



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

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.01 Информатика и вычислительная техника

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

О Т Ч Е Т

по лабораторной работе № 3

Название: Классы, наследование, полиморфизм. Вариант 9

Дисциплина: Языки программирования для работы с большими
данными

Студент ИУ6-22М
(Группа)

К.Ю. Каташинский
(Подпись, дата) (И.О. Фамилия)

Преподаватель

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

Москва, 2024

Цель работы

Освоить принципы ООП на языке программирования Java.

Ход работы

Задание 1. 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.

Код программы представлен в листинге 1, результат – на рисунке 1.

Листинг 1 – Код программы задания 1

```
package Lab3;

class BinaryEquation {
    private int[] numbers;

    public BinaryEquation(int[] numbers) {
        this.numbers = numbers;
    }

    public BinaryEquation(int a, int b, int c) {
        this.numbers = new int[]{a, b, c};
    }

    private int getD() {
        int a = numbers[0];
        int b = numbers[1];
        int c = numbers[2];
        return b * b - 4 * a * c;
    }

    public double[] search() {
        int a = numbers[0];
        int b = numbers[1];
        return new double[]{
            (-b + Math.sqrt(getD())) / (2 * a),
            (-b - Math.sqrt(getD())) / (2 * a)
        };
    }
}
```

```

    }

    public double searchCenter() {
        int a = numbers[0];
        int b = numbers[1];
        return (double) -b / (2 * a);
    }

    public double[][] searchIntervals() {
        int a = numbers[0];
        double point = searchCenter();
        if (a > 0) {
            return new double[][]{
                {Integer.MIN_VALUE, point},
                {point, Integer.MAX_VALUE}
            };
        }
        return new double[][]{
            {point, Integer.MAX_VALUE},
            {Integer.MIN_VALUE, point}
        };
    }
}

public class Variant19 {
    public static void main(String[] args) {
        BinaryEquation[] bes = new BinaryEquation[]{
            new BinaryEquation(1, -3, 2),
            new BinaryEquation(1, 3, 2),
            new BinaryEquation(1, -5, 6),
            new BinaryEquation(1, 5, 6),
        };

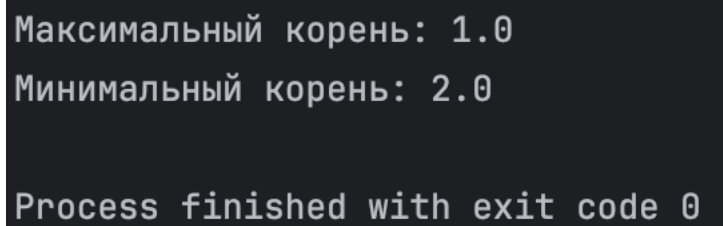
        double[] arr = bes[0].search();
        double min = arr[0], max = arr[1];
    }
}

```

```

        if (min < max) {
            double t = min;
            min = max;
            max = t;
        }
        for(int i = 1; i < bes.length; ++i) {
            double[] tempA = bes[i].search();
            double cMin = arr[0], cMax = arr[1];
            if (cMin < cMax) {
                double t = cMin;
                cMin = cMax;
                cMax = t;
            }
            if (cMin < min) {
                min = cMin;
            }
            if (cMax > max) {
                max = cMax;
            }
        }
        System.out.println("Максимальный корень: " + max);
        System.out.println("Минимальный корень: " + min);
    }
}

```



```

Максимальный корень: 1.0
Минимальный корень: 2.0

Process finished with exit code 0

```

Рисунок 1 – Результат задания 1

Задание 2. Определить класс Булева матрица (BoolMatrix) размерности ($n \times m$). 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 представлен в листинге 2, результат – на рисунке 2.

Листинг 2 – Код программы задания 2

```
package Lab3;

class BinaryMatrix {
    private int n, m;
    private boolean[][] matrix;
    public BinaryMatrix(boolean[][] matrix, int n, int m) {
        this.matrix = matrix;
        this.n = n;
        this.m = m;
    }
    public BinaryMatrix(int n, int m) {
        this.matrix = new boolean[n][m];
    }
    public boolean[][] get() {
        return this.matrix;
    }
    public void sum(BinaryMatrix bm) {
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < m; ++j) {
                if (!this.matrix[i][j] && !bm.matrix[i][j])
                {
                    this.matrix[i][j] = false;
                    continue;
                }
                this.matrix[i][j] = true;
            }
        }
    }
}
```

```

}

public void multiply(BinaryMatrix bm) {
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            if (this.matrix[i][j] && bm.matrix[i][j]) {
                this.matrix[i][j] = true;
                continue;
            }
            this.matrix[i][j] = false;
        }
    }
}

public void inverse() {
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            this.matrix[i][j] = !this.matrix[i][j];
        }
    }
}

public int countTrue() {
    int count = 0;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            if (this.matrix[i][j]) {
                ++count;
            }
        }
    }
    return count;
}

public void sort() {

```

```

int[][][] arr = new int[n][2];
for (int i = 0; i < n; ++i) {
    int count = 0;
    for (int j = 0; j < m; ++j) {
        if (this.matrix[i][j]) {
            ++count;
        }
    }
    arr[i][0] = count;
    arr[i][1] = i;
}
for (int i = 0; i < n - 1; ++i) {
    for (int j = 0; j < n - i - 1; ++j) {
        if (arr[j][0] > arr[j + 1][0]) {
            var t = arr[j];
            arr[j] = arr[j+1];
            arr[j+1] = t;
        }
    }
}
var newMatrix = new boolean[n][m];
for (int i = 0; i < n; ++i) {
    var pos = arr[i][1];
    newMatrix[i] = this.matrix[pos].clone();
}
this.matrix = newMatrix;
}
}

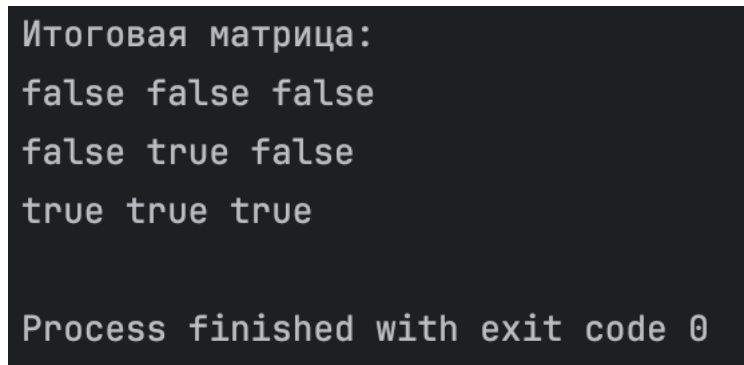
public class Variant110 {
    public static void main(String[] args) {
        BinaryMatrix bm = new BinaryMatrix(new boolean[][]{

```

```

        {true, true, true},
        {false, true, false},
        {false, false, false}
    }, 3, 3);
bm.sort();
var r = bm.get();
System.out.println("Итоговая матрица: ");
for (int i = 0; i < 3; ++i) {
    for (int j = 0; j < 3; ++j) {
        System.out.printf(r[i][j] + " ");
    }
    System.out.println();
}
}
}

```



```

Итоговая матрица:
false false false
false true false
true true true

Process finished with exit code 0

```

Рисунок 2 – Результат задания 2

Задание 3. Product: id, Наименование, UPC, Производитель, Цена, Срок хранения, Количество. Создать массив объектов. Вывести:

- а) список товаров для заданного наименования;
- б) список товаров для заданного наименования, цена которых не превосходит заданную;
- с) список товаров, срок хранения которых больше заданного.

Код программы представлен в листинге 3, результат – на рисунке 3.

Листинг 3 – Код программы задания 3

```
package Lab3;
```



```

import java.util.Objects;

class Product {
    private final int id;
    private int price;
    private int count;
    private final int days;
    private final String name;
    private final String upc;
    private final String creator;

    public Product(int id, int price, int count, int days,
String name, String upc, String creator) {
        this.id = id;
        this.price = price;
        this.count = count;
        this.days = days;
        this.name = name;
        this.upc = upc;
        this.creator = creator;
    }

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

    public void setCount(int count) {
        this.count = count;
    }

    public String getName() {
        return this.name;
    }

    public int getPrice() {
        return this.price;
    }
}

```

```

    }

    public int getCount() {
        return this.count;
    }

    public int getDays() {
        return this.days;
    }

    @Override
    public String toString() {
        return String.format("%d - %s - %s - %s - %d",
this.id, this.name, this.upc, this.creator, this.days);
    }
}

public class Variant29 {
    public static void main(String[] args) {
        var products = new Product[]{
            new Product(1, 2, 3, 4, "n1", "u2", "c3"),
            new Product(2, 2, 3, 10, "n1", "u3", "c3")
        };

        var name = "n1";
        var price = 5;
        var days = 7;
        System.out.println("a");
        for (Product pr : products) {
            if (pr.getName().equals(name)) {
                System.out.println(pr);
            }
        }

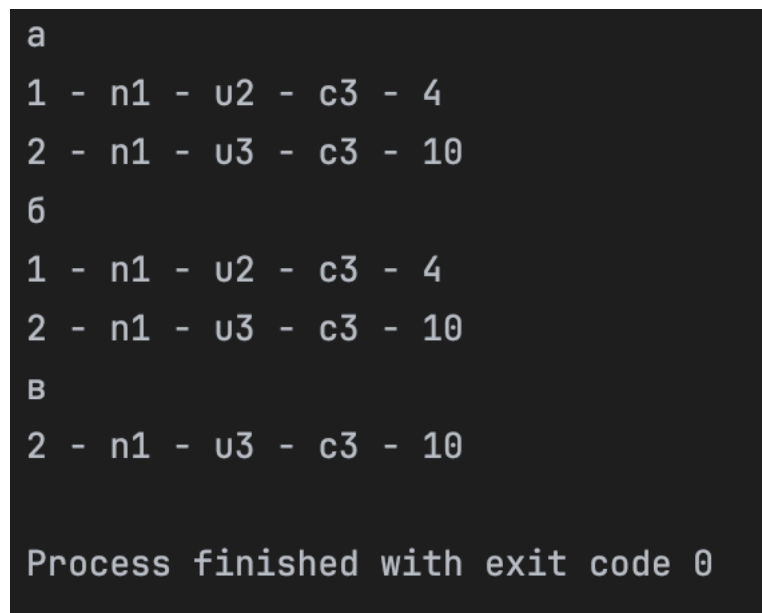
        System.out.println("ö");
        for (Product pr : products) {

```

```

        if (pr.getName().equals(name) && pr.getPrice()
<= price) {
            System.out.println(pr);
        }
    }
    System.out.println("В");
    for (Product pr : products) {
        if (pr.getDays() > days) {
            System.out.println(pr);
        }
    }
}
}

```



```

a
1 - n1 - u2 - c3 - 4
2 - n1 - u3 - c3 - 10
b
1 - n1 - u2 - c3 - 4
2 - n1 - u3 - c3 - 10
В
2 - n1 - u3 - c3 - 10

Process finished with exit code 0

```

Рисунок 3 – Результат задания 3

Задание 4. Train: Пункт назначения, Номер поезда, Время отправления, Число мест (общих, купе, плацкарт, люкс). Создать массив объектов. Вывести:

- список поездов, следующих до заданного пункта назначения;
- список поездов, следующих до заданного пункта назначения и отправляющихся после заданного часа;
- список поездов, отправляющихся до заданного пункта назначения и имеющих общие места.

Код программы представлен в листинге 4, результат – на рисунке 4.

Листинг 4 – Код программы задания 4

```
package Lab3;

class Train {
    private final String destination;
    private final String number;
    private int departureTime;
    private final int generalCount;
    private final int compartmentCount;
    private final int reservedCount;
    private final int suiteCount;

    public Train(String destination, String number, int
departureTime, int generalCount, int compartmentCount, int
reservedCount, int suiteCount) {
        this.destination = destination;
        this.number = number;
        this.departureTime = departureTime;
        this.generalCount = generalCount;
        this.compartmentCount = compartmentCount;
        this.reservedCount = reservedCount;
        this.suiteCount = suiteCount;
    }

    public void setDepartureTime(int departureTime) {
        this.departureTime = departureTime;
    }

    public String getDestination() {
        return this.destination;
    }

    public int getDepartureTime() {
        return this.departureTime;
    }
}
```

```

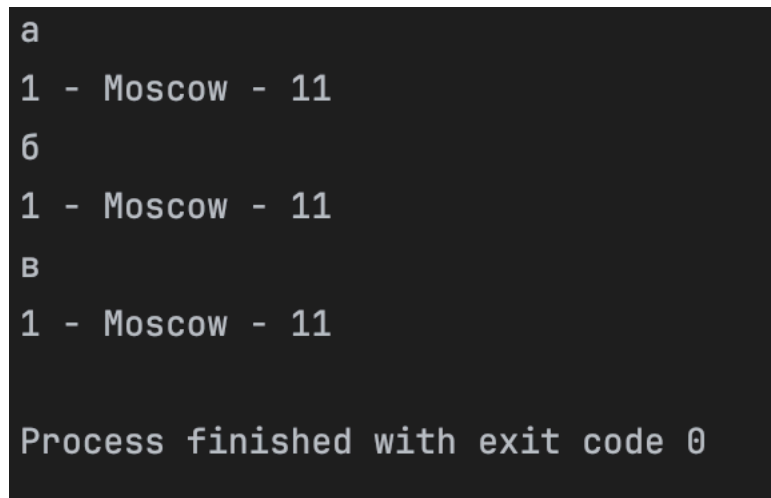
    public int getGeneralCount() {
        return this.generalCount;
    }
    @Override
    public String toString() {
        return String.format("%s - %s - %d", this.number,
this.destination, this.departureTime);
    }
}
public class Variant210 {
    public static void main(String[] args) {
        var trains = new Train[]{
            new Train("Moscow", "1", 11, 50, 0, 10, 0),
            new Train("London", "2", 5, 0, 50, 10, 0)
        };
        var destination = "Moscow";
        var departureTime = 10;
        var generalCount = 0;
        System.out.println("a");
        for(Train tr : trains) {
            if (tr.getDestination().equals(destination)) {
                System.out.println(tr);
            }
        }
        System.out.println("б");
        for(Train tr : trains) {
            if (tr.getDestination().equals(destination) &&
tr.getDepartureTime() > departureTime) {
                System.out.println(tr);
            }
        }
    }
}

```

```

        System.out.println("В");
        for(Train tr : trains) {
            if (tr.getDestination().equals(destination) &&
tr.getGeneralCount() > generalCount) {
                System.out.println(tr);
            }
        }
    }
}

```



```

а
1 - Moscow - 11
б
1 - Moscow - 11
в
1 - Moscow - 11

Process finished with exit code 0

```

Рисунок 4 – Результат задания 4

Задание 5. Создать объект класса Фотоальбом, используя класс Фотография. Методы: задать название фотографии, дополнить фотоальбом фотографией, вывести на консоль количество фотографий.

Код программы представлен в листинге 5, результат – на рисунке 5.

Листинг 5 – Код программы задания 5

```

package Lab3;

import java.util.Arrays;
import java.util.Objects;

class Photo {
    private String name, sizes;
    public Photo(String name, String sizes) {
        this.name = name;
    }
}

```

```

        this.sizes = sizes;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return this.name;
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Photo)) return false;
        return Objects.equals(((Photo) o).name, name);
    }
    @Override
    public int hashCode() {
        return Objects.hash(name);
    }
    @Override
    public String toString() {
        return this.name;
    }
}

class PhotoAlbum {
    private Photo[] photos;
    public PhotoAlbum() {
        photos = new Photo[]{};
    }
    public void setName(String name, int index) {
        if (index < 0 || index >= this.photos.length)
return;

```

```

        this.photos[index].setName(name);
    }
    public void append(Photo p) {
        var t = new Photo[this.photos.length + 1];
        System.arraycopy(this.photos, 0, t, 0,
photos.length);
        t[t.length - 1] = p;
        this.photos = t;
    }
    public void printCount() {
        System.out.println("Количество: " +
this.photos.length);
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof PhotoAlbum)) return false;
        return Arrays.equals(((PhotoAlbum) o).photos,
photos);
    }
    @Override
    public int hashCode() {
        return Objects.hash((Object) photos);
    }
    @Override
    public String toString() {
        return this.photos[0].getName();
    }
}
public class Variant39 {
    public static void main(String[] args) {

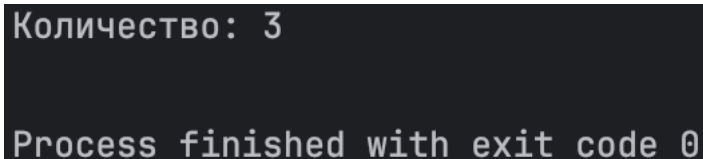
```



```

        PhotoAlbum album = new PhotoAlbum();
        album.append(new Photo("1", "1x2"));
        album.append(new Photo("1", "2x2"));
        album.append(new Photo("1", "3x2"));
        album.printCount();
    }
}

```



```

Количество: 3
Process finished with exit code 0

```

Рисунок 5 – Результат задания 5

Задание 6. 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 представлен в листинге 6, результат – на рисунке 6.

Листинг 6 – Код программы задания 6

```

package Lab3;
import java.util.Objects;
class Day {
    private int number;
    public Day(int number) {
        this.number = number;
    }
    public int getNumber() {
        return this.number;
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Day)) return false;
        return Objects.equals(((Day) o).number, number);
    }
}

```

```

    }

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

    @Override
    public String toString() {
        return Integer.toString(this.number);
    }
}

class Month {
    private int number;

    public Month(int number) {
        this.number = number;
    }

    public int getNumber() {
        return this.number;
    }

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

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

    @Override
    public String toString() {
        return Integer.toString(this.number);
    }
}

```

```

    }
}
class Year {
    private Day day;
    private Month month;
    private int number;
    public Year(Day d, Month m, int number) {
        this.day = d;
        this.month = m;
        this.number = number;
    }
    public void printWeek() {
        var week = (this.month.getNumber() - 1) * 4 +
this.day.getNumber() / 2 + 1;
        System.out.println("Неделя: " + week);
    }
    public void printDifference(Year y) {
        var days = Math.abs(this.day.getNumber() -
y.day.getNumber());
        var months = Math.abs(this.month.getNumber() -
y.month.getNumber());
        System.out.println("Количество дней: " + days +
months * 30);
        System.out.println("Количество месяцев: " +
months);
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Year)) return false;
        return Objects.equals(((Year) o).number, number);
    }
}

```

```

    }

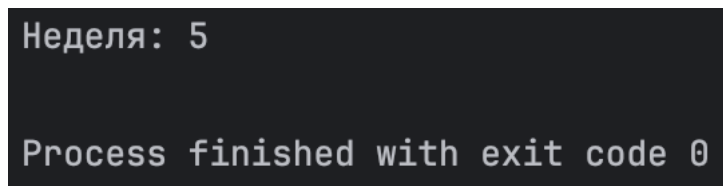
    @Override
    public int hashCode() {
        return Objects.hash(number);
    }

    @Override
    public String toString() {
        return String.format("%s.%s.%d", this.day,
this.month, this.number);
    }
}

public class Variant310 {
    public static void main(String[] args) {
        Year year = new Year(new Day(1), new Month(2),
2024);

        year.printWeek();
    }
}

```



```

Неделя: 5

Process finished with exit code 0

```

Рисунок 6 – Результат задания 6

Задание 7. Построить модель программной системы. 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 представлен в листинге 7, результат – на рисунке 7.

Листинг 7 – Код программы задания 7

```

package Lab3;

import java.util.Objects;

```

```

class ShopProduct {
    private String name;
    public ShopProduct(String name) {
        this.name = name;
    }
}

class User {
    private String name;
    private boolean banned = false;
    public User(String name) {
        this.name = name;
        this.banned = false;
    }
    public void ban(boolean banned) {
        this.banned = banned;
    }
    public String getName() {
        return this.name;
    }
}

class Order {
    private int id;
    private ShopProduct[] list;
    private User user;
    private boolean payed;
    public Order(int id, ShopProduct[] list, User user) {
        this.id = id;
        this.list = list;
        this.user = user;
        this.payed = false;
    }
}

```

```

    public int getID() {
        return id;
    }

    public void pay() {
        this.payed = true;
    }
}

class Shop {
    private ShopProduct[] products = {};
    private Order[] orders = {};
    private User[] users = {};

    public void addProduct(ShopProduct p) {
        var t = new ShopProduct[this.users.length + 1];
        System.arraycopy(this.products, 0, t, 0,
products.length);
        t[t.length - 1] = p;
        this.products = t;
    }

    public void addOrder(Order o) {
        var t = new Order[this.users.length + 1];
        System.arraycopy(this.orders, 0, t, 0,
orders.length);
        t[t.length - 1] = o;
        this.orders = t;
    }

    public void addUser(User u) {
        var t = new User[this.users.length + 1];
        System.arraycopy(this.users, 0, t, 0,
users.length);
        t[t.length - 1] = u;
        this.users = t;
    }
}

```

```

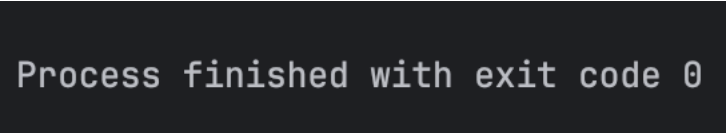
    }

    public User findUser(String name) {
        for (var u : users) {
            if (Objects.equals(u.getName(), name)) {
                return u;
            }
        }
        return null;
    }

    public Order findOrder(int id) {
        for (var o : orders) {
            if (o.getID() == id) {
                return o;
            }
        }
        return null;
    }
}

public class Variant49 {
    public static void main(String[] args) {
        var shop = new Shop();
        var product = new ShopProduct("product");
        var user = new User("user");
        shop.addProduct(product);
        shop.addUser(user);
        shop.addOrder(new Order(1, new
ShopProduct[]{product}, user));
    }
}

```



```
Process finished with exit code 0
```

Рисунок 7 – Результат задания 7

Задание 8. Построить модель программной системы. 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 представлен в листинге 8, результат – на рисунке 8.

Листинг 8 – Код программы задания 8

```
package Lab3;
import java.util.Objects;
class Request {
    private String destination, date;
    public Request(String dest, String d) {
        this.destination = dest;
        this.date = d;
    }
    public String getDestination() {
        return this.destination;
    }
    public String getDate() {
        return this.date;
    }
}
class OfficeTrain {
    private String number, destination, date;
    private int price;
```



```

    public OfficeTrain(String n, String dest, String d, int
p) {
        this.number = n;
        this.destination = dest;
        this.date = d;
        this.price = p;
    }
    public String getDestination() {
        return this.destination;
    }
    public String getDate() {
        return this.date;
    }
    public int getPrice() {
        return this.price;
    }
}
class BoxOffice {
    private OfficeTrain[] trains = {};
    private Request[] requests = {};
    public void addTrain(OfficeTrain tr) {
        var t = new OfficeTrain[this.trains.length + 1];
        System.arraycopy(this.trains, 0, t, 0,
trains.length);
        t[t.length - 1] = tr;
        this.trains = t;
    }
    public void addRequest(Request r) {
        var t = new Request[this.requests.length + 1];
        System.arraycopy(this.requests, 0, t, 0,
requests.length);

```

```

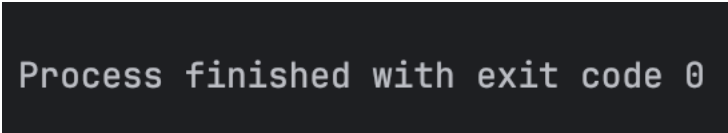
        t[t.length - 1] = r;
        this.requests = t;
    }

    public OfficeTrain[] findTrains(Request r) {
        OfficeTrain[] res = {};
        for (var t:trains) {
            if (!Objects.equals(t.getDestination(),
r.getDestination())) {
                continue;
            }
            if (!Objects.equals(t.getDate(), r.getDate()))
{
                continue;
            }
            var temp = new OfficeTrain[res.length + 1];
            System.arraycopy(res, 0, temp, 0, res.length);
            temp[temp.length - 1] = t;
            res = temp;
        }
        return res;
    }
}

public class Variant410 {
    public static void main(String[] args) {
        var office = new BoxOffice();
        var train = new OfficeTrain("abc", "Moscow",
"01.01.2024", 500);
        var request = new Request("Moscow", "01.01.2024");
        office.addTrain(train);
        office.addRequest(request);
    }
}

```

}



```
Process finished with exit code 0
```

Рисунок 8 – Результат задания 8

Вывод

В ходе выполнения лабораторной работы были освоены принципы ООП на языке программирования Java.