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»

(МТУСИ)

Кафедра «Программная инженерия»

дисциплина «Рефакторинг баз данных и приложений»

Лабораторная работа № 3

**«Изучение рефакторинга приложений»**

Выполнил: студент группы БПИ2303

Берездовец Илья Сергеевич

2025

**Цель работы:**

1. Ознакомиться с основными принципами и задачами рефакторинга.

2. Научиться выявлять проблемные участки кода (code smells) и устранять их.

3.Применить техники рефакторинга для улучшения читаемости, структуры и производительности кода.

4. Развить навыки анализа и улучшения существующего кода.

**Ход работы:**

1. **Открыл проект до рефакторинга:**



Рис. 1 – файл shop.py

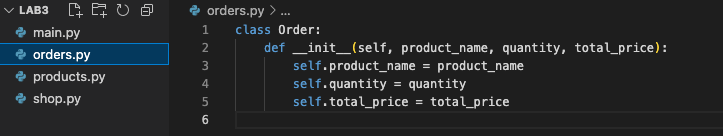


Рис. 2 – файл orders.py

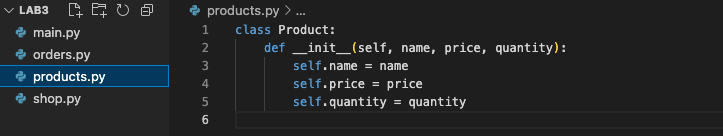


Рис. 3 – файл products.py

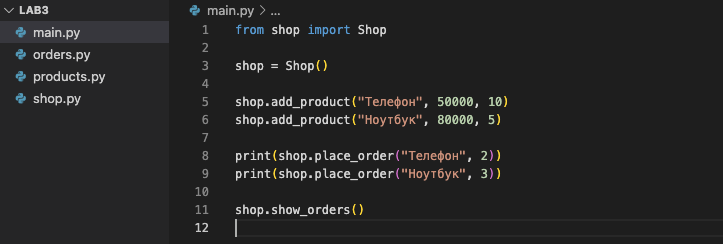


Рис. 4 – файл main.py

1. **Выявил проблемы, которые подлежат рефакторингу**.

1. Дублирование кода:

Проверка наличия товара повторяется при оформлении заказа.

2. Отсутствие обработки исключений:

Нет защиты от ввода некорректных данных (например, отрицательного количества товара).

3. Плохая читаемость и нарушение SRP (Single Responsibility Principle):

Shop выполняет сразу несколько задач: хранит товары, оформляет заказы и уменьшает их количество.

1. **Провел рефакторинг кода.**

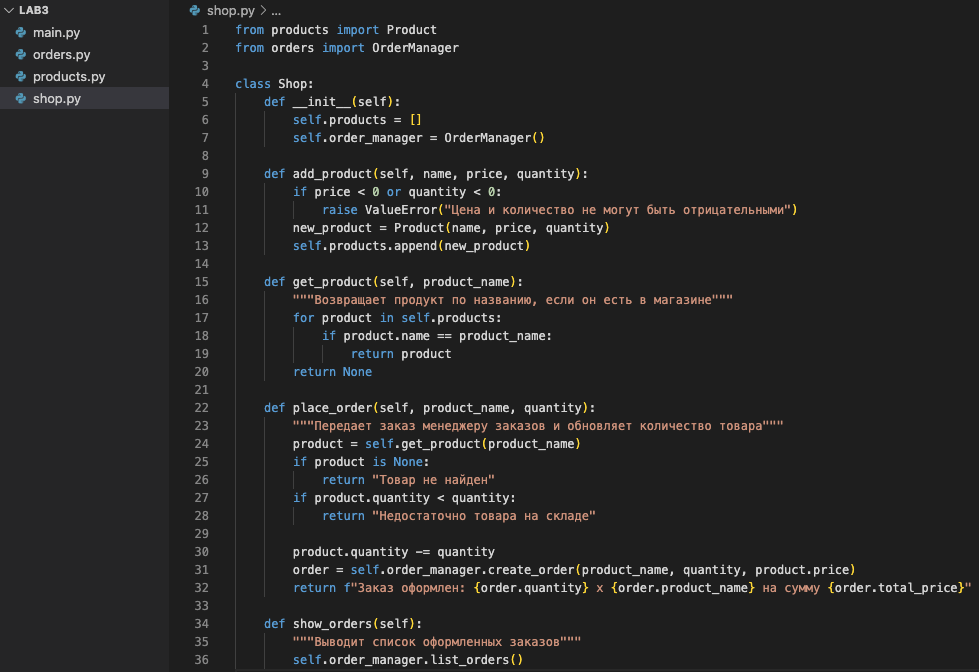


Рис. 5 – shop.py после рефакторинга

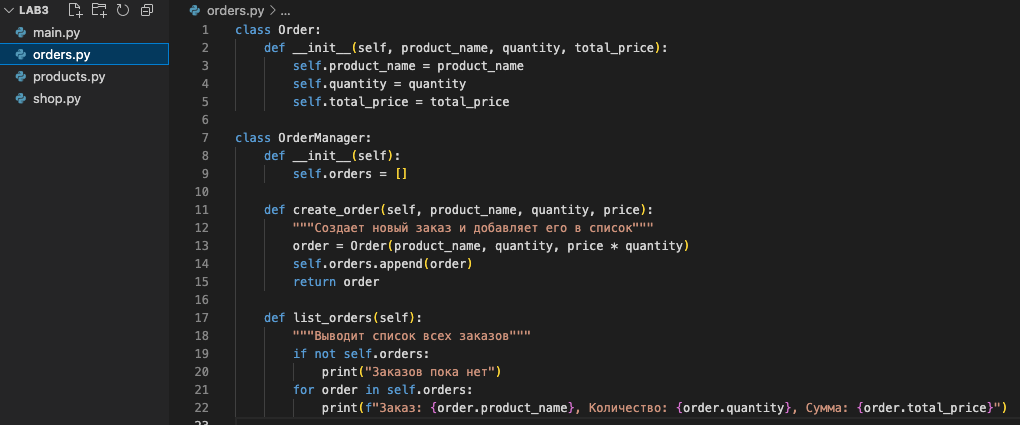


Рис. 6 – orders.py после рефакторинга

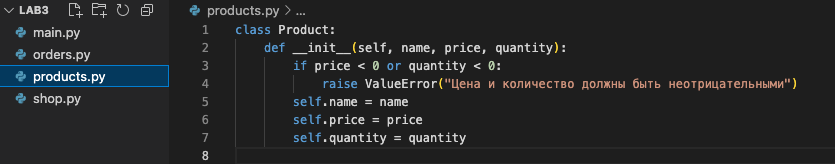


Рис. 7 – products.py после рефакторинга

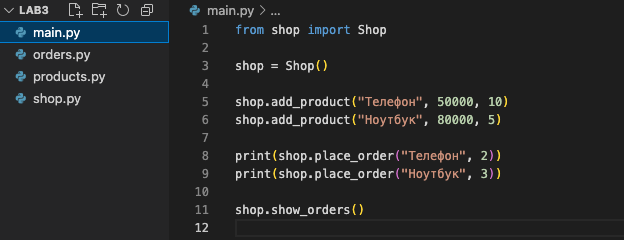


Рис. 8 – main.py

1. **Улучшения после рефакторинга:**

1. Код стал более читаемым и разделенным по ответственности:

Shop управляет товарами.

OrderManager управляет заказами.

Product хранит данные о продукте с валидацией.

2. Устранено дублирование кода:

Логика заказов теперь находится в OrderManager, а не в Shop.

Функция get\_product() теперь возвращает продукт, вместо поиска вручную.

3. Добавлены проверки:

Product теперь не может иметь отрицательную цену или количество.

Shop.add\_product() тоже проверяет корректность данных.

1. **Добавил тесты:**

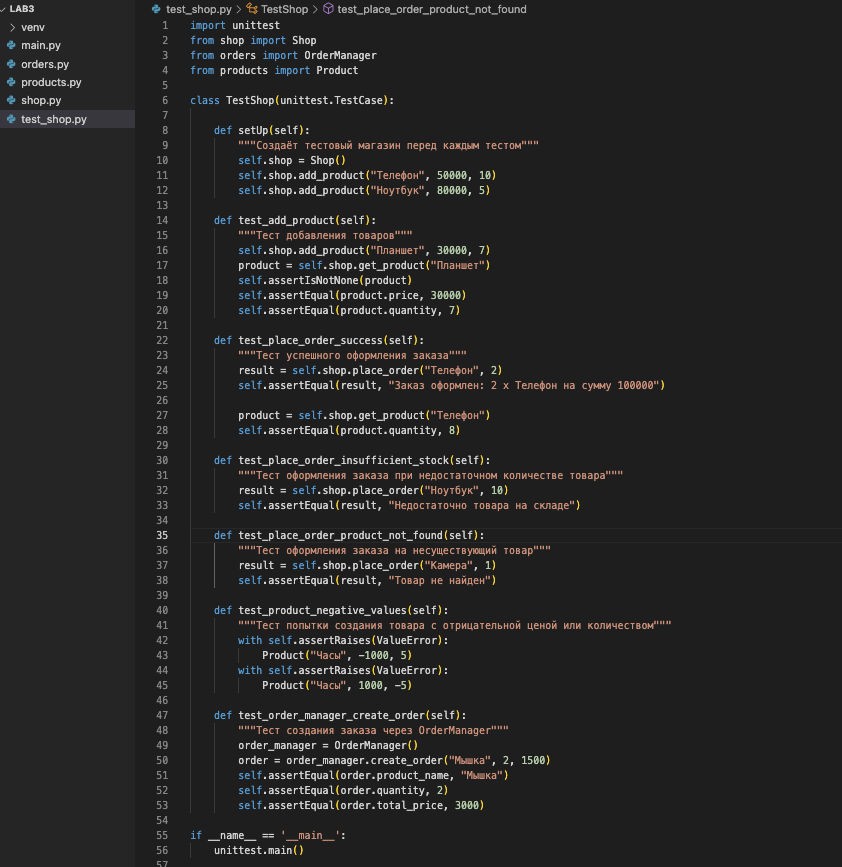
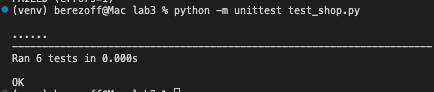
­

Рис. 9 – файл test\_shop.py (тесты)

Тесты прошли успешно:



**Вывод:**

В ходе выполнения лабораторной работы был проведен рефакторинг исходного кода с целью улучшения его читаемости и структуры. Были устранены проблемы дублирования кода, большие функции были разделены на более мелкие, а имена переменных и методов приведены к более понятному виду. Добавлена обработка ошибок для повышения надежности. В завершение были написаны и успешно выполнены юнит-тесты для проверки функциональности основных методов классов. Работа позволила развить навыки рефакторинга и улучшить качество кода.

**Ссылка на GitHub**: https://github.com/iberezaa