

МАГИСТЕРСКАЯ ПРОГРАММА 09.04.01/07 Интеллектуальные системы анализа,
обработки и интерпретации больших данных

Вариант 15

Дисциплина: Языки программирования для работы с большими данными

Преподаватель	П.В. Степанов
(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

Цель работы:

Получение навыков работы со Stream API в Java.

Выполнение:

Задание 1:

Использовать ТОЛЬКО методы Stream API. Циклов и условий быть не должно.

1. Задана коллекция строк. Вернуть последний элемент и третий элемент коллекции.
2. Задана коллекция строк. Вернуть первый элемент коллекции, а также существуют ли все совпадения с шаблоном. Шаблон можно выбрать произвольно.

Листинг выполнения подзадачи 1 (файл lr911.java)

```
package lr911;

import java.util.Arrays;
import java.util.Collection;
import java.util.stream.Stream;

public class lr911 {
    public static void main(String[] args) {
        Collection<String> collection = Arrays.asList("a1", "a2", "a3", "a4");
        Stream<String> my_stream = collection.stream();
        System.out.println("Last = " + my_stream.skip(collection.size()-
1).findFirst().get());
        my_stream = collection.stream();
        System.out.println("Third = " + my_stream.skip(2).findFirst().get());
    }
}
```



```
lr911 x
C:\Users\stale\.jdk\openjdk-17.0.2\bin\jav
Last = a4
Third = a3

Process finished with exit code 0
```

Рисунок 1 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл lr912.java)

```
package lr91;  
  
import java.util.Arrays;  
import java.util.Collection;  
import java.util.stream.Stream;  
  
public class lr912 {  
    public static void main(String[] args) {  
        Collection<String> collection = Arrays.asList("a1", "a2", "a4", "a4");  
        Stream<String> my_stream = collection.stream();  
        System.out.println("first = " + my_stream.findFirst().get());  
        my_stream = collection.stream();  
        System.out.println("Template 4 = " + my_stream.filter(s ->  
s.contains("4")).toList());  
    }  
}
```

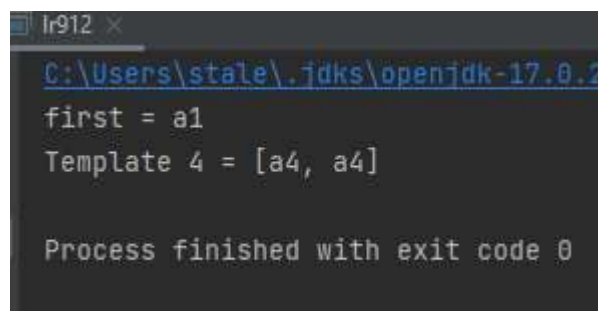


Рисунок 2 - Результат выполнения кода решения подзадачи 2

Задание 2:

Использовать ТОЛЬКО методы Stream API. Циклов и условий быть не должно.

1. Задана коллекция строк. Отсортировать значения по алфавиту и убрать повторы
2. Задана коллекция вида:

(Класс People: имя и возраст)

```
Collection<People> peoples = Arrays.asList(  
    new People("Ivan", 16),  
    new People("Petr", 23),  
    new People("Maria", 42)  
);
```

Отсортировать по имени в обратном алфавитном порядке.

Листинг выполнения подзадачи 1 (файл lr921.java)

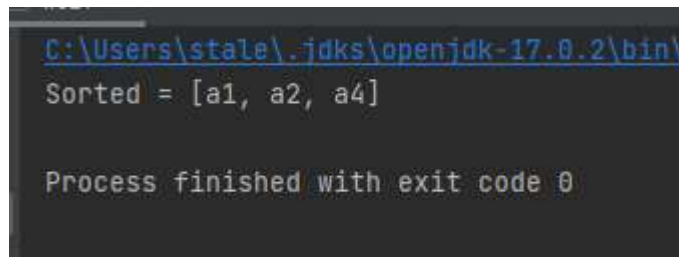
```

package lr92;

import java.util.Arrays;
import java.util.Collection;
import java.util.stream.Stream;

public class lr921 {
    public static void main(String[] args) {
        Collection<String> collection = Arrays.asList("a1", "a2", "a4", "a4");
        Stream<String> my_stream = collection.stream();
        System.out.println("Sorted = " + my_stream.sorted().distinct().toList());
    }
}

```



```

C:\Users\stale\.jdk\openjdk-17.0.2\bin\
Sorted = [a1, a2, a4]

Process finished with exit code 0

```

Рисунок 3 - Результат выполнения кода решения подзадачи 1

Листинг выполнения подзадачи 2 (файл lr922.java)

```

package lr92;

import java.util.Arrays;
import java.util.Collection;
import java.util.Objects;
import java.util.stream.Collectors;
import java.util.stream.Stream;

class People{
    private String name;
    private Integer age;

    public People() {
    }

    public People(String name, Integer age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
    }

    @Override

```

```

    public String toString() {
        return "People{" +
            "name='" + name + '\'' +
            ", age=" + age +
            '}';
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        People people = (People) o;
        return Objects.equals(name, people.name) && Objects.equals(age, people.age);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name, age);
    }
}

public class lr922 {
    public static void main(String[] args) {
        Collection<People> peoples = Arrays.asList(
            new People("Ivan", 16),
            new People("Petr", 23),
            new People("Maria", 42)
        );
        Stream<People> my_stream = peoples.stream();
        System.out.println(my_stream.sorted((o1,o2) -> -
o1.getName().compareTo(o2.getName())).collect(Collectors.toList()));
    }
}

```

```

C:\Users\stale\.jdk\openjdk-17.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA\lib\idea_rt.jar" -Dfile.encoding=UTF-8
[People{name='Petr', age=23}, People{name='Maria', age=42}, People{name='Ivan', age=16}]

Process finished with exit code 0

```

Рисунок 4 - Результат выполнения кода решения подзадачи 2

Вывод:

При выполнении лабораторной работы были получены навыки работы со Stream API в Java.

