|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

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

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 9 |

**Название:** Stream API

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | С.В. Астахов | |
|  | (Группа) |  | (Подпись, дата) | | (И.О. Фамилия) |
|  |  |  |  | |  |
| Преподаватель |  |  |  | | П.В. Степанов |
|  |  |  | (Подпись, дата) | | (И.О. Фамилия) |

Москва, 2024

**Цель работы:** освоить базовые принципы работы со Stream API на языке Java.

**Вариант: 1.**

**Задание 1:** Задана коллекция строк. Вернуть последний элемент и третий элемент коллекции.

Код решения приведен в листинге 1.

Листинг 1 — реализация решения

|  |
| --- |
| package org.example;  import java.io.\*;  import java.util.\*;  import java.util.concurrent.atomic.AtomicBoolean;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  import java.util.stream.Collectors;  import java.util.stream.IntStream;  public class Main {  public static void main(String[] args) throws IOException, InterruptedException {  System.out.println("=== Stram API work example ===\n\n");  ArrayList<String> st = new ArrayList<>();  st.add("st1");  st.add("st2");  st.add("st3");  st.add("st4");  st.add("st5");  st.add("st6");  System.out.printf("ArrayList: %s\n", st);  System.out.printf("ArrayList[last]: %s\n", st.stream().reduce((first, second) -> second).orElse(null));  System.out.printf("ArrayList[2]: %s\n", st.stream().skip(2).findFirst().get());  }  } |

Результат исполнения программы приведен в листинге 2.

Листинг 2 — результат исполнения программы

|  |
| --- |
| === Stram API work example ===  ArrayList: [st1, st2, st3, st4, st5, st6]  ArrayList[last]: st6  ArrayList[2]: st3 |

**Задание 2:** Задана коллекция строк. Вернуть первый элемент коллекции, а также существуют ли все совпадения с шаблоном. Шаблон можно выбрать произвольно.

Код решения приведен в листинге 3.

Листинг 3 — реализация решения

|  |
| --- |
| package org.example;  import java.io.\*;  import java.util.\*;  import java.util.concurrent.atomic.AtomicBoolean;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  import java.util.stream.Collectors;  import java.util.stream.IntStream;  public class Main {  public static void main(String[] args) throws IOException, InterruptedException {  System.out.println("=== Stram API work example ===\n\n");  ArrayList<String> st = new ArrayList<>();  st.add("st1");  st.add("st2");  st.add("st3");  st.add("apple");  st.add("st5");  st.add("st6");  String rg1 = ".\*app.\*";  String rg2 = ".\*pineapple.\*";  System.out.printf("ArrayList: %s\n", st);  System.out.printf("ArrayList[first]: %s\n", st.stream().findFirst().get());  System.out.printf("ArrayList[regex='%s']: %b\n", rg1, st.stream().anyMatch(row -> row.matches(rg1)));  System.out.printf("ArrayList[regex='%s']: %b\n", rg2, st.stream().anyMatch(row -> row.matches(rg2)));  }  } |

Результат исполнения программы приведен в листинге 4.

Листинг 4 — результат исполнения программы

|  |
| --- |
| === Stram API work example ===  ArrayList: [st1, st2, st3, apple, st5, st6]  ArrayList[first]: st1  ArrayList[regex='.\*app.\*']: true  ArrayList[regex='.\*pineapple.\*']: false |

**Задание 3:** Задана коллекция строк. Отсортировать значения по алфавиту и убрать повторы.

Код решения приведен в листинге 5.

Листинг 5 — реализация решения

|  |
| --- |
| package org.example;  import java.io.\*;  import java.util.\*;  import java.util.concurrent.atomic.AtomicBoolean;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  import java.util.stream.Collectors;  import java.util.stream.IntStream;  public class Main {  public static void main(String[] args) throws IOException, InterruptedException {  System.out.println("=== Stram API work example ===\n\n");  ArrayList<String> st = new ArrayList<>();  st.add("st1");  st.add("st2");  st.add("st3");  st.add("apple");  st.add("st5");  st.add("st5");  st.add("st6");  String rg1 = ".\*app.\*";  String rg2 = ".\*pineapple.\*";  System.out.printf("ArrayList: %s\n", st);  System.out.printf("ArrayList[sorted, uniq]: %s\n", st  .stream().collect(Collectors.toSet())  .stream().collect(Collectors.toList())  .stream().sorted()  .collect(Collectors.toList()));  }  } |

Результат исполнения программы приведен в листинге 6.

Листинг 6 — результат исполнения программы

|  |
| --- |
| === Stram API work example ===  ArrayList: [st1, st2, st3, apple, st5, st5, st6]  ArrayList[sorted, uniq]: [apple, st1, st2, st3, st5, st6] |

**Задание 4:** Задана коллекция: (Класс People: имя и возраст); Отсортировать по имени в обратном алфавитном порядке.

Код решения приведен в листинге 7.

Листинг 7 — реализация решения

|  |
| --- |
| package org.example;  import java.io.\*;  import java.util.\*;  import java.util.concurrent.atomic.AtomicBoolean;  import java.util.regex.Matcher;  import java.util.regex.Pattern;  import java.util.stream.Collectors;  import java.util.stream.IntStream;  public class Main {  public static void main(String[] args) throws IOException, InterruptedException {  System.out.println("=== Stram API work example ===\n\n");  Collection<People> peoples = Arrays.asList(  new People("Ivan", 16),  new People("Petr", 23),  new People("Maria", 42)  );  Comparator<Integer> customComparator = (a, b) -> {  // Custom comparison logic between a and b  // Return a negative value if a should come before b, positive if a should come after b, and 0 if they are equal  return a - b; // Example: Sorting in ascending order  };  System.out.printf("ArrayList: %s\n", peoples);  System.out.printf("List[reversed]: %s\n", peoples  .stream()  .sorted(Comparator.comparing(People::getName))  .collect(Collectors.toList()).reversed()  );  }  }  class People {  public String name;  public int age;  public String toString(){  return String.format("%s-%d", name, age);  }  public People(String name, int age) {  this.name = name;  this.age = age;  }  public String getName() {  return name;  }  } |

Результат исполнения программы приведен в листинге 8.

Листинг 8 — результат исполнения программы

|  |
| --- |
| === Stram API work example ===  ArrayList: [Ivan-16, Petr-23, Maria-42]  List[reversed]: [Petr-23, Maria-42, Ivan-16] |

**Вывод:** в ходе лабораторной работы были освоены базовые принципы работы со Stream API на языке Java.