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 (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 №4

Название: Внутренние_классы,_Интерфейсы

Дисциплина: Языки программирования для работы с большими данными

Студент ИУ6-12М (Группа) _____ (Подпись, дата) В.А. Трофимов (И.О. Фамилия)

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

Москва, 2023

Задания:

1.6. Создать класс Shop (магазин) с внутренним классом, с помощью объектов которого можно хранить информацию об отделах, товарах и услуг.

1.7. Создать класс Справочная Служба Общественного Транспорта с внутренним классом, с помощью объектов которого можно хранить информацию о времени, линиях маршрутов и стоимости проезда.

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

2.6. interface Корабль <- abstract class Военный Корабль <- class Авианосец.

2.7. interface Врач <- class Хирург <- class Нейрохирург.

Код для решения задания 1.6:

```
package bdjava.lab4;

import static java.lang.System.out;

public class Program1 {
    public static void main(String[] args) {
        Shop shop = new Shop("Абибас");
        Shop.Department dep = shop.new Department("Зимняя обувь");
        dep.setDescription("ПОлезные вещи для зимы");
        Shop.Product prod = shop.new Product("Сапоги", 2000);
        prod.setDescription("Модные, молодежные");
        prod.setDepartment(dep);
        Shop.Service serv = shop.new Service("Чистка обуви", 300);
        serv.setDescription("Быстро и качественно");
        out.println("-----------");
        out.println("Отдел:");
        out.println(dep);
        out.println("-----------");
        out.println("Товар:");
        out.println(prod);
        out.println("-----------");
        out.println("Услуга:");
```

```
        out.println(serv);

    }
}

class Shop {
    private String name;
    class Department {
        private final int ID;
        private final String title;
        private String description;
        private static int numOfDepartaments;
        Department (String title) {
            this.ID = Department.numOfDepartaments;
            Department.numOfDepartaments++;
            this.title = title;
        }
        public int getID() {
            return ID;
        }
        public String getDescription() {
            return description;
        }
        public void setDescription(String description) {
            this.description = description;
        }
        @Override
        public String toString() {
            return ID + "\n" +
                    title + "\n" +
                    description + "\n";
        }
    }
    class Product {
        private final int ID;
        private final String title;
        private int cost;
```

```
        private String description;
        private Department department;
        private static int numOfProducts;
        Product (String title) {
            this.ID = Product.numOfProducts;
            Product.numOfProducts++;
            this.title = title;
        }
        Product (String title, int cost) {
            this(title);
            this.cost = cost;
        }
        public void setCost(int cost) {
            this.cost = cost;
        }
        public int getCost() {
            return cost;
        }
        public int getID() {
            return ID;
        }
        public String getDescription() {
            return description;
        }
        public void setDescription(String description) {
            this.description = description;
        }
        public Department getDepartment() {
            return department;
        }
        public void setDepartment(Department department) {
            this.department = department;
        }
        @Override
        public String toString() {
            return ID + "\n" +
                    title + "\n" +
```

```
                    cost + "py6\n" +
                    description + "\n" +
                    department + "\n";
        }
    }
    class Service {
        private final int ID;
        private final String title;
        private int cost;
        private String description;
        private static int numOfServices;
        Service (String title) {
            this.ID = Service.numOfServices;
            Service.numOfServices++;
            this.title = title;
        }
        Service (String title, int cost) {
            this(title);
            this.cost = cost;
        }
        public void setCost(int cost) {
            this.cost = cost;
        }
        public int getCost() {
            return cost;
        }
        public int getID() {
            return ID;
        }
        public String getDescription() {
            return description;
        }
        public void setDescription(String description) {
            this.description = description;
        }
        @Override
        public String toString() {
```

```
            return ID + "\n" +
                    title + "\n" +
                    cost + "руб\n" +
                    description + "\n";
        }
    }
    Shop (String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

Код для решения задания 1.7:

```
package bdjava.lab4;

import java.util.*;

import static java.lang.System.out;

public class Program2 {

    public static void main(String[] args) {
        Calendar calendar = new GregorianCalendar();
        calendar.set(Calendar.YEAR, 2023);
        calendar.set(Calendar.MONTH, 0);
        calendar.set(Calendar.DAY_OF_MONTH, 25);
        calendar.set(Calendar.HOUR_OF_DAY, 19);
        calendar.set(Calendar.MINUTE, 42);
        calendar.set(Calendar.SECOND, 12);

        PublicTransportInformationService serv = new
```

```
PublicTransportInformationService();
        PublicTransportInformationService.Transport bus = serv.new
Transport("OP344TT322", "bus");
        bus.setCost(100);
        bus.setRouteLineId(16);
        bus.addToSchedule("London", calendar);
        out.println(bus);
    }
}

class PublicTransportInformationService {
    class Transport {
        private final String num;
        private final String type;
        private HashMap <String, Calendar> schedule;
        private int routeLineId;
        private  int cost;
        Transport (String num, String type) {
            this.num = num;
            this.type = type;
            schedule = new HashMap<String, Calendar>();
        }
        public int getCost() {
            return cost;
        }
        public void setCost(int cost) {
            this.cost = cost;
        }
        public int getRouteLineId() {
            return routeLineId;
        }
        public void setRouteLineId(int routeLineId) {
            this.routeLineId = routeLineId;
        }
        public String getNum() {
            return num;
        }
```

```
        public String getType() {
            return type;
        }
        public HashMap<String, Calendar> getSchedule() {
            return schedule;
        }
        public void addToSchedule(String halt, Calendar time) {
            this.schedule.put(halt, time);
        }
        public void setSchedule(HashMap<String, Calendar> schedule) {
            this.schedule = schedule;
        }

        @Override
        public String toString() {
            StringBuilder buf = new StringBuilder("Transport{" +
                    "num='" + num + '\'' +
                    ", type='" + type + '\'' +
                    ", schedule=(\n");
            for (String key : schedule.keySet())
                buf.append("<" + key + " : " + schedule.get(key).getTime() +
">\n");
            buf.append(")\n" +
                    ", routeLineId=" + routeLineId +
                    ", cost=" + cost +
                    '}');
            return buf.toString();
        }
    }
}
```

Код для решения задания 2.6:

```
package bdjava.lab4;

import java.util.ArrayList;
```

```
import static java.lang.System.out;

public class Program3 {
    public static void main(String[] args) {

    }
}

interface Ship {
    public boolean store (Object obj);
    public boolean swim (double dlatitude, double dlongitude);
    public void dropAnchor ();
    public void raiseAnchor ();
    public String sendSignal (String str);
    public void getSignal (String str);
}

abstract class Warship implements Ship {
    private double longitude;
    private double latitude;
    private boolean isAnchorUp;
    @Override
    public boolean swim(double dlatitude, double dlongitude) {
        this.latitude += dlatitude;
        this.longitude += dlongitude;
        return true;
    }
    @Override
    public void dropAnchor() {
        isAnchorUp = false;
    }
    @Override
    public void raiseAnchor() {
        isAnchorUp = true;
    }
    @Override
    public void getSignal(String str) {
```

```
        out.println("Получено сообщение: " + str);
    }
    @Override
    public String sendSignal(String str) {
        return str;
    }
}

class AircraftCarrier extends Warship {
    private ArrayList <Airplane> airplanes;
    @Override
    public boolean store(Object obj) {
        airplanes = new ArrayList<Airplane>();
        airplanes.add((Airplane) obj);
        return true;
    }
    public ArrayList<Airplane> getAirplanes() {
        return airplanes;
    }

}

class Airplane {
    private final int num;
    private final String model;
    Airplane (int num, String model) {
        this.num = num;
        this.model = model;
    }

    public int getNum() {
        return num;
    }

    public String getModel() {
        return model;
```

```
    }
}
```

Код для решения задания 2.7:

```
package bdjava.lab4;

import java.util.PrimitiveIterator;

import static java.lang.System.out;

public class Program4 {
    public static void main(String[] args) {

    }
}

interface Doctor {
    public String examine (String complaints, Patient p);
    public boolean treat (String examination, Patient p);
    public String writePrescription (String examination, Patient p);
}

abstract class Surgeon implements Doctor {
    public boolean cut (String examination, Patient p) {
        return true;
    }
    public boolean sewUp (String examination, Patient p) {
        return true;
    }
    @Override
    public String writePrescription(String examination, Patient p) {
        return "Выписанный рецепт";
    }
}

class Neurosurgeon extends Surgeon {
```

```
    @Override
    public boolean treat(String examination, Patient p) {
        out.println("Лечим со знаниями нейрохирургии");
        return true;
    }
    @Override
    public String examine(String complaints, Patient p) {
        return "Обследуем со знаниями нейрохирургии";
    }
}

class Patient {

}
```

Вывод:

Были созданы классы "Shop" и "Справочная Служба Общественного Транспорта" с внутренними классами для хранения информации. Также были реализованы абстрактные классы и интерфейсы для классов "Корабль", "Военный Корабль", "Авианосец", "Врач", "Хирург" и "Нейрохирург" с использованием наследования и полиморфизма.