ЦЕЛ НА УПРАЖНЕНИЕТО

Да подобрите качеството на следния код, без да променяте неговата функционалност:

- добавяне на поръчки
- изчисляване на цена
- поддържане на ДДС
- промо кодове
- отстъпка

Вашата задача е да направите кода:

- по-четим
- по-разделен (SRP)
- по-тестируем
- по-разширяем
- по-сигурен

Анализ на проблемите (code smells)

Прегледайте кода и открийте **поне 10 проблема**.
Всяко от тези трябва да бъде отбелязано:

Очевидни проблеми в кода:

1. Смесване на UI логика с бизнес логика
2. Глобално състояние (static List<Order> allOrders)
3. CalculatePrice е твърде дълъг и прави много неща
4. Липса на валидация на входа
5. Дублиран код
6. Лоши имена на променливи (o, x, t)
7. Промо кодовете са case-sensitive (бъг)
8. Търсене на order чрез бавен loop
9. Липса на NULL проверки
10. Липса на инкапсулация (полетата на Order са public)
11. Няма интерфейси → не може да се тества качествено

12. Няма разделение на слоеве (UI / Service / Repository)
13. Парите се държат в double, което е опасно
14. Няма отделен PriceCalculator клас
15. Прекалено дълъг Main метод

Задача 1: Изнесете модела (Order, OrderItem) в отделни файлове

Насоки:

- Създайте папка Models
- Направете свойствата с public getter-и и setter-и
- Сменете public int id с public int Id { get; set; }

Задача 2: Изнесете бизнес логи

Задача 3: Премахнете глобалните static полета

Създайте OrderRepository

Задача 4: Подобрете валидирането

Примери:

- ако поръчката няма продукти → хвърли грешка
- ако quantity ≤ 0 → грешка
- ако price ≤ 0 → грешка
- ако discount < 0 или > 1 → грешка

Задача 5: Преработете CalculatePrice

Ред на изчисленията: Да се използват добре именувани локални променливи

1. Сума на продуктите
2. Отстъпка
3. Промо код
4. ДДС