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, искусственный интеллект и системы управления**

КАФЕДРА **Компьютерные системы и сети**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Интеллектуальные системы анализа, обработки и интерпретации больших данных**

ОТЧЕТ
по лабораторной работе №4

Название: Внутренние классы и интерфейсы

Дисциплина Языки программирования для работы с большими данными

Студент	ИУ6–22М		М.Э.Хабаров
	(Группа)	(Подпись, дата)	(И.О. Фамилия)
Преподаватель			П.В. Степанов
		(Подпись, дата)	(И.О. Фамилия)

2024 г

Цель: изучить работу с внутренними классами и интерфейсами на языке Java.

Задание №1

Формулировка задания и код программы представлены в листинге 1:

```
/*
    Создать класс Computer (компьютер) с внутренним классом, с помощью
объектов
    которого можно хранить информацию об операционной системе, процессоре
    и оперативной памяти.
*/
class Computer {
    private String brand;
    private String model;
    private Specs specs;

    public Computer(String brand, String model) {
        this.brand = brand;
        this.model = model;
        this.specs = new Specs();
    }

    public Specs getSpecs() {
        return this.specs;
    }

    public void displayInfo() {
        System.out.println("Computer: " + brand + " " + model);
        System.out.println("Operating System: " + specs.operatingSystem);
        System.out.println("Processor: " + specs.processor);
        System.out.println("RAM: " + specs.ram + "GB");
    }

    class Specs {
        private String operatingSystem;
        private String processor;
        private int ram;

        public void setOperatingSystem(String operatingSystem) {
            this.operatingSystem = operatingSystem;
        }

        public void setProcessor(String processor) {
            this.processor = processor;
        }

        public void setRam(int ram) {
            this.ram = ram;
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Computer myComputer = new Computer("Dell", "Inspiron");

        Computer.Specs myComputerSpecs = myComputer.getSpecs();
        myComputerSpecs.setOperatingSystem("Windows 10");
        myComputerSpecs.setProcessor("Intel Core i5");
        myComputerSpecs.setRam(8);
```

```
        myComputer.displayInfo();
    }
}
```

Задание №2

Формулировка задания и код программы представлены в листинге 2:

```
import java.util.ArrayList;

/*
    Создать класс Park (парк) с внутренним классом, с помощью объектов
которого
    можно хранить информацию об аттракционах, времени их работы и стоимости.
*/
class Park {
    private String name;
    private Attractions attractions;

    public Park(String name) {
        this.name = name;
        this.attractions = new Attractions();
    }

    public Attractions getAttractions() {
        return this.attractions;
    }

    public void displayInfo() {
        System.out.println("Park: " + name);
        System.out.println("List of attractions: " +
attractions.attractions_list);
    }

    class Attractions {
        private ArrayList<String> attractions_list = new ArrayList<>();

        public void addAttraction(String attraction) {
            this.attractions_list.add(attraction);
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Park park = new Park("DisneyLand");

        Park.Attractions parkAttractions = park.getAttractions();
        parkAttractions.addAttraction("roller coaster");
        parkAttractions.addAttraction("ferris wheel");
        parkAttractions.addAttraction("house of horror");

        park.displayInfo();
    }
}
```

Задание №3

Формулировка задания и код программы представлены в листинге 3:

```java
/*
    interface Корабль <- class Грузовой Корабль <- class Танкер.
*/

// Интерфейс Ship
interface Ship {
    void sail();
}

// Класс CargoShip реализует интерфейс Ship
class CargoShip implements Ship {
    @Override
    public void sail() {
        System.out.println("Cargo ship is sailing.");
    }
}

// Класс Tanker наследует от CargoShip
class Tanker extends CargoShip {
    @Override
    public void sail() {
        System.out.println("Tanker is sailing.");
    }
}

public class Main {
    public static void main(String[] args) {
        Ship cargoShip = new CargoShip();
        Ship tanker = new Tanker();

        cargoShip.sail();
        tanker.sail();
    }
}
```

Задание №4

Формулировка задания и код программы представлены в листинге 4:

```java
/*
    interface Мебель <- abstract class Шкаф <- class Книжный Шкаф
 */

// Интерфейс Furniture
interface Furniture {
    void description();
}

// Абстрактный класс Closet реализует интерфейс Furniture
abstract class Closet implements Furniture {
    @Override
    public void description() {
        System.out.println("This is a closet.");
    }

    abstract void type();
}

// Класс Bookshelf наследует от абстрактного класса Closet
class Bookshelf extends Closet {
    @Override
```

```
    void type() {
        System.out.println("This is a bookshelf.");
    }
}

public class Main {
    public static void main(String[] args) {
        Furniture bookshelf = new Bookshelf();

        bookshelf.description();
        ((Closet) bookshelf).type(); // Приведение типа для вызова метода
type()
    }
}
```

Вывод: в результате выполнения лабораторной работы были освоены основные принципы внутренних классов и интерфейсов на языке Java.