**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Ордена трудового Красного Знамени федеральное государственное бюджетное**

**образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Математическая кибернетика и информационные технологии

Лабораторная работа №2

**Объектно-ориентированное программирование.**

**по дисциплине  
«Информационные технологии и программирование»**

Выполнил: студент гр. БВТ2403

Титов Д.С.

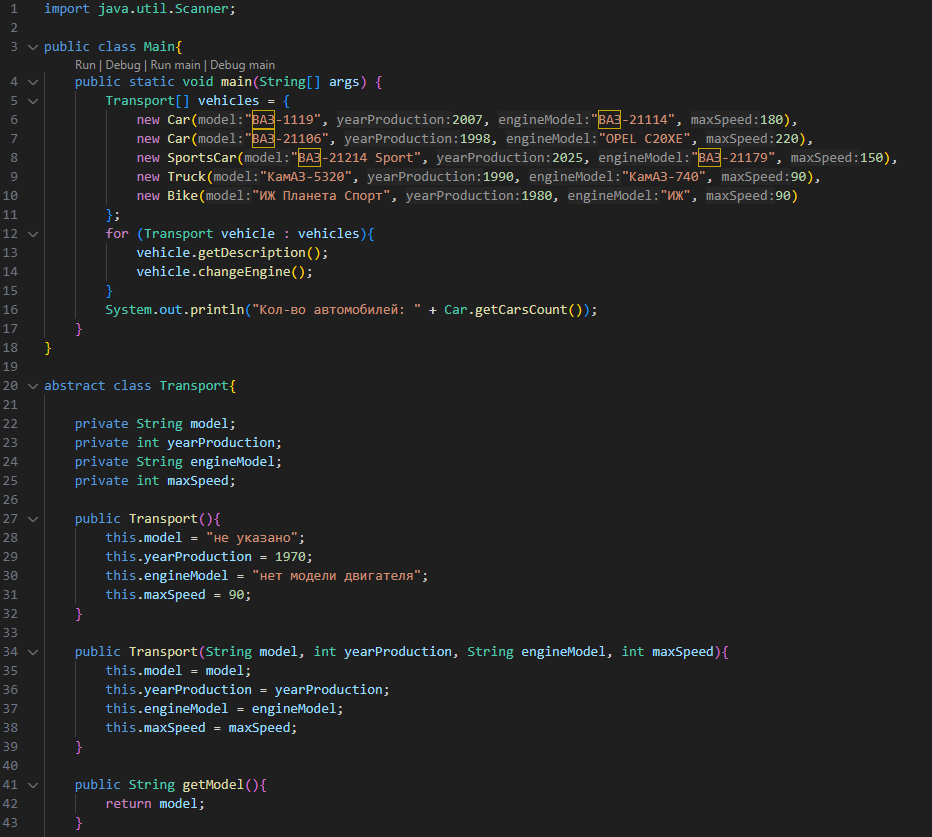
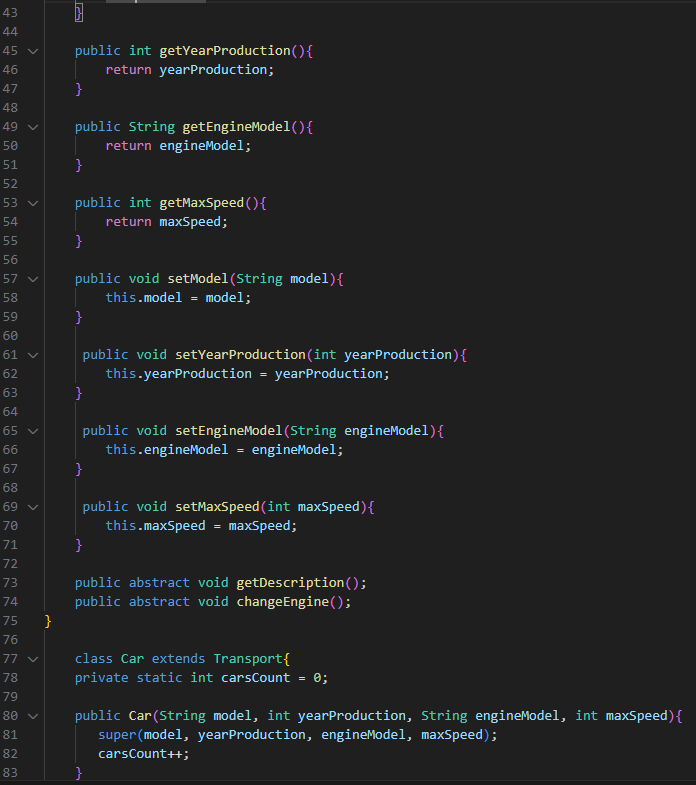
Проверил:

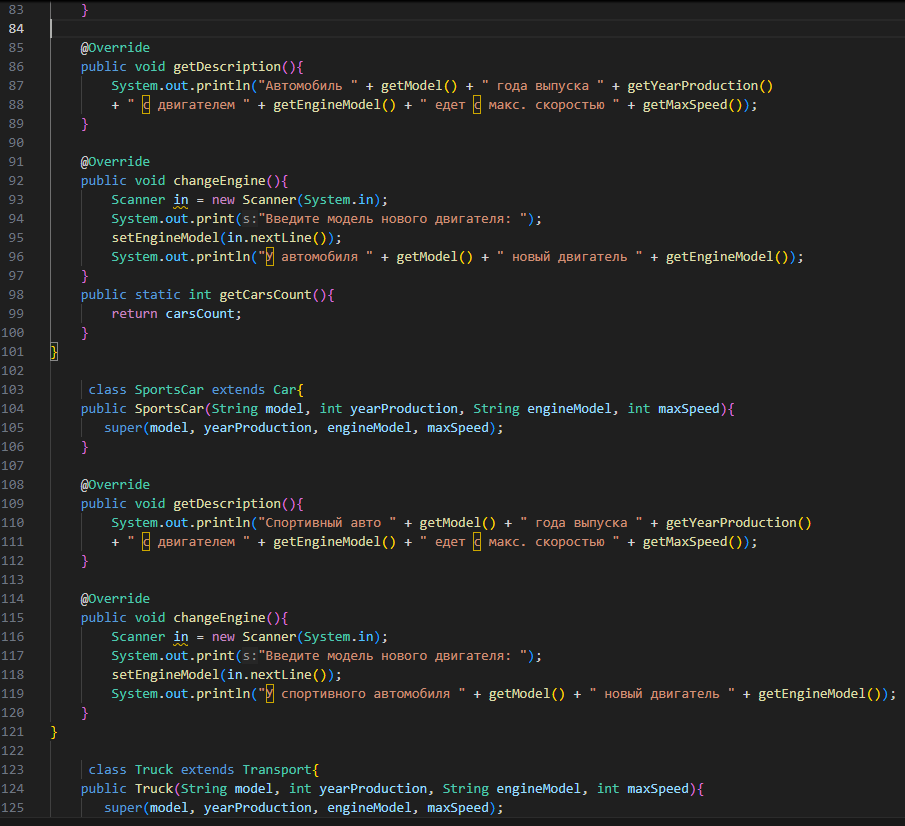
Москва, 2025 г.

**Цель работы:** изучение и практическое применение основных принципов объектно-ориентированного программирования (ООП) на языке Java, включая работу с классами, объектами, инкапсуляцией и наследованием, для разработки структурированных программных решений.

**Основная часть:**

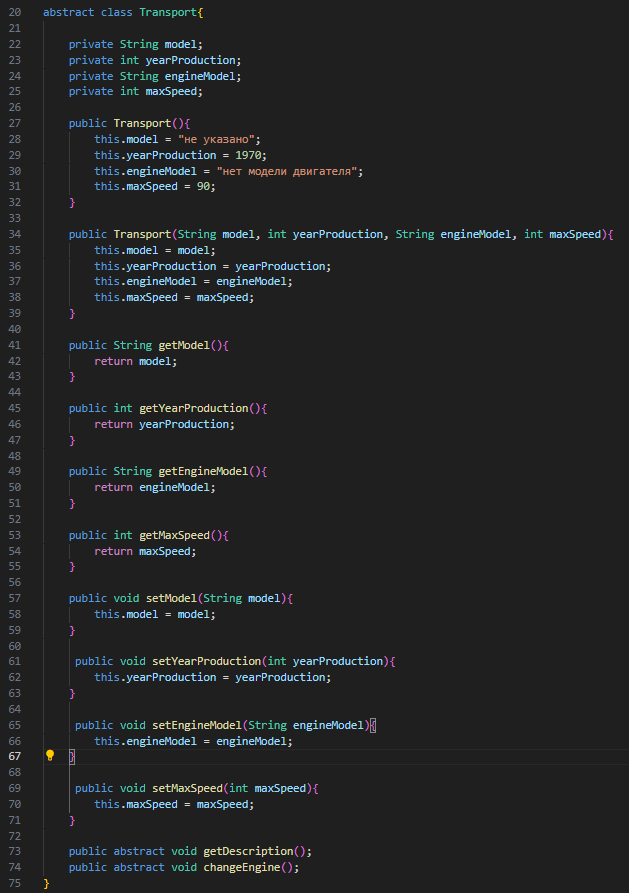
Задание. Создайте программу, демонстрирующую основные принципы объектно-ориентированного программирования: абстракцию, наследование, полиморфизм и инкапсуляцию.



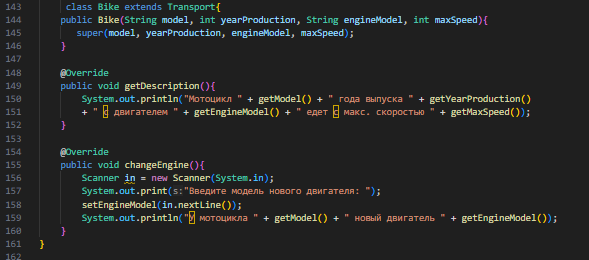


Для выполнения задания был разработан иерархический набор классов, моделирующих различные виды транспортных средств. В основе иерархии лежит абстрактный класс Transport, содержащий общие поля: модель, год выпуска, модель двигателя и максимальную скорость. Все поля объявлены как private, что обеспечивает инкапсуляцию, а доступ к ним осуществляется через соответствующие геттеры и сеттеры. Класс Transport также содержит два абстрактных метода — getDescription() и changeEngine(), которые должны быть реализованы во всех дочерних классах, что демонстрирует принцип абстракции.

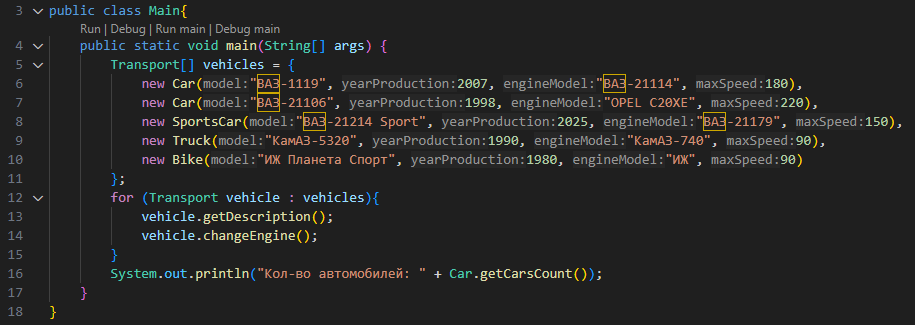


От абстрактного класса Transport наследуются четыре конкретных класса: Car, SportsCar, Truck и Bike. Класс Car расширяет функциональность базового класса и включает статическое поле carsCount для подсчёта количества созданных экземпляров автомобилей. При создании каждого объекта класса Car значение этого счётчика увеличивается на единицу. Класс SportsCar наследуется от Car, что иллюстрирует многоуровневое наследование, и переопределяет методы getDescription() и changeEngine() с учётом специфики спортивного автомобиля. Классы Truck и Bike наследуются непосредственно от Transport и также переопределяют оба абстрактных метода, адаптируя их под особенности грузовика и мотоцикла соответственно.





В методе main() класса Main создаётся массив объектов типа Transport, содержащий экземпляры всех четырёх производных классов. С помощью цикла for-each для каждого элемента массива вызываются методы getDescription() и changeEngine(). Благодаря полиморфизму при вызове этих методов автоматически выполняются их реализации, соответствующие конкретному типу объекта. В процессе выполнения программы пользователю предлагается ввести новую модель двигателя для каждого транспортного средства, после чего выводится сообщение об успешной замене. В завершение программа выводит общее количество созданных автомобилей, используя статический метод getCarsCount() класса Car.



Таким образом, реализованная программа наглядно демонстрирует применение ключевых принципов объектно-ориентированного программирования: инкапсуляция обеспечивается сокрытием внутреннего состояния объектов, абстракция — через использование абстрактного базового класса, наследование — построением иерархии классов, а полиморфизм — единообразной обработкой объектов разных типов через единый интерфейс.

Лабораторную работу я загрузил в репозиторий в GitHub по ссылке: https://github.com/sol2033/ITIP\_labs.git

**Вывод:** я изучил и практически применил основные принципы объектно-ориентированного программирования (ООП) на языке Java, включая работу с классами, объектами, инкапсуляцией и наследованием, для разработки структурированных программных решений.