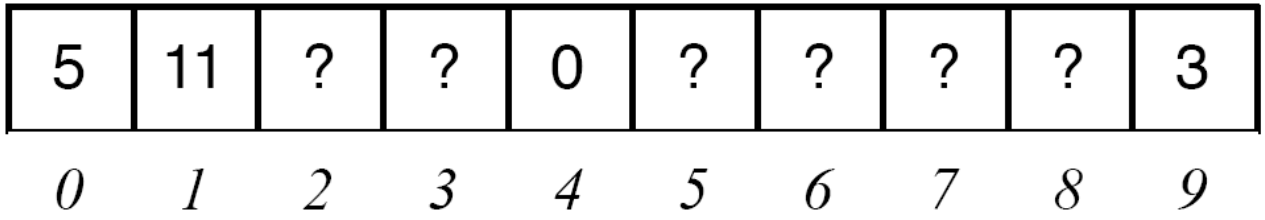
Técnicas de Programação II

Aula 04 – Arrays

Edirlei Soares de Lima
<edirlei.lima@uniriotec.br>

Arrays

- **Array** é um mecanismo que nos permite armazenar um conjunto de valores na memória do computador.



- Em Java, arrays são objetos. Consequentemente, uma variável do tipo array é na verdade uma referência para um objeto.

```
int[] myArray;
```

Arrays – Declaração

- Ao declararmos um array em Java:

```
int[] myArray;
```

  - Nenhuma área de memória para o array é alocada, apenas uma referência para um array de inteiros é definida;
  - Nenhuma referência ao tamanho do array é feita na declaração;
  - Apenas na criação do array é que iremos alocar espaço em memória e, consequentemente, iremos definir o seu tamanho.

Arrays – Criação

- Antes de utilizar um array, ele deve ser explicitamente criado:

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

- Podemos declarar e criar um array:

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

- Podemos declarar, criar e inicializar um array:

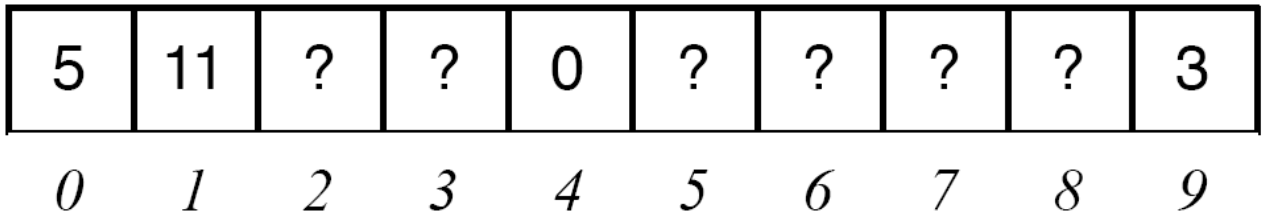
```
int[] myArray = new int[]{10, 20, 30, 40};
```

Arrays – Incialização

- Os elementos do vetor podem ser acessados através dos seus índices:

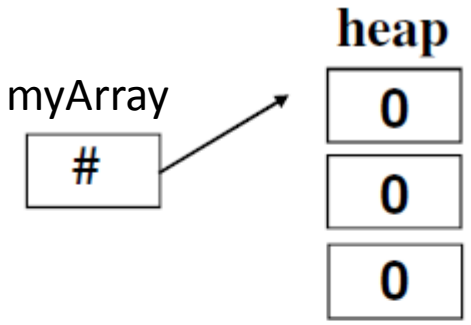
```
int[] myArray = new int[10];

myArray[0] = 5;
myArray[1] = 11;
myArray[4] = 0;
myArray[9] = 3;
```

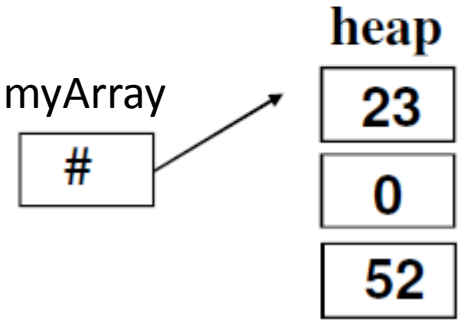


Arrays – Inicialização

```
int[] myArray = new int[3];
```

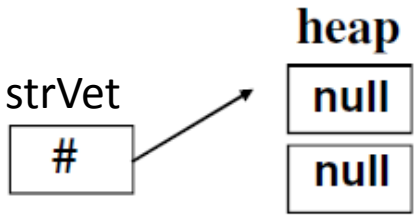



```
myArray[0] = 23;
myArray[1] = 0;
myArray[2] = 52;
```

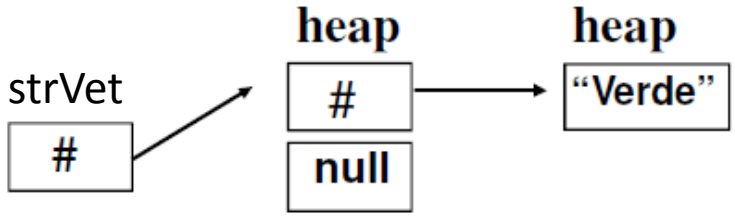


Arrays de Referências para Objetos

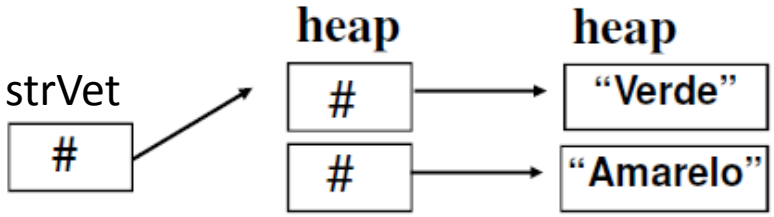
```
String[] strVet = new String[2];
```




```
strVet[0] = new String("Verde");
```




```
strVet[1] = new String("Amarelo");
```



Tamanho de um Array

- Em Java, arrays são objetos. A propriedade **length** de um array representa o seu tamanho:

```
int[] vet = new int[]{10, 20, 30, 40};

System.out.println("Tamanho: " + vet.length);
```

```
for (x = 0; x < myArray.length; x++)
{
  System.out.println(myArray[x]);
}
```

Arrays – Foreach

- A linguagem Java também fornece uma outra versão do comando "for" para iterar sobre coleções de objetos:

```
int[] vet = new int[]{10, 20, 30, 40};

for (int elem : vet)
  System.out.println(elem);
```

```
String[] vet = new String[]{"João", "Maria", "Pedro"};

for (String str : vet)
  System.out.println(str);
```

Arrays e Funções

- É possível passar arrays por parâmetro para funções:

```
public static void exibeArray(int[] vet)
{
  for (int val : vet)
  {
    System.out.println(val);
  }
}

public static void main(String[] args)
{
  int[] myArray = new int[]{1,2,3,4,5,6};

  exibeArray(myArray);
}
```

Arrays e Funções

- Funções podem alterar o conteúdo de arrays:

```
public static void dobraArray(int[] vet)
{
  int x;
  for (x = 0; x < vet.length; x++)
  {
    vet[x] = vet[x] * 2;
  }
}

public static void main(String[] args)
{
  int[] myArray = new int[]{1,2,3,4,5,6};
  dobraArray(myArray);
  exibeArray(myArray);
}
```

Arrays e Funções

- Funções podem retornar arrays:

```
public static int[] maiores(int[] vet, int limite){
  int[] vet_maiores;
  int cont = 0, i = 0;
  for (int val : vet){
    if (val > limite)
      cont++;   //conta total de elementos maiores
  }
  vet_maiores = new int[cont];
  for (int val : vet){
    if (val > limite){
      vet_maiores[i] = val;
      i++;
    }
  }
  return vet_maiores;
}
...
```

Arrays e Funções

```
...

public static void main(String[] args)
{
  int[] myArray1 = new int[]{1,2,3,4,5,6};
  int[] myArray2;

  myArray2 = maiores(myArray1, 3);

  exibeArray(myArray2);
}
```

Arrays Multidimensionais

- Em Java, arrays podem ter mais de uma dimenção:

```
int[][] mat;
```

3	1	8	6	1
7	2	5	4	9
1	9	3	1	2
5	8	6	7	3
6	4	9	2	1

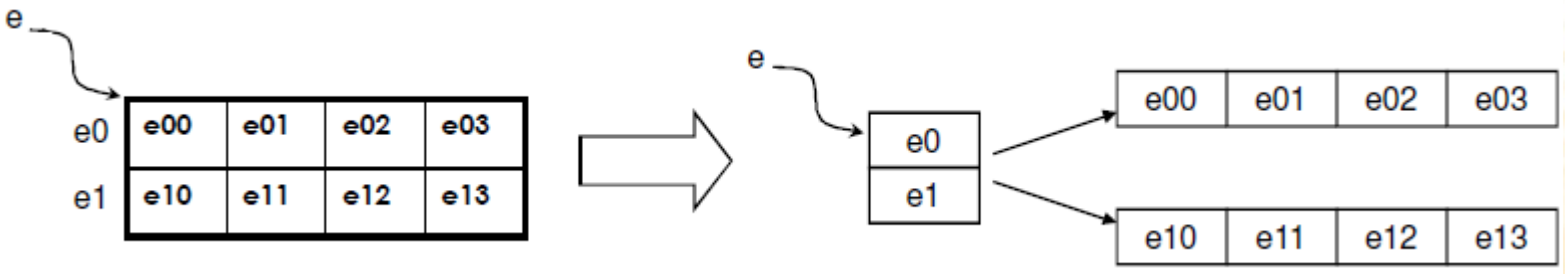
Arrays Multidimensionais – Criação

- Da mesma forma que arrays simples, arrays multidimensionais também devem ser explicitamente criados:

```
int[][] mat = new int[3][3];
```

- Eles também podem ser criados e inicializados:

```
int[][] mat = new int[][]{{10,20,30,40},
                          {40,50,60,70}};
```



Arrays Multidimensionais

- Os elementos do array multidimensional podem ser acessados através dos seus índices:

```
int[][] mat = new int[3][3];

mat[0][0] = 5;
mat[0][2] = 1;
mat[2][1] = 8;
```

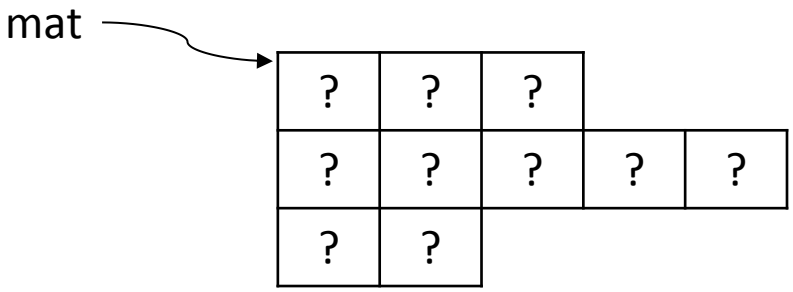
5	?	1
?	?	?
?	8	?

Arrays Multidimensionais

- Java permite a criação de arrays com dimensões internas diferentes:

```
String[][] mat = new String[3][];

mat[0] = new String[3];
mat[1] = new String[5];
mat[2] = new String[2];
```



Arrays Multidimensionais – Foreach

- Para percorrer um array de múltiplas dimensões internas de tamanhos diferentes é possível usar o for para iterar sobre as coleções de objetos:

```
for (String[] vet : mat)
{
  for (String str : vet)
  {
    System.out.println(str);
  }
}
```

Classe Utilitária Arrays

- A classe utilitária Arrays (java.util.Arrays) fornece vários métodos para a manipulação de arrays:
  - **fill** : preenche um vetor com um determinado valor;
  - **equals** : compara se o conteúdo de dois arrays é exatamente igual;
  - **copyOf** : copia o conteúdo de um array para outro array;
  - **binarySearch** : busca por um determinado elemento dentro de um array;
  - **sort** : ordena um array;

Classe Arrays – Método `fill`

- Preencher um vetor de inteiros com zeros:

```
int[] myArray = new int[50];

Arrays.fill(myArray, 0);
```

- Preencher um vetor de Strings com strings vazias:

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

Arrays.fill(myArray, "");
```

Classe Arrays – Método `equals`

- Comparar dois vetores de inteiros:

```
int[] myArray1 = new int[]{1,2,3,4,5};

int[] myArray2 = new int[]{1,2,3,4,5};

if (Arrays.equals(myArray1, myArray2))
  System.out.println("São iguais!");
else
  System.out.println("Não são iguais!");
```

Classe Arrays – Método `copyOf`

- Copiar o conteúdo de um array para outro:

```
int[] myArray1 = new int[]{1,2,3,4,5};
int[] myArray2 = new int[5];


myArray2 = Arrays.copyOf(myArray1, myArray1.length);
```

- Copiar o conteúdo de um array para outro em um intervalo:

```
int[] myArray1 = new int[]{1,2,3,4,5};
int[] myArray2 = new int[5];


myArray2 = Arrays.copyOfRange(myArray1, 2, 4);
```

Classe Arrays – Método `binarySearch`

- Busca binária em um array:
  - **Atenção:** o array precisa estar ordenado.

```
int[] myArray1 = new int[]{1,2,3,4,5,6,7,8,9,10};

int idx = Arrays.binarySearch(myArray1, 9);

if (idx > -1)
  System.out.println("Valor encontrado no indice: " + idx);
```

```
String[] nomes = new String[]{"João", "Maria",
                              "Paulo", "Pedro"};

int idx = Arrays.binarySearch(nomes, "Maria");

if (idx > -1)
  System.out.println("Valor encontrado no indice: " + idx);
```

Classe Arrays – Método `sort`

- Ordenar um vetor de inteiros:

```
int[] myArray1 = new int[]{2,9,6,7,82,3,1,4,56,4,8};

Arrays.sort(myArray1);

for (int val : myArray1)
  System.out.println(val);
```

- Ordenar um vetor de Strings:

```
String[] nomes = new String[]{"Pedro", "Antonio",
                              "Joao", "Maria"};
Arrays.sort(nomes);

for (String str : nomes)
  System.out.println(str);
```

Exercícios

Lista de Exercícios 05 – Arrays

http://uniriodb2.uniriotec.br