МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

КАФЕДРА ИИТ

ОТЧЁТ

по лабораторной работе №3

Выполнил:

А. Н. Марзан,

студент 3 курса

группы ПО-9

Проверил:

А. А. Крощенко,

«08» 02 2024 г.

**Цель:** научиться создавать и использовать классы в программах на языке программирования Java

**Вариант 12**

**Ход работы**

**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()

2) Равносторонний треугольник, заданный длинами сторон - предусмотреть возможность определения площади и периметра, а также логический метод, определяющий существует или такой треугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа.

**Задание 2**

Разработать автоматизированную систему на основе некоторой структуры данных, манипулирующей объектами пользовательского класса. Реализовать требуемые функции обработки данных

Требования к выполнению

* Задание посвящено написанию классов, решающих определенную задачу автоматизации;
* Данные для программы загружаются из файла (формат произволен). Файл создать и написать вручную.

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:**

**Main.java**

import java.util.List;  
  
public class Main  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println();  
 System.*out*.println("Task 1:");  
 EquilateralTriangle triangle1 = new EquilateralTriangle(5.0);  
 EquilateralTriangle triangle2 = new EquilateralTriangle(7.5);  
  
 System.*out*.println("Triangle 1: " + triangle1);  
 System.*out*.println("Perimeter of Triangle 1: " + triangle1.calculatePerimeter());  
 System.*out*.println("Area of Triangle 1: " + triangle1.calculateArea());  
 System.*out*.println("Is Triangle 1 valid? " + triangle1.isTriangleValid());  
  
 System.*out*.println();  
  
 System.*out*.println("Triangle 2: " + triangle2);  
 System.*out*.println("Perimeter of Triangle 2: " + triangle2.calculatePerimeter());  
 System.*out*.println("Area of Triangle 2: " + triangle2.calculateArea());  
 System.*out*.println("Is Triangle 2 valid? " + triangle2.isTriangleValid());  
  
 System.*out*.println();  
  
 System.*out*.println("Are Triangle 1 and Triangle 2 equal? " + triangle1.equals(triangle2));  
  
  
 System.*out*.println();  
 System.*out*.println("Task 2:");  
  
 CarRentalSystem rentalSystem = new CarRentalSystem();  
  
 rentalSystem.addCar(new Car(1, "Toyota", "Corolla", 2018, Color.*RED*, 20000,  
 "AB12345", 123));  
 rentalSystem.addCar(new Car(2, "BMW", "X5", 2020, Color.*BLUE*, 50000,  
 "CD67890", 456, "John Doe",  
 "1234567890"));  
 rentalSystem.addCar(new Car(3, "Mercedes", "E-Class", 2019, Color.*WHITE*, 35000,  
 "EF24680", 789));  
  
 List<Car> allCars = rentalSystem.getCars();  
 System.*out*.println("All Cars:");  
 for (Car car : allCars)  
 {  
 System.*out*.println(car);  
 }  
  
 List<Car> toyotaCars = rentalSystem.getCarsByBrand("Toyota");  
 System.*out*.println("\nToyota Cars:");  
 for (Car car : toyotaCars)  
 {  
 System.*out*.println(car);  
 }  
  
 List<Car> oldCars = rentalSystem.getCarsByModelAndYears("Corolla", 3);  
 System.*out*.println("\nOld Toyota Corolla Cars (>3 years):");  
 for (Car car : oldCars)  
 {  
 System.*out*.println(car);  
 }  
  
 List<Car> expensiveCars = rentalSystem.getCarsByYearAndPrice(2019, 30000);  
 System.*out*.println("\nExpensive 2019 Cars (>30000):");  
 for (Car car : expensiveCars)  
 {  
 System.*out*.println(car);  
 }  
  
 List<Car> rentedCars = rentalSystem.getRentedCars();  
 System.*out*.println("\nRented Cars:");  
 for (Car car : rentedCars)  
 {  
 System.*out*.println(car);  
 }  
  
 List<Car> rentedCarsWithTenantsInfo = rentalSystem.getRentedCarsWithTenantsInfo();  
 System.*out*.println("\nRented Cars with Tenants Info:");  
 for (Car car : rentedCarsWithTenantsInfo)  
 {  
 System.*out*.println(car);  
 }  
 }  
}

**Car.java**

import java.util.ArrayList;  
import java.util.List;  
  
enum Color  
{  
 *YELLOW*,  
 *RED*,  
 *GREEN*,  
 *BLUE*,  
 *PURPLE*,  
 *WHITE*,  
 *METALLIC*}  
public class Car  
{  
 int id;  
 String mark;  
 String model;  
 int yearOfRelease;  
 Color color;  
 int price;  
 String registrationNumber;  
 int carNumber;  
 String fullNameOfPersonWhoRentedIt = "NoName";  
 String passportNumberOfTenant = "NoPassport";  
 public Car(int id, String mark, String model, int yearOfRelease,  
 Color color, int price, String registrationNumber,  
 int carNumber)  
 {  
 this.id = id;  
 this.mark = mark;  
 this.model = model;  
 this.yearOfRelease = yearOfRelease;  
 this.color = color;  
 this.price = price;  
 this.registrationNumber = registrationNumber;  
 this.carNumber = carNumber;  
 }  
 public Car(int id, String mark, String model, int yearOfRelease,  
 Color color, int price, String registrationNumber,  
 int carNumber, String fullNameOfPersonWhoRentedIt,  
 String passportNumberOfTenant)  
 {  
 this(id, mark, model, yearOfRelease, color, price, registrationNumber,  
 carNumber);  
 this.fullNameOfPersonWhoRentedIt = fullNameOfPersonWhoRentedIt;  
 this.passportNumberOfTenant = passportNumberOfTenant;  
  
 }  
  
 @Override  
 public String toString()  
 {  
 return "Id: " + id + " Mark: " + mark + " Model: " + model + " Year of release: " + yearOfRelease  
 + " Color: " + color + " Price: " + price + " Registration number: " + registrationNumber  
 + " Car number: " + carNumber + " Name of person who rented it: " + fullNameOfPersonWhoRentedIt  
 + " Passport number of tenant:" + passportNumberOfTenant;  
 }  
}  
  
class CarRentalSystem  
{  
 private List<Car> cars;  
  
 public CarRentalSystem() {  
 cars = new ArrayList<>();  
 }  
  
 public void addCar(Car car) {  
 cars.add(car);  
 }  
  
 public List<Car> getCars() {  
 return cars;  
 }  
  
 public List<Car> getCarsByBrand(String mark) {  
 List<Car> carsByMark = new ArrayList<>();  
 for (Car car : cars) {  
 if (car.mark.equalsIgnoreCase(mark)) {  
 carsByMark.add(car);  
 }  
 }  
 return carsByMark;  
 }  
  
 public List<Car> getCarsByModelAndYears(String model, int years)  
 {  
 List<Car> carsByModelAndYears = new ArrayList<>();  
 int currentYear = java.time.Year.*now*().getValue();  
 for (Car car : cars)  
 {  
 if (car.model.equalsIgnoreCase(model) && (currentYear - car.yearOfRelease) > years)  
 {  
 carsByModelAndYears.add(car);  
 }  
 }  
 return carsByModelAndYears;  
 }  
  
 public List<Car> getCarsByYearAndPrice(int year, int price) {  
 List<Car> carsByYearAndPrice = new ArrayList<>();  
 for (Car car : cars) {  
 if (car.yearOfRelease == year && car.price > price) {  
 carsByYearAndPrice.add(car);  
 }  
 }  
 return carsByYearAndPrice;  
 }  
  
 public List<Car> getRentedCars() {  
 List<Car> rentedCars = new ArrayList<>();  
 for (Car car : cars) {  
 if (!car.fullNameOfPersonWhoRentedIt.equalsIgnoreCase("NoName")) {  
 rentedCars.add(car);  
 }  
 }  
 return rentedCars;  
 }  
  
 public List<Car> getRentedCarsWithTenantsInfo() {  
 List<Car> rentedCarsWithTenantsInfo = new ArrayList<>();  
 for (Car car : cars) {  
 if (!car.fullNameOfPersonWhoRentedIt.equalsIgnoreCase("NoName")) {  
 rentedCarsWithTenantsInfo.add(car);  
 }  
 }  
 return rentedCarsWithTenantsInfo;  
 }  
}

**EquilateralTriangle.java**

public class EquilateralTriangle  
{  
 private double sideLength;  
  
 public EquilateralTriangle(double sideLength)  
 {  
 this.sideLength = sideLength;  
 }  
  
 public double getSideLength()  
 {  
 return sideLength;  
 }  
  
 public void setSideLength(double sideLength)  
 {  
 this.sideLength = sideLength;  
 }  
  
 public double calculatePerimeter()  
 {  
 return 3 \* sideLength;  
 }  
  
 public double calculateArea()  
 {  
 return (Math.*sqrt*(3) / 4) \* sideLength \* sideLength;  
 }  
  
 public boolean isTriangleValid()  
 {  
 return sideLength > 0;  
 }  
  
 @Override  
 public String toString()  
 {  
 return "EquilateralTriangle{" +  
 "sideLength=" + sideLength +  
 '}';  
 }  
  
 @Override  
 public boolean equals(Object obj)  
 {  
 if (this == obj)  
 {  
 return true;  
 }  
 if (obj == null || getClass() != obj.getClass())  
 {  
 return false;  
 }  
 EquilateralTriangle other = (EquilateralTriangle) obj;  
 return Double.*compare*(other.sideLength, sideLength) == 0;  
 }  
}

**Вывод программы:**

Task 1:

Triangle 1: EquilateralTriangle{sideLength=5.0}

Perimeter of Triangle 1: 15.0

Area of Triangle 1: 10.825317547305481

Is Triangle 1 valid? true

Triangle 2: EquilateralTriangle{sideLength=7.5}

Perimeter of Triangle 2: 22.5

Area of Triangle 2: 24.356964481437334

Is Triangle 2 valid? true

Are Triangle 1 and Triangle 2 equal? false

Task 2:

All Cars:

Id: 1 Mark: Toyota Model: Corolla Year of release: 2018 Color: RED Price: 20000 Registration number: AB12345 Car number: 123 Name of person who rented it: NoName Passport number of tenant:NoPassport

Id: 2 Mark: BMW Model: X5 Year of release: 2020 Color: BLUE Price: 50000 Registration number: CD67890 Car number: 456 Name of person who rented it: John Doe Passport number of tenant:1234567890

Id: 3 Mark: Mercedes Model: E-Class Year of release: 2019 Color: WHITE Price: 35000 Registration number: EF24680 Car number: 789 Name of person who rented it: NoName Passport number of tenant:NoPassport

Toyota Cars:

Id: 1 Mark: Toyota Model: Corolla Year of release: 2018 Color: RED Price: 20000 Registration number: AB12345 Car number: 123 Name of person who rented it: NoName Passport number of tenant:NoPassport

Old Toyota Corolla Cars (>3 years):

Id: 1 Mark: Toyota Model: Corolla Year of release: 2018 Color: RED Price: 20000 Registration number: AB12345 Car number: 123 Name of person who rented it: NoName Passport number of tenant:NoPassport

Expensive 2019 Cars (>30000):

Id: 3 Mark: Mercedes Model: E-Class Year of release: 2019 Color: WHITE Price: 35000 Registration number: EF24680 Car number: 789 Name of person who rented it: NoName Passport number of tenant:NoPassport

Rented Cars:

Id: 2 Mark: BMW Model: X5 Year of release: 2020 Color: BLUE Price: 50000 Registration number: CD67890 Car number: 456 Name of person who rented it: John Doe Passport number of tenant:1234567890

Rented Cars with Tenants Info:

Id: 2 Mark: BMW Model: X5 Year of release: 2020 Color: BLUE Price: 50000 Registration number: CD67890 Car number: 456 Name of person who rented it: John Doe Passport number of tenant:1234567890

**Вывод:** научился создавать и использовать классы в программах на языке программирования Java.