Aluna: Analu Sorbara

1. Quais são as motivações para o uso dos genéricos em Java?
A principal motivação é permitir maior flexibilidade ao criar classes, permitindo que essas possam ser utilizadas para mais de um tipo de variável ou classe.

2. Um tipo primitivo pode ser usado como argumento de tipo?
Não, somente tipos compostos, ou seja, classes e interfaces.

3. Mostre como declarar uma classe chamada TesteDoisParametros que use dois parâmetros genéricos.

```
package generics;

public class TesteDoisParametros <T,S> {
    private final T varT;
    private final S varS;

    public TesteDoisParametros(T varT, S varS) {
        this.varT = varT;
        this.varS = varS;
    }

    public void print() {
        System.out.println(varT);
        System.out.println(varS);
    }
}
```

```
package generics;

public class GenericsTest {
    public static void main(String[] args) {
        TesteDoisParametros<String, Integer> testeDoisParametros = new
TesteDoisParametros<>("teste", 3);
        testeDoisParametros.print();
    }
}
```

4. Agora, altere TesteDoisParametros para que seu segundo parâmetro de tipo seja subclasse do primeiro parâmetro de tipo.
package generics;

```
package generics;

public class TesteDoisParametros <T,S extends T> {
    private final T varT;
    private final S varS;
```

```
    public TesteDoisParametros(T varT, S varS) {
        this.varT = varT;
        this.varS = varS;
    }

    public void print() {
        System.out.println(varT);
        System.out.println(varS);
    }
}
```

```
package generics;

public class GenericsTest {
    public static void main(String[] args) {
        TesteDoisParametros<String, String> testeDoisParametros = new
TesteDoisParametros<>("teste", "teste 2");
        testeDoisParametros.print();
    }
}
```

5. No que diz respeito aos genéricos, o que é o símbolo "?" e o que ele faz?
O símbolo ? permite que omitimos o tipo do termo genérico ao definir uma variável, deixando esse trabalho para o compilador.

6. O argumento curinga pode ser limitado?
Sim, pode ser limitado há uma subclasse utilizando a palavra extends.

7. Um método genérico chamado MeuGenerico( ) tem um parâmetro de tipo. Além disso, MeuGenerico( ) tem um parâmetro cujo tipo é o do parâmetro de tipo. Ele também retorna um objeto desse parâmetro de tipo. Mostre como declarar MeuGenerico( ).

```
public <S> S meuGenerico(S valor) {
     return valor;
}
```

8. Dada a interface genérica a seguir
interface IGenericoIF<T, V extends T> { // ...
mostre a declaração de uma classe chamada MinhaClasse que implemente IGenericoIF.

```
public class IGenericoIFImpl<T, V extends T> implements IGenericoIF<T, V> {
    //...
}
```

9. Dada uma classe genérica chamada Contador<T>, mostre como criar um objeto de seu tipo bruto.

Contador contador = new Contador<>();

10. Existem parâmetros de tipo no tempo de execução?
Não, todos os parâmetros de tipo são validados em tempo de compilação.

11. O que é <>?
É um símbolo para indicar parâmetros de tipo para determinada variável e também para indicar para uma classe e uma interface que será utilizado um parâmetro de tipo. Quando vazio, é uma forma de deixar implícito o tipo de uma nova instância.

12. Como a linha a seguir pode ser simplificada?

```
   MinhaClasse<Double,String> obj = new MinhaClasse<Double,String>(1.1,"Hi");
```

```
MinhaClasse<Double,String> obj = new MinhaClasse<>(1.1, "Hi");
// ou
MinhaClasse<?,?> obj2 = new MinhaClasse<>(1.1, "Hi");
```