Рефакторинг текста:

```
# Классы данных
class Computer:
    def __init__(self, id, name, price, os_id):
        self.id = id
        self.name = name
        self.price = price
        self.os_id = os_id


class OperatingSystem:
    def __init__(self, id, name):
        self.id = id
        self.name = name


class ComputerOSRelation:
    def __init__(self, computer_id, os_id):
        self.computer_id = computer_id
        self.os_id = os_id


# Запрос 1: Компьютеры, название которых заканчивается на "Book", и их ОС
def get_computers_ending_with_book(computers, operating_systems):
    return [
        (comp.name, os.name)
        for comp in computers
        for os in operating_systems
        if comp.name.endswith("Book") and comp.os_id == os.id
    ]


# Запрос 2: ОС и средняя стоимость компьютеров, отсортированные по цене
def get_os_with_average_price(computers, operating_systems):
    os_avg_price = {}
    for os in operating_systems:
        comp_prices = [comp.price for comp in computers if comp.os_id == os.id]
        if comp_prices:
            os_avg_price[os.name] = sum(comp_prices) / len(comp_prices)

    return sorted(os_avg_price.items(), key=lambda x: x[1])


# Запрос 3: ОС, название которых начинается с "L", и список компьютеров с ними
def get_os_starting_with_l(computers, operating_systems):
    return [
        (os.name, [comp.name for comp in computers if comp.os_id == os.id])
        for os in operating_systems
        if os.name.startswith("L")
    ]
```

```
# Тестовые данные
computers = [
    Computer(1, "Lenovo ThinkPad", 800, 1),
    Computer(2, "HP EliteBook", 1200, 2),
    Computer(3, "Dell Inspiron", 900, 1),
    Computer(4, "Asus ZenBook", 1000, 3),
    Computer(5, "Acer Predator", 1500, 2),
]

operating_systems = [
    OperatingSystem(1, "Windows 10"),
    OperatingSystem(2, "Linux Ubuntu"),
    OperatingSystem(3, "macOS"),
]

# Выполнение запросов
if __name__ == "__main__":
    print("Запрос 1:")
    for name, os_name in get_computers_ending_with_book(computers,
operating_systems):
        print(f"Компьютер: {name}, ОС: {os_name}")

    print("\nЗапрос 2:")
    for os_name, avg_price in get_os_with_average_price(computers,
operating_systems):
        print(f"ОС: {os_name}, Средняя стоимость: {avg_price}")

    print("\nЗапрос 3:")
    for os_name, comp_list in get_os_starting_with_l(computers,
operating_systems):
        print(f"ОС: {os_name}, Компьютеры: {', '.join(comp_list)}")
```

Модульные тесты:

```
import unittest

class TestComputerFunctions(unittest.TestCase):
    def setUp(self):
        # Тестовые данные
        self.computers = [
            Computer(1, "Lenovo ThinkPad", 800, 1),
            Computer(2, "HP EliteBook", 1200, 2),
            Computer(3, "Dell Inspiron", 900, 1),
            Computer(4, "Asus ZenBook", 1000, 3),
            Computer(5, "Acer Predator", 1500, 2),
        ]
        self.operating_systems = [
            OperatingSystem(1, "Windows 10"),
            OperatingSystem(2, "Linux Ubuntu"),
            OperatingSystem(3, "macOS"),
        ]

    def test_get_computers_ending_with_book(self):
        result = get_computers_ending_with_book(self.computers,
self.operating_systems)
        expected = [("HP EliteBook", "Linux Ubuntu"), ("Asus ZenBook", "macOS")]
        self.assertEqual(result, expected)

    def test_get_os_with_average_price(self):
        result = get_os_with_average_price(self.computers,
self.operating_systems)
        expected = [("Windows 10", 850), ("macOS", 1000), ("Linux Ubuntu", 1350)]
        self.assertEqual(result, expected)

    def test_get_os_starting_with_l(self):
        result = get_os_starting_with_l(self.computers, self.operating_systems)
        expected = [("Linux Ubuntu", ["HP EliteBook", "Acer Predator"])]
        self.assertEqual(result, expected)


if __name__ == "__main__":
    unittest.main()
```