**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

по дисциплине: «Введение в разработку программного обеспечения»

на тему: «Развитие представлений о разработке программ. Объектно-ориентированное программирование в *Java*»

Выполнил: студент гр. ИТП-21

Рылушкин Б.А.

Принял: преподаватель-стажер

Башаримов Ю.С.

Гомель 2024

**Цель работы:** получить базовые знания о разработке программного обеспечения с использованием объектно-ориентированного программирования на Java, а также в создании иерархии классов, основанной на UML-диаграмме.

**Вариант 5**

**Задание:**

1.1 Создать класс Car, Engine и Driver.

1.2 Класс Driver содержит поля - ФИО, стаж вождения.

1.3 Класс Engine содержит поля - мощность, производитель.

1.4 Класс Car содержит поля - марка автомобиля, класс автомобиля,

вес, водитель типа Driver, мотор типа Engine.

1.5 Вывести водителей со стажем более 5 лет.

1.6 Вывести автомобили советского производства.

**Ход работы**

1. Был скачен *IntelliJ IDEA Community* для работы с *java*.
2. Создали проект.
3. Создали класс Driver который представляет книгу и содержит поля *fullname*, *experience*.
4. Создали класс car который содержит поля: марка автомобиля, класс автомобиля, вес, водитель типа Driver, мотор типа Engine.
5. Так же создали класс *Engine*. Он включает:

Поля:

* *power*: значение мощности двигателя.
* *manufacturer*: строка, указывающая производителя двигателя.

Методы:

* Конструктор для инициализации объекта с названием.
* Метод *printExperiencedDrivers(List<Car> cars))* возвращает список водителей со стажем больше 5 лет.
* Метод *printSovietCars(List<Car> cars)* возвращает список машин советского производства.

1. И создали класс *Main* который является самым главным классом, содержащим метод *main*, который выполняет программу. В этом классе есть:

* Создаются три водителя: " Литошик Максим", " Алексей Николайчик" и "Саня Фаер".
* Создание машин: "ВАЗ 2101","Москвич 408" и "Жигули".

На рисунке 1 представлено выполнение задания.

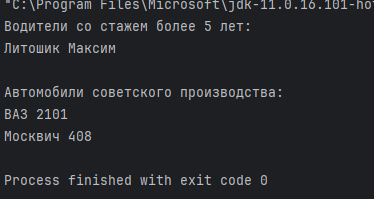


Рисунок 1 – Результат выполнения программы

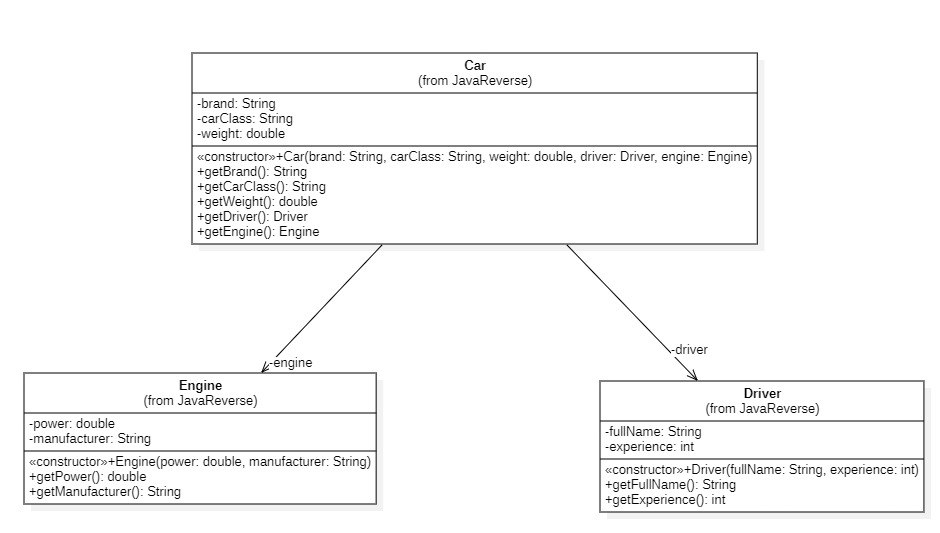


Рисунок 2 – Иерархия классов на *UML* – диаграмме

Листинг программы представлен в приложении А.

**Вывод:** в результате выполнения лабораторной работы была реализована иерархия классов с использованием наследования и композиции. Код был записан в соответствии с принятыми стандартами.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

**Main.java:**

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Главный класс для запуска приложения.

\*/

public class Main {

public static void main(String[] args) {

// Создаем водителей

Driver driver1 = new Driver("Литошик Максим", 6);

Driver driver2 = new Driver("Алексей Николайчик", 5);

Driver driver3 = new Driver("Саня Фаер", 2);

// Создаем двигатели

Engine engine1 = new Engine(150, "ВАЗ");

Engine engine2 = new Engine(200, "Москвич");

// Создаем автомобили

List<Car> cars = new ArrayList<>();

cars.add(new Car("ВАЗ 2101", "Легковой", 850, driver1, engine1));

cars.add(new Car("Москвич 408", "Легковой", 950, driver2, engine2));

cars.add(new Car("Жигули", "Легковой", 900, driver3, engine1));

// Вывод водителей со стажем более 5 лет

VehicleMethods.printExperiencedDrivers(cars);

// Вывод автомобилей советского производства

VehicleMethods.printSovietCars(cars);

}

}  
  
**VehicleClasses.java:**

class Driver {

private String fullName; // ФИО водителя

private int experience; // Стаж вождения

public Driver(String fullName, int experience) {

this.fullName = fullName;

this.experience = experience;

}

public String getFullName() {

return fullName;

}

public int getExperience() {

return experience;

}

}

/\*\*

\* Класс, представляющий двигатель автомобиля.

\*/

class Engine {

private double power; // Мощность двигателя

private String manufacturer; // Производитель двигателя

public Engine(double power, String manufacturer) {

this.power = power;

this.manufacturer = manufacturer;

}

public double getPower() {

return power;

}

public String getManufacturer() {

return manufacturer;

}

}

/\*\*

\* Класс, представляющий автомобиль.

\*/

class Car {

private String brand; // Марка автомобиля

private String carClass; // Класс автомобиля

private double weight; // Вес автомобиля

private Driver driver; // Водитель

private Engine engine; // Двигатель

public Car(String brand, String carClass, double weight, Driver driver, Engine engine) {

this.brand = brand;

this.carClass = carClass;

this.weight = weight;

this.driver = driver;

this.engine = engine;

}

public String getBrand() {

return brand;

}

public String getCarClass() {

return carClass;

}

public double getWeight() {

return weight;

}

public Driver getDriver() {

return driver;

}

public Engine getEngine() {

return engine;

}

}  
  
**VegicleMethods.java:**

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Класс, содержащий методы для работы с автомобилями.

\*/

public class VehicleMethods {

/\*\*

\* Выводит водителей со стажем более 5 лет.

\*/

public static void printExperiencedDrivers(List<Car> cars) {

System.out.println("Водители со стажем более 5 лет:");

for (Car car : cars) {

if (car.getDriver().getExperience() > 5) {

System.out.println(car.getDriver().getFullName());

}

}

}

/\*\*

\* Выводит автомобили советского производства.

\*/

public static void printSovietCars(List<Car> cars) {

System.out.println("\nАвтомобили советского производства:");

for (Car car : cars) {

if (car.getBrand().contains("ВАЗ") || car.getBrand().contains("Москвич")) {

System.out.println(car.getBrand());

}

}

}

}