Параметризованные типы

Пример:

```
public int abs(int x) {return x>0?x:-x;}
```

Вопросы:

- Что делает этот метод?

- Является ли следующий код ошибочным:

```
double d; double m = abs(d);
```

Определение и свойства параметризованных типов

Параметризация позволяет создавать классы, интерфейсы и методы, в которых тип обрабатываемых данных задается как параметр.

Параметром может быть только ссылочный тип

Параметром может быть шаблон

Синтаксис параметров:

<Тип1 или Шаблон, Тип2 или Шаблон, ...>

Пример класса

```
public class Subject <T1, T2> {
   private T1 name;
   private T2 id;
   public Subject() {}
   public Subject(T1 names, T2 ids ) {
      id = ids;
      name = names;
   } }

Subject<String,Integer> sub =new Subject<String,Integer>();

char ch[] = {‘J’,’a’,’v’,’a’};
Subject<char[],Double> sub2 = new Subject<char[],Double>(ch, 71.0 );
```

Пример интерфейса

```
public interface Generator<T> { T next(); }
public class Coffee {}

public class CoffeeGenerator implements
                                        Generator<Coffee> {
  private Coffee[] types = {…}
  public Coffee next() {
     Random rand = new Random();
     return types[rand.nextInt(types.length)]
  } }
```

Пример библиотечных интерфейсов

```
Interface Comparable<T> {
    int compareTo(T o)
}
java.util.function
Interface Function<T,R> {
    R apply(T t)
}
```

Шаблоны типов

? extends Type : семейство наследников типа Type.

? super Type : семейство предков типа Type.

? : набор любых типов.

Пример:

```
public interface Collection<E> {
boolean   addAll (Collection<? extends E> c);
}
```