Проект: «Система управления транспортом логистической компании»

Цель: Смоделировать парк транспорта (легковые, грузовики, дроны), рассчитать стоимость поездок, сформировать отчёты. Показать в работе принципы ООП без привязки к фреймворкам.

Теоретическая база

1) Класс и объект

Класс — это «чертёж», объект — конкретный экземпляр по этому чертежу.
В проекте: классы транспорта (Легковой/Грузовик/Дрон), объекты — конкретные модели.

2) Инкапсуляция

Данные скрыты внутри класса; наружу — только «безопасные» методы (геттеры/сеттеры, сервисные методы).
В проекте: модель, расход, пробег — закрытые свойства; изменение пробега — через метод.

3) Наследование

Общие признаки и поведение уносим в базовый тип, конкретика — в наследниках.
В проекте: базовый «Транспорт» + наследники «Легковой», «Грузовик», «Дрон».

4) Абстракция

Выделяем только существенные характеристики и «контракты» поведения.
В проекте: базовый «Транспорт» задаёт обязательный контракт «как посчитать стоимость поездки».

5) Полиморфизм

Один и тот же вызов метода у разных объектов ведёт себя по-разному (разные формулы, разные доплаты).
В проекте: «рассчитать стоимость поездки» — общий вызов, но разные результаты у Легкового/Грузовика/Дрона.

Что надо сделать.

Шаг 1. Определить доменную модель

- Базовый тип «Транспорт»: общие характеристики (модель, расход топлива, пробег).
- Наследники:
  - **Легковой**: базовый расход.
  - **Грузовик**: учёт веса груза (чем тяжелее — тем дороже/больше расход).
  - **Дрон**: учёт надбавки за энергоёмкость/эксплуатацию (например, +20%).

Ожидаемый результат шага: список сущностей и их свойств (таблица, UML-набросок).

Шаг 2. Описать «контракт» поведения

- У любого транспорта должно быть **обязательное поведение**: «рассчитать стоимость поездки» по расстоянию и цене топлива.
- Контракт закрепляется в базовом типе; конкретные формулы — в наследниках.

Ожидаемый результат шага: текстовое описание контрактов (1–2 абзаца) и схема, где видно обязательный метод.

Шаг 3. Настроить инкапсуляцию

- Свойства транспорта (модель, расход, пробег) — **непосредственно извне не изменяются**.

- Пробег меняется только через публичный метод «проехать(км)».
- Цена/расстояние поступают как параметры при расчёте (не хранятся внутри объекта — это признаки сценария, а не состояния).

Ожидаемый результат шага: список «допустимых действий» над объектом (Что можно? Что нельзя?).

Шаг 4. Придумать формулы расчёта

- **Легковой**: стоимость = (дистанция / 100) × расход × цена топлива.
- **Грузовик**: стоимость = (дистанция / 100) × (расход + 0.5 × тоннаж) × цена топлива.
- **Дрон**: стоимость = (дистанция / 100) × расход × цена топлива × 1.2 (надбавка).

Ожидаемый результат шага: краткая памятка с формулами (1 блок).

Шаг 5. Спроектировать отчёт менеджера парка

- На вход: набор транспортных средств, единые «дистанция» и «цена топлива».
- На выход: сводная таблица с расчётами и итоговой суммой.

Ожидаемый результат шага (пример вывода):

Модель: Toyota Camry | Расход: 8.0 л/100 км
Стоимость поездки (150 км, 200/л): 2400.0

Модель: Volvo FH | Расход: 25.0 л/100 км | Тоннаж: 10 т
Стоимость поездки (150 км, 200/л): 4125.0

Модель: DJI Air 2S | Расход: 5.0 л/100 км | Надбавка: 20%
Стоимость поездки (150 км, 200/л): 1800.0

ИТОГО по парку: 8325.0

Где принципы ООП в этом варианте:

- Класс/объект — конкретные модели транспорта;
- Инкапсуляция — свойства закрыты, расчёт через методы;
- Наследование — общий базовый тип + три наследника;
- Абстракция — единый контракт расчёта стоимости;
- Полиморфизм — общий вызов «посчитать стоимость» даёт разные цифры.