

Genéricos

Métodos e classes genéricas.

Autor: Prof. Manuel F. Paradela Ledón.

Métodos genéricos

A linguagem de programação Java permite a definição de métodos genéricos. Por exemplo, este método permitirá a visualização de vetores de elementos de qualquer tipo **E**:

```
public <E> void visualizar(E vetor[]) {  
    for (int i = 0; i < vetor.length; i++) {  
        System.out.print(vetor[i] + "\n");  
    }  
    System.out.println();  
}
```

Assim, podemos usar este mesmo método para visualizar vetores de elementos Integer, Double, Float, String e de qualquer outra classe.

Métodos genéricos

Veja como utilizar o método genérico mostrado no slide anterior:

```
Double vet1[] = {4.5, 6.3, 1.2, 78.3, 0.15, 4.3};
```

```
Integer vet2[] = {4, 6, 1, 78, 15, 3};
```

```
AtletaVelocidade atlet[] = {  
    new AtletaVelocidade("Luiza","Brasil","F", 11.4f, 11.9f),  
    new AtletaVelocidade("Pedro","Brasil","M", 10.4f, 10.9f),  
    new AtletaVelocidade("Gilbert","EUA","M", 12.5f, 12.6f),  
    new AtletaVelocidade("Stephan","Canada","M", 11.0f, 10.98f),  
    new AtletaVelocidade("Ana","Argentina","F", 13.14f, 13.7f)  
};
```

```
visualizar(vet1);
```

```
visualizar(vet2);
```

```
visualizar(atlet);
```

Métodos genéricos - Exemplo com ordenação

```
public <E extends Comparable<E>> boolean bubbleSort(E vetor [])  
{ // Este método ordena um vetor de tipo genérico.  
    if (vetor == null) return false;  
    for (int i = 0; i < vetor.length - 1; i++) {  
        for (int j = 0; j < vetor.length - 1 - i; j++) {  
            if (vetor[j].compareTo(vetor[j+1]) > 0) {  
                //A comparação anterior funciona porque as classes Integer,  
                //Double, AtletaVelocidade... possuem o método compareTo.  
                E tmp = vetor[j];  
                vetor[j] = vetor[j+1];  
                vetor[j+1] = tmp;  
            }  
        }  
    }  
    return true;  
}
```

Um único método de ordenação
para ordenar "qualquer vetor"...

Métodos genéricos - Exemplo com ordenação

Veja estes requerimentos na classe AtletaVelocidade:

```
public class AtletaVelocidade extends Atleta implements Comparable {  
  
    public int compareTo (Object outroAtleta) { // cabeçalho obrigatório  
        //usamos o nome do atleta como critério para efetuar a ordenação:  
        if(getNome().compareTo(((AtletaVelocidade)outroAtleta).getNome()) < 0 )  
            return -1;  
        else if(getNome().compareTo(((AtletaVelocidade)outroAtleta).getNome()) == 0 )  
            return 0;  
        else return 1;  
    }  
  
    ...  
}
```

Métodos genéricos - Exemplo com ordenação

```
bubbleSort(vet1);  
System.out.println("Vetor ordenado:");  
visualizar(vet1);
```

```
bubbleSort(vet2);  
System.out.println("Vetor ordenado:");  
visualizar(vet2);
```

```
bubbleSort(atlet);  
System.out.println("Vetor ordenado pelos nomes dos atletas:");  
visualizar(atlet);
```

Um único método de ordenação para ordenar "qualquer vetor": um vetor de Integer, outro de Double e outro com objetos da classe AtletaVelocidade.

Classes genéricas

A linguagem de programação Java permite a definição de classes genéricas. Por exemplo, uma classe Pilha ou Fila que defina o tipo de elemento específico que podemos guardar dentro da estrutura de dados.

```
class Pilha <E> {  
    // classe genérica: a pilha armazenará elementos do tipo E  
    private Node topo = null;  
  
    ...  
}
```

Classes genéricas - Exemplo Pilha

```
class Pilha <E> {  
    private Node topo = null;  
  
    public Pilha() {  
    }  
  
    public Node push (E novo) {  
        Node nodo = new Node();  
        nodo.setValue(novo);  
        nodo.setNext(null);  
        if (topo == null) topo = nodo;  
        else {  
            nodo.setNext(topo);  
            topo = nodo;  
        }  
        return nodo;  
    }  
}
```


Classes genéricas - Exemplo Pilha

```
public E pop() {  
    if (topo == null) return null;  
    E value = (E) topo.getValue();  
    Node temp = topo;  
    topo = topo.getNext();  
    temp = null;  
    return value;  
}  
  
public boolean isEmpty() {  
    if(topo == null) return true; else return false;  
}  
  
public E top() {  
    if(topo == null) return null; else return (E)topo.getValue();  
}  
}
```

Classes genéricas – Exemplo Pilha

```
Pilha <Integer> pilha = new Pilha <Integer> ();
```

```
pilha.push(5);
```

```
pilha.push(12);
```

```
pilha.push(6);
```

```
pilha.push(4);
```

```
pilha.push(7);
```

```
// Mas, não poderíamos inserir outros tipos de objetos:
```

```
// pilha.push(7.3f); // classe Float
```

```
// pilha.push(6.4); // classe Double
```

```
// pilha.push("São Paulo"); // classe String
```

```
// o que seria uma vantagem se queremos uma pilha com
```

```
// elementos do mesmo tipo.
```

```
// A inserção de objetos polimorfos seria possível sem genéricos:
```

```
// Pilha pilha = new Pilha();
```

Classes genéricas - outro exemplo

```
class ListaOrd <E> {  
    private int size;      //contador de objetos  
    private Node head;    //cabeça da lista ordenada  
    ...  
}
```

```
ListaOrd <Aluno>listaAlunos = new ListaOrd <Aluno>();  
listaAlunos.insert(new Aluno("8888", "Caio Silva", 54, 'M'));  
listaAlunos.insert(new Aluno("7777", "Roberto Ledon", 92, 'M'));  
...
```

```
//listaAlunos.insert(new String("Um objeto String")); //provocará erro  
//listaAlunos.insert(new Atleta("Luana Souza", "Brasil")); //provocará erro
```

Classes genéricas - Exemplo com ArrayList

Como a classe ArrayList foi definida desta forma:

```
java.util Class ArrayList<E>
```

//podemos usar o ArrayList para adicionar objetos polimorfos:

```
ArrayList lista1 = new ArrayList();
```

```
lista1.add(9); //Integer
```

```
lista1.add(1.3f); //Float
```

```
lista1.add(9.8); //Double
```

```
lista1.add("Luis Lima"); //String
```

//ou utilizar o ArrayList para adicionar objetos de uma classe específica:

```
ArrayList <Integer> lista2 = new ArrayList <Integer> ();
```

```
lista2.add(9); //Integer
```

```
lista2.add(2); //Integer
```

/* mas não podemos inserir objetos de outras classes:

```
    lista2.add(8.4f); //Float
```

```
    lista2.add(9.8); //Double
```

```
    lista2.add("Luiza"); //String
```

```
*/
```

Exemplos para analisar

Veja os projetos NetBeans completos fornecidos:

- Pilha1
- Pilha1_com_genericos
- OrdenacaoBubbleSortGenericos