Машанло Рамина IT-125

**Лаб.9**

1. Я создала абстрактный класс Payment, который задаёт общий интерфейс для платёжных систем с методами pay и refund. Подклассы CreditCardPayment и CryptoPayment реализуют эти методы по-разному: первый выполняет оплату и возврат с помощью кредитной карты, а второй — с использованием криптовалюты. Так я показала абстракцию и полиморфизм, когда один интерфейс имеет разные реализации.

2. Я создала абстрактный класс Course, который определяет структуру любого курса через методы start, get_materials и end. Подклассы PythonCourse и MathCourse реализуют их по-своему: у каждого курса свой набор материалов и логика запуска и завершения. Это пример наследования и переопределения методов.

3. Я создала абстрактный класс Delivery, где есть методы calculate_cost и deliver. Подклассы AirDelivery, GroundDelivery и SeaDelivery по-разному рассчитывают стоимость доставки в зависимости от типа транспорта. Это демонстрирует полиморфизм, когда один интерфейс имеет разные реализации.

4. Я создала класс BankAccount с приватными полями владельца, баланса и PIN-кода. Методы deposit, withdraw и change_pin проверяют правильность PIN перед операцией, что обеспечивает инкапсуляцию и безопасность данных. Так я защитила доступ к счёту от прямого изменения.

5. Я создала класс UserProfile с приватными полями email и пароль, а также защищённым полем _status. Метод login проверяет правильность данных, upgrade_to_premium меняет статус пользователя, а get_info возвращает информацию о профиле. Это пример инкапсуляции и контроля доступа через методы.

6. Я создала класс Product, где хранится название, цена и приватная скидка. Метод set_discount устанавливает скидку только при наличии прав администратора, а get_price возвращает цену с учётом скидки.

Так я реализовала инкапсуляцию и валидацию данных, защищая цену от прямого изменения.

7. Я создала классы TextFile, ImageFile и AudioFile, у которых есть одинаковый метод open, открывающий разные типы файлов. Функция open_all вызывает метод open для каждого объекта, не зная его точный тип. Это пример полиморфизма, когда разные классы реализуют одинаковый метод по-своему.

8. Я создала классы Car, Truck и Bicycle, где у каждого есть метод move, рассчитывающий время движения исходя из скорости. Функция simulate_transport вызывает одинаковый метод у разных транспортных средств, демонстрируя полиморфизм.

9. Я создала классы Student, Teacher и Administrator, у которых одинаковый метод access_portal, но он возвращает разное сообщение в зависимости от роли. Это ещё один пример полиморфизма, когда одинаковый метод реализован по-разному у разных объектов.