МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ОТЧЁТ

по лабораторной работе №5

Выполнил:
студент группы ПО-9
Дарашкевич Д.И.

Проверил:
Крощенко А.А.

Брест 2024

Цель работы: приобрести практические навыки в области объектно-ориентированного проектирования.

Вариант 4

Задание 1:

Реализовать абстрактные классы или интерфейсы, а также наследование и полиморфизм для следующих классов:

4) interface Учебное Заведение ← class Колледж ← class Университет.

Код программы:

EducationalInstitution

```
public interface EducationalInstitution {
    void addStudent(String studentName);

    void removeStudent(String studentName);

    void getStudentInfo();
}
```

College

```
public class College implements EducationalInstitution {
    @Override
    public void addStudent(String studentName) {
        System.out.println("Студент " + studentName + " добавлен в колледж.");
    }

    @Override
    public void removeStudent(String studentName) {
        System.out.println("Студент " + studentName + " удален из колледжа.");
    }

    @Override
    public void getStudentInfo() {
        System.out.println("Информация о студентах в колледже:");
    }
}
```

University

```
public class University implements EducationalInstitution {
    @Override
    public void addStudent(String studentName) {
        System.out.println("Студент " + studentName + " добавлен в университет.");
    }

    @Override
    public void removeStudent(String studentName) {
        System.out.println("Студент " + studentName + " удален из университета.");
    }

    @Override
    public void getStudentInfo() {
        System.out.println("Информация о студентах в университете:");
    }
}
```

Main

```
public class Main {
    public static void main(String[] args) {
        College college = new College();

        college.addStudent("Иванов");
        college.addStudent("Петров");
        college.addStudent("Сидоров");

        college.getStudentInfo();
```

```
        college.removeStudent("Петров");

        college.getStudentInfo();

        University university = new University();

        university.addStudent("Козлов");
        university.addStudent("Михайлов");

        university.getStudentInfo();

        university.removeStudent("Михайлов");

        university.getStudentInfo();
    }
}
```

Входные данные:

```
        college.addStudent("Иванов");
        college.addStudent("Петров");
        college.addStudent("Сидоров");

        college.getStudentInfo();

        college.removeStudent("Петров");

        college.getStudentInfo();

        University university = new University();

        university.addStudent("Козлов");
        university.addStudent("Михайлов");

        university.getStudentInfo();

        university.removeStudent("Михайлов");

        university.getStudentInfo();
```

Результат работы программы:

```
C:\Users\Legion\.jdks\openjdk-22.0.1\bin\java.exe "-javaagent:D:\IntelliJ IDEA 20
 -classpath "C:\Users\Legion\Desktop\6 семестр\СПП\lab5\1\out\production\1" Main
Студент Иванов добавлен в колледж.
Студент Петров добавлен в колледж.
Студент Сидоров добавлен в колледж.
Информация о студентах в колледже:
Студент Петров удален из колледжа.
Информация о студентах в колледже:
Студент Козлов добавлен в университет.
Студент Михайлов добавлен в университет.
Информация о студентах в университете:
Студент Михайлов удален из университета.
Информация о студентах в университете:

Process finished with exit code 0
```

Задание 2:

В следующих заданиях требуется создать суперкласс (абстрактный класс, интерфейс) и определить общие методы для данного класса. Создать подклассы, в которых добавить

специфические свойства и методы. Часть методов переопределить. Создать массив объектов суперкласса и заполнить объектами подклассов. Объекты подклассов идентифицировать конструктором по имени или идентификационному номеру. Использовать объекты подклассов для моделирования реальных ситуаций и объектов. 4) Создать суперкласс Грузоперевозчик и подклассы Самолет, Поезд, Автомобиль. Определить время и стоимость перевозки для указанных городов и расстояний.

Код программы:

Carrier

```
public abstract class Carrier {
    protected String type;
    protected String route;
    protected double distance;
    protected double transportTime;
    protected double transportCost;

    public Carrier(String type, String route, double distance) {
        this.type = type;
        this.route = route;
        this.distance = distance;
    }

    public abstract void calculateTime();

    public abstract void calculateCost();

    public void displayTransportInfo() {
        System.out.println("Carrier type: " + type);
        System.out.println("Route: " + route);
        System.out.println("Distance: " + distance + " km");
        System.out.println("Transport time: " + transportTime + " hours");
        System.out.println("Transport cost: " + transportCost + " RUB");
    }
}
```

Airplane

```
public class Airplane extends Carrier {
    private double speed;

    public Airplane(String route, double distance, double speed) {
        super("Airplane", route, distance);
        this.speed = speed;
    }

    @Override
    public void calculateTime() {
        transportTime = distance / speed;
    }

    @Override
    public void calculateCost() {
        transportCost = distance * 10;
    }
}
```

Train

```
public class Train extends Carrier {
    private double speed;

    public Train(String route, double distance, double speed) {
        super("Train", route, distance);
        this.speed = speed;
    }
```

```
    @Override
    public void calculateTime() {
        transportTime = distance / speed;
    }

    @Override
    public void calculateCost() {
        transportCost = distance * 5;
    }
}
```

Car

```
public class Car extends Carrier {
    private double speed;

    public Car(String route, double distance, double speed) {
        super("Car", route, distance);
        this.speed = speed;
    }

    @Override
    public void calculateTime() {
        transportTime = distance / speed;
    }

    @Override
    public void calculateCost() {
        transportCost = distance * 7; // Example cost calculation for a car
    }
}
```

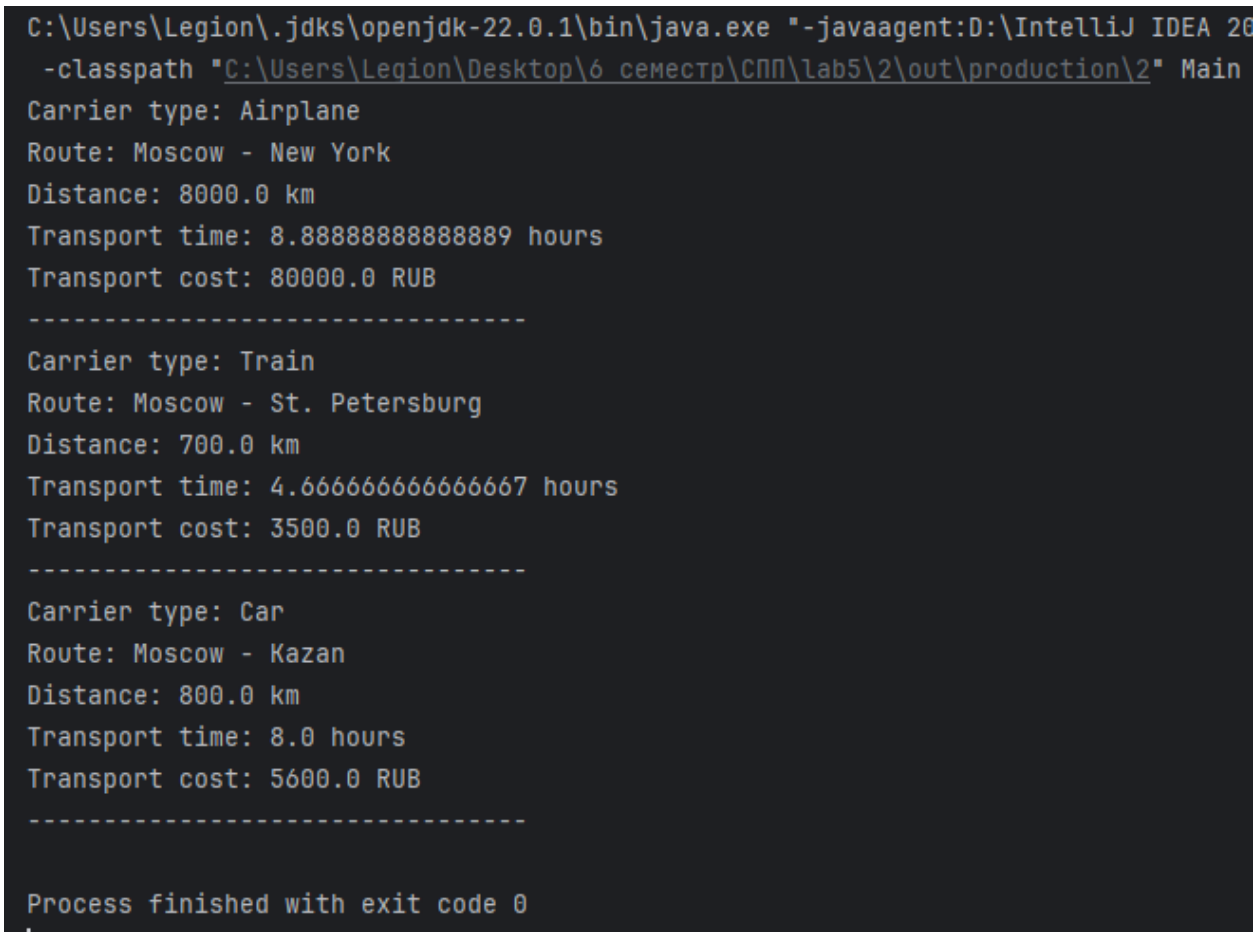
Main

```
public class Main {
    public static void main(String[] args) {
        Carrier[] carriers = new Carrier[3];

        carriers[0] = new Airplane("Moscow - New York", 8000, 900);
        carriers[1] = new Train("Moscow - St. Petersburg", 700, 150);
        carriers[2] = new Car("Moscow - Kazan", 800, 100);

        for (Carrier carrier : carriers) {
            carrier.calculateTime();
            carrier.calculateCost();
            carrier.displayTransportInfo();
            System.out.println("---------------------------------");
        }
    }
}
```

Входные данные:

```
        carriers[0] = new Airplane("Moscow - New York", 8000, 900);
        carriers[1] = new Train("Moscow - St. Petersburg", 700, 150);
        carriers[2] = new Car("Moscow - Kazan", 800, 100);
```

Результат работы программы:



Задание 3:

В задании 3 ЛР №4, где возможно, заменить объявления суперклассов объявлениями абстрактных классов или интерфейсов.

Система Вступительные экзамены. Абитуриент регистрируется на Факультет, сдает Экзамены. Преподаватель выставляет Оценку. Система подсчитывает средний балл и определяет Абитуриентов, зачисленных в учебное заведен.

Код программы:

Applicant

```
public class Applicant {
    private String name;
    private double averageGrade;

    public Applicant(String name, double averageGrade) {
        this.name = name;
        this.averageGrade = averageGrade;
    }

    public String getName() {
        return name;
    }

    public double getAverageGrade() {
        return averageGrade;
    }
}
```

Exam

```
abstract class Exam {
    private String name;

    public Exam(String name) {
        this.name = name;
```

```
    }

    public String getName() {
        return name;
    }

    public abstract void conductExam();

    public abstract void gradeExam(Applicant applicant);
}
```

Faculty

```
class Faculty {
    private String name;

    public Faculty(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

Teacher

```
interface Teacher {
    void gradeExam(Exam exam, Applicant applicant);
}
```

EnrollmentSystem

```
import java.util.ArrayList;
import java.util.List;

public class EnrollmentSystem {
    private List<Applicant> applicants;

    public EnrollmentSystem() {
        this.applicants = new ArrayList<>();
    }

    public void registerApplicant(Applicant applicant) {
        applicants.add(applicant);
    }

    public List<Applicant> determineAdmittedApplicants(double passingGrade) {
        List<Applicant> admittedApplicants = new ArrayList<>();
        for (Applicant applicant : applicants) {
            if (applicant.getAverageGrade() >= passingGrade) {
                admittedApplicants.add(applicant);
            }
        }
        return admittedApplicants;
    }
}
```

Main

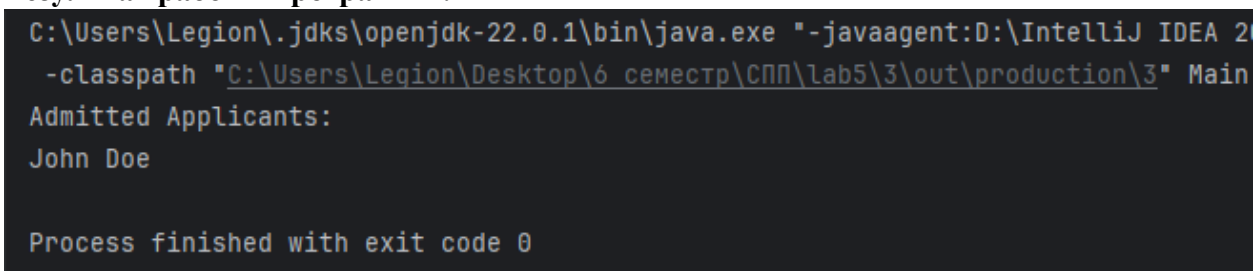
```
import java.util.List;

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

        Applicant applicant1 = new Applicant("John Doe", 85.0);
        Applicant applicant2 = new Applicant("Jane Smith", 60.5);
        Applicant applicant3 = new Applicant("Alice Johnson", 65.0);
        admissionSystem.registerApplicant(applicant1);
        admissionSystem.registerApplicant(applicant2);
```

```
        admissionSystem.registerApplicant(applicant3);

        List<Applicant> admittedApplicants =
admissionSystem.determineAdmittedApplicants(70.0);

        System.out.println("Admitted Applicants:");
        for (Applicant admittedApplicant : admittedApplicants) {
            System.out.println(admittedApplicant.getName());
        }
    }
}
```

Входные данные:

```
Applicant applicant1 = new Applicant("John Doe", 85.0);
Applicant applicant2 = new Applicant("Jane Smith", 60.5);
Applicant applicant3 = new Applicant("Alice Johnson", 65.0);
```

Результат работы программы:



Вывод: в ходе выполнения данной лабораторной работы я приобрел практические навыки в области объектно-ориентированного проектирования.