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

О Т Ч Е Т

по лабораторной работе №3

Название: Классы, наследование, полиморфизм

Дисциплина: Языка программирования для работы с большими данными

Студент	ИУ6-23М		Д.Д. Капитонов
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

Москва, 2022

Вариант 1, номера 9 и 10.

9. Определить класс Квадратное уравнение. Класс должен содержать несколько конструкторов. Реализовать методы для поиска корней, экстремумов, а также интервалов убывания/возрастания. Создать массив объектов и определить наибольшие и наименьшие по значению корни.

10. Определить класс Булева матрица (BoolMatrix) размерности (n x m). Класс должен содержать несколько конструкторов. Реализовать методы для логического сложения (дизъюнкции), умножения и инверсии матриц. Реализовать методы для подсчета числа единиц в матрице и упорядочения строк в лексикографическом порядке

Код программы:

```
package dan.lab;

import com.sun.org.apache.xpath.internal.operations.Bool;

import java.util.Arrays;
import java.util.Random;

public class lab3_1 {
   public static class kv_yr
   {
      int a,b,c;
      double x1,x2;
      public kv_yr(int a, int b, int c)
      {
         this.a = a;
         this.b = b;
         this.c = c;
      }
      public kv_yr(int x)
      {
         Random r = new Random();
         this.a =  r.nextInt(12);
         this.b =  r.nextInt(12);
         this.c =  r.nextInt(12);
         this.x1 = x;
      }

      public void kv_print()
      {
         System.out.printf("Уравнение имеет вид: %d*x^2+%d*x+%d=0\n",this.a, this.b,this.c);
```

```
    }
    public String kv_korni()
    {
        double d,x1,x2;
        String result;
        d = Math.pow(this.b,2) - 4*this.a*this.c;
        if (d<0)
        {
            result = "У уравнения отсутствуют корни";
            this.x1 = 666;
            this.x2 = 666;
        } else if (d == 0)
        {
            x1 = (-b+Math.sqrt(d))/(2*this.a);
            result = "Корень уравнения = " + x1;
            this.x1 = x1;
            this.x2 = this.x1;
        } else
        {
            this.x1 = (-b+Math.sqrt(d))/(2*this.a);
            this.x2 = (-b-Math.sqrt(d))/(2*this.a);
            result = "Корни уравнения: " + this.x1+" "+this.x2;


        }

        return result;
    }
    public String kv_voz_yb()
    {
        String result;

        result = "Экстремум функции находится в точке - " + -this.b/this.a+"\n";
        result += "Функция возрастает на промежутки от минус бесконечности до "+ -
this.b/this.a+ " и убывает начиная с " + -this.b/this.a + " до + бесконечнгсти";
        return result;
    }
}
public static class BoolMatrix {
    int n;
    int m;
    int true_count;
    boolean[][] matrix;

    public BoolMatrix(int n, int m) {
        Random r = new Random();
        this.n = n;
        this.m = m;
        this.matrix = new boolean[n][m];
        for (int i = 0; i != n; i++) {
            for (int j = 0; j != m; j++) {
                this.matrix[i][j] = r.nextBoolean();
```

```
            if (this.matrix[i][j] == true)
            {
                this.true_count++;
            }
        }
    }
}
public BoolMatrix() {
    Random r = new Random();
    this.n = r.nextInt(10)+1;
    this.m = r.nextInt(10)+1;
    this.matrix = new boolean[n][m];
    for (int i = 0; i != n; i++) {
        for (int j = 0; j != m; j++) {
            this.matrix[i][j] = r.nextBoolean();
            if (this.matrix[i][j] == true)
            {
                this.true_count++;
            }
        }
    }
}
public void m_print()
{
    System.out.printf("\nМатрица размерностью %d на %d\n", this.n, this.m);
    for (int i=0; i!=this.n; i++)
    {
        for (int j=0;j!=this.m;j++)
        {
            System.out.printf("%2b\t", this.matrix[i][j]);
        }
        System.out.println("");
    }
    System.out.printf("В матрицы содержиться %d элемента(ов) равных 1\n",
this.true_count);
}
public static void m_summ(BoolMatrix m1, BoolMatrix m2)
{
    System.out.println("\nЛогическое сложение матриц");
    for (int i=0;i!=m1.n;i++)
    {
        for (int j=0; j!=m1.m; j++)
        {
            System.out.printf("%1b\t", m1.matrix[i][j] | m2.matrix[i][j]);
        }
        System.out.println("");
    }
}
public static void m_umn(BoolMatrix m1, BoolMatrix m2)
{
    System.out.println("\nЛогическое умножение матриц");
    for (int i=0;i!=m1.n;i++)
```

```
    {
        for (int j=0; j!=m1.m; j++)
        {
            System.out.printf("%1b\t", m1.matrix[i][j] & m2.matrix[i][j]);
        }
        System.out.println("");
    }
}
public void m_invers()
{
    System.out.println("\nИнверсия матрицы\nБыло:");
    this.m_print();
    for (int i=0; i!=this.n; i++)
    {
        for (int j=0; j!=this.m; j++)
        {
            this.matrix[i][j] = !this.matrix[i][j];
        }
    }

    System.out.println("Стало:");
    this.tr_count();
    this.m_print();

}
public void tr_count()
{
    this.true_count =0;
    for (int i = 0; i != n; i++) {
        for (int j = 0; j != m; j++) {
            if (this.matrix[i][j])
            {
                this.true_count++;
            }
        }
    }
}
public void b_sort()
{
    System.out.println("\nСортировка матрицы в лексикографическом порядке\nБыло:");
    this.m_print();
    boolean[] cash = new boolean[this.m];
    int count_1=0, count_2=0;
    for (int ii=0;ii!=this.n;ii++) {
        for (int i = 0; i != this.n - 1; i++) {
            for (int j = 0; j != this.m; j++) {
                if (this.matrix[i][j]) {
                    count_1 += (int) Math.pow(this.m - j, 2);
                }
                if (this.matrix[i + 1][j]) {
                    count_2 += (int) Math.pow(this.m - j, 2);
                }
```

```
                }
                if (count_1 < count_2) {
                    for (int n = 0; n != this.m; n++) {
                        cash[n] = this.matrix[i][n];
                    }
                    for (int n = 0; n != this.m; n++) {
                        this.matrix[i][n] = this.matrix[i + 1][n];
                    }
                    for (int n = 0; n != this.m; n++) {
                        this.matrix[i + 1][n] = cash[n];
                    }
                }
                count_1 = 0;
                count_2 = 0;
            }
        }
        System.out.println("Стало:");
        this.m_print();
    }
}

public static void main(String[] args) {
    System.out.println("Вариант 1 №9");
    kv_yr n = new kv_yr(1,3,-4);
    n.kv_print();
    System.out.println(n.kv_korni());
    System.out.println(n.kv_voz_yb());
    int m=2;
    kv_yr[] mass = new kv_yr[m];
    Random r = new Random();
    double max=0, min=9999999;
    for (int i=0; i!=m; i++)
    {
        mass[i] = new kv_yr(r.nextInt(20)+1,r.nextInt(20)+1,r.nextInt(20)-21);
        //mass[i].kv_print();
        System.out.println(mass[i].kv_korni());
        if (mass[i].x1>max && mass[i].x1!=666)
        {
            max = mass[i].x1;
        }
        if (mass[i].x1<min && mass[i].x1!=666)
        {
            min = mass[i].x1;
        }

    }
    System.out.println("Максимальный корень x1 = "+max+". Минимальный корень x1 = "+min);

    System.out.println("\nВариант 1 №10");
    BoolMatrix m_1 = new BoolMatrix(6,6);
    m_1.m_print();
```

```
        BoolMatrix m_2 = new BoolMatrix(6,6);
        m_2.m_print();
        BoolMatrix.m_summ(m_1,m_2);
        BoolMatrix.m_umn(m_1,m_2);
        m_1.m_invers();
        m_1.b_sort();




    }
}
```

Результат выполнения программы:

```
Вариант 1 №9
Уравнение имеет вид: 1*x^2+3*x+-4=0
Корни уравнения: 1.0 -4.0
Экстремум функции находится в точке - -3
Функция возрастает на промежутки от минус бесконечности до -3 и убывает начиная с -3 до + бесконечнгсти
Корни уравнения: 2.420695782710004 -2.7540291160433377
Корни уравнения: 1.25 -2.0
Максимальный корень x1 = 2.420695782710004. Минимальный корень x1 = 1.25

Вариант 1 №10

Матрица размерностью 6 на 6
false   false   false   false   false   false
true    false   true    false   false   false
false   false   true    true    false   false
false   true    true    false   false   true
false   true    false   true    false   true
false   true    true    true    false   true
В матрицы содержиться 14 элемента(ов) равных 1

Матрица размерностью 6 на 6
true    false   false   true    false   false
true    true    false   true    false   false
true    true    true    true    true    false
true    false   false   false   true    true
true    true    true    false   true    false
false   true    false   false   false   false
В матрицы содержиться 18 элемента(ов) равных 1

Логическое сложение матриц
true    false   false   true    false   false
true    true    true    true    false   false
true    true    true    true    true    false
true    true    true    false   true    true
true    true    true    true    true    true
false   true    true    true    false   true

Логическое умножение матриц
false   false   false   false   false   false
true    false   false   false   false   false
false   false   true    true    false   false
false   false   false   false   false   true
false   true    false   false   false   false
false   true    false   false   false   false
```

```
Инверсия матрицы
Было:

Матрица размерностью 6 на 6
false   false   false   false   false   false
true    false   true    false   false   false
false   false   true    true    false   false
false   true    true    false   false   true
false   true    false   true    false   true
false   true    true    true    false   true
В матрицы содержиться 14 элемента(ов) равных 1
Стало:

Матрица размерностью 6 на 6
true    true    true    true    true    true
false   true    false   true    true    true
true    true    false   false   true    true
true    false   false   true    true    false
true    false   true    false   true    false
true    false   false   false   true    false
В матрицы содержиться 22 элемента(ов) равных 1

Сортировка матрицы в лексикографическом порядке
Было:

Матрица размерностью 6 на 6
true    true    true    true    true    true
false   true    false   true    true    true
true    true    false   false   true    true
true    false   false   true    true    false
true    false   true    false   true    false
true    false   false   false   true    false
В матрицы содержиться 22 элемента(ов) равных 1
Стало:

Матрица размерностью 6 на 6
true    true    true    true    true    true
true    true    false   false   true    true
true    false   true    false   true    false
true    false   false   true    true    false
true    false   false   false   true    false
false   true    false   true    true    true
В матрицы содержиться 22 элемента(ов) равных 1
```

Вариант 2, номера 9 и 10.

9. Product: id, Наименование, UPC, Производитель, Цена, Срок хранения, Количество. Создать массив объектов. Вывести: a) список товаров для заданного наименования; b) список товаров для заданного наименования, цена

которых не превосходит заданную; c) список товаров, срок хранения которых больше заданного.
10. Train: Пункт назначения, Номер поезда, Время отправления, Число мест (общих, купе, плацкарт, люкс). Создать массив объектов. Вывести: a) список поездов, следующих до заданного пункта назначения; b) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа; c) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места

Код программы:

```
package daniil.lab;
import java.util.Scanner;
import java.util.Random;

public class lab3_2 {

   public static class Product {
      //Наименование, UPC, Производитель, Цена, Срок хранения, Количество
      String name, creator;
      boolean upc;
      double cost;
      int safe_day, count;

      public Product(String name, String creator, boolean upc, double cost, int safe_day, int count)
{
         this.name = name;
         this.creator = creator;
         this.upc = upc;
         this.cost = cost;
         this.safe_day = safe_day;
         this.count = count;
      }

      public Product() {
         System.out.println("Добавление нового продукта в базу");
         Scanner in = new Scanner(System.in);
         int n = in.nextInt();
         System.out.print("Наименование - ");
         this.name = in.nextLine();
         System.out.print("Производитель - ");
         this.creator = in.nextLine();
         System.out.print("UPS (T|F) - ");
         this.upc = in.nextBoolean();
         System.out.print("Цена - ");
         this.cost = in.nextInt();
         System.out.print("Срок хранения (суток) - ");
         this.safe_day = in.nextInt();
         System.out.print("Количество - ");
         this.count = in.nextInt();
      }

      public void Product_print() {
```

```
        System.out.println("Информация о продукте:");
        System.out.printf("Наименование - %s\nПроизоводитель - %s\nUPS - %b\nЦена - %f\nСрок хранения - %d\n" +
                "Количество - %d\n", this.name, this.creator, this.upc, this.cost, this.safe_day, this.count);
        System.out.println("");
    }

    public static Product[] Pr_mass_create(int n) {
        Product[] mass = new Product[n];
        Random r = new Random();
        String[] rand_creator = {"KIA", "HYUNDAI", "AUDI", "BMW", "NISSAN"};
        String[] rand_name = {"K5", "SOLARIS", "RS6", "X7", "X-TRAIL"};
        for (int i = 0; i != n; i++) {
            mass[i] = new Product(rand_name[r.nextInt(4)], rand_creator[r.nextInt(4)], false, r.nextDouble() * 100, i * i + 1, i + 10);
        }
        return mass;
    }

    public static void select(Product[] mass, int n, String name, int cost, int safe_day) {
        for (int i = 0; i != n; i++) {
            if (!name.equals("") && cost == 0) {
                if (mass[i].name.equals(name)) {
                    mass[i].Product_print();
                }
            }
            if (!name.equals("") && cost > 0) {
                if (mass[i].name.equals(name) && mass[i].cost <= cost) {
                    mass[i].Product_print();
                }
            }
            if (name.equals("") && cost == 0 && safe_day > 0) {
                if (mass[i].safe_day > safe_day) {
                    mass[i].Product_print();
                }
            }
        }
    }

}

public static class Train {
    //Пункт назначения, Номер поезда, Время отправления, Число мест (общих, купе, плацкарт, люкс)
    String arrival;
    int train_no;
    int time_out;
    int total_place, kupe_place, pla_place, luxe_place;

    public Train() {
```

```
        String[] arriaval_places = {"Омкс", "Рязань", "Москва", "Питер", "Надым", "Тула",
"Колыма"};
        Random r = new Random();
        this.arrival = arriaval_places[r.nextInt(6)];
        this.train_no = r.nextInt(10000) + 1;
        this.time_out = r.nextInt(23) + 1;
        this.kupe_place = r.nextInt(40);
        this.pla_place = r.nextInt(100) + 1;
        this.luxe_place = r.nextInt(25);
        this.total_place = this.kupe_place + this.pla_place + this.luxe_place;

    }

    public void Tr_print() {
        System.out.println("Инфрмация о поезде:");
        System.out.printf("Пункт названчения - %s\nНомер поезда - %d\nЧас отправления -
%d\nОбщее число мест - %d\nМеста купе - %d\n" +
                "Места плацкарт - %d\nМеста люкс - %d\n\n", this.arrival, this.train_no,
this.time_out, this.total_place, this.kupe_place, this.pla_place, this.luxe_place);

    }

    public static Train[] create_mass(int n) {
        Train[] mass = new Train[n];
        for (int i = 0; i != n; i++) {
            mass[i] = new Train();
        }
        return mass;
    }
    public static void select(Train[] mass, int n, String arrival, int time_out, int total_place)
    {
        for (int i=0; i!=n; i++)
        {
            if (time_out == 0 && total_place == 0)
            {
                if (mass[i].arrival.equals(arrival))
                {
                    mass[i].Tr_print();
                }
            }else if (time_out > 0 && total_place == 0)
            {
                if (mass[i].time_out > time_out && mass[i].arrival.equals(arrival))
                {
                    mass[i].Tr_print();
                }
            } else
            {
                if (mass[i].total_place > total_place && mass[i].arrival.equals(arrival))
                {
                    mass[i].Tr_print();
                }
```

```
            }
        }
    }

}
public static void main(String[] args) {
    int n=8;
    System.out.println("Вариант 2 №9");
    Product[] mass;
    mass = Product.Pr_mass_create(n);
    for (int i=0; i!=n;i++)
    {

        mass[i].Product_print();
    }
    System.out.println("Запрос поиска (а) Поиск К5");
    Product.select(mass, n, "К5",0,0);
    System.out.println("Запрос поиска (б) Поиск К5 стоимостью меньше 50");
    Product.select(mass, n, "К5",50,0);
    System.out.println("Запрос поиска (с) срок хранения больше 10");
    Product.select(mass, n, "",0,10);
    System.out.println("\nВариант 2 №10");
    Train[] masstr;
    masstr = Train.create_mass(n);
    for (int i=0; i!=n;i++)
    {
        masstr[i].Tr_print();
    }
    System.out.println("Запрос (а) список поездов, следующих до заданного пункта назначения");
    Train.select(masstr,n,"Москва",0,0);
    System.out.println("Запрос (б) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа");
    Train.select(masstr,n,"Питер",10,0);
    System.out.println("Запрос (с) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места");
    Train.select(masstr,n,"Надым",0,100);




}
}
```

Результат выполнения программы:

```
Вариант 2 №9
Информация о продукте:
Наименование - K5
Произоводитель - HYUNDAI
UPS - false
Цена - 73,198164
Срок хранения - 1
Количество - 10

Информация о продукте:
Наименование - X7
Произоводитель - BMW
UPS - false
Цена - 74,478733
Срок хранения - 2
Количество - 11

Информация о продукте:
Наименование - SOLARIS
Произоводитель - AUDI
UPS - false
Цена - 56,787696
Срок хранения - 5
Количество - 12

Информация о продукте:
Наименование - X7
Произоводитель - BMW
UPS - false
Цена - 18,307039
Срок хранения - 10
Количество - 13

Информация о продукте:
Наименование - RS6
Произоводитель - AUDI
UPS - false
Цена - 59,158288
Срок хранения - 17
Количество - 14

Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 77,630127
Срок хранения - 26
Количество - 15

Информация о продукте:
Наименование - RS6
Произоводитель - BMW
UPS - false
Цена - 54,388484
Срок хранения - 50
Количество - 17

Запрос поиска (a) Поиск K5
Информация о продукте:
Наименование - K5
Произоводитель - HYUNDAI
UPS - false
Цена - 73,198164
Срок хранения - 1
Количество - 10

Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 77,630127
Срок хранения - 26
Количество - 15

Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 10,274399
Срок хранения - 37
Количество - 16

Запрос поиска (б) Поиск K5 стоимостью меньше 50
Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 10,274399
Срок хранения - 37
Количество - 16

Запрос поиска (c) срок хранения больше 10
Информация о продукте:
Наименование - RS6
Произоводитель - AUDI
UPS - false
Цена - 59,158288
Срок хранения - 17
Количество - 14
```

```
Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 77,630127
Срок хранения - 26
Количество - 15

Информация о продукте:
Наименование - K5
Произоводитель - KIA
UPS - false
Цена - 10,274399
Срок хранения - 37
Количество - 16

Информация о продукте:
Наименование - RS6
Произоводитель - BMW
UPS - false
Цена - 54,388484
Срок хранения - 50
Количество - 17


Вариант 2 №10
Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 955
Час отправления - 11
Общее число мест - 45
Места купе - 6
Места плацкарт - 15
Места люкс - 24

Инфрмация о поезде:
Пункт названчения - Питер
Номер поезда - 786
Час отправления - 6
Общее число мест - 63
Места купе - 36
Места плацкарт - 20
Места люкс - 7

Инфрмация о поезде:
Пункт названчения - Омкс
Номер поезда - 731
Час отправления - 5
Общее число мест - 136
Места купе - 23
Места плацкарт - 91
Места люкс - 22

Инфрмация о поезде:
Пункт названчения - Омкс
Номер поезда - 731
Час отправления - 5
Общее число мест - 136
Места купе - 23
Места плацкарт - 91
Места люкс - 22

Инфрмация о поезде:
Пункт названчения - Омкс
Номер поезда - 5187
Час отправления - 5
Общее число мест - 129
Места купе - 36
Места плацкарт - 71
Места люкс - 22

Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 9510
Час отправления - 5
Общее число мест - 49
Места купе - 39
Места плацкарт - 10
Места люкс - 0

Инфрмация о поезде:
Пункт названчения - Питер
Номер поезда - 4426
Час отправления - 10
Общее число мест - 93
Места купе - 32
Места плацкарт - 53
Места люкс - 8

Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 7422
Час отправления - 8
Общее число мест - 114
Места купе - 19
Места плацкарт - 76
Места люкс - 19

Инфрмация о поезде:
Пункт названчения - Питер
Номер поезда - 3636
Час отправления - 5
Общее число мест - 92
Места купе - 8
```

```
Запрос (a) список поездов, следующих до заданного пункта назначения
Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 955
Час отправления - 11
Общее число мест - 45
Места купе - 6
Места плацкарт - 15
Места люкс - 24

Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 9510
Час отправления - 5
Общее число мест - 49
Места купе - 39
Места плацкарт - 10
Места люкс - 0

Инфрмация о поезде:
Пункт названчения - Москва
Номер поезда - 7422
Час отправления - 8
Общее число мест - 114
Места купе - 19
Места плацкарт - 76
Места люкс - 19

Запрос (б) список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа
Запрос (с) список поездов, отправляющихся до заданного пункта назначения и имеющих общие места

Process finished with exit code 0
```

Вариант 3, номера 9 и 10.

9. Создать объект класса Фотоальбом, используя класс Фотография. Методы: задать название фотографии, дополнить фотоальбом фотографией, вывести на консоль количество фотографий.

10. Создать объект класса Год, используя классы Месяц, День. Методы: задать дату, вывести на консоль день недели по заданной дате, рассчитать количество дней, месяцев в заданном временном промежутке.

Код программы:

```
package daniil.lab;

import java.util.*;
import java.time.format.DateTimeFormatter;
import java.time.LocalDate;
import java.time.Period;


public class lab3_3 {
    public static class photo
    {
```

```
    private String ph_name;
    private String f_pass;
    private photo(String ph_name, String f_pass) {
        this.ph_name = ph_name;
        this.f_pass = f_pass;
    }

    private photo() {

    }
    public boolean equals(photo o2){
        if (this.hashcode(o2)) {
            return this.equals(o2);
        }else{
            return false;
        }
    }
    public boolean hashcode(photo o2){
        return this.hashCode() == o2.hashCode();
    }
    public String toString(){
        return "Название фотогравии - "+ this.ph_name+ ",  директория  - "+ this.f_pass;
    }
    private void ph_print(){
        System.out.println(this.ph_name+this.f_pass);
    }

}
public static class photoalbum extends photo
{
    private String alb_name;
    private ArrayList<photo> album;

    public photoalbum(String alb_name, ArrayList<photo> album) {
        this.alb_name = alb_name;
        this.album = album;
    }
    public photoalbum(String alb_name) {
        this.alb_name = alb_name;
        this.album = new ArrayList<>();

    }
    public boolean equals(photoalbum o2){
        if (this.hashcode(o2)) {
            return this.equals(o2);
        }else{
            return false;
        }
    }
    public boolean hashcode(photoalbum o2){
        return this.hashCode() == o2.hashCode();
    }
```

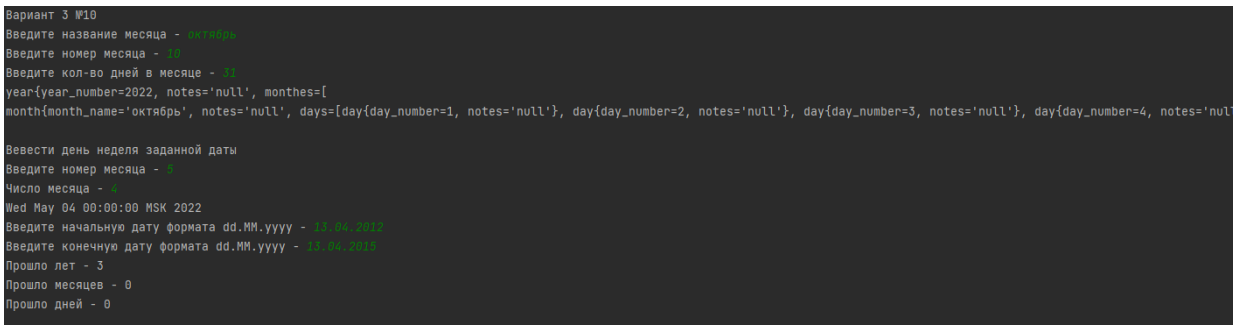
```
    private void add_photo(photo new_p){
        this.album.add(new_p);
    }
    public String toString(){
        String result = "";
        result+="Название альбома - "+ this.alb_name;
        for (photo ph:this.album) {
          result+= "\n" + ph.toString();
        }
        return result;
    }
    private int ph_count(){
        return this.album.size();
    }
}



public static void main(String[] args) {
    System.out.println("Вариант 3 №9");
    photoalbum album = new photoalbum("Новый альбом");
    photo ph_1 = new photo("photo1", "newfolder1");
    photo ph_2 = new photo("photo2", "newfolder2");
    album.add_photo(ph_1);
    album.add_photo(ph_2);
    System.out.println(album.ph_count());
    System.out.println(album.toString());

    System.out.println("\nВариант 3 №10");
    year year2022 = new year(2022);
    for (int i=0; i!=1;i++){
        String name;
        int number, count;
        Scanner in = new Scanner(System.in);
        System.out.print("Введите название месяца - ");
        name = in.nextLine();
        System.out.print("Введите номер месяца - ");
        number = in.nextInt();
        System.out.print("Введите кол-во дней в месяце - ");
        count = in.nextInt();
        month new_month = new month(name, number);
        for (int j=1; j<=count;j++){
            day new_day = new day(j);
            new_month.add_day(new_day);
        }
        year2022.add_new_month(new_month);
    }
    System.out.println(year2022.toString());
    System.out.println("");
    System.out.println("Ввести день неделя заданной даты");
    Scanner in = new Scanner(System.in);
    int m_n, d_n;
```

```
    Scanner in1 = new Scanner(System.in);
    System.out.print("Введите номер месяца - ");
    m_n = in1.nextInt();
    System.out.print("Число месяца - ");
    d_n = in1.nextInt();
    Calendar show_data = new GregorianCalendar(2022, m_n-1, d_n);
    System.out.println(show_data.getTime());
    Calendar show_data1 = new GregorianCalendar(2022, m_n-1, d_n);
    String begin, end;
    System.out.print("Введите начальную дату формата dd.MM.yyyy - ");

    begin = in.nextLine();
    System.out.print("Введите конечную дату формата dd.MM.yyyy - ");
    end = in.nextLine();
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd.MM.yyyy");
    LocalDate startDate = LocalDate.parse(begin, formatter);
    LocalDate endDate = LocalDate.parse(end, formatter);
    Period period = Period.between(startDate, endDate);
    System.out.println("Прошло лет - "+period.getYears());
    System.out.println("Прошло месяцев - "+period.getMonths());
    System.out.println("Прошло дней - "+period.getDays());
}

public static class day{
    int day_number;
    String notes;

    public day(int day_number) {
        this.day_number = day_number;
    }

    public day(int day_number, String notes) {
        this.day_number = day_number;
        this.notes = notes;
    }

    public day(){}

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        day day = (day) o;
        return day_number == day.day_number && Objects.equals(notes, day.notes);
    }

    @Override
    public int hashCode() {
        return Objects.hash(day_number, notes);
    }

    @Override
```

```
    public String toString() {
        return "day{" +
                "day_number=" + day_number +
                ", notes='" + notes + '\'' +
                '}';
    }
}
public static class month extends day{
    String month_name;
    int month_number;
    String notes;
    ArrayList<day> days;

    public month(String month_name, int month_number, String notes, ArrayList<day> days) {
        this.month_name = month_name;
        this.month_number = month_number;
        this.notes = notes;
        this.days = days;
    }

    public month(String month_name, int month_number) {
        this.month_name = month_name;
        this.month_number = month_number;
        this.days = new ArrayList<>();
    }
     public void add_day(day newday){
        this.days.add(newday);
     }


    public month(){}


    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        if (!super.equals(o)) return false;
        month month = (month) o;
        return Objects.equals(month_name, month.month_name) && Objects.equals(notes,
month.notes) && Objects.equals(days, month.days);
    }

    @Override
    public int hashCode() {
        return Objects.hash(super.hashCode(), month_name, notes, days);
    }

    @Override
    public String toString() {
        return "\nmonth{" +
                "month_name='" + month_name + '\'' +
```

```
                ", notes='" + notes + '\'' +
                ", days=" + days.toString() +
                '}';
    }
  }
  public static class year extends month{
    int year_number;
    String notes;
    ArrayList<month> monthes;

    public year(int year_number, String notes, ArrayList<month> monthes) {
      this.year_number = year_number;
      this.notes = notes;
      this.monthes = monthes;
    }

    public year(int year_number) {
      this.year_number = year_number;
      this.monthes = new ArrayList<>();
    }

    public void add_new_month(month new_month){
      this.monthes.add(new_month);
    }

    @Override
    public boolean equals(Object o) {
      if (this == o) return true;
      if (o == null || getClass() != o.getClass()) return false;
      if (!super.equals(o)) return false;
      year year = (year) o;
      return year_number == year.year_number && Objects.equals(notes, year.notes);
    }

    @Override
    public int hashCode() {
      return Objects.hash(super.hashCode(), year_number, notes);
    }

    @Override
    public String toString() {
      return "year{" +
              "year_number=" + year_number +
              ", notes='" + notes + '\'' +
              ", monthes=" + monthes.toString() +
              '}';
    }
  }
}
```

Результат выполнения программы:



Вариант 4, номер 9.

9. Система Интернет-магазин. Администратор добавляет информацию о Товаре. Клиент делает и оплачивает Заказ на Товары. Администратор регистрирует Продажу и может занести неплательщиков в «черный список».

Код программы:

```
package dan.lab;

import java.io.*;
import java.io.FileOutputStream;
import java.util.ArrayList;
import java.util.Scanner;

public class Lab3_4 {
   public static class Item implements Serializable{
      private static final long serialVersionUID = 2609875448065146411L;
      private String name;
      private int cost;
      private int count;
      public ArrayList<Item> items_sell;

      public Item() {
         items_sell = new ArrayList<>();

      }

      public Item(String name, int cost, int count) {
         this.name = name;
         this.cost = cost;
         this.count = count;
      }

      public void add_item(String name, int cost, int count){
         Item cash = new Item(name,cost, count);
         this.items_sell.add(cash);
      }
      public void add_count_item(String name, int cost, int count){
         int n = this.search_item(name, cost);
         if (n!=-1){
            items_sell.get(n).count+=count;
```

```
        } else{
            System.out.println("Такого товара не найдено");
        }
    }
    public void remove_item(String name, int cost){
        int n = this.search_item(name, cost);
        if (n !=-1) {
            this.items_sell.remove(n);
        }else{
            System.out.println("Такого товара не найдено");
        }
    }
    public int search_item(String name, int cost){
        for (int i=0; i!=items_sell.size();i++)
        {
            if(items_sell.get(i).name.equals(name) && items_sell.get(i).cost == cost){
                return i;
            }
        }
        return -1;
    }

    public String toString(int n) {
        if (n!=-1) {
            return "item{" +
                    "Название товара='" + this.items_sell.get(n).name + '\'' +
                    ", цена=" + this.items_sell.get(n).cost + '\'' +
                    ", кол-во=" + this.items_sell.get(n).count +
                    '}';
        }else{
            return "Такого товара не найдено";
        }
    }

    @Override
    public String toString() {
        return "item{" +
                "Название товара='" + name + '\'' +
                ", цена=" + cost +
                ", кол-во=" + count +
                '}';
    }
}

public static class Client implements Serializable{
    private String FIO;
    private boolean ban;
    private int balance;
    public ArrayList<Client> clients;

    public Client(String FIO, int balance) {
        this.FIO = FIO;
```

```
    this.ban = false;
    this.balance = balance;
}

public Client() {
    clients = new ArrayList<>();
}

public void add_new_client(String FIO, int balance){
    Client cash = new Client(FIO, balance);
    clients.add(cash);
}

public int search_client(String name){
    for(int i=0;i!=this.clients.size();i++){
        if (this.clients.get(i).FIO.equals(name)){
            return i;
        }
    }
    return -1;
}

public void remove_client(String name)
{
    int n = this.search_client(name);
    if (n !=-1){
        clients.remove(n);
    }else{
        System.out.println("такого пользователя не найдено");
    }
}

public void add_money(int n, int money){
    if (n!=-1){
        this.balance+=money;
    }else{
        System.out.println("такого пользователя не найдено");
    }
}

public void ban_unban_client(String name, boolean ban){
    int n = this.search_client(name);
    if (n!=-1){
        clients.get(n).ban = ban;
    }else{
        System.out.println("такого пользователя не найдено");
    }
}


public String toString(int n) {
    if (n!=-1) {
```

```
            return "client{" +
                    "ФИО='" + clients.get(n).FIO + '\'' +
                    ", забанен=" + clients.get(n).ban +
                    ", баланс=" + clients.get(n).balance +
                    '}';
        }else{
            return "Такого пользователя не найдено";
        }
    }

    @Override
    public String toString() {
        return "client{" +
                "ФИО='" + FIO + '\'' +
                ", забанен=" + ban +
                ", баланс=" + balance +
                '}';
    }
}



public static void main(String[] args) throws IOException, ClassNotFoundException {
    System.out.println("Вариант 4 №9");
    System.out.println("Старт системмы Интернет-магазин");

    Item itemss = new Item();
    itemss.add_item("IPhone", 6000, 5);
    FileOutputStream items_file_w = new FileOutputStream("itemsobj.txt");
    ObjectOutputStream itemsobj_w = new ObjectOutputStream(items_file_w);
    itemsobj_w.writeObject(itemss);
    itemsobj_w.close();

    Client clientss = new Client();
    clientss.add_new_client("Вася", 2000);
    FileOutputStream clients_file_w = new FileOutputStream("clientsobj.txt");
    ObjectOutputStream clientsobj_w = new ObjectOutputStream(clients_file_w);
    clientsobj_w.writeObject(clientss);
    clientsobj_w.close();


    FileInputStream items_file = new FileInputStream("itemsobj.txt");
    ObjectInputStream itemsobj = new ObjectInputStream(items_file);
    Object cash_i = itemsobj.readObject();
    Item items = (Item) cash_i;
    itemsobj.close();
    System.out.println("Товары загружены");

    FileInputStream clients_file = new FileInputStream("clientsobj.txt");
    ObjectInputStream clientsobj = new ObjectInputStream(clients_file);
    Object cash_c = clientsobj.readObject();
    Client clients_list = (Client) cash_c;
```

```
        clientsobj.close();
        System.out.println("Клиенты загружены");
while (true) {
    System.out.println("Вы клиент или администратор? 1- клиент, 2 - администратор, 3 - выйти");
    Scanner in_1 = new Scanner(System.in);
    int who = in_1.nextInt();
    String cl_name = "";
    int cl_id = 0;
    while (who != 3 && who != 55 && who != 66) {
        Scanner in = new Scanner(System.in);
        switch (who) {
            case 1:
                System.out.println("Введите своё имя");
                cl_name = (in.nextLine());
                cl_id = clients_list.search_client(cl_name);
                System.out.println("Информация о клиенте:" + cl_id);
                System.out.println(clients_list.toString(cl_id));
                if (cl_id == -1) {
                    who = -1;
                } else {
                    if (!clients_list.clients.get(cl_id).ban) {
                        who = 55;
                    } else {
                        who = -1;
                        System.out.println("Вы были забанены. Обратитесь к администратору");
                    }

                }
                break;
            case 2:
                System.out.println("Введите пароль");
                String pass = in.nextLine();
                if (pass.equals("123")) {
                    System.out.println("Добро пожаловать!");
                    who = 66;
                } else {
                    System.out.println("Пароль не верный");
                    who = -1;
                }
                break;
            default:
                System.out.println("Такого значения нет, введите снова");
                who = -1;
                break;
        }
        if (who == -1) {
            System.out.println("Вы клиент или администратор? 1- клиент, 2 - администратор, 3 - выйти");
            who = in.nextInt();
        }
    }
```

```
   if (who != 3) {
      System.out.println("Добро пожаловать в в систему. Доступные команды:");
      boolean check = true;
      while (check) {
         if (who == 55) {
            System.out.println("Клиент, введите цифры для выполнения действия:\n 1 - Пополнить балланс, 2 - Купить товар, 3 - выйти");
            Scanner in_c = new Scanner(System.in);
            switch (Integer.parseInt(in_c.nextLine())) {
               case 1:
                  System.out.println("На какую сумму вы хотите пополнить балланс?" + cl_name + cl_id);
                  in_c.reset();
                  int money = in_c.nextInt();
                  clients_list.clients.get(cl_id).add_money(cl_id, money);
                  System.out.println("Успешно");
                  System.out.println(clients_list.clients.get(cl_id).toString());
                  break;
               case 2:
                  System.out.println("Введите название товара и его цену для покупки");
                  in_c.reset();
                  String it_name = in_c.nextLine();
                  in_c.reset();
                  int it_cost = in_c.nextInt();
                  int it_id = items.search_item(it_name, it_cost);
                  if (it_id != -1) {
                     if (items.items_sell.get(it_id).count > 0) {
                        if (items.items_sell.get(it_id).cost <= clients_list.clients.get(cl_id).balance) {
                           clients_list.clients.get(cl_id).balance -= items.items_sell.get(it_id).cost;
                           items.items_sell.get(it_id).count -= 1;
                           System.out.println("Покупка совершена успешно");
                        } else {
                           System.out.println("У вас недостаточно средств для покупки");
                        }
                     } else {
                        System.out.println("Товара нет в наличии");
                     }
                  } else{
                     System.out.println("Такого товара нет");
                  }
                  break;
               case 3:
                  check = false;
                  break;
               default:
                  System.out.println("Такого значения нет, введите снова");
                  break;

            }
         } else {
            System.out.println("Администратор, введите цифры для выполнения действия:\n 1 - Забанить клиента, 2 - Добавить товар, 3 - Информация о товаре, 4 - Выйти");
```

```
            Scanner in_a = new Scanner(System.in);
            switch (Integer.parseInt(in_a.nextLine())) {
               case 1:
                  System.out.println("Чтоба занести клиента в чёрный список введите его имя
и статус");
                  in_a.reset();
                  String name = in_a.nextLine();
                  boolean status = Boolean.parseBoolean(in_a.nextLine());
                  clients_list.ban_unban_client(name, status);
                  System.out.println("Успешно");
                  break;
               case 2:
                  System.out.println("Чтобы добавить товар введите его название, цену и кол-
во");
                  in_a.reset();
                  String iname = in_a.nextLine();
                  int icost = Integer.parseInt(in_a.nextLine());
                  int icount = Integer.parseInt(in_a.nextLine());
                  items.add_item(iname,icost,icount);
                  System.out.println("Успешно");
                  System.out.println(items.toString(items.search_item(iname,icost)));
                  break;
               case 3:
                  System.out.println("Введите название товара и его цену");
                  in_a.reset();
                  String it_name = in_a.nextLine();
                  in_a.reset();
                  int it_cost = in_a.nextInt();
                  int it_id = items.search_item(it_name, it_cost);
                  if (it_id != -1) {
                     System.out.println(items.items_sell.get(it_id).toString());
                  } else {
                     System.out.println("Такого товара нет");
                  }
                  break;
               case 4:
                  check = false;
                  break;
               default:
                  break;


            }

         }
      }

   }


   //items.add_item("IPhone", 6000, 5);
//    FileOutputStream items_file_w = new FileOutputStream("itemsobj.txt");
```

```
//    ObjectOutputStream itemsobj_w = new ObjectOutputStream(items_file_w);
//    itemsobj_w.writeObject(items);
//    itemsobj_w.close();
//
//
//    //clients_list.add_new_client("Вася", 2000);
//    FileOutputStream clients_file_w = new FileOutputStream("clientsobj.txt");
//    ObjectOutputStream clientsobj_w = new ObjectOutputStream(clients_file_w);
//    clientsobj_w.writeObject(clients_list);
//    clientsobj_w.close();
}

    }
}
```

Результат выполнения программы:

```
Вариант 4 №9
Старт систетмы Интернет-магазин
Товары загружены
Клиенты загружены
Вы клиент или администратор? 1- клиент, 2 - администратор, 3 - выйти
2
Введите пароль
123
Добро пожаловать!
Добро пожаловать в в систему. Доступные команды:
Администратор, введите цифры для выполнения действия:
 1 - Забанить клиента, 2 - Добавить товар, 3 - Информация о товаре, 4 - Выйти
2
Чтобы добавить товар введите его название, цену и кол-во
Айфон
123
32
Успешно
item{Название товара='Айфон', цена=123', кол-во=32}
Администратор, введите цифры для выполнения действия:
 1 - Забанить клиента, 2 - Добавить товар, 3 - Информация о товаре, 4 - Выйти
```

Вариант 4, номер 10.

10. Система Железнодорожная касса. Пассажир делает Заявку на станцию назначения, время и дату поездки. Система регистрирует Заявку и осуществляет поиск подходящего Поезда. Пассажир делает выбор Поезда и получает Счет на оплату. Администратор вводит номера Поездов, промежуточные и конечные станции, цены

Код программы:

```
package dan.lab;

import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class Lab3_4_10 {
    public static class Passanger implements Serializable{
        private String FIO;
        private int balance;
        public ArrayList<Passanger> pas_list;

        public Passanger(String FIO, int balance) {
            this.FIO = FIO;
            this.balance = balance;
        }

        public Passanger() {
            pas_list = new ArrayList<>();
        }

        @Override
        public String toString() {
            return "passanger{" +
                    "FIO='" + FIO + '\'' +
                    ", balance=" + balance +
                    '}';
        }

        public String toString(int n) {
            if (n!=-1) {
                return "passanger{" +
                        "FIO='" + pas_list.get(n).FIO + '\'' +
                        ", balance=" + pas_list.get(n).balance +
                        '}';
            }else{
                return "Пассажир не найден";
            }
        }

        public void add_pas(String FIO, int balance){
            Passanger cash = new Passanger(FIO, balance);
            this.pas_list.add(cash);
        }

        public int search_pas(String name){
            for (int i=0;i!=this.pas_list.size();i++){
                if (this.pas_list.get(i).FIO.equals(name)){
                    return i;
                }
            }
            return -1;
```

```
    }

    public void add_money_pass(int value){
        this.balance+=value;
    }
}

public static class Train implements Serializable{
    private String depin;
    private String depout;
    private int day;
    private int t_out;
    private int t_in;
    private int cost;
    ArrayList<Train> train_list;

    public Train(String depin, String depout, int day, int t_out, int t_in, int cost) {
        this.depin = depin;
        this.depout = depout;
        this.day = day;
        this.t_out = t_out;
        this.t_in = t_in;
        this.cost = cost;
    }

    public Train() {
        train_list = new ArrayList<>();
    }

    @Override
    public String toString() {
        return "train{" +
                "depin='" + depin + '\'' +
                ", depout='" + depout + '\'' +
                ", day=" + day +
                ", t_out=" + t_out +
                ", t_in=" + t_in +
                ", cost=" + cost +
                '}';
    }
    public String toString(int n) {
        if (n!=-1) {
            return "train{" +
                    "depin='" + train_list.get(n).depin + '\'' +
                    ", depout='" + train_list.get(n).depout + '\'' +
                    ", day=" + train_list.get(n).day +
                    ", t_out=" + train_list.get(n).t_out +
                    ", t_in=" + train_list.get(n).t_in +
                    ", cost=" + train_list.get(n).cost +
                    '}';
        }else
        {
```

```
            return "Такого поезда нет";
        }
    }

    public void add_train(String depin, String depout, int day, int t_out, int t_in, int cost){
        Train cash = new Train(depin, depout, day, t_out, t_in, cost);
        this.train_list.add(cash);
    }

    public int searchg_train(String depout, int t_out){
        for(int i=0; i!=this.train_list.size();i++){
            if (this.train_list.get(i).depout.equals(depout) && this.train_list.get(i).t_out==t_out){
                return i;
            }
        }
        return -1;
    }


}

public static void main(String[] args) throws IOException, ClassNotFoundException {
    System.out.println("Вариант 4 №10");


    Passanger c_pass = new Passanger();
    c_pass.add_pas("Вася", 10000);
    FileOutputStream pass_file_w = new FileOutputStream("passobj.txt");
    ObjectOutputStream passobj_w = new ObjectOutputStream(pass_file_w);
    passobj_w.writeObject(c_pass);
    passobj_w.close();

    Train c_train = new Train();
    c_train.add_train("Питер","Москва",4,6,20,1200);
    FileOutputStream train_file_w = new FileOutputStream("trainobj.txt");
    ObjectOutputStream trainobj_w = new ObjectOutputStream(train_file_w);
    trainobj_w.writeObject(c_train);
    trainobj_w.close();

    FileInputStream pass_file = new FileInputStream("passobj.txt");
    ObjectInputStream passobj = new ObjectInputStream(pass_file);
    Object cash_i = passobj.readObject();
    Passanger passangers = (Passanger) cash_i;
    passobj.close();
    System.out.println("Пассажиры загруженны");

    FileInputStream train_file = new FileInputStream("trainobj.txt");
    ObjectInputStream trainobj = new ObjectInputStream(train_file);
    Object cash_c = trainobj.readObject();
    Train trains = (Train) cash_c;
    trainobj.close();
    System.out.println("Поезда загруженны");
```

```
        while (true) {
        Scanner in_1 = new Scanner(System.in);
        System.out.println("Вы пассажир или администратор? 1 - пассажир, 2 - администратор, 3 - выйти");
        int who = in_1.nextInt();
        String p_name="";
        int p_id = 0;

           while (who != 3 && who != 55 && who != 66) {
              Scanner in = new Scanner(System.in);
              switch (who) {
                 case 1:
                    System.out.println("Пассажир, введите своё имя");
                    p_name = in.nextLine();
                    p_id = passangers.search_pas(p_name);
                    if (p_id != -1) {
                       System.out.println("Добро пожаловать!");
                       who = 55;
                    } else {
                       who = -1;
                    }
                    System.out.println(passangers.toString(p_id));
                    break;
                 case 2:
                    System.out.println("Введите пароль");
                    if (in.nextLine().equals("123")) {
                       System.out.println("Успешно");
                       who = 66;
                    } else {
                       System.out.println("Пароль введён не верно");
                       who = -1;
                    }
                    break;
                 case 3:
                    who = 3;
                 default:
                    System.out.println("Такого пункта меню нет в списке");
                    who = -1;
              }
              if (who == -1) {
                 System.out.println("Вы пассажир или администратор? 1 - пассажир, 2 - администратор, 3 - выйти");
                 who = in.nextInt();

              }
           }

           if (who != 3) {
              System.out.println("Добро пожаловать в систему покупки билетов");
              boolean check = true;
              while (check) {
                 if (who == 55) {
```

```
                System.out.println("Пассажир, выбери пункт меню для совершения действий\n1 - купить билет, 2 - пополнить балланс, 3 - выйти");
                Scanner in_p = new Scanner(System.in);
                switch (Integer.parseInt(in_p.nextLine())) {
                    case 1:
                        System.out.println("Введите город отправления и время отправления");
                        in_p.reset();
                        String city = in_p.nextLine();
                        int t_out = Integer.parseInt(in_p.nextLine());
                        System.out.println("Информация о поезде");
                        int tr_id = trains.searchg_train(city, t_out);
                        System.out.println(trains.toString(tr_id));
                        System.out.println("Купить билет на этот поезд? 1 - да, 2 - нет");
                        switch (in_p.nextLine()) {
                            case "1":
                                if (passangers.pas_list.get(p_id).balance >= trains.train_list.get(tr_id).cost) {
                                    passangers.pas_list.get(p_id).balance -= trains.train_list.get(tr_id).cost;
                                    System.out.println("Покупка совершена");
                                } else {
                                    System.out.println("У вас недостаточно средств");
                                }
                                break;
                            case "2":
                                System.out.println("Отмена покупки");
                                break;
                            default:
                                System.out.println("Такого пункта меню нет");
                                break;
                        }
                        break;
                    case 2:
                        System.out.println("Введите сумму поплнения");
                        int value = Integer.parseInt(in_p.nextLine());
                        passangers.pas_list.get(p_id).add_money_pass(value);
                        System.out.println(passangers.pas_list.get(p_id).toString());
                    case 3:
                        check = false;
                        break;
                    default:
                        System.out.println("Такого пункта меню нет");
                        break;
                }
            } else {
                System.out.println("Администратор, выбери пункт меню для совершения действий\n1 - добавить поезд, 2 - посмотрет информацию о поезде, 3 - выйти");
                Scanner in_a = new Scanner(System.in);
                switch (in_a.nextLine()) {
                    case "1":
                        System.out.println("Введите:");
                        System.out.print("Место прибытия - ");
```

```
                    String d_in = in_a.nextLine();
                    System.out.print("Место отбытия - ");
                    String d_out = in_a.nextLine();
                    System.out.print("День отправки - ");
                    int day = Integer.parseInt(in_a.nextLine());
                    System.out.print("Время отправки - ");
                    int t_out = Integer.parseInt(in_a.nextLine());
                    System.out.print("Время прибытия - ");
                    int t_in = Integer.parseInt(in_a.nextLine());
                    System.out.print("Стоимость - ");
                    int cost = Integer.parseInt(in_a.nextLine());
                    trains.add_train(d_in, d_out, day, t_out, t_in, cost);
                    System.out.println("Новый рейс успешно добавлен");
                    System.out.println(trains.train_list.get(trains.searchg_train(d_out,
t_out)).toString());
                    break;
                case "2":
                    System.out.println("Введите:");
                    System.out.print("Место отбытия - ");
                    d_out = in_a.nextLine();
                    System.out.print("Время отправки - ");
                    t_out = Integer.parseInt(in_a.nextLine());
                    System.out.println(trains.train_list.get(trains.searchg_train(d_out,
t_out)).toString());
                    break;
                case "3":
                    check = false;
                    break;
                default:
                    System.out.println("Такого пункта меню нет");
                    break;
            }
        }
    }
}


//      //passanger c_pass = new passanger();
//      //c_pass.add_pas("Вася", 10000);
//      FileOutputStream pass_file_w = new FileOutputStream("passobj.txt");
//      ObjectOutputStream passobj_w = new ObjectOutputStream(pass_file_w);
//      passobj_w.writeObject(passangers);
//      passobj_w.close();
//
//      //train c_train = new train();
//      //c_train.add_train("Питер","Москва",4,6,20,1200);
//      FileOutputStream train_file_w = new FileOutputStream("trainobj.txt");
//      ObjectOutputStream trainobj_w = new ObjectOutputStream(train_file_w);
//      trainobj_w.writeObject(trains);
//      trainobj_w.close();
    }
```

```
    }
}
```

Результат выполнения программы:

```
Вариант 4 №10
Пассажиры загружены
Поезда загружены
Вы пассажир или администратор? 1 - пассажир, 2 - администратор, 3 - выйти
2
Введите пароль
123
Успешно
Добро пожаловать в систему покупки билетов
Администратор, выбери пункт меню для совершения действий
1 - добавить поезд, 2 - посмотрет информацию о поезде, 3 - выйти
1
Введите:
Место прибытия - Питер
Место отбытия - Москва
День отправки - 12
Время отправки - 2
Время прибытия - 4
Стоимость - 1200
Новый рейс успешно добавлен
train{depin='Питер', depout='Москва', day=12, t_out=2, t_in=4, cost=1200}
Администратор, выбери пункт меню для совершения действий
1 - добавить поезд, 2 - посмотрет информацию о поезде, 3 - выйти
```

Вывод: научились работать с наследованием, полиморфизмом и тд в java.