# Coleções Genéricas

Estruturas de Dados e Algoritmos

# Coleções Genéricas

- As classes que estamos criando estão limitadas ao armazenamento de um tipo de dados específico no atributo vetor.

```
13   public class VetorInteiro {
14        int[] vetor;
15
16        VetorInteiro(int tamanho){
17           vetor = new int[10];
18        }
19   }
```

- Por exemplo, na classe acima, somente podemos armazenar valores inteiros na variável vetor.

# Coleções Genéricas

- Uma característica essencial das coleções de tipos abstratos de dados é a de que deveríamos poder usá-las para quaisquer tipos de dados

- O Java possui um mecanismo especial para realizar a parametrização de tipos, permitindo a generalização essa capacidade de generalização.

# Coleções Genéricas

- Para parametrizar algum tipo pertencente a uma determinada classe utilizamos a notação <Item> após o nome da classe.  Isso define o Item como um parâmetro de tipo, ou seja, um representante simbólico do tipo concreto que será utilizado pelo cliente.

- Podemos ler  Pilha<Item>  como "pilha de items"

- Tipos genéricos permite a criação de um código mais limpo do lado do cliente, e mais fácil de entender e debugar.

# Coleções Genéricas

- É possível generalizar o tipo do vetor que armazenará da classe da seguinte forma:

```
public class VetorGenerico<Item> {
    Item[] vetor;

    public VetorGenerico(int tamanho){
        vetor = (Item[]) new Object[tamanho];
    }
}
```

# Coleções Genéricas

- É possível generalizar o tipo do vetor que armazenará da classe da seguinte forma:

```
public class VetorGenerico<Item> {
    Item[] vetor;

    public VetorGenerico(int tamanho){
        vetor = (Item[]) new Object[tamanho];
    }
}
```

# Coleções Genéricas

- É possível generalizar o tipo do vetor que armazenará da classe da seguinte forma:

```
public class VetorGenerico<Item> {
    Item[] vetor;

    public VetorGenerico(int tamanho){
        vetor = (Item[]) new Object[tamanho];
    }
}
```

# Coleções Genéricas

- É possível generalizar o tipo do vetor que armazenará da classe da seguinte forma:

```
public class VetorGenerico<Item> {
    Item[] vetor;

    public VetorGenerico(int tamanho){
        vetor = (Item[]) new Object[tamanho];
    }
}
```

# Coleções Genéricas

- A utilização da nova classe permite agora informar qual o tipo que se deseja para o atributo Item[ ] vetor

```
18      public static void main(String[] args) {
19          VetorInteiro vetint = new VetorInteiro(10);
20          vetint.vetor[0] = 1;
21
            VetorGenerico<String> vetgenerico = new VetorGenerico<String>(10);
23          vetgenerico.vetor[0] = "casa";
24      }
25  }
```

# Coleções Genéricas

- A utilização da nova classe permite agora informar qual o tipo que se deseja para o atributo Item[ ] vetor

```
18      public static void main(String[] args) {
19          VetorInteiro vetint = new VetorInteiro(10);
20          vetint.vetor[0] = 1;
21
            VetorGenerico<String> vetgenerico = new VetorGenerico<String>(10);
23          vetgenerico.vetor[0] = "casa";
24      }
25  }
```

# Referências utilizadas


- Sedgewick, Robert, and Kevin Wayne. Algorithms. Addison-Wesley Professional, 2011.