]);
```

Inicialização de vetores

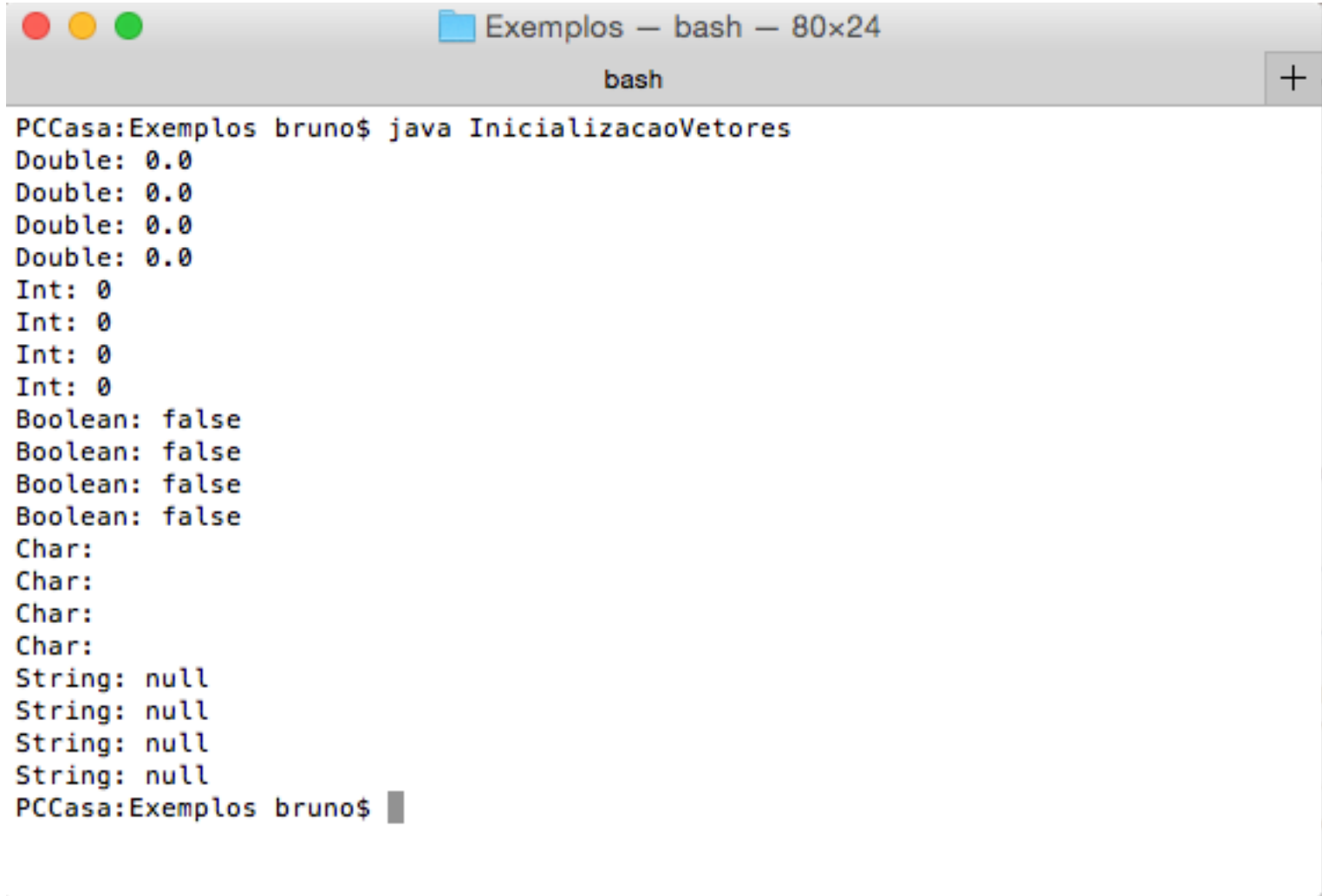
- ▶ Quando não declaramos valores iniciais explicitamente, Java se encarrega de preencher os vetores com valores iniciais
- ▶ Isso **não** é o desejável
 - ▶ Declarar explicitamente os valores iniciais é uma boa prática de programação
 - ▶ Facilita o entendimento do seu código por outros programadores (já sabem o que vão encontrar inicialmente nos vetores)
 - ▶ Evita alguns erros

```

1  import java.util.Scanner;
2  public class InicializacaoVetores
3  {
4      public static void main (String [] args)
5      {
6          Scanner teclado = new Scanner (System.in);
7          double[] vetorDouble = new double[4];
8          int[] vetorInt = new int [4];
9          boolean [] vetorBool = new boolean [4];
10         char [] vetorChar = new char [4];
11         String [] vetorString = new String [4];
12
13         for(int i = 0; i < vetorDouble.length; i++)
14             System.out.println("Double: " + vetorDouble[i]);
15
16         for(int i = 0; i < vetorInt.length; i++)
17             System.out.println("Int: " + vetorInt[i]);
18
19         for(int i = 0; i < vetorBool.length; i++)
20             System.out.println("Boolean: " + vetorBool[i]);
21
22         for(int i = 0; i < vetorChar.length; i++)
23             System.out.println("Char: " + vetorChar[i]);
24
25         for(int i = 0; i < vetorString.length; i++)
26             System.out.println("String: " + vetorString[i]);
27     }
28 }

```

Resultados



```
Exemplos — bash — 80x24
bash
PCCasa:Exemplos bruno$ java InicializacaoVetores
Double: 0.0
Double: 0.0
Double: 0.0
Double: 0.0
Int: 0
Int: 0
Int: 0
Int: 0
Boolean: false
Boolean: false
Boolean: false
Boolean: false
Char:
Char:
Char:
Char:
String: null
String: null
String: null
String: null
PCCasa:Exemplos bruno$
```

Inicialização de vetores

- ▶ **Tipos primitivos são inicializados com valores padrão**
 - ▶ int: 0; double e float: 0.0;
 - ▶ char: vazia “
 - ▶ boolean: false
- ▶ **Tipos de classe não são inicializados!**
 - ▶ Valor null
 - ▶ Indica uma posição nula, vazia, sem nenhum objeto ou valor válido naquela posição

Inicialização de vetores

- ▶ Ainda, é possível preencher apenas algumas das posições de um vetor com valores válidos
 - ▶ Requer atenção!

```
1 public class InicializacaoVetoresV2
2 {
3     public static void main (String [] args)
4     {
5         String [] vetorString = new String [20];
6
7         for(int i = 0; i < 5; i++)
8             vetorString[i] = "Posição " + i + " preenchida!";
9
10        for(int i = 0; i < vetorString.length; i++)
11            System.out.println("String: " + vetorString[i]);
12
13    }
14 }
```

Inicialização de vetores

- ▶ Ainda, é possível preencher apenas algumas das posições de um vetor com valores válidos
 - ▶ Requer atenção!

```
1 public class InicializacaoVetoresV3
2 {
3     public static void main (String [] args)
4     {
5         String [] vetorString = new String [20];
6
7         for(int i = 0; i < 5; i++)
8             vetorString[i] = "Posição " + i + " preenchida!";
9
10        for(int i = 0; i < vetorString.length; i++)
11            System.out.println("String: " + vetorString[i].toUpperCase());
12
13    }
14 }
```

Gera erro ao tentar manipular posições entre 5 e 19!
Não foram inicializadas (valor nulo)!

Inicialização de vetores

- ▶ Ainda, é possível preencher apenas algumas das posições de um vetor com valores válidos
- ▶ Requer atenção!

```
1 public class InicializacaoVetoresV4
2 {
3     public static void main (String [] args)
4     {
5         String [] vetorString = new String [20];
6
7         for(int i = 0; i < 5; i++)
8             vetorString[i] = "Posição " + i + " preenchida!";
9
10        for(int i = 0; i < vetorString.length; i++)
11        {
12            if(vetorString[i] != null)
13                System.out.println("String: " + vetorString[i].toUpperCase());
14            else
15                System.out.println("String vazia");
16        }
17    }
18 }
19 }
```

Exercícios - Recapitulação

- ▶ Nos exemplos a seguir, encontre os erros e indique uma possível correção para todos eles

```
1  import java.util.Scanner;
2  public class ExercicioVetores
3  {
4      public static void main (String [] args)
5      {
6          double[] vetorDouble = new double[4];
7          for(int i = 0; i <=4; i++)
8              System.out.println("Valor " + i + " : " + vetorDouble[i]);
9      }
10 }
```

Resposta: for(int i = 0; i < 4; i++) OU for(int i = 0; i <= 3; i++) OU MELHOR AINDA
for(int i = 0; i < vetorDouble.length; i++)

Exercícios - Recapitulação

```
1 public class ExercicioVetores2
2 {
3     public static void main (String [] args)
4     {
5         double[] vetorDouble = new double[4.0];
6         for(int i = 0; i < 4; i++)
7             System.out.println("Valor " + i + " : " + vetorDouble[i]);
8     }
9 }
```

Resposta: `double[] vetorDouble = new double[4];`

Capacidade de vetores é INTEIRA. Não aceita valores de outros tipos

Exercícios - Recapitulação

```
1 public class ExercicioVetores3
2 {
3     public static void main (String [] args)
4     {
5         String[] vetorString = new String[5];
6         for(int i = 0; i < vetorString.length; i++){
7             System.out.println("Valor " + i + " : " + vetorString[i]);
8             System.out.println("Valor " + i + " : " + vetorString[i].toLowerCase());
9         }
10    }
11 }
```

Resposta: String é tipo classe, logo não é inicializado. Na linha 8, quando tentamos usar um método de um objeto String que estaria armazenado em uma posição do vetor, uma exceção é lançada – é encontrado um valor nulo ao invés de um String válido. A solução é inicializar os valores do vetor de String ANTES de usar os objetos ali armazenados.

Lembrete - Erros: tempo de compilação x tempo de execução

- ▶ Temos dois tipos de erros básicos em Java: aqueles que ocorrem em tempo de compilação e aqueles que ocorrem em tempo de execução
 - ▶ Tempo de compilação: detectados pelo compilador
 - ▶ Gerados por erros sintáticos
 - ▶ Compilação não termina, logo o “.class” não é gerado
 - ▶ Ex: ExercicioVetores2
 - ▶ Tempo de execução: não são detectados pelo compilador
 - ▶ Gerados por erros de lógica, geralmente
 - ▶ Compilação termina, “.class” é gerado
 - ▶ Lançam exceções
 - ▶ Ex: ExercicioVetores e ExercicioVetores3

Exercício

- ▶ Faça um programa que receba um conjunto de n valores numéricos em ponto flutuante. Depois, mostre a média e o desvio padrão destes números. O desvio padrão pode ser computado por meio da fórmula:

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{n}}$$

Ao final, mostre quais dos números se encontram dentro do limite de um desvio padrão abaixo ou acima da média ($\mu \pm \sigma$)

Exercício

- ▶ Faça um programa em Java que receba como argumento, da linha de comando um conjunto de nomes de alunos e notas dos mesmos em uma prova. A seguir, mostre o nome e a nota dos alunos com a menor e a maior nota. O primeiro parâmetro recebido pelo programa é a quantidade de alunos na turma. Abaixo, um exemplo de uso do programa:

```
>> java NotasComNomes 3 João 5,0 Maria 6,0 José 7,5
```

João tem a menor nota: 5,0.

José tem a maior nota: 7,5.

Exercício

- ▶ Normalização de vetores numéricos consiste em modificar os valores armazenados de maneira a deixar todas as posições do vetor dentro de um intervalo predeterminado. A transformação mais simples consiste na normalização linear, da seguinte forma:

$$z_i = \frac{x_i - \min}{\max - \min}$$

Considerando um vetor de n posições, i varia de 0 a $n-1$; \min consiste no menor valor e \max no maior valor do vetor. Faça um programa em Java que leia n valores de um vetor numérico e exiba, ao final, o vetor resultante de sua normalização linear.

Exercício

- ▶ Monte um programa em Java que peça ao usuário para digitar n Strings e mostre, ao final, quais Strings são iguais.
Exemplo:

Quantas Strings serão digitadas?

>> 3

Digite a String 1:

>> Abracadabra

Digite a String 2:

>> Abracadabra

Digite a String 3:

>> Não é abracadabra

Strings iguais: 1, 2 - Abracadabra

Quantas Strings serão digitadas?

>> 2

Digite a String 1:

>> Abracadabra

Digite a String 2:

>> Não é abracadabra

Todas as Strings são diferentes.