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 «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 5

по курсу «Языки и методы программирования»

«Монады в языке Java»

Студент группы ИУ9-21Б Горбунов А. Д.

Преподаватель Посевин Д. П.

Москва 2023

1 Задание

Множество строк вида «Фамилия Имя Отчество, год рождения» с операциями:

1. порождение потока имён;

2. поиск в множестве самого старшего обладателя указанной фамилии.

Проверить работу первой операции нужно путём поиска наиболее часто встречающегося имени.

2 Результаты

Исходный код программы представлен в листинге 1, 2, 3, 4, 5

Листинг 1 — класс Test

```
public class Test {
    public static void main(String[] args) {
        PersonTable t = new PersonTable();
        t.add("a", "aa", "aaa", 1);
        t.add("b", "bb", "bbb", 10);
        t.add("c", "asf", "ccc", 3);
        t.add("d", "dd", "ddd", 4);
        t.add("gc", "rf", "sv", 5);
        t.add("gac", "cc", "fr", 2);
        t.add("gac", "cc", "fr", 7);
        t.add(new Person("as","cc","kef", 8));
        t.firstNameStream().forEach(x-> System.out.println(x.getKey()));
        System.out.println(t.getMaxAge("gac").get().age);
    }
}
```

Листинг 2 — класс PersonTable

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Optional;
import java.util.stream.Stream;
public class PersonTable {
    HashMap<String, Person> Table;
    HashMap<String, Integer> firstnames;
```

Листинг 3 — класс PersonTable(продолжение)

```
    private int maxAge = -1;
    private int maxNumbersFirstname = 1;
    PersonTable() {
        Table = new HashMap<>();
        firstnames = new HashMap<>();
    }
    void add(Person p) {
        if(firstnames.containsKey(p.firstname)){
            Integer f = firstnames.get(p.firstname);
            firstnames.remove(p.firstname);
            firstnames.put(p.firstname, f+1);
            if(maxNumbersFirstname <f+1){
                maxNumbersFirstname = f+1;
            }

        }else{
            firstnames.put(p.firstname, 1);
        }
        Table.put(new String[]{p.firstname, p.lastname,p.surname}, p);

        if(maxAge< p.age){
            maxAge=p.age;
        }
    }
    void add(String lastname, String firstname, String surname, int age)
    {

        if(firstnames.containsKey(firstname)){
            Integer f = firstnames.get(firstname);
            firstnames.remove(firstname);
            firstnames.put(firstname, f+1);
            if(maxNumbersFirstname <f+1){maxNumbersFirstname = f+1;}
        }else{firstnames.put(firstname, 1); }
        Table.put(new String[]{firstname, lastname, surname}, new Person
    (lastname, firstname, surname, age));
        if(maxAge< age){maxAge= age;}
    }
    public Stream<Map.Entry<String, Integer>> firstNameStream() {
        ArrayList<Map.Entry<String, Integer>> result = new ArrayList<>()
    ;
        firstnames.entrySet().stream().filter(x-> firstnames.get(x.
    getKey()) == getMaxNumbersFirstname()).forEach(x -> result.add(x));
        return result.stream();
    }
    public Optional<Person> getMaxAge(String lastname) {
        Optional<Person> result = Optional.empty();
        Optional<Map.Entry<String[], Person>> tmp = Table.entrySet().
    stream().sorted(new NameComparator()).filter(x-> (x.getKey()[1].
    equals(lastname))).findFirst();
        if (tmp.isPresent()) {
            result = Optional.ofNullable(tmp.get().getValue());}
        return result;
    }
    public int getMaxNumbersFirstname(){
        return maxNumbersFirstname;}
}
```

Листинг 4 — класс Person

```
public class Person {
    int age;
    String firstname, lastname, surname;
    public Person (String lastname, String firstname, String surname,
    int age) {
        this.lastname = lastname;
        this.firstname = firstname;
        this.surname = surname;
        this.age = age;
    }
}
```

Листинг 5 — класс NameComparator

```
import java.util.*;
class NameComparator implements Comparator<Map.Entry<String[], Person>>
    {
    public int compare(Map.Entry<String[], Person>  a, Map.Entry<String
    [], Person> b) {
        int a0, b0;
        a0 = a.getValue().age;
        b0 = b.getValue().age;
        if (a0 < b0) { return 1; }
        if (a0 == b0) { return 0; }
        return -1;
    }
}
```

Результат запуска представлен на рисунке 1, 2, 3.

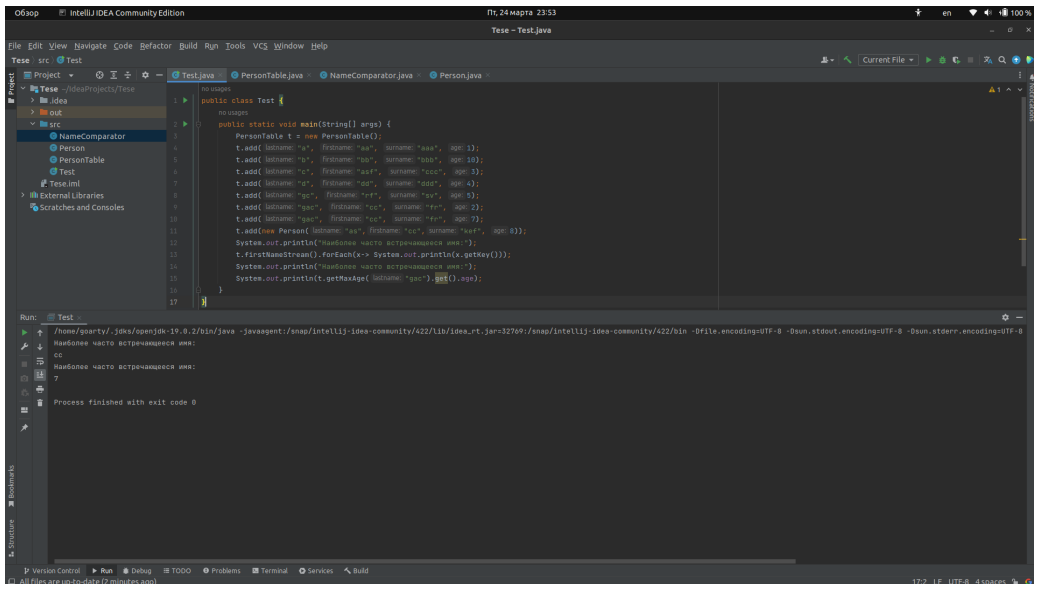

Рис. 1 — Вывод программы(3 три повторяющихся имени cc и две повторяющиеся фамилии gac(age 7 и 2))

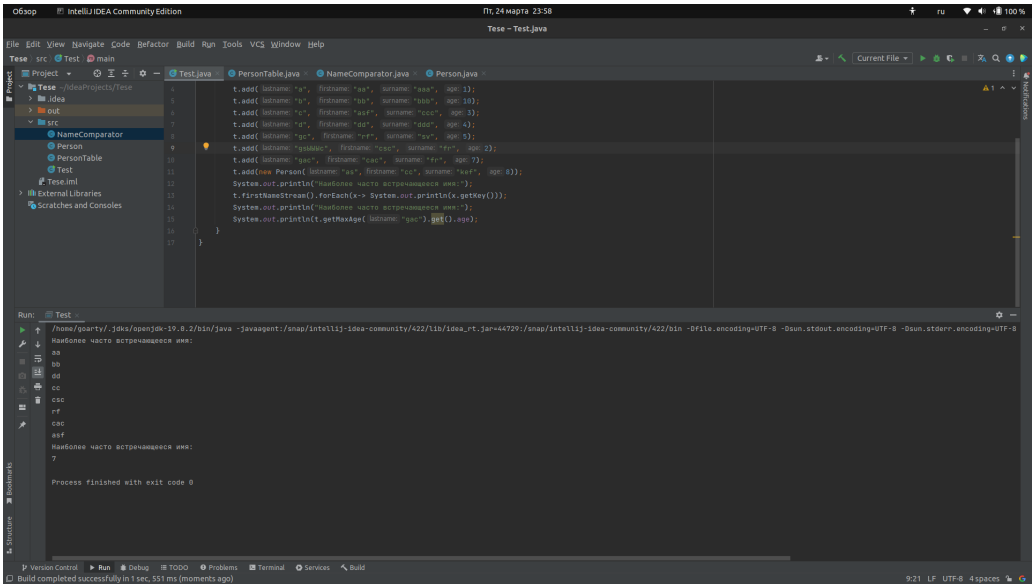

Рис. 2 — Вывод программы(нет одинаковых имён и только одна фамилия gas (age 7))

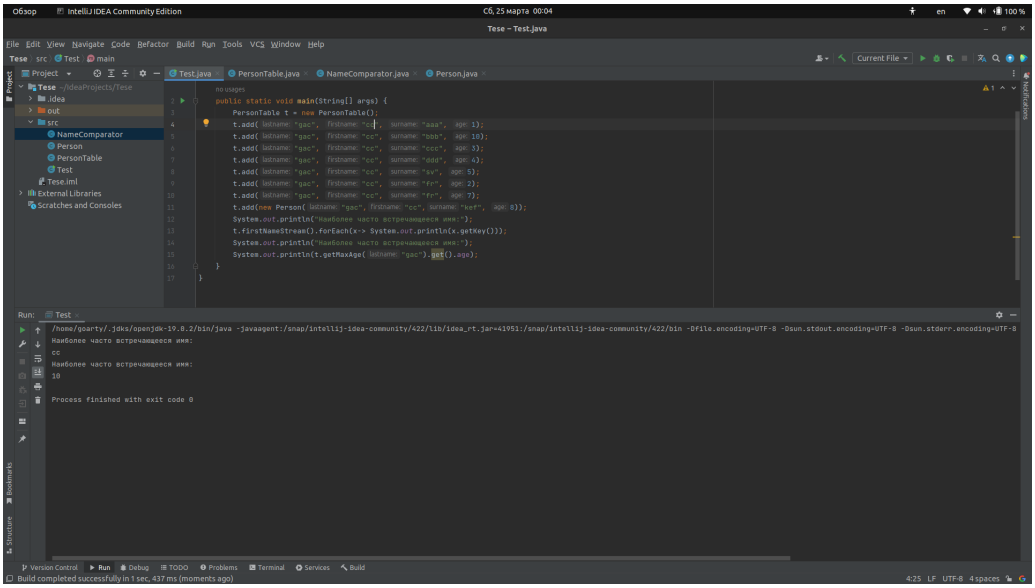

Рис. 3 — Вывод программы(всех зовут cc и у всех фамилия gas)