|  |  |
| --- | --- |
|  | **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**

**Отчет**

**по лабораторной работе № 9**

**Название:** Stream API

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

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

Москва, 2023

1. ****Введение и задание****

**Целью лабораторной работы является формирование навыков использования Stream API при работе с языком программирования java, изучение соответствующих синтаксических конструкций и особенностей работы.**

**Номер в списке – 10.**

**Вариант 1.**

**Задание 10.** Задана коллекция:

(Класс Student: имя и рейтинг)

Collection<Student> students = Arrays.asList( new Student("Ivan", 40), new Student("Petr", 60), new Student("Olga", 70) );

Вернуть список студентов имя которых начинается на P и рейтинг находится в интервале 40-60.

**Задание 11.** Коллекция из 10 задания. Вернуть средний балл.

**Вариант 2.**

**Задание 10.** Задана коллекция чисел. С помощью метода reduce вернуть сумму чисел, которые > 1

**Задание 11. Задана коллекция:**

**(Класс People: имя и возраст, пол (enum))Collection<People> peoples = Arrays.asList( new People("Ivan", 16, Sex.MAN), new People("Petr", 23, Sex.MAN), new People("Maria", 42, Sex.WOMAN));**

**Найти самого старшего человека мужского пола.**

1. ****Ход работы****

**Выполнение задания №10 первого варианта лабораторной работы:**

**import java.util.Arrays;**

**import java.util.Collection;**

**import java.util.stream.Collectors;**

**public class Main {**

**public static void main(String[] args) {**

**Collection<Student> students = Arrays.asList(**

**new Student("Ivan", 40),**

**new Student("Petr", 60),**

**new Student("Olga", 70)**

**);**

**System.out.println(students.stream().filter(s -> s.name().charAt(0) == 'P'**

**&& s.rate() <= 60**

**&& s.rate() >= 40)**

**.collect(Collectors.toList()));**

**}**

**}**

**record Student(String name, int rate) {}**

**Выполнение задания №11 первого варианта лабораторной работы:**

**import java.util.ArrayList;**

**import java.util.Calendar;**

**import java.util.Collection;**

**import java.util.stream.IntStream;**

**import java.util.Random;**

**public class Main {**

**public static final Random RNG = new Random(Calendar.getInstance().getTimeInMillis());**

**public static void main(String[] args) {**

**Collection<Assignement> assignements = new ArrayList<>();**

**IntStream.range(0, 10).forEach(i -> assignements.add(new Assignement(i, RNG.nextInt(60,100))));**

**System.out.println(assignements.stream().mapToDouble(Assignement::mark).average().orElse(0.0));**

**}**

**}**

**record Assignement(int info, int mark){}**

**Выполнение задания №10 второго варианта лабораторной работы:**

**import java.util.Arrays;**

**import java.util.List;**

**public class Main {**

**public static void main(String[] args) {**

**List<Integer> numbers = Arrays.asList(4, 8, 15, 16, 23, 42);**

**System.out.println(numbers.stream()**

**.filter(number -> number > 10)**

**.reduce(0, Integer::sum));**

**}**

**}**

**Выполнение задания №11 второго варианта лабораторной работы:**

**import java.util.Arrays;**

**import java.util.Collection;**

**public class Main {**

**public static void main(String[] args) {**

**Collection<Person> people = Arrays.asList(**

**new Person("Ivan", 16, Person.Gender.MALE),**

**new Person("Petr", 23, Person.Gender.MALE),**

**new Person("Maria", 42, Person.Gender.FEMALE)**

**);**

**System.out.println(people.stream().filter(p -> p.gender() == Person.Gender.MALE)**

**.max((p1, p2) -> Integer.compare(p1.age(), p2.age())));**

**}**

**}**

**record Person(String name, int age, Gender gender) {**

**public enum Gender {**

**MALE,**

**FEMALE,**

**OTHER**

**}**

**}**

1. ****Заключение****

**В процессе выполнения лабораторной работы были сформированы навыки использования Stream API при работе с языком программирования java, были изучены соответствующие синтаксические конструкции и особенности работы.**