Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

По дисциплине «Современные платформы программирования»

Выполнила:

Студентка 3 курса

Группы ПО-3

Пивчик В.Г.

Проверил:

Крощенко А.А.

Брест 2020 г.

Цель работы:

Приобрести практические навыки в области объектно - ориентированного проектирования.

Вариант 9

Постановка задачи:

Задание 1:

Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

Создать **класс Mobile** с внутренним классом, с помощью объектов которого можно хранить информацию о моделях телефонов и их свойствах.

Задание 2:

Реализовать агрегирование. При создании класса агрегируемый класс объявляется как атрибут (локальная переменная, параметр метода). Включить в каждый класс 2-3 метода на выбор. Продемонстрировать использование разработанных классов.

Создать **класс Автомобиль**, используя класс Колесо.

Задание 3:

Построить модель программной системы с применением отношений (обобщения, агрегации, ассоциации, реализации) между классами. Задать атрибуты и методы классов. Реализовать (если необходимо) дополнительные классы. Продемонстрировать работу разработанной 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.

Ход работы

Текст программы:

Задание 1

Код программы

Mobile

```
package com.company;

public class Mobile {
    private String model;
    private Parameters parameters;
    public Mobile() {
    }

    public Mobile(String model) {
        this.model = model;
    }

    public void getOperatingSystem() {
        if (this.model.toLowerCase().contains("iphone")) {
System.out.println("This device works on iOS");
        } else {
            System.out.println("This device works on Android");
        }
    }
    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public Parameters getParameters() {
        return parameters;
    }

    public void setParameters(Parameters parameters) {
        this.parameters = parameters;
    }


    public class Parameters {
        private String screen;
        private Integer memory;
        private Integer megapixels;
        private Integer ram;
        private Integer power;
        public void getAllParameters()    {
            if (this.screen != null &&    !this.screen.isEmpty()) {
                System.out.println("Screen resolution: " + this.screen + "
px");
            }
            if (this.memory != null) {
                System.out.println("Memory space: " + this.memory + " GB");
```

```
            }
            if (this.megapixels != null) {
                System.out.println("Camera: " + this.megapixels + " MP");
            }
            if (this.ram != null) {
                System.out.println("RAM: " + this.ram + " GB");
            }
            if (this.power != null) {
                System.out.println("Battery power: " + this.power + "
MaH");
            }
        }

        public String getScreen() {
            return screen;
        }

        public void setScreen(String screen) {
            this.screen = screen;
        }

        public Integer getMemory() {
            return memory;
        }

        public void setMemory(Integer memory) {
            this.memory = memory;
        }

        public Integer getMegapixels() {
            return megapixels;
        }

        public void setMegapixels(Integer megapixels) {
            this.megapixels = megapixels;
        }

        public Integer getRam() {
            return ram;
        }

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

        public Integer getPower() {
            return power;
        }

        public void setPower(Integer power) {
            this.power = power;
        }
    }
}
```

Main

```
public class Main {

    public static void main(String[] args) {
        // mobile one
        Mobile iphone = new Mobile();
```

```
        Mobile.Parameters iphoneParameters = iphone.new Parameters();
        iphone.setModel("Iphone 7");
        System.out.println(iphone.getModel());
        iphone.setParameters(iphoneParameters);
iphoneParameters.setMegapixels(12); iphoneParameters.setRam(2);
iphoneParameters.setMemory(128);
        iphone.getOperatingSystem();
        iphoneParameters.getAllParameters();
        // mobile two
        Mobile samsung = new Mobile("Samsung S10+");
        Mobile.Parameters samsungParameters = samsung.new Parameters();
        System.out.println(samsung.getModel());
        samsung.setParameters(samsungParameters);
        samsungParameters.setMegapixels(16)
        ; samsungParameters.setRam(4);
        samsungParameters.setMemory(64);
        samsungParameters.setPower(2000);
        samsungParameters.setScreen("2000x4000");
        samsung.getOperatingSystem();
        samsungParameters.getAllParameters();
    }
}
```

Рисунок с результатом работы программы

```
Iphone 7
This device works on iOS
Memory space: 128 GB
Camera: 12 MP
RAM: 2 GB
Samsung S10+
This device works on Android
Screen resolution: 2000x4000 px
Memory space: 64 GB
Camera: 16 MP
RAM: 4 GB
Battery power: 2000 MaH
```

Задание 2

Код программы

Car

```
package com.company;

import java.util.ArrayList;

public class Car {
    private String model;
    private Integer year;
    private ArrayList<Wheel> wheels;

    public Car(String model, Integer year) { this.model = model;
        this.year = year;
        this.wheels = new ArrayList<>();
    }
```

```
    public void ride() {
        if(isCorrectlyMade()){
            System.out.println("Ready to go!");
        } else {
            System.out.println("Something went wrong");
        }
    }

    public boolean isCorrectlyMade() {
        if (this.wheels == null || this.wheels.size() != 4) { return false;
        }
        boolean isCorrectlyMade = true;
        for (int i = 0; i < this.wheels.size() && isCorrectlyMade; i++) {
            for (int j = 0; j < this.wheels.size() && isCorrectlyMade; j++)
{
                if (!wheels.get(i).equals(wheels.get(j))) { isCorrectlyMade
= false;
                }
            }
        }
        return isCorrectlyMade;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }

    public Integer getYear() {
        return year;
    }

    public void setYear(Integer year) {
        this.year = year;
    }

    public ArrayList<Wheel> getWheels() {
        return wheels;
    }

    public void addWheel(Integer wheelSize, String seasonType) {
        this.wheels.add(new Wheel(wheelSize, seasonType));
    }

    public class Wheel {
        private Integer wheelSize;
        private String seasonType;

        public Wheel(Integer wheelSize, String seasonType) {
            this.wheelSize = wheelSize;
            this.seasonType = seasonType;
        }

        public Integer getWheelSize() {
            return wheelSize;
        }

        public void setWheelSize(Integer wheelSize) {
            this.wheelSize = wheelSize;
        }
```

```
        public String getSeasonType() {
            return seasonType;
        }

        public void setSeasonType(String seasonType) {
            this.seasonType = seasonType;
        }

        @Override
        public boolean equals(Object o) {
            if (this == o) return true;
            if (o == null || getClass() != o.getClass()) return false;
Wheel wheel = (Wheel) o;
            return wheelSize.equals(wheel.wheelSize) &&
seasonType.equals(wheel.seasonType);
        }
    }
}
```

Main

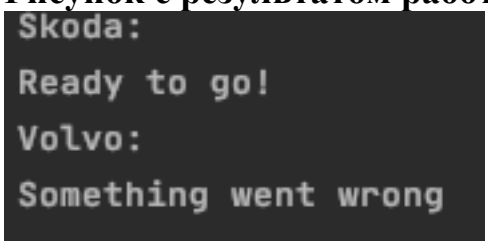
```
package com.company;

public class Main {

    public static void main(String[] args) {
        // car one
        Car skoda = new Car("Skoda", 2018);
        for (int i = 0; i < 4; i++) {
            skoda.addWheel(55, "Summer");
        }
        System.out.println("Skoda:");
        skoda.ride();

        // car two
        Car volvo = new Car("Volvo", 1999);
        for (int i = 0; i < 3; i++) {
            volvo.addWheel(55, "Winter");
        }
        volvo.addWheel(57, "Winter");
        System.out.println("Volvo:");
        volvo.ride();
    }
}
```

Рисунок с результатом работы программы



- **Задание 3**
  Код программы

Main

```
package com.company;
import java.util.ArrayList;
import java.util.Arrays;

public class Main {
    public static void main(String[] args) {
        RailwayCash railwayCash = new RailwayCash();
        Admin admin = new Admin(railwayCash);
        admin.addTrain(
                "2019-10-21 14:00",
                701,
                500,
                new ArrayList<String>(Arrays.asList("Жабинка", "Берёза",
"Барановичи", "Минск")),
                13.50f);
        admin.addTrain(
                "2019-10-21 14:00",
                703,
                500,
                new ArrayList<String>(Arrays.asList("Барановичи",
"Минск")),
                15.50f);
        Passenger passenger = new Passenger();
        Request request = passenger.createRequest("Барановичи", "2019-10-21
14:00");
        Bill bill =
passenger.chooseTrain(railwayCash.findTrainsByRequest(request));
        System.out.println(bill);
    }
}
```

Admin

```
package com.company;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;

public class Admin {
    private RailwayCash railwayCash;

    public Admin(RailwayCash railwayCash) {
        this.railwayCash = railwayCash;
    }

    public void addTrain(
            String dayAndTime,
            Integer number,
            Integer seatsAmount,
            ArrayList<String> stations,
            Float pricePerSeat
    ) {
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-
dd HH:mm");
        Train train = new Train(
                LocalDateTime.parse(dayAndTime, formatter),
```

```
                number,
                seatsAmount,
                stations,
                pricePerSeat
        );
        railwayCash.addTrains(train);
    }
}
```

Bill

```
package com.company;

public class Bill {
    private Float price;
    private Integer seatNumber;

    public Float getPrice() {
        return price;
    }

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

    public Integer getSeatNumber() {
        return seatNumber;
    }

    public void setSeatNumber(Integer seatNumber) {
        this.seatNumber = seatNumber;
    }

    @Override
    public String toString() {
        return "Bill{" + "price=" + price + ", seatNumber=" + seatNumber +
'}';
    }
}
```

Passenger

```
package com.company;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Scanner;

public class Passenger {

    public Request createRequest(String destination, String date) {
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-
dd HH:mm");
        Request request = new Request();
        request.setDayAndTime(LocalDateTime.parse(date, formatter));
        request.setDestination(destination);
        return request;
    }

    public Bill chooseTrain(ArrayList<Train> trains) {
        Scanner scanner = new Scanner(System.in);
```

```
        for (Train train : trains) {
            System.out.println(train);
        }
        System.out.println("Choose a train number: ");
        Integer chosenNumber = scanner.nextInt();
        boolean trainIsNotFound = true;
        Train chosenTrain = null;
        for    (int i = 0; i < trains.size() && trainIsNotFound; i++) {
            if (trains.get(i).getNumber().equals(chosenNumber)) {
                chosenTrain = trains.get(i);
                chosenTrain.reserveSeat();
                trainIsNotFound = false;
            }
        }
        if (chosenTrain != null) {
            Bill bill = new Bill();
            bill.setPrice(chosenTrain.getPricePerSeat());
            bill.setSeatNumber(chosenTrain.getOccupiedSeatsAmount());
            return bill;
        } else {
            throw new RuntimeException("Train is not found!");
        }
    }
```

RailwayCash

```
package com.company;
import java.util.ArrayList;
import java.util.Collections;

public class RailwayCash {
    private ArrayList<Train> trainsList;

    public RailwayCash() {
        this.trainsList = new ArrayList<>();
    }

    public void addTrains(Train... trains) {
        Collections.addAll(trainsList, trains);
    }

    public ArrayList<Train> findTrainsByRequest(Request request) {
        ArrayList<Train> suitableTrains = new ArrayList<>();
        for (Train train : trainsList) {
            if (train.getDayAndTime().isEqual(request.getDayAndTime())
                    &&
train.getStations().contains(request.getDestination())
                    && train.hasFreeSeats()) {
                suitableTrains.add(train);
            }
        }
        return suitableTrains;
    }
}
```

Request

```
package com.company;
import java.time.LocalDateTime;

public class Request {
```

```
    private String destination;
    private LocalDateTime dayAndTime;

    public String getDestination() {
        return destination;
    }

    public void setDestination(String destination) {
        this.destination = destination;
    }

    public LocalDateTime getDayAndTime() {
        return dayAndTime;
    }

    public void setDayAndTime(LocalDateTime dayAndTime) {
        this.dayAndTime = dayAndTime;
    }
}
```

Train

```
package com.company;


import java.time.LocalDateTime;

import java.util.ArrayList;


public class Train {
    private LocalDateTime dayAndTime;
    private Integer number;
    final private Integer seatsAmount;
    private Integer occupiedSeatsAmount;
    private final ArrayList<String> stations;
    private Float pricePerSeat;

    public Train(
            LocalDateTime dayAndTime,
            Integer number,
            Integer seatsAmount,
            ArrayList<String> stations,
            Float pricePerSeat
            ) {
        this.dayAndTime = dayAndTime;
        this.number = number;
        this.seatsAmount = seatsAmount;
        this.occupiedSeatsAmount = 0;
        this.stations = stations;
        this.pricePerSeat = pricePerSeat;
    }

    public Float getPricePerSeat() {
        return pricePerSeat;
    }

    public void setPricePerSeat(Float pricePerSeat) {
        this.pricePerSeat = pricePerSeat;
    }

    public LocalDateTime getDayAndTime() {
        return dayAndTime;
    }
```

```
    public void setDayAndTime(LocalDateTime dayAndTime) {
        this.dayAndTime = dayAndTime;
    }

    public Integer getNumber() {
        return number;
    }

    public void setNumber(Integer number) {
        this.number = number;
    }

    public ArrayList<String> getStations() {
        return stations;
    }

    public Integer getSeatsAmount() {
        return seatsAmount;
    }

    public Integer getOccupiedSeatsAmount() {
        return occupiedSeatsAmount;
    }

    public void reserveSeat(){
        this.occupiedSeatsAmount++;
    }

    public boolean hasFreeSeats() {
        return occupiedSeatsAmount < seatsAmount;
    }

    @Override
    public String toString() {
        return "Train{" +
                "dayAndTime=" + dayAndTime +
                ", number=" + number +
                ", seatsAmount=" + seatsAmount +
                ", occupiedSeatsAmount=" + occupiedSeatsAmount + ",
pricePerSeat=" + pricePerSeat + '}';
    }
}
```

Рисунок с результатом работы программы

```
Train{dayAndTime=2019-10-21T14:00, number=701, seatsAmount=500, occupiedSeatsAmount=0, pricePerSeat=13.5}
Train{dayAndTime=2019-10-21T14:00, number=703, seatsAmount=500, occupiedSeatsAmount=0, pricePerSeat=15.5}
Choose a train number:
701
Bill{price=13.5, seatNumber=1}
```

Выводы:

Я приобрела практические навыки в области объектно-ориентированного проектирования.