Родовые типы Java (Generics)

Рассматриваемые вопросы

1. Параметризованные классы
2. Diamond оператор
3. Универсальные методы (Generic methods)
4. Ограничения на допустимые типы
5. Ковариантность, контравариантность и инвариантность
6. Wildcards

Параметризованные классы

```
public class ClassName<T1, T2, ...> {
    // ...
}
```

Diamond оператор

```
Pair<Integer, String> pair = new Pair<>(6, "Apr");

Pair<Integer, String> pair = new Pair(6, "Apr");
```

Diamond оператор

```
List list = new LinkedList();
list.add("First");
list.add("Second");
List<String> list2 = list;
for (Iterator<String> itemItr = list2.iterator(); itemItr.hasNext(); )
    System.out.println( itemItr.next() );
```

Diamond оператор

```
List list = new LinkedList<>();
list.add("First");
list.add("Second");
List<String> list2 = list;
for (Iterator<String> itemItr = list2.iterator(); itemItr.hasNext(); )
    System.out.println( itemItr.next() );
```

Diamond оператор

При выполнении

list.add(10);

без <>: java.lang.ClassCastException

с <>: ошибка компиляции

Универсальные методы (Generic methods)

Generic-метод определяет базовый набор операций, которые будут применяться к разным типам данных.

<T **extends** Тип> returnType method(T arg) { }

<T> T[] method(**int** count, T arg) { }

Универсальные методы (Generic methods)

Ковариантность — это сохранение иерархии наследования исходных типов в производных типах в том же порядке.

Множество<Животные> = Множество<Кошки>

Универсальные методы (Generic methods)

Контравариантность — это обращение иерархии исходных типов на противоположную в производных типах.

Множество<Кошки> = Множество<Животные>

Универсальные методы (Generic methods)

Инвариантность — отсутствие наследования между производными типами.

Универсальные методы (Generic methods)

Массивы – ковариантны

Дженерики инвариантны

Универсальные методы (Generic methods)

```
List<Integer> ints = Arrays.asList(1,2,3);
List<Number> nums = ints; // compile-time error
nums.set(2, 3.14);
assert ints.toString().equals("[1, 2, 3.14]");
```

Универсальные методы (Generic methods)

```
List<String>[] lsa = new List<String>[10]; // не верно
Object[] oa = lsa; //OK, List<String> - подтип Object
List<Integer> li = new ArrayList<Integer>();
li.add(new Integer(3));
oa[0] = li;
Strings = lsa[0].get(0);
```

Wildcards

```
List<Integer> ints = new ArrayList<Integer>();
List<? extends Number> nums = ints;
```

Это ковариантность, т.к.

List<Integer> — подтип List<? extends Number>

Wildcards

```
List<Number> nums = new ArrayList<Number>();
List<? super Integer> ints = nums;
```

Это контравариантность, т.к.

List<Number> - подтип List<? super Integer>.

Вопросы для самоконтроля

1. Для чего введены Generics?
2. В чем отличие Generics в методах и в классах?
3. Что такое Diamond-оператор?
4. Что такое WildCards?
5. Какой тип данных виден для Generics в Runtime?