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 **09.04.01/12 Интеллектуальный анализ больших данных в системах поддержки принятия решений.**

О Т Ч Е Т

по лабораторной работе № 3

Вариант № 7

Название: классы, наследование, полиморфизм

Дисциплина: языки программирования для работы с большими данными

Студент ИУ6-23М (Группа) _____ (Подпись, дата) Ф.А. Лучкин (И.О. Фамилия)

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

Москва, 2024

Цель: освоить принципы работы с классами, наследованием и полиморфизмом на языке программирования Java.

Задание 1: определить класс Дробь в виде пары (m,n). Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения и деления дробей. Объявить массив из k дробей, ввести/вывести значения для массива дробей. Создать массив объектов и передать его в метод, который изменяет каждый элемент массива с четным индексом путем добавления следующего за ним элемента массива.

Код класса Fraction:

```
package lab3_var1_7;

import static lab1_var2_7.Main.getGreatestCommonDivisor;


public class Fraction {
    private int m;
    private int n;

    public Fraction() {
        this.setM(1);
        this.setN(1);
    }
    public Fraction(int m, int n) {
        this.setM(m);
        this.setN(n);
    }

    public int getM() {
        return this.m;
    }

    public void setM(int m) {
        this.m = m;
    }

    public int getN() {
        return this.n;
    }

    public void setN(int n) {
        this.n = n;
    }

    public void print() {
        System.out.printf("%d/%d ", m, n);
    }

    public static Fraction add(Fraction first, Fraction second) {
        int denominator = first.getN() * second.getN();
        int numerator = first.getM() * second.getN() + first.getN() *
second.getM();
        return signFix(numerator, denominator);
    }

    public static Fraction subtract(Fraction first, Fraction second) {
```

```
        int denominator = first.getN() * second.getN();
        int numerator = first.getM() * second.getN() - first.getN() *
second.getM();
        return signFix(numerator, denominator);
    }

    public static Fraction multiply(Fraction first, Fraction second) {
        int numerator = first.getM() * second.getM();
        int denominator = first.getN() * second.getN();
        return signFix(numerator, denominator);
    }

    public static Fraction divide(Fraction first, Fraction second) {
        int numerator = first.getM() * second.getN();
        int denominator = first.getN() * second.getM();
        return signFix(numerator, denominator);
    }

    private static Fraction signFix(int numerator, int denominator) {
        int commonDivisor = getGreatestCommonDivisor(numerator, denominator);
        numerator = numerator/ commonDivisor;
        denominator = denominator / commonDivisor;
        if (denominator < 0) {
            denominator = denominator * -1;
            numerator = numerator * -1;
        }
        return new Fraction(numerator, denominator);
    }
}
```

Код класса Main:

```
package lab3_var1_7;

import java.util.Random;


public class Main {
    public static void main(String[] args) {
        Random random = new Random();

        /*Fraction first = new Fraction();
        Fraction second = new Fraction(1, 2);
        operationsDemonstration(first, second);

        first.setN(4);
        second.setM(3);
        second.setN(8);
        operationsDemonstration(first, second);

        first.setM(5);
        first.setN(6);
        second.setM(6);
        second.setN(7);
        operationsDemonstration(first, second);
        System.out.println("TASK:");*/

        int arrSize = random.nextInt(9) + 2;
        Fraction[] fractions = new Fraction[arrSize];

        System.out.println("Source array:");
        for (int i = 0; i < arrSize; i++) {
            int n = random.nextInt(8) + 2;
            int m = random.nextInt(n) + 1;
            fractions[i] = new Fraction(m, n);
```

```
            fractions[i].print();
        }

        System.out.println("\nResult array:");
        for (Fraction fraction : taskMethod(fractions)) {
            fraction.print();
        }

    }

    private static Fraction[] taskMethod(Fraction[] fractions) {
        for (int i = 0; i < fractions.length; i++) {
            if ((i % 2 == 0) && (i < fractions.length - 1)) {
                fractions[i] = Fraction.add(fractions[i], fractions[i+1]);
            }
        }
        return fractions;
    }

    private static void operationsDemonstration(Fraction first, Fraction second) {
        Fraction fractionSum = Fraction.add(first, second);
        Fraction fractionDiff = Fraction.subtract(first, second);
        Fraction fractionMultiple = Fraction.multiply(first, second);
        Fraction fractionQuotient = Fraction.divide(first, second);
        System.out.printf("First fraction: %d/%d\n", first.getM(), first.getN());
        System.out.printf("Second fraction: %d/%d\n", second.getM(),
second.getN());
        System.out.printf("Sum fraction: %d/%d\n", fractionSum.getM(),
fractionSum.getN());
        System.out.printf("Diff fraction: %d/%d\n", fractionDiff.getM(),
fractionDiff.getN());
        System.out.printf("Multiple fraction: %d/%d\n",
                fractionMultiple.getM(), fractionMultiple.getN());
        System.out.printf("Quotient of fractions: %d/%d\n--------------------\n",
                fractionQuotient.getM(), fractionQuotient.getN());
    }
}
```

Работа программы показана на рисунке 1.

```
Source array:
1/5 1/9 2/3 3/5 1/7 1/2 8/9
Result array:
14/45 1/9 19/15 3/5 9/14 1/2 8/9
```

Рисунок 1 – Работа программы

Задание 2: определить класс Комплекс. Класс должен содержать несколько конструкторов. Реализовать методы для сложения, вычитания, умножения, деления, присваивания комплексных чисел. Создать два вектора размерности n из комплексных координат. Передать их в метод, который выполнит их сложение.

Код класса ComplexNumber:

```
package lab3_var1_8;


public class ComplexNumber {
    private double re;
    private double im;

    public ComplexNumber() {
        this.setRe(1);
        this.setIm(0);
    }
    public ComplexNumber(double re, double im) {
        this.setRe(re);
        this.setIm(im);
    }

    public double getRe() {
        return this.re;
    }

    public void setRe(double re) {
        this.re = re;
    }

    public double getIm() {
        return this.im;
    }

    public void setIm(double im) {
        this.im = im;
    }

    public void setComplexNumber(double re, double im) {
        this.re = re;
        this.im = im;
    }

    public void print() {
        System.out.printf("(%.3f ; %.3f)", re, im);
    }

    public static ComplexNumber add(ComplexNumber first, ComplexNumber second) {
        return new ComplexNumber(
                first.getRe() + second.getRe(),
                first.getIm() + second.getIm()
        );
    }

    public static ComplexNumber subtract(ComplexNumber first, ComplexNumber
second) {
        return new ComplexNumber(
                first.getRe() - second.getRe(),
                first.getIm() - second.getIm()
        );
    }

    public static ComplexNumber multiply(ComplexNumber first, ComplexNumber
second) {
        double operationRe = first.getRe() * second.getRe() - first.getIm() *
second.getIm();
        double operationIm = first.getRe() * second.getIm() + first.getIm() *
second.getRe();
        return new ComplexNumber(operationRe, operationIm);
```

```
    }

    public static ComplexNumber divide(ComplexNumber first, ComplexNumber second)
{
        double operationRe = (first.getRe() * second.getRe() +
                first.getIm() * second.getIm());
        double operationIm = (first.getIm() * second.getRe() -
                first.getRe() * second.getIm());
        double divisor = second.getRe() * second.getRe() +
                second.getIm() * second.getIm();

        return new ComplexNumber(operationRe / divisor, operationIm / divisor);
    }
}
```

Код класса Main:

```
package lab3_var1_8;

import java.util.Random;


public class Main {
    public static void main(String[] args) {
        Random random = new Random();

        /*ComplexNumber first = new ComplexNumber();
        ComplexNumber second = new ComplexNumber(1, 2);
        operationsDemonstration(first, second);

        first.setIm(4);
        second.setRe(3);
        second.setIm(8);
        operationsDemonstration(first, second);

        first.setRe(5);
        first.setIm(6);
        second.setRe(6);
        second.setIm(7);
        operationsDemonstration(first, second);*/
        //System.out.println("TASK:");

        int n = random.nextInt(9) + 2;
        ComplexNumber[] firstVector = new ComplexNumber[n];
        ComplexNumber[] secondVector = new ComplexNumber[n];

        System.out.println("Source vectors:");
        for (int i = 0; i < n; i++) {
            int re = random.nextInt(10) - random.nextInt(10);
            int im = random.nextInt(10) - random.nextInt(10);
            firstVector[i] = new ComplexNumber(re, im);

            re = random.nextInt(10) - random.nextInt(10);
            im = random.nextInt(10) - random.nextInt(10);
            secondVector[i] = new ComplexNumber(re, im);
        }
        for (ComplexNumber numbers : firstVector) {
            numbers.print();
            System.out.print(' ');
        }
        System.out.print('\n');
        for (ComplexNumber numbers : secondVector) {
            numbers.print();
            System.out.print(' ');
```

```
        }

        System.out.println("\nResult vector:");
        for (ComplexNumber numbers : taskMethod(firstVector, secondVector)) {
            numbers.print();
            System.out.print(' ');
        }
    }

    private static ComplexNumber[] taskMethod(ComplexNumber[] firstVector,
ComplexNumber[] secondVector) {
        ComplexNumber[] resultVector = new ComplexNumber[firstVector.length];
        for (int i = 0; i < firstVector.length; i++) {
            resultVector[i] = ComplexNumber.add(firstVector[i], secondVector[i]);
        }
        return resultVector;
    }

    private static void operationsDemonstration(ComplexNumber first, ComplexNumber
second) {
        ComplexNumber fractionSum = ComplexNumber.add(first, second);
        ComplexNumber fractionDiff = ComplexNumber.subtract(first, second);
        ComplexNumber fractionMultiple = ComplexNumber.multiply(first, second);
        ComplexNumber fractionQuotient = ComplexNumber.divide(first, second);
        System.out.printf("First number: (%.3f ; %.3f)\n", first.getRe(),
first.getIm());
        System.out.printf("Second number: (%.3f ; %.3f)\n", second.getRe(),
second.getIm());
        System.out.printf("Sum number: (%.3f ; %.3f)\n", fractionSum.getRe(),
fractionSum.getIm());
        System.out.printf("Diff number: (%.3f ; %.3f)\n", fractionDiff.getRe(),
fractionDiff.getIm());
        System.out.printf("Multiple number: (%.3f ; %.3f)\n",
                fractionMultiple.getRe(), fractionMultiple.getIm());
        System.out.printf("Quotient of numbers: (%.3f ; %.3f)\n------------------
-\n",
                fractionQuotient.getRe(), fractionQuotient.getIm());
    }
}
```

Работа программы показана на рисунке 2.

```
Source vectors:
(-4,000 ; 2,000) (-2,000 ; -1,000) (2,000 ; 5,000) (2,000 ; -8,000) (-4,000 ; -3,000)
(7,000 ; -3,000) (-5,000 ; -2,000) (2,000 ; 0,000) (2,000 ; -4,000) (-8,000 ; -1,000)
Result vector:
(3,000 ; -1,000) (-7,000 ; -3,000) (4,000 ; 5,000) (4,000 ; -12,000) (-12,000 ; -4,000)
```

Рисунок 2 – Работа программы

Задание 3: создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. Phone: id, Фамилия, Имя, Отчество, Адрес, Номер кредитной карточки, Дебет, Кредит, Время городских и междугородных разговоров. Создать массив объектов. Вывести: а) сведения об

абонентах, у которых время внутригородских разговоров превышает заданное; b) сведения об абонентах, которые пользовались междугородной связью; c) сведения об абонентах в алфавитном порядке.

Код класса Phone:

```
package lab3_var2_7;


public class Phone implements Comparable<Phone> {
    private int id;
    private String lastname;
    private String firstname;
    private String surname;
    private String address;
    private int creditCard;
    private double debit;
    private double credit;

    private int cityCallsMinutes;
    private int longDistanceCallsMinutes;

    public Phone(int id, String lastname, String firstname, String surname,
                 String address, int creditCard, double debit, double credit,
                 int cityCallsMinutes, int longDistanceCallsMinutes) {
        this.setId(id);
        this.setLastname(lastname);
        this.setFirstname(firstname);
        this.setSurname(surname);
        this.setAddress(address);
        this.setCreditCard(creditCard);
        this.setDebit(debit);
        this.setCredit(credit);
        this.setCityCallsMinutes(cityCallsMinutes);
        this.setLongDistanceCallsMinutes(longDistanceCallsMinutes);
    }

    @Override
    public String toString() {
        return String.format("""
                Phone = {
                \tid: %d; lastname: %s; firstname: %s; surname: %s;
                \taddress: %s; creditCard: %d;
                \tdebit: %.3f; credit: %.3f;
                \tcityCallsMinutes: %d; longDistanceCallsMinutes: %d
                }""",
                id, lastname, firstname, surname, address,creditCard, debit,
credit, cityCallsMinutes, longDistanceCallsMinutes);
    }

    @Override
    public int compareTo(Phone p)
    {
        int result = this.getLastname().compareTo(p.getLastname());
        if (result != 0) {
            return result;
        } else {
            result = this.getFirstname().compareTo(p.getFirstname());
        }
        if (result != 0) {
            return result;
        } else {
            return this.getSurname().compareTo(p.getSurname());
```

```
        }
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getLastname() {
        return lastname;
    }

    public void setLastname(String lastname) {
        this.lastname = lastname;
    }

    public String getFirstname() {
        return firstname;
    }

    public void setFirstname(String firstname) {
        this.firstname = firstname;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public int getCreditCard() {
        return creditCard;
    }

    public void setCreditCard(int creditCard) {
        this.creditCard = creditCard;
    }

    public double getDebit() {
        return debit;
    }

    public void setDebit(double debit) {
        this.debit = debit;
    }

    public double getCredit() {
        return credit;
    }

    public void setCredit(double credit) {
        this.credit = credit;
```

```
    }

    public int getCityCallsMinutes() {
        return cityCallsMinutes;
    }

    public void setCityCallsMinutes(int cityCallsMinutes) {
        this.cityCallsMinutes = cityCallsMinutes;
    }

    public int getLongDistanceCallsMinutes() {
        return longDistanceCallsMinutes;
    }

    public void setLongDistanceCallsMinutes(int longDistanceCallsMinutes) {
        this.longDistanceCallsMinutes = longDistanceCallsMinutes;
    }
}
```

Код класса Main:

```
package lab3_var2_7;

import java.util.Arrays;
import java.util.Random;
import java.util.Scanner;
import static java.lang.System.*;


public class Main {
    public static void main(String[] args) {
        Random random = new Random();
        Scanner scanner = new Scanner(in);

        int n = random.nextInt(5) + 2;
        out.printf("Input info about %d users\n", n);
        Phone[] users = new Phone[n];

        for (int i = 0; i < n; i++) {
            int id = i + 1;
            out.printf("Input lastname for user №%d:\n", id);
            String lastname = scanner.nextLine();
            out.printf("Input firstname for user №%d:\n", id);
            String firstname = scanner.nextLine();
            out.printf("Input surname for user №%d:\n", id);
            String surname = scanner.nextLine();
            out.printf("Input address for user №%d:\n", id);
            String address = scanner.nextLine();

            int creditCard = id * 1000;
            int debit = random.nextInt(1000000);
            int credit = random.nextInt(1000000);
            int cityCallsMinutes = random.nextInt(30);
            int longDistanceCallsMinutes = random.nextInt(30) * (i % 3);

            users[i] = new Phone(id, lastname, firstname, surname,
                                 address, creditCard, debit, credit,
                                 cityCallsMinutes, longDistanceCallsMinutes);
        }
        out.println("Source array:");
        printAll(users);
        out.println("------------------\nSet limit for city calls:");
        int limit = scanner.nextInt();
        out.println("------------------\nExceeded the time limit:");
        exceedingCityCallsTime(limit, users);
        out.println("------------------\nLong distance calls usage:");
```

```
        longDistanceCallsUsage(users);
        out.println("------------------\nSorted by full name:");
        Arrays.sort(users);
        printAll(users);
    }

    private static void printAll(Phone[] users) {
        for (Phone user : users) {
            out.println(user);
        }
    }
    private static void exceedingCityCallsTime(int limit, Phone[] users) {
        int flag = 0;
        for (Phone user : users) {
            if (user.getCityCallsMinutes() > limit) {
                out.println(user);
                flag = 1;
            }
        }
        if (flag == 0) out.println("No one");
    }

    private static void longDistanceCallsUsage(Phone[] users) {
        int flag = 0;
        for (Phone user : users) {
            if (user.getLongDistanceCallsMinutes() > 0) {
                out.println(user);
                flag = 1;
            }
        }
        if (flag == 0) out.println("No one");
    }
}
```

Работа программы показана на рисунках 3-4.

```
Input info about 2 users
Input lastname for user №1:
Luchkin
Input firstname for user №1:
Fedor
Input surname for user №1:
Antonovich
Input address for user №1:
Borovaya 8
Input lastname for user №2:
Kochikina
Input firstname for user №2:
Lada
Input surname for user №2:
Viacheslavovna
Input address for user №2:
Borovaya 8
Source array:
Phone = {
    id: 1; lastname: Luchkin; firstname: Fedor; surname: Antonovich;
    address: Borovaya 8; creditCard: 1000;
    debit: 751494,000; credit: 309032,000;
    cityCallsMinutes: 22; longDistanceCallsMinutes: 0
}
Phone = {
    id: 2; lastname: Kochikina; firstname: Lada; surname: Viacheslavovna;
    address: Borovaya 8; creditCard: 2000;
    debit: 17262,000; credit: 899737,000;
    cityCallsMinutes: 4; longDistanceCallsMinutes: 23
}
-----------------
```

Рисунок 3 – Работа программы

```
Set limit for city calls:
10
-----------------
Exceeded the time limit:
Phone = {
    id: 1; lastname: Luchkin; firstname: Fedor; surname: Antonovich;
    address: Borovaya 8; creditCard: 1000;
    debit: 751494,000; credit: 309032,000;
    cityCallsMinutes: 22; longDistanceCallsMinutes: 0
}
-----------------
Long distance calls usage:
Phone = {
    id: 2; lastname: Kochikina; firstname: Lada; surname: Viacheslavovna;
    address: Borovaya 8; creditCard: 2000;
    debit: 17262,000; credit: 899737,000;
    cityCallsMinutes: 4; longDistanceCallsMinutes: 23
}
-----------------
Sorted by full name:
Phone = {
    id: 2; lastname: Kochikina; firstname: Lada; surname: Viacheslavovna;
    address: Borovaya 8; creditCard: 2000;
    debit: 17262,000; credit: 899737,000;
    cityCallsMinutes: 4; longDistanceCallsMinutes: 23
}
Phone = {
    id: 1; lastname: Luchkin; firstname: Fedor; surname: Antonovich;
    address: Borovaya 8; creditCard: 1000;
    debit: 751494,000; credit: 309032,000;
    cityCallsMinutes: 22; longDistanceCallsMinutes: 0
}
```

Рисунок 4 – Работа программы

Задание 4: создать классы, спецификации которых приведены ниже. Определить конструкторы и методы setТип(), getТип(), toString(). Определить дополнительно методы в классе, создающем массив объектов. Задать критерий выбора данных и вывести эти данные на консоль. Car: id, Марка, Модель, Год выпуска, Цвет, Цена, Регистрационный номер. Создать массив объектов. Вывести: a) список автомобилей заданной марки; b) список автомобилей заданной модели, которые эксплуатируются больше n лет; c) список автомобилей заданного года выпуска, цена которых больше указанной.

Вычислить определитель матрицы.

Код модуля Car:

```java
package lab3_var2_8;

import java.text.DateFormat;
import java.text.SimpleDateFormat;


public class Car {
    private int id;
    private String brand;
    private String model;
    private int yearOfManufacture;
    private String color;
    private double price;
```

```
    private String regNum;

    public Car(int id, String brand, String model, int yearOfManufacture,
               String color, double price, String regNum) {
        this.setId(id);
        this.setBrand(brand);
        this.setModel(model);
        this.setYearOfManufacture(yearOfManufacture);
        this.setColor(color);
        this.setPrice(price);
        this.setRegNum(regNum);
    }

    @Override
    public String toString() {
        DateFormat dateFormat = new SimpleDateFormat("yyyy.MM.dd");
        return String.format("""
                Phone = {
                \tid: %d;
                \tbrand: %s;
                \tmodel: %s;
                \tyearOfManufacture: %s;
                \tcolor: %s;
                \tprice: %.3f;
                \tregNum: %s;
                }""",
                id, brand, model, yearOfManufacture, color, price, regNum);
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getBrand() {
        return brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public int getYearOfManufacture() {
        return yearOfManufacture;
    }

    public void setYearOfManufacture(int yearOfManufacture) {
        this.yearOfManufacture = yearOfManufacture;
    }

    public String getColor() {
        return color;
    }
```

```
    public void setColor(String model) {
        this.color = color;
    }

    public double getPrice() {
        return price;
    }

    public void setPrice(Double price) {
        this.price = price;
    }

    public String getRegNum() {
        return regNum;
    }

    public void setRegNum(String regNum) {
        this.regNum = regNum;
    }
}
```

Код модуля Main:

```
package lab3_var2_8;


import java.util.Calendar;
import java.util.Objects;
import java.util.Random;
import java.util.Scanner;

import static java.lang.System.*;


public class Main {
    public static void main(String[] args) {
        Random random = new Random();
        Scanner scanner = new Scanner(in);

        int n = random.nextInt(5) + 2;
        out.printf("Input info about %d cars\n", n);
        Car[] cars = new Car[n];

        for (int i = 0; i < n; i++) {
            int id = i + 1;
            out.printf("Input brand of car №%d:\n", id);
            String brand = scanner.nextLine();
            out.printf("Input model of car №%d:\n", id);
            String model = scanner.nextLine();
            out.printf("Input color of car №%d:\n", id);
            String color = scanner.nextLine();
            out.printf("Input regNum of car №%d:\n", id);
            String regNum = scanner.nextLine();

            int price = random.nextInt(3000000) + 1000000;
            int yearOfManufacture = random.nextInt(24) + 2000;

            cars[i] = new Car(id, brand, model, yearOfManufacture, color, price,
regNum);
        }

        printAll(cars);
        out.println("------------------\nSet brand for filter №1:");
        String brand = scanner.nextLine();
        out.printf("------------------\nFiltered by brand '%s':\n", brand);
```

```
        brandFilter(brand, cars);
        out.println("------------------\nSet model for filter №2:");
        String model = scanner.nextLine();
        out.println("------------------\nSet count of years for filter №2:");
        int yearCount = scanner.nextInt();
        out.printf("------------------\nCars with model '%s' older than %d:\n",
model, yearCount);
        oldModelFilter(model, yearCount, cars);
        out.println("------------------\nSet year of manufacture for filter №3:");
        int year = scanner.nextInt();
        out.println("------------------\nSet car price for filter №3:");
        double price = scanner.nextDouble();
        out.printf("------------------\nCars released in %d that more expensive
than %.3f:\n", year, price);
        yearPriceFilter(year, price, cars);

    }

    private static void printAll(Car[] cars) {
        for (Car car : cars) {
            out.println(car);
        }
    }

    private static void brandFilter(String brand, Car[] cars) {
        int flag = 0;
        for (Car car : cars) {
            if (Objects.equals(car.getBrand(), brand)) {
                out.println(car);
                flag = 1;
            }
        }
        if (flag == 0) out.println("No one");
    }

    private static void oldModelFilter(String model, int yearCount, Car[] cars) {
        int flag = 0;
        for (Car car : cars) {
            if ((Objects.equals(car.getModel(), model)) &&
                    ((Calendar.getInstance().get(Calendar.YEAR) -
car.getYearOfManufacture()) > yearCount)) {
                out.println(car);
                flag = 1;
            }
        }
        if (flag == 0) out.println("No one");
    }

    private static void yearPriceFilter(int year, double price, Car[] cars) {
        int flag = 0;
        for (Car car : cars) {
            if (Objects.equals(car.getYearOfManufacture(), year) &&
                    (car.getPrice() > price)) {
                out.println(car);
                flag = 1;
            }
        }
        if (flag == 0) out.println("No one");
    }
}
```

Работа программы показана на рисунках 5-6.

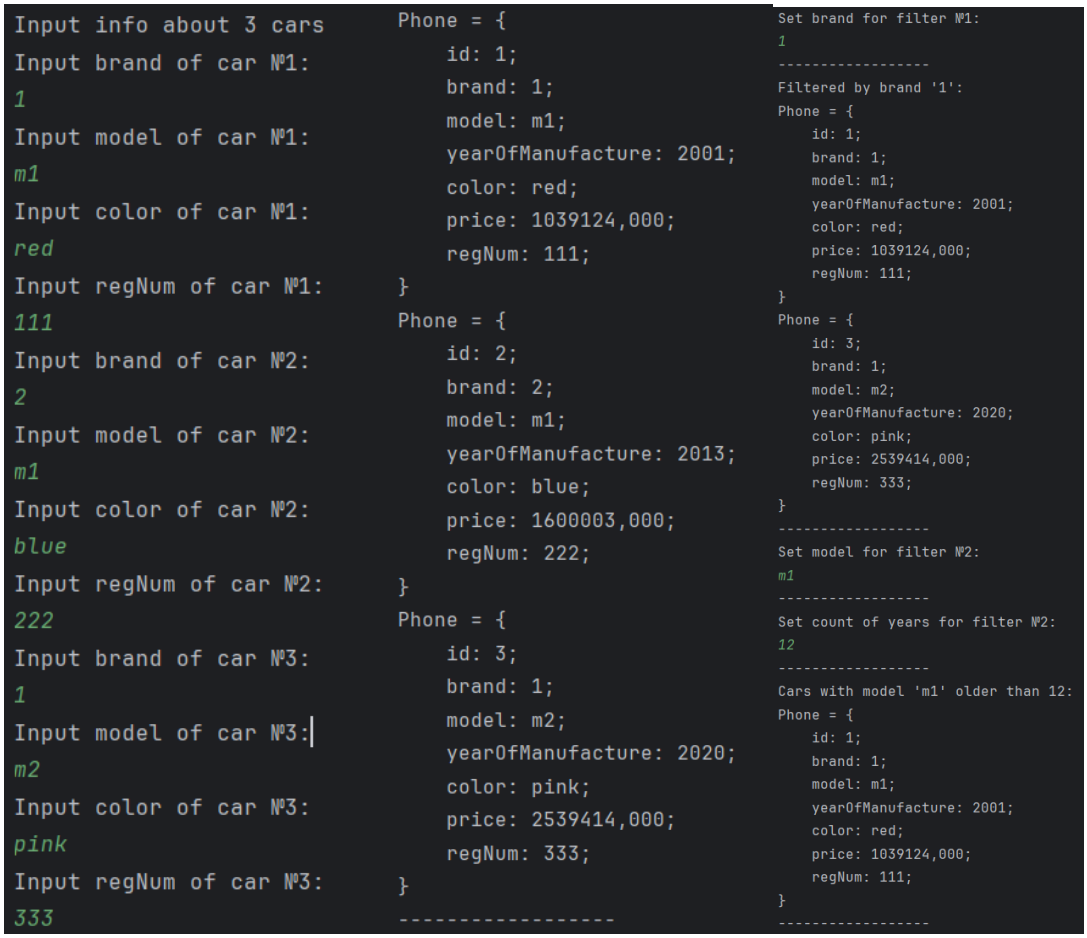

Рисунок 5 – Работа программы

Рисунок 6 – Работа программы

Задание 5: создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString(). Создать объект класса Дерево, используя классы Лист. Методы: зацвести, опасть листьям, покрыться инеем, пожелтеть листьям.

Код модуля Leaf:

```
package lab3_var3_7;

import java.util.Objects;


public class Leaf {
    private String state;

    public Leaf(String state){
        this.setState(state);
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    @Override
    public  oolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Leaf leaf)) return false;
        return Objects.equals(leaf.getState(), getState());
    }

    @Override
    public int hashCode() {
        return Objects.hash(getState());
    }

    @Override
    public String toString() {
        return String.format("leaf: {state: '%s'}", getState());
    }
}
```

Код модуля Tree:

```
package lab3_var3_7;

import java.util.Objects;
import java.util.Random;


public class Tree {

    private int leafCount;
    private Leaf[] leaves;
```

```
public Tree(){
    bloom();
}

public Tree(int leafCount){
    this.leafCount = leafCount;
    this.leaves = new Leaf[leafCount];
    for (int I = 0; I < this.leafCount; i++) {
        this.leaves[i] = new Leaf("blooms");
    }
}

public Leaf[] getLeaves() {
    return leaves;
}

public void setLeaves(Leaf[] newLeaves) {
    this.leafCount = newLeaves.length;
    this.leaves = newLeaves;
}

@Override
public  oolean equals(Object o) {
    if (this == o) return true;
    if (!(o instanceof Tree tree)) return false;

    Leaf[] firstLeaves = getLeaves();
    Leaf[] secondLeaves = tree.getLeaves();
    if (firstLeaves.length != secondLeaves.length) return false;

     oolean flag = true;
    int I = 0;
    while (flag && I < leafCount) {
        if (!firstLeaves[i].equals(secondLeaves[i])) flag = false;
        I = I + 1;
    }
    return flag;
}

@Override
public int hashCode() {
    return Objects.hash((Object) getLeaves());
}

@Override
public String toString() {
    StringBuilder result = new StringBuilder("Tree: {\n");
    for (Leaf leaf : getLeaves()) {
        result.append("\t").append(leaf.toString()).append("\n");
    }
    result.append("}");
    return result.toString();
}

public void bloom() {
    Random random = new Random();
    this.leafCount = random.nextInt(10) + 2;
    this.leaves = new Leaf[this.leafCount];
    for (int I = 0; I < this.leafCount; i++) {
        this.leaves[i] = new Leaf("blooms");
    }
}

public void frost() {
```

```
        for (Leaf leaf : this.leaves) {
            leaf.setState("covered with frost");
        }
    }

    public void turnYellow() {
        for (Leaf leaf : this.leaves) {
            leaf.setState("turned yellow");
        }
    }

    public void fall() {
        setLeaves(new Leaf[0]);
    }
}
```

Код модуля Main:

```
package lab3_var3_7;

import static java.lang.System.out;


public class Main {
    public static void main(String[] args) {
        out.println("Random tree with 5 leaves");
        Tree tree = new Tree(5);
        System.out.println(tree);

        out.println("\nHandmade tree");
        Leaf[] leaves = new Leaf[3];
        leaves[0] = new Leaf("lol");
        leaves[1] = new Leaf("kek");
        leaves[2] = new Leaf("cheburek");
        tree.setLeaves(leaves);
        System.out.println(tree);

        out.println("\nRandom tree");
        tree = new Tree();
        System.out.println(tree);

        tree.frost();
        System.out.println(tree);
        tree.turnYellow();
        System.out.println(tree);
        tree.fall();
        System.out.println(tree);
        tree.bloom();
        System.out.println(tree);
    }
}
```

Работа программы показана на рисунках 7-8.

```
Random tree with 5 leaves
Tree: {
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
}

Handmade tree
Tree: {
    leaf: {state: 'lol'}
    leaf: {state: 'kek'}
    leaf: {state: 'cheburek'}
}
```

Рисунок 7 – Работа программы

```
Random tree
Tree: {
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
}
Tree: {
    leaf: {state: 'covered with frost'}
    leaf: {state: 'covered with frost'}
    leaf: {state: 'covered with frost'}
    leaf: {state: 'covered with frost'}
}
Tree: {
    leaf: {state: 'turned yellow'}
    leaf: {state: 'turned yellow'}
    leaf: {state: 'turned yellow'}
    leaf: {state: 'turned yellow'}
}
Tree: {
}
Tree: {
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
    leaf: {state: 'blooms'}
}
```

Рисунок 8 – Работа программы

Задание 6: создать приложение, удовлетворяющее требованиям, приведенным в задании. Аргументировать принадлежность классу каждого создаваемого метода и корректно переопределить для каждого класса методы equals(), hashCode(), toString(). Создать объект класса Пианино, используя класс Клавиша. Методы: настроить, играть на пианино, нажимать клавишу.

Код модуля Key:

```
package lab3_var3_8;

import java.util.Objects;


public class Key {
    private String state;

    public Key(String state){
        this.setState(state);
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Key key)) return false;
        return Objects.equals(key.getState(), getState());
    }

    @Override
    public int hashCode() {
        return Objects.hash(getState());
    }

    @Override
    public String toString() {
        return String.format("key: {state: '%s'}", getState());
    }

}
```

Код модуля Piano:

```
package lab3_var3_8;

import java.util.Objects;
import java.util.Random;


public class Piano {
    // out of tune | in tune
    private int keyCount;
    private Key[] keys;
```

```
    public Piano() {
        Random random = new Random();
        this.keyCount = random.nextInt(10) + 2;
        this.keys = new Key[this.keyCount];
        this.tune();
    }

    public Piano(int keyCount){
        this.keyCount = keyCount;
        this.keys = new Key[this.keyCount];
        this.tune();
    }

    public Key[] getKeys() {
        return keys;
    }

    public void setKeys(Key[] newKeys) {
        this.keyCount = newKeys.length;
        this.keys = newKeys;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Piano piano)) return false;

        Key[] firstKeys = getKeys();
        Key[] secondKeys = piano.getKeys();
        if (firstKeys.length != secondKeys.length) return false;

        boolean flag = true;
        int i = 0;
        while (flag && i < keyCount) {
            if (!firstKeys[i].equals(secondKeys[i])) flag = false;
            i = i + 1;
        }
        return flag;
    }

    @Override
    public int hashCode() {
        return Objects.hash((Object) getKeys());
    }

    @Override
    public String toString() {
        StringBuilder result = new StringBuilder("Piano: {\n");
        for (Key key : getKeys()) {
            result.append("\t").append(key.toString()).append("\n");
        }
        result.append("}");
        return result.toString();
    }

    public void tune() {
        for (int i = 0; i < this.keyCount; i++) {
            this.keys[i] = new Key("in tune");
        }
    }

    public void pushKey(int keyNumber) {
        Random random = new Random();
        if (random.nextBoolean() && random.nextBoolean())
```

```
this.keys[keyNumber].setState("out of tune");
    }

    public void play() {
        Random random = new Random();
        int pushCount = random.nextInt(keyCount / 2);
        for (int i = 0; i < pushCount; i++) {
            pushKey(random.nextInt(keyCount));
        }
    }
}
```

Код модуля Main:

```
package lab3_var3_8;

import static java.lang.System.out;


public class Main {
    public static void main(String[] args) {
        out.println("Random piano with 5 keys");
        Piano piano = new Piano(5);
        System.out.println(piano);

        out.println("\nHandmade piano");
        Key[] keys = new Key[3];
        keys[0] = new Key("lol");
        keys[1] = new Key("kek");
        keys[2] = new Key("cheburek");
        piano.setKeys(keys);
        System.out.println(piano);

        out.println("\nRandom piano");
        piano = new Piano();
        System.out.println(piano);

        for (int i = 0; i < 20; i++) {
            piano.play();
        }
        System.out.println(piano);
        piano.tune();
        System.out.println(piano);

        for (int i = 0; i < 20; i++) {
            piano.pushKey(piano.getKeys().length - 1);
            piano.pushKey(0);
        }
        System.out.println(piano);
    }
}
```

Работа программы показана на рисунках 9-10.

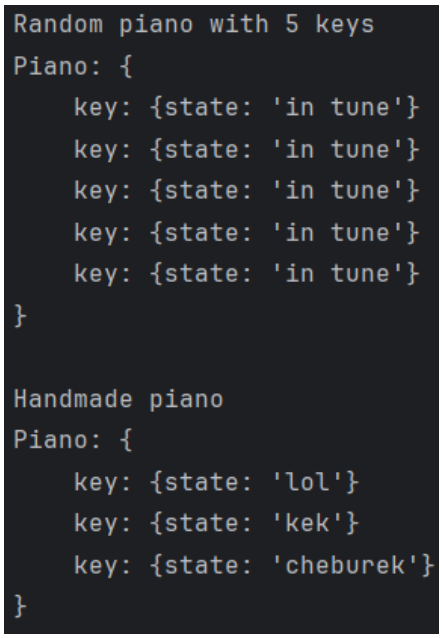

Рисунок 9 – Работа программы

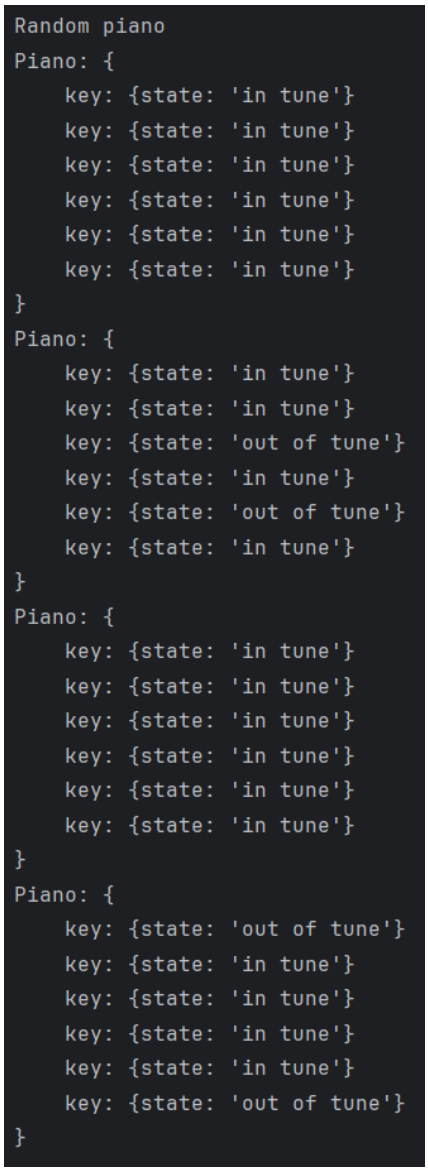

Рисунок 10 – Работа программы

Задание 7: построить модель программной системы. Система Телефонная станция. Абонент оплачивает Счет за разговоры и Услуги, может попросить Администратора сменить номер и отказаться от услуг. Администратор изменяет номер, Услуги и временно отключает Абонента за неуплату.

Код модуля Service:

```
package lab3_var4_7;

import java.util.Random;


public class Service {
    private String name;
    private double cost;

    public Service(String name, double cost){
        this.setName(name);
        this.setCost(cost);
    }

    public Service(String name){
        Random random = new Random();
        this.setName(name);
        this.setCost(random.nextInt(10) + 10);
    }

    public String getName() {
        return name;
    }

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

    public double getCost() {
        return cost;
    }

    public void setCost(double cost) {
        this.cost = cost;
    }

    @Override
    public String toString() {
        return String.format("service: {name: '%s'; cost: %.3f}", getName(),
getCost());
    }
}
```

Код модуля User:

```
package lab3_var4_7;

import java.util.Objects;


public class User {

    private int number;
```

```
    private double balance;
    private double debtForCalls;
    private Service[] services;
    private String state;
    public User(int number, double balance, double debtForCalls, Service[]
services) {
        this.setNumber(number);
        this.setBalance(balance);
        this.setDebtForCalls(debtForCalls);
        this.setServices(services);
        this.setState("On");
    }

    public int getNumber() {
        return number;
    }

    public void setNumber(int number) {
        this.number = number;
    }

    public double getBalance() {
        return balance;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }

    public double getDebtForCalls() {
        return debtForCalls;
    }

    public void setDebtForCalls(double debtForCalls) {
        this.debtForCalls = debtForCalls;
    }

    public Service[] getServices() {
        return services;
    }

    public void setServices(Service[] newServices) {
        this.services = newServices;
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    @Override
    public String toString() {
        StringBuilder result = new StringBuilder(
                String.format("""
                        User: {
                        \tnumber: %d;
                        \tbalance: %.3f;
                        \tdebtForCalls: %.3f;
                        \tstate: %s;
                        """,
                        getNumber(), getBalance(), getDebtForCalls(), getState()
                )
```

```
        );
        for (Service service : getServices()) {
            result.append("\t").append(service.toString()).append("\n");
        }
        result.append("}");
        return result.toString();
    }

    public void checkBalance() {
        double debt = debtForCalls;
        for (Service service : this.services) {
            debt = debt + service.getCost();
        }
        if (debt > getBalance()) {
            setState("Off");
        } else {
            setBalance(getBalance() - debt);
            setState("On");
        }
    }

    public void serviceOff(String name) {
        int counter = 0;
        for (Service service : this.services) {
            if (!Objects.equals(service.getName(), name)) counter++;
        }
        Service[] newServices = new Service[counter];

        counter = 0;
        for (Service service : this.services) {
            if (!Objects.equals(service.getName(), name)) {
                newServices[counter] = service;
                counter++;
            }
        }
        setServices(newServices);
    }
}
```

Код модуля Main:

```
package lab3_var4_7;

import static java.lang.System.out;


public class Main {
    public static void main(String[] args) {

        int number = 666;
        double balance = 100;
        double debtForCalls = 50;
        Service[] services = new Service[3];
        services[0] = new Service("lol");
        services[1] = new Service("kek");
        services[2] = new Service("cheburek");

        out.println("Source User");
        User user = new User(number, balance, debtForCalls, services);
        out.println(user);

        out.println("\nChanged User");
        user.setNumber(777);
        user.serviceOff("kek");
```

```
        out.println(user);

        out.println("\nAfter two balance operations");
        user.checkBalance();
        out.println(user);
        user.checkBalance();
        out.println(user);
    }
}
```

Работа программы показана на рисунке 11.

```
Source User
User: {
    number: 666;
    balance: 100,000;
    debtForCalls: 50,000;
    state: On;
    service: {name: 'lol'; cost: 12,000}
    service: {name: 'kek'; cost: 10,000}
    service: {name: 'cheburek'; cost: 15,000}
}

Changed User
User: {
    number: 777;
    balance: 100,000;
    debtForCalls: 50,000;
    state: On;
    service: {name: 'lol'; cost: 12,000}
    service: {name: 'cheburek'; cost: 15,000}
}

After two balance operations
User: {
    number: 777;
    balance: 23,000;
    debtForCalls: 50,000;
    state: On;
    service: {name: 'lol'; cost: 12,000}
    service: {name: 'cheburek'; cost: 15,000}
}
User: {
    number: 777;
    balance: 23,000;
    debtForCalls: 50,000;
    state: Off;
    service: {name: 'lol'; cost: 12,000}
    service: {name: 'cheburek'; cost: 15,000}
}
```

Рисунок 11 – Работа программы

Задание 8: построить модель программной системы. Система Телефонная станция. Система Автобаза. Диспетчер распределяет заявки на Рейсы между Водителями и назначает для этого Автомобиль. Водитель может сделать заявку на ремонт. Диспетчер может отстранить Водителя от работы. Водитель делает отметку о выполнении Рейса и состоянии Автомобиля.

Код модуля Voyage:

```
package lab3_var4_8;


public class Voyage {
    private int state;

    public Voyage(int state){
        this.setState(state);
    }

    public int getState() {
        return state;
    }

    public void setState(int state) {
        this.state = state;
    }

    @Override
    public String toString() {
        return String.format("voyage: {state: %d/10}", getState());
    }
}
```

Код модуля Car:

```
package lab3_var4_8;


public class Car {
    private int state;

    public Car(int state){
        this.setState(state);
    }

    public int getState() {
        return state;
    }

    public void setState(int state) {
        this.state = state;
    }

    @Override
    public String toString() {
        return String.format("car: {state: %d/10}", getState());
    }
}
```

Код модуля Driver:

```
package lab3_var4_8;


public class Driver {

    private String state;
    private Voyage voyage;
    private Car car;

    public Driver(Voyage voyage, Car car) {
        this.setState("On");
        this.setVoyage(voyage);
        this.setCar(car);
    }

    public int[] getVoyageResults() {
        int[] result = new int[3];
        result[0] = getVoyage().getState();
        result[1] = getCar().getState();
        result[2] = makeRepairRequest(result[1]);
        return result;
    }

    private int makeRepairRequest(int carState) {
        return (carState < 4) ? 1 : 0;
    }

    public String getState() {
        return state;
    }

    public void setState(String state) {
        this.state = state;
    }

    public Voyage getVoyage() {
        return voyage;
    }

    public void setVoyage(Voyage voyage) {
        this.voyage = voyage;
    }

    public Car getCar() {
        return car;
    }

    public void setCar(Car car) {
        this.car = car;
    }

    @Override
    public String toString() {
        return String.format(
                "driver: {\n\t%s; \n\t%s;\n\tstate: '%s'\n}",
                getVoyage().toString(), getCar().toString(), getState()
        );
    }
}
```

Код модуля Main:

```
package lab3_var4_8;

import java.util.Arrays;
import static java.lang.System.out;


public class Main {
    public static void main(String[] args) {
        Voyage firstVoyage = new Voyage(-1);
        Voyage secondVoyage = new Voyage(-1);
        Car firstCar = new Car(10);
        Car secondCar = new Car(9);
        Driver firstDriver = new Driver(firstVoyage, firstCar);
        Driver secondDriver = new Driver(secondVoyage, secondCar);

        out.println("Default drivers");
        out.println(firstDriver);
        out.println(secondDriver);

        out.println("\nDrivers after voyages");
        firstVoyage.setState(4);
        firstCar.setState(3);
        secondVoyage.setState(10);
        secondCar.setState(8);
        out.println(firstDriver);
        out.println(secondDriver);

        out.println("\nDrivers voyages results [voyage state, car state, 1(0) -
car is broken(is fine)]");
        out.print(Arrays.toString(firstDriver.getVoyageResults())+"; ");
        out.print(Arrays.toString(secondDriver.getVoyageResults()));

        out.println("\n\nDrivers after inspection");
        inspectionAndRemovalFromWork(firstDriver);
        inspectionAndRemovalFromWork(secondDriver);
        out.println(firstDriver);
        out.println(secondDriver);
    }

    private static void inspectionAndRemovalFromWork(Driver driver) {
        if (driver.getVoyageResults()[2] == 1) {
            driver.setState("Off");
        } else driver.setState("On");
    }
}
```

Работа программы показана на рисунке 12.

```
Default drivers
driver: {
    voyage: {state: -1/10};
    car: {state: 10/10};
    state: 'On'
}
driver: {
    voyage: {state: -1/10};
    car: {state: 9/10};
    state: 'On'
}

Drivers after voyages
driver: {
    voyage: {state: 4/10};
    car: {state: 3/10};
    state: 'On'
}
driver: {
    voyage: {state: 10/10};
    car: {state: 8/10};
    state: 'On'
}

Drivers voyages results [voyage state, car state, 1(0) - car is broken(is fine)]
[4, 3, 1]; [10, 8, 0]

Drivers after inspection
driver: {
    voyage: {state: 4/10};
    car: {state: 3/10};
    state: 'Off'
}
driver: {
    voyage: {state: 10/10};
    car: {state: 8/10};
    state: 'On'
}
```

Рисунок 12 – Работа программы

Ссылка на git-репозиторий: https://github.com/FedorLuchkin/Java_bmstu

Вывод: были освоены принципы работы с классами, наследованием и полиморфизмом на языке программирования Java.