**Министерство образования Республики Беларусь**
**Учреждение образования**
**«Брестский государственный технический университет»**
**Кафедра ИИТ**

**Лабораторная работа №3**
По дисциплине: «ССП»
Вариант - 12

**Выполнил:**
Студент 3 курса
Группы ПО-8
Иванюк М.С.
**Проверил:**
Крощенко А.А

**Брест, 2024**

**Лабораторная работа №3**

**Цель работы:** научится создавать и использовать классы в программах на языке программирования Java.

**Задание 1:**

Реализовать простой класс. Требования к выполнению
• Реализовать пользовательский класс по варианту.
• Создать другой класс с методом main, в котором будут находится примеры использования пользовательского класса. Для каждого класса
• Создать поля классов
• Создать методы классов
• Добавьте необходимые get и set методы (по необходимости)
• Укажите соответствующие модификаторы видимости
• Добавьте конструкторы • Переопределить методы toString() и equals()

**Равносторонний треугольник, заданный длинами сторон** – Предусмотреть возможность определения площади и периметра, а так же логический метод, определяющий существует или такой треугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа.

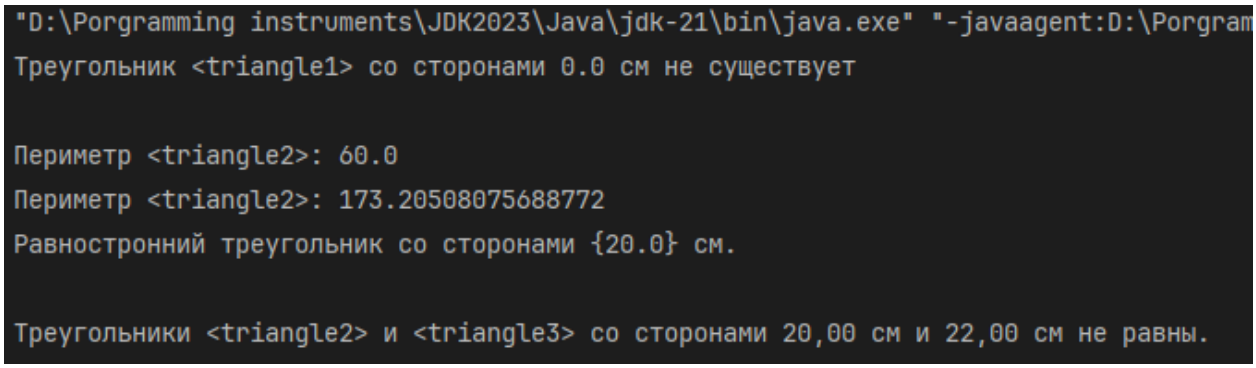
**Код программы:**

```
package Lab3;

import java.util.Scanner;

public class task1 {

    public static void main(String[] args){
        double sideLength = 30.0;
        EquilateralTriangle triangle1 = new EquilateralTriangle();
        if(triangle1.isEquilateralTriangleExist()){
            System.out.println("Периметр <triangle1>: " +
triangle1.calculatePerimeter());
            System.out.println("Площадь <triangle1>: " +
triangle1.calculateArea());
            System.out.println(triangle1.toString());
        }
        else{
            System.out.println("Треугольник <triangle1> со сторонами " +
triangle1.getSideLength() +" см не существует");
        }

        EquilateralTriangle triangle2 = new EquilateralTriangle(20.00);
        if(triangle2.isEquilateralTriangleExist()){
            System.out.println("\nПериметр <triangle2>: " +
triangle2.calculatePerimeter());
            System.out.println("Периметр <triangle2>: " +
triangle2.calculateArea());
            System.out.println(triangle2);
        }
        else{
            System.out.println("Треугольник <triangle2> со стронами " +
```

```
triangle2.getSideLength() +" см не существует");
        }

        EquilateralTriangle triangle3 = new EquilateralTriangle(22.00);

        if(triangle2.equals(triangle3)){
            System.out.printf("\nТреугольники <triangle2> и <triangle3> со
сторонами %.2f см и %.2f см
равны.",triangle2.getSideLength(),triangle3.getSideLength());
        }
        else{
            System.out.printf("\nТреугольники <triangle2> и <triangle3> со
сторонами %.2f см и %.2f см не
равны.",triangle2.getSideLength(),triangle3.getSideLength());
        }
    }
}

 class EquilateralTriangle{
    private double sidesLength;

    public EquilateralTriangle(){
        this.sidesLength=0.0;
    }

    public EquilateralTriangle(double sideLength){
        this.sidesLength=sideLength;
    }

    public void setSideLength(double sideLength){
        this.sidesLength=sideLength;
    }

    public double getSideLength(){
        return this.sidesLength;
    }

    public double calculatePerimeter(){
        return this.sidesLength*3;
    }

    public double calculateArea(){
        return (Math.pow(this.sidesLength,2)*Math.sqrt(3))/4;
    }

    public boolean isEquilateralTriangleExist(){
        return this.sidesLength>0;
    }

    @Override
    public String toString() {
        return "Равносторонний треугольник со сторонами {"+
                this.sidesLength + "} см.";
    }

    @Override
    public boolean equals(Object otherObject){
        if(this == null){
           return false;
        } else if(this == otherObject){
            return true;
        } else {
            EquilateralTriangle otherEquilateralTriangle =
(EquilateralTriangle) otherObject;
```

```
            return this.getSideLength() ==
otherEquilateralTriangle.getSideLength();
        }
    }
}
```

**Результат работы программы:**




**Задание 2:**

**Автоматизированная система проката автомобилей.**

Составить программу, которая хранит и обрабатывает информацию о прокате автомобилей. О каждом автомобиле (Car) содержится следующая информация:

• id;
• Марка;
• Модель;
• Год выпуска;
• Цвет;
• Цена;
• Регистрационный номер;
• Номер машины.
• ФИО лица, взявшего на прокат (при наличии);
• Номер паспорта лица-арендатора (при наличии).

Программа должна обеспечить вывод списков:

• автомобилей;
• автомобилей заданной марки;
• автомобилей заданной модели, которые эксплуатируются больше n лет;
• автомобилей заданного года выпуска, цена которых больше указанной;
• автомобилей, взятых на прокат;
• автомобилей, взятых на прокат с выводом личной информации об арендаторах.

**Код программы:**

**Task2.java:**

```java
package Lab3;

import java.util.Scanner;

public class task2 {
    public static void main(String[] args) {
        final String FILE_PATH = "C:/Users/proro/OneDrive/Рабочий
стол/CarInfo.txt";

        CarRent carRentalSystem = new CarRent(FILE_PATH);

        while(true){
            int choice;
            System.out.println("(1) Вывод списка всех автомобилей.");
            System.out.println("(2) Вывод списка всех автомобилей заданной
марки.");
            System.out.println("(3) Вывод списка автомобилей заданной модели,
которые эксплуатируются больше n лет.");
            System.out.println("(4) Вывод списка всех автомобилей заданного
года выпуска, цена которых больше указанной.");
            System.out.println("(5) Вывод списка всех автомобилей, взятых на
прокат.");
            System.out.println("(6) Вывод списка всех автомобилей,взятых на
прокат с выводом личной информации об арендаторах.");
            System.out.println("(7) Выход.");

            System.out.println("Выберите пункт меню: ");
            Scanner scanner = new Scanner(System.in);
            choice = scanner.nextInt();
            switch (choice) {
                case 1:
                    carRentalSystem.printAllCars();
                    break;
                case 2:
                    System.out.println("Введите марку автомобиля: ");
                    scanner.nextLine();
                    String brand = scanner.nextLine();
                    carRentalSystem.printCarsByBrand(brand);
                    break;
                case 3:
                    scanner.nextLine();
                    System.out.println("Введите модель автомобиля: ");
                    String model = scanner.nextLine();
                    System.out.println("Введите количество лет эксплуатации:
");
                    int  years = scanner.nextInt();

                    carRentalSystem.printOlderCarsByModel(model,years);
                    break;
                case 4:
                    System.out.println("Введите год автомобиля: ");
                    int year = scanner.nextInt();
                    System.out.println("Введите цену: ");
                    double price = scanner.nextDouble();
                    carRentalSystem.printCarsByYearAndPrice(year,price);
                    break;
                case 5:
                    carRentalSystem.printRentedCars();
                    break;
                case 6:
                    carRentalSystem.printRentedCarsWithRenterInfo();
                    break;
                case 7:
                    return;
```

```
                default:
                    System.out.println("Неверный пункт меню");
                    break;
            }
        }
    }
}
```

## Car.java:

```
package Lab3;

class Car{
    private int carId;
    private String carBrand;
    private String carModel;
    private int year;
    private String carColor;
    private double carPrice;
    private String regNumber;
    private String carNumber;
    private String renterFIO;
    private String renterPassportNumber;

    public Car(int carId, String carBrand, String carModel, int year, String
carColor, double carPrice, String regNumber, String carNumber, String
renterFIO, String renterPassportNumber) {
        this.carId = carId;
        this.carBrand = carBrand;
        this.carModel = carModel;
        this.year = year;
        this.carColor = carColor;
        this.carPrice = carPrice;
        this.regNumber = regNumber;
        this.carNumber = carNumber;
        this.renterFIO = renterFIO;
        this.renterPassportNumber = renterPassportNumber;
    }

    public Car(){
        this.carId = 0;
        this.carBrand = "Undefined";
        this.carModel = "Undefined";
        this.year = 0;
        this.carColor = "Undefined";
        this.carPrice = 0.0;
        this.regNumber = "Undefined";
        this.carNumber = "Undefined";
        this.renterFIO = "Undefined";
        this.renterPassportNumber = "Undefined";
    }

    public void setCarId(int carId){
        this.carId = carId;
    }
    public void setCarBrand(String carBrand){
        this.carBrand=carBrand;
    }
    public void setCarModel(String carModel) {
        this.carModel = carModel;
    }
    public void setYear(int year) {
```

```
        this.year = year;
    }
    public void setCarColor(String carColor) {
        this.carColor = carColor;
    }
    public void setCarPrice(double carPrice) {
        this.carPrice = carPrice;
    }
    public void setRegNum(String regNumber) {
        this.regNumber = regNumber;
    }
    public void setCarNumber(String carNumber) {
        this.carNumber = carNumber;
    }
    public void setRenterFIO(String renterFIO) {
        this.renterFIO = renterFIO;
    }
    public void setRenterPassportNumber(String renterPassportNumber) {
        this.renterPassportNumber = renterPassportNumber;
    }
    public int getCarId() {
        return carId;
    }
    public String getCarBrand() {
        return carBrand;
    }
    public String getCarModel() {
        return carModel;
    }
    public int getYear() {
        return year;
    }
    public String getCarColor() {
        return carColor;
    }
    public double getCarPrice() {
        return carPrice;
    }
    public String getRegNumber() {
        return regNumber;
    }
    public String getCarNumber() {
        return carNumber;
    }
    public String getRenterFullName() {
        return renterFIO;
    }
    public String getRenterPassportNumber() {
        return renterPassportNumber;
    }
    public void printCarInfo(){
        System.out.println("id: "+ this.carId);
        System.out.println("brand: "+ this.carBrand);
        System.out.println("model: "+ this.carModel);
        System.out.println("year: "+ this.year);
        System.out.println("color: "+ this.carColor);
        System.out.println("price: "+ this.carPrice + "$");
        System.out.println("registration number: "+ this.regNumber);
        System.out.println("number: "+ this.carNumber);
        System.out.println("Renter FullName: "+ this.renterFIO);
        System.out.println("Renter passport number: "+
this.renterPassportNumber + "\n");
    }
}
```

## CarRent.java:

```
package Lab3;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.time.Year;
import java.util.ArrayList;

class CarRent {
    private ArrayList<Car> cars;

    public CarRent(String filePath) {

        this.cars = new ArrayList<>();

        File file = new File(filePath);

        try {
            BufferedReader reader = new BufferedReader(new FileReader(file));
            String line;
            Car car = null;
            while ((line = reader.readLine()) != null) {
                String[] arg = line.split(":");

                if (arg.length == 2) {
                    String key = arg[0];
                    String value = arg[1];
                    switch (key) {
                        case "id":
                            car = new Car();
                            car.setCarId(Integer.parseInt(value));
                            break;
                        case "Brand":
                            car.setCarBrand(value);
                            break;
                        case "Model":
                            car.setCarModel(value);
                            break;
                        case "Year":
                            car.setYear(Integer.parseInt(value));
                            break;
                        case "Color":
                            car.setCarColor(value);
                            break;
                        case "Price":
                            car.setCarPrice(Double.parseDouble(value));
                            break;
                        case "RegNumber":
                            car.setRegNum(value);
                            break;
                        case "CarNumber":
                            car.setCarNumber(value);
                            break;
                        case "RenterFIO":
                            car.setRenterFIO(value);
                            break;
                        case "RenterPassportNumber":
                            car.setRenterPassportNumber(value);
                            cars.add(car);
```

```
                            break;
                        default:
                            System.out.println("Invalid data format: " +
key);
                            break;
                    }
                }
            }
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }

    //Вывод списка всех автомобилей
    public void printAllCars() {
        System.out.println("Cars information: ");
        for (Car car : this.cars) {
            car.printCarInfo();
        }
    }

    //Вывод списка всех автомобилей заданной марки
    public void printCarsByBrand(String carBrand) {
        boolean isFind = false;
        for (Car car : this.cars) {
            if (car.getCarBrand().equalsIgnoreCase(carBrand)) {
                isFind = true;
                car.printCarInfo();
            }
        }
        if (!isFind) {
            System.out.println("Ничего не найдено.");
        }
    }

    //Вывод списка автомобилей заданной модели, которые эксплуатируются
больше n лет;
    public void printOlderCarsByModel(String model, int years) {
        boolean isFind = false;
        int currentYear = Year.now().getValue();
        for (Car car : this.cars) {
            if (car.getCarModel().equalsIgnoreCase(model) && currentYear -
car.getYear() > years) {
                isFind = true;
                car.printCarInfo();
            }
        }
        if (!isFind) {
            System.out.println("Ничего не найдено.");
        }
    }

    //Вывод списка всех автомобилей заданного года выпуска, цена которых
больше указанной;
    public void printCarsByYearAndPrice(int year, double price) {
        int currentYear = Year.now().getValue();
        if(year>currentYear){
            System.out.println("Введите верные данные.");
            return;
        }

        for (Car car : this.cars) {
            if (car.getYear() == year && car.getCarPrice() > price) {
                car.printCarInfo();
```

```
            }
        }
    }

    //Вывод списка всех автомобилей, взятых на прокат
    public void printRentedCars() {
        boolean isFind = false;
        System.out.println("Автомобили взятые на прокат: ");
        for (Car car : this.cars) {
            if (!car.getRenterFullName().equals("-") &&
!car.getRenterPassportNumber().equals("-")) {
                System.out.println("id: " + car.getCarId() + "\nbrand: " +
car.getCarBrand() + "\nmodel: " + car.getCarModel());
                isFind = true;
            }
        }
        if (!isFind) {
            System.out.println("Ничего не найдено.");
        }
    }

    //Вывод списка всех автомобилей,взятых на прокат с выводом личной
информации об арендаторах.
    public void printRentedCarsWithRenterInfo() {
        boolean isFind = false;
        System.out.println("Автомобили взятые на прокат: ");
        for (Car car : this.cars) {
            if (!car.getRenterFullName().equals("-") &&
!car.getRenterPassportNumber().equals("-")) {
                System.out.println("id: " + car.getCarId() + "\nbrand: " +
car.getCarBrand() + "\nmodel: " + car.getCarModel()
                + "\nRenter FullName: " + car.getRenterFullName() + "\nRenter
passport number: " + car.getRenterPassportNumber() + "\n");
                isFind = true;
            }
        }
        if (!isFind) {
            System.out.println("Ничего не найдено.");
        }
    }
}
```

## Результат работы программы:

```
"D:\Porgramming instruments\JDK2023\Java\jdk-21\bin\java.exe" "-javaagent:D:\Porgramming instrum
(1) Вывод списка всех автомобилей.
(2) Вывод списка всех автомобилей заданной марки.
(3) Вывод списка автомобилей заданной модели, которые эксплуатируются больше n лет.
(4) Вывод списка всех автомобилей заданного года выпуска, цена которых больше указанной.
(5) Вывод списка всех автомобилей, взятых на прокат.
(6) Вывод списка всех автомобилей,взятых на прокат с выводом личной информации об арендаторах.
(7) Выход.
Выберите пункт меню:
1
Cars information:
id: 1
brand: Volkswagen
model: Passat b4
year: 1997
color: Green
price: 3000.0$
registration number: AB98289189
number: 4908 EB-1
Renter FullName: Daniil Zamaletdinov Aleksandrovich
Renter passport number: 556987233

id: 2
brand: Volkswagen
model: Passat b5
year: 2004
color: Green
price: 5000.0$
registration number: AB98289189
number: 7777 AA-1
Renter FullName: Maksim Yvaniuk Sergeevich
Renter passport number: AB3171230
```

```
 id: 3
 brand: Audi
 model: A4
 year: 2012
 color: Red
 price: 12000.0$
 registration number: AC23123EDS22
 number: 8888 AA-1
 Renter FullName: -
 Renter passport number: -

 id: 4
 brand: Porshe
 model: 911
 year: 2019
 color: Blue
 price: 150000.0$
 registration number: AC23DFDDF22
 number: 9999 BB-1
 Renter FullName: Bondarenko Kirill Andreevich
 Renter passport number: AB3232323

 id: 5
 brand: Opel
 model: Grandland X
 year: 2021
 color: Grey
 price: 20500.0$
 registration number: BV33DFDDF99
 number: 0777 KB-1
 Renter FullName: Buben Stanislav Olegovich
 Renter passport number: AB6669033
```

```
id: 6
brand: Mercedes-Benz
model: Maybach X222
year: 2018
color: Black
price: 275000.0$
registration number: BV33DFDDF99
number: 1111 II-1
Renter FullName: Buvin Dmitriy Aleksandrovich
Renter passport number: AB9991233
```

```
Выберите пункт меню:
2
Введите марку автомобиля:
Opel
id: 5
brand: Opel
model: Grandland X
year: 2021
color: Grey
price: 20500.0$
registration number: BV33DFDDF99
number: 0777 KB-1
Renter FullName: Buben Stanislav Olegovich
Renter passport number: AB6669033
```

```
Выберите пункт меню:
3
Введите модель автомобиля:
911
Введите количество лет эксплуатации:
3
id: 4
brand: Porshe
model: 911
year: 2019
color: Blue
price: 150000.0$
registration number: AC23DFDDF22
number: 9999 BB-1
Renter FullName: Bondarenko Kirill Andreevich
Renter passport number: AB3232323
```

```
Выберите пункт меню:
4
Введите год автомобиля:
2018
Введите цену:
10000
id: 6
brand: Mercedes-Benz
model: Maybach X222
year: 2018
color: Black
price: 275000.0$
registration number: BV33DFDDF99
number: 1111 II-1
Renter FullName: Buvin Dmitriy Aleksandrovich
Renter passport number: AB9991233
```

```
Выберите пункт меню:
5
Автомобили взятые на прокат:
id: 1
brand: Volkswagen
model: Passat b4
id: 2
brand: Volkswagen
model: Passat b5
id: 4
brand: Porshe
model: 911
id: 5
brand: Opel
model: Grandland X
id: 6
brand: Mercedes-Benz
model: Maybach X222
```

```
Выберите пункт меню:
6
Автомобили взятые на прокат:
id: 1
brand: Volkswagen
model: Passat b4
Renter FullName: Daniil Zamaletdinov Aleksandrovich
Renter passport number: 556987233

id: 2
brand: Volkswagen
model: Passat b5
Renter FullName: Maksim Yvaniuk Sergeevich
Renter passport number: AB3171230

id: 4
brand: Porshe
model: 911
Renter FullName: Bondarenko Kirill Andreevich
Renter passport number: AB3232323

id: 5
brand: Opel
model: Grandland X
Renter FullName: Buben Stanislav Olegovich
Renter passport number: AB6669033

id: 6
brand: Mercedes-Benz
model: Maybach X222
Renter FullName: Buvin Dmitriy Aleksandrovich
Renter passport number: AB9991233
```

**Вывод:** в ходе выполнения лабораторной работы я научился создавать и использовать классы в программах на языке программирования Java.

.