МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

КАФЕДРА ИИТ

Отчет по лабораторной работе №4

Выполнил:

Студент 4 курса

Группы АС-50

Куц Д.А.

Проверил:

Крощенко А.А.

Брест 2020

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

Задание 1. Реализовать указанный класс, включив в него вспомогательный внутренний класс или классы. Реализовать 2-3 метода (на выбор). Продемонстрировать использование реализованных классов.

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

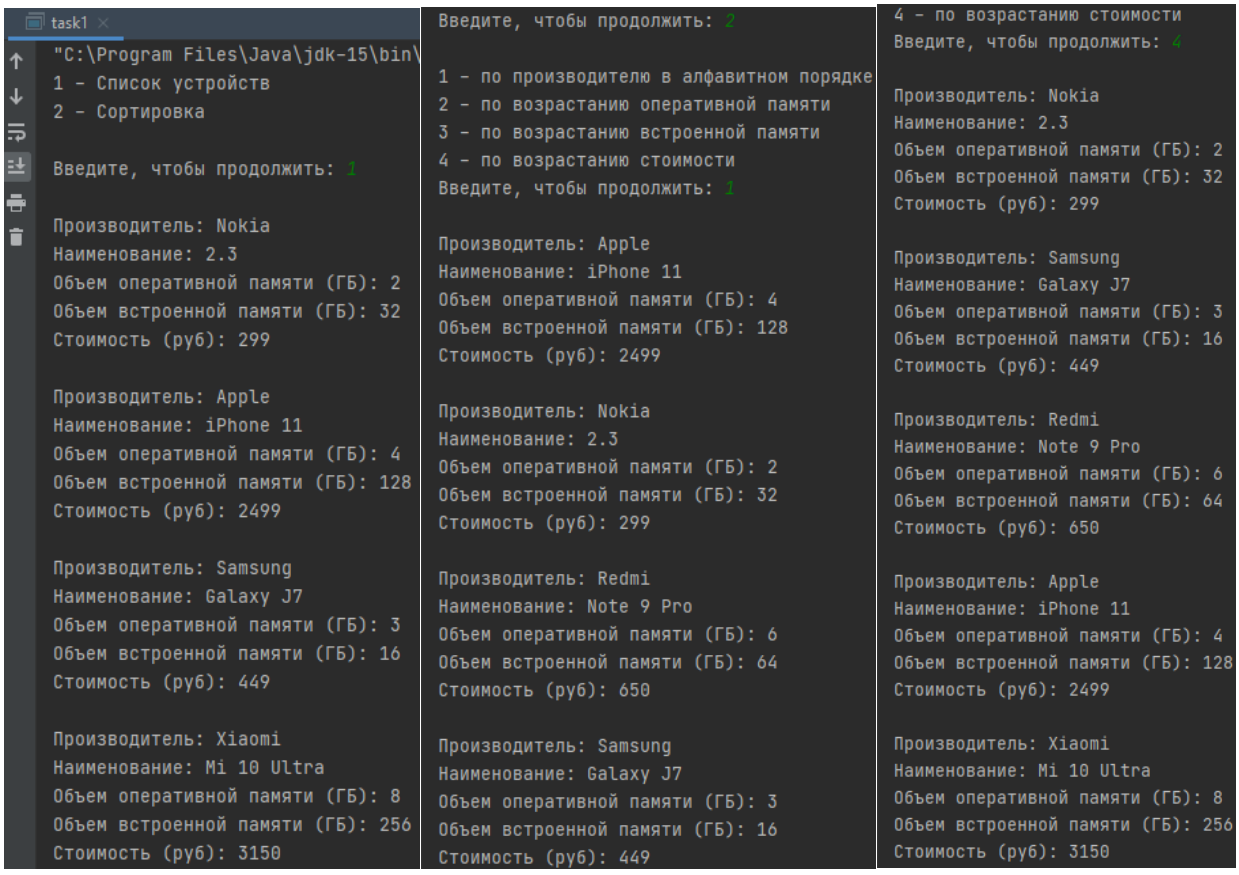
Код:

```
import java.util.*;

class Mobile {

    static class MobileSpec {

        private final int id, RAM, ROM, price;
        private final String manufacturer, name;

        public MobileSpec(int id, String manufacturer, String name, int RAM,
int ROM, int price) {
            this.id = id;
            this.manufacturer = manufacturer;
            this.name = name;
            this.RAM = RAM;
            this.ROM = ROM;
            this.price = price;
        }

        public int getId() { return id; }

        public String getManufacturer() { return manufacturer; }

        public int getRAM() { return RAM; }

        public int getROM() { return ROM; }

        public int getPrice() { return price; }

        public void printElement() {
            System.out.println("\nПроизводитель: " + manufacturer +
"\nНаименование: " + name
                    + "\nОбъем оперативной памяти (ГБ): " + RAM + "\nОбъем
встроенной памяти (ГБ): " + ROM
                    + "\nСтоимость (руб): " + price);
        }
    }

    private final ArrayList<MobileSpec> phoneList = new ArrayList<>();

    public Mobile() {
        phoneList.add(new MobileSpec(1,"Nokia","2.3", 2, 32, 299));
        phoneList.add(new MobileSpec(2,"Apple","iPhone 11", 4, 128, 2499));
        phoneList.add(new MobileSpec(3,"Samsung","Galaxy J7", 3, 16, 449));
        phoneList.add(new MobileSpec(4,"Xiaomi","Mi 10 Ultra", 8, 256,
3150));
        phoneList.add(new MobileSpec(5,"Redmi","Note 9 Pro", 6, 64, 650));
    }

    public void printMobileList() { for (MobileSpec mobileSpec : phoneList)
```

```
mobileSpec.printElement(); }

    private final Map<Integer, Integer> RAMMap = new HashMap<>();
    private final Map<Integer, Integer> ROMMap = new HashMap<>();
    private final Map<Integer, Integer> priceMap = new HashMap<>();
    private final Map<String, Integer> manufacturerMap = new HashMap<>();

    public void sortMobileList() {

        System.out.println("\n1 - по производителю в алфавитном порядке");
        System.out.println("2 - по возрастанию оперативной памяти");
        System.out.println("3 - по возрастанию встроенной памяти");
        System.out.println("4 - по возрастанию стоимости");

        Scanner scanner = new Scanner(System.in);
        System.out.print("Введите, чтобы продолжить: ");
        int menu = scanner.nextInt();
        switch (menu) {
            case 1 -> {
                for (MobileSpec spec : phoneList)
manufacturerMap.put(spec.getManufacturer(), spec.getId());
                Map<String, Integer> sortManufacturer = new
TreeMap<>(manufacturerMap);
                for (Map.Entry<String, Integer> entry :
sortManufacturer.entrySet())
                    for (MobileSpec spec : phoneList)
                        if (entry.getValue() == spec.getId())
spec.printElement();
            }
            case 2 -> {
                for (MobileSpec spec : phoneList) RAMMap.put(spec.getRAM(),
spec.getId());
                Map<Integer, Integer> sortRAM = new TreeMap<>(RAMMap);
                for (Map.Entry<Integer, Integer> entry : sortRAM.entrySet())
                    for (MobileSpec spec : phoneList)
                        if (entry.getValue() == spec.getId())
spec.printElement();
            }
            case 3 -> {
                for (MobileSpec spec : phoneList) ROMMap.put(spec.getROM(),
spec.getId());
                Map<Integer, Integer> sortROM = new TreeMap<>(ROMMap);
                for (Map.Entry<Integer, Integer> entry : sortROM.entrySet())
                    for (MobileSpec spec : phoneList)
                        if (entry.getValue() == spec.getId())
spec.printElement();
            }
            case 4 -> {
                for (MobileSpec spec : phoneList)
priceMap.put(spec.getPrice(), spec.getId());
                Map<Integer, Integer> sortPrice = new TreeMap<>(priceMap);
                for (Map.Entry<Integer, Integer> entry :
sortPrice.entrySet())
                    for (MobileSpec spec : phoneList)
                        if (entry.getValue() == spec.getId())
spec.printElement();
            }
        }
    }
}

public class task1 {

    public static void main(String[] args) {
```

```
        Mobile mobile = new Mobile();
        System.out.println("1 - Список устройств");
        System.out.println("2 - Сортировка\n");

        Scanner scanner = new Scanner(System.in);
        System.out.print("Введите, чтобы продолжить: ");
        int menu = scanner.nextInt();
        switch (menu) {
            case 1: mobile.printMobileList(); break;
            case 2: mobile.sortMobileList(); break;
            default: break;
        }
    }
}
```

Результат:



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.

9) Создать класс Автомобиль, используя класс Колесо.

Код:

```
import java.util.Scanner;

class Wheel {
    int rotationSpeed = 0;
```

```
    public int getRotationSpeed() { return rotationSpeed; }

    public void setRotationSpeed(int rotationSpeed) { this.rotationSpeed =
rotationSpeed; }
}

enum carList {Renault, BMW, Lamborghini}

class Car {
    Wheel wheel = new Wheel();
    String selectCar;
    boolean isEngineStarted = false;
    int turnGear;

    public Car(String selectCar) { this.selectCar = selectCar; }

    public void setEngineStarted(boolean engineStarted) { isEngineStarted =
engineStarted; }

    public void startEngine() { setEngineStarted(true); }

    public void stopEngine() { setEngineStarted(false); }

    public void turnOnGear(int gear) { turnGear = gear; }

    public void drive() {
        if (!isEngineStarted) System.out.println("Двигатель выключен");
        else {
            switch (turnGear) {
                case 0 -> System.out.println("Двигатель запущен \nНейтральная
передача. Скорость 0 км/ч");
                case 1 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(20);
                        case "BMW" -> wheel.setRotationSpeed(30);
                        case "Lamborghini" -> wheel.setRotationSpeed(40);
                    }
                    System.out.println("Первая передача. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
                case 2 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(40);
                        case "BMW" -> wheel.setRotationSpeed(60);
                        case "Lamborghini" -> wheel.setRotationSpeed(80);
                    }
                    System.out.println("Вторая передача. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
                case 3 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(60);
                        case "BMW" -> wheel.setRotationSpeed(90);
                        case "Lamborghini" -> wheel.setRotationSpeed(120);
                    }
                    System.out.println("Третья передача. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
                case 4 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(90);
                        case "BMW" -> wheel.setRotationSpeed(150);
                        case "Lamborghini" -> wheel.setRotationSpeed(200);
```

```
                    }
                    System.out.println("Четвертая передача. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
                case 5 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(140);
                        case "BMW" -> wheel.setRotationSpeed(200);
                        case "Lamborghini" -> wheel.setRotationSpeed(300);
                    }
                    System.out.println("Пятая передача. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
                case -1 -> {
                    switch (selectCar) {
                        case "Renault" -> wheel.setRotationSpeed(20);
                        case "BMW" -> wheel.setRotationSpeed(40);
                        case "Lamborghini" -> wheel.setRotationSpeed(50);
                    }
                    System.out.println("Задний ход. Скорость " +
wheel.getRotationSpeed() + " км/ч");
                }
            }
        }
    }

    public void TestDrive() {
        drive();
        startEngine();
        turnOnGear(0); drive();
        turnOnGear(-1); drive();
        turnOnGear(1); drive();
        turnOnGear(2); drive();
        turnOnGear(3); drive();
        turnOnGear(4); drive();
        turnOnGear(5); drive();
        stopEngine(); drive();
    }
}

public class task2 {
    public static void main(String[] args) {
        Car car1 = new Car(carList.Renault.toString());
        Car car2 = new Car(carList.BMW.toString());
        Car car3 = new Car(carList.Lamborghini.toString());

        System.out.println("1 - Renault Clio");
        System.out.println("2 - BMW M3 E46");
        System.out.println("3 - Lamborghini Huracan Evo");

        Scanner scanner = new Scanner(System.in);
        System.out.print("Выберите автомобиль: ");
        int menu = scanner.nextInt();
        System.out.println();
        switch (menu) {
            case 1 -> car1.TestDrive();
            case 2 -> car2.TestDrive();
            case 3 -> car3.TestDrive();
        }
    }
}
```

Результат:

```
"C:\Program Files\Java\jdk-15\bin\jav
1 - Renault Clio
2 - BMW M3 E46
3 - Lamborghini Huracan Evo
Выберите автомобиль: 1

Двигатель выключен
Двигатель запущен
Нейтральная передача. Скорость 0 км/ч
Задний ход. Скорость 20 км/ч
Первая передача. Скорость 20 км/ч
Вторая передача. Скорость 40 км/ч
Третья передача. Скорость 60 км/ч
Четвертая передача. Скорость 90 км/ч
Пятая передача. Скорость 140 км/ч
Двигатель выключен

Process finished with exit code 0
```

```
Выберите автомобиль: 2

Двигатель выключен
Двигатель запущен
Нейтральная передача. Скорость 0 км/ч
Задний ход. Скорость 40 км/ч
Первая передача. Скорость 30 км/ч
Вторая передача. Скорость 60 км/ч
Третья передача. Скорость 90 км/ч
Четвертая передача. Скорость 150 км/ч
Пятая передача. Скорость 200 км/ч
Двигатель выключен

Process finished with exit code 0
```

```
Выберите автомобиль: 3

Двигатель выключен
Двигатель запущен
Нейтральная передача. Скорость 0 км/ч
Задний ход. Скорость 50 км/ч
Первая передача. Скорость 40 км/ч
Вторая передача. Скорость 80 км/ч
Третья передача. Скорость 120 км/ч
Четвертая передача. Скорость 200 км/ч
Пятая передача. Скорость 300 км/ч
Двигатель выключен

Process finished with exit code 0
```

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.

9) 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:

```
import java.util.ArrayList;
import java.util.Scanner;

class Administrator {
    ArrayList<Train> trainList = new ArrayList<>();
```

```
    String departurePoint, departureTime, departureDate, destinationPoint,
destinationTime;
    ArrayList<String> stationList, stationTimeList;

    public void setDeparturePoint(String departurePoint) {
this.departurePoint = departurePoint; }

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

    public void setDepartureDate(String departureDate) { this.departureDate =
departureDate; }

    public void setDestinationTime(String destinationTime) {
this.destinationTime = destinationTime; }

    public void setDestinationPoint(String destinationPoint) {
this.destinationPoint = destinationPoint; }

    public void setStationList(ArrayList<String> stationList) {
this.stationList = stationList; }

    public void setStationTimeList(ArrayList<String> stationTimeList) {
this.stationTimeList = stationTimeList; }

    public void createTrain() {
        trainList.add(new Train(destinationPoint, destinationTime,
departurePoint, departureTime, departureDate,
                stationList, stationTimeList));
    }

    void printTrain(String date, String point) {
        int i = 0;
        for (Train train : trainList) {
            if(train.departureDate.equals(date))
                if (train.destinationPoint.equals(point))
System.out.println("Поезд №" + i + ", Отбытие: "
                        + train.departurePoint + " " + train.departureTime +
" " + train.departureDate + ", Прибытие: "
                        + train.destinationPoint + " " +
train.destinationTime + " " + train.departureDate
                        + "\nМаршрут: " + train.stationList + ", Время
прибытия по станциям: " + train.stationTimeList);
            i++;
        }
    }
}

class Passenger {
    ArrayList<Request> requestsList = new ArrayList<>();
    String date, point, time, destination;

    public void setDate(String date) { this.date = date; }

    public void setPoint(String point) { this.point = point; }

    public void setTime(String time) { this.time = time; }

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

    public void createRequest() { requestsList.add(new Request(date, point,
time, destination)); }
```

```
    void printRequest() {
        int i = 0;
        for (Request request: requestsList) {
            System.out.println("Заявка №" + i + ", Дата: " +
request.departureDate + ", Время: " + request.departureTime
                    + ", Пункт отбытия: " + request.departurePoint + ", Пункт
назначения: " + request.destinationPoint);
            i++;
        }
    }
}

class Train {
    String destinationPoint, destinationTime, departurePoint, departureTime,
departureDate;
    ArrayList<String> stationList, stationTimeList;

    public Train(String destinationPoint, String destinationTime, String
departurePoint, String departureTime,
                 String departureDate, ArrayList<String> stationList,
ArrayList<String> stationTimeList) {
        this.destinationPoint = destinationPoint;
        this.destinationTime = destinationTime;
        this.departurePoint = departurePoint;
        this.departureTime = departureTime;
        this.departureDate = departureDate;
        this.stationList = stationList;
        this.stationTimeList = stationTimeList;
    }
}

class Request {
    String departureDate, departurePoint, departureTime, destinationPoint;

    public Request(String departureDate, String departurePoint, String
departureTime, String destinationPoint) {
        this.departureDate = departureDate;
        this.departurePoint = departurePoint;
        this.departureTime = departureTime;
        this.destinationPoint = destinationPoint;
    }
}

public class task3 {
    public static void main(String[] args) {
        String date, point;
        Passenger passenger = new Passenger();
        passenger.setDestination("Минск");
        passenger.setDate("11.12.2020");
        passenger.setPoint("Брест");
        passenger.setTime("06:00");
        passenger.createRequest();

        passenger.setDestination("Брест");
        passenger.setDate("13.12.2020");
        passenger.setPoint("Минск");
        passenger.setTime("19:30");
        passenger.createRequest();

        passenger.printRequest();

        System.out.print("Введите номер заявки: ");
        Scanner scanner = new Scanner(System.in);
        int requestNumber = scanner.nextInt();
```

```
        System.out.println("");

        date = passenger.requestsList.get(requestNumber).departureDate;
        point = passenger.requestsList.get(requestNumber).destinationPoint;

        Administrator administrator = new Administrator();
        administrator.setDeparturePoint("Брест");
        administrator.setDepartureDate("11.12.2020");
        administrator.setDepartureTime("06:21");
        administrator.setDestinationPoint("Минск");
        administrator.setDestinationTime("09:23");
        ArrayList<String> stations1 = new ArrayList<>();
        stations1.add("Жабинка");
        stations1.add("Береза");
        stations1.add("Барановичи");
        administrator.setStationList(stations1);
        ArrayList<String> stationsTimes = new ArrayList<>();
        stationsTimes.add("06:44");
        stationsTimes.add("07:28");
        stationsTimes.add("08:30");
        administrator.setStationTimeList(stationsTimes);
        administrator.createTrain();

        administrator.setDeparturePoint("Минск");
        administrator.setDepartureDate("13.12.2020");
        administrator.setDepartureTime("19:47");
        administrator.setDestinationPoint("Брест");
        administrator.setDestinationTime("23:02");
        ArrayList<String> stations2 = new ArrayList<>();
        stations2.add("Барановичи");
        stations2.add("Ивацевичи");
        stations2.add("Жабинка");
        administrator.setStationList(stations2);
        ArrayList<String> stationsTimes2 = new ArrayList<>();
        stationsTimes2.add("21:13");
        stationsTimes2.add("21:45");
        stationsTimes2.add("22:43");
        administrator.setStationTimeList(stationsTimes2);
        administrator.createTrain();

        administrator.printTrain(date, point);

        Scanner railwayNumber = new Scanner(System.in);
        System.out.print("Введите номер поезда: ");
        int r_number = railwayNumber.nextInt();
        int checkNumber = (int) (Math.random() * 200000) + 100000;
        int seat = (int) (Math.random() * 200) + 10;
        int ticketCost = (int) (Math.random() * 5) + 20;
        System.out.println("\nСчет №" + checkNumber);
        System.out.println("Оплата маршрута №" + r_number);
        System.out.println("Место №" + seat);
        System.out.println("Стоимость билета: " + ticketCost);
        System.out.println("Приятной поездки!");
    }
}
```

Результат:

```
"C:\Program Files\Java\jdk-15\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ
Заявка №0, Дата: 11.12.2020, Время: 06:00, Пункт отбытия: Брест, Пункт назначения: Минск
Заявка №1, Дата: 13.12.2020, Время: 19:30, Пункт отбытия: Минск, Пункт назначения: Брест
Введите номер заявки: 1

Поезд №1, Отбытие: Минск 19:47 13.12.2020, Прибытие: Брест 23:02 13.12.2020
Маршрут: [Барановичи, Ивацевичи, Жабинка], Время прибытия по станциям: [21:13, 21:45, 22:43]
Введите номер поезда: 1

Счет №206310
Оплата маршрута №1
Место №109
Стоимость билета: 23
Приятной поездки!

Process finished with exit code 0
```

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