Tipos curinga (wildcard types)

http://educandoweb.com.br

Prof. Dr. Nelio Alves

Generics são invariantes

List<Object> não é o supertipo de qualquer tipo de lista:

```
List<Object> myObjs = new ArrayList<Object>();
List<Integer> myNumbers = new ArrayList<Integer>();
myObjs = myNumbers; // erro de compilação
```

O supertipo de qualquer tipo de lista é List<?>. Este é um tipo curinga:

```
List<?> myObjs = new ArrayList<Object>();
List<Integer> myNumbers = new ArrayList<Integer>();
myObjs = myNumbers;
```

Com tipos curinga podemos fazer métodos que recebem um genérico de "qualquer tipo":

```
package application;

import java.util.Arrays;
import java.util.List;

public class Program {

    public static void main(String[] args) {
        List<Integer> myInts = Arrays.asList(5, 2, 10);
        printList(myInts);
    }

    public static void printList(List<?> list) {
        for (Object obj : list) {
            System.out.println(obj);
        }
    }
}
```

Porém não é possível adicionar dados a uma coleção de tipo curinga

```
package application;

import java.util.ArrayList;
import java.util.List;

public class Program {

    public static void main(String[] args) {

        List<?> list = new ArrayList<Integer>();
        list.add(3); // erro de compilação
    }
}
```

O compilador não sabe qual é o tipo específico do qual a lista foi instanciada.