Кузнецов Павел ИУ5-34Б

Отчёт РК2 по дисциплине “Парадигмы и конструкции языков программирования”

Рефакторинг кода программы:

Main.py

```
class Computer:
    def __init__(self, comp_id, name, price):
        self.comp_id = comp_id
        self.name = name
        self.price = price


class DisplayClass:
    def __init__(self, display_id, name):
        self.display_id = display_id
        self.name = name


class ComputerDisplay:
    def __init__(self, comp_id, display_id):
        self.comp_id = comp_id
        self.display_id = display_id


# Создание тестовых данных
computers = [
    Computer(1, 'Dell XPS', 1200),
    Computer(2, 'Apple MacBook', 2400),
    Computer(3, 'Lenovo ThinkPad', 900),
    Computer(4, 'Asus ROG', 1500),
]

display_classes = [
    DisplayClass(1, 'Advanced Display Technology'),
    DisplayClass(2, 'Basic Display Package'),
    DisplayClass(3, 'Advanced Display Tech'),
]

computer_displays = [
    ComputerDisplay(1, 1),
    ComputerDisplay(2, 2),
    ComputerDisplay(3, 1),
    ComputerDisplay(4, 3),
    ComputerDisplay(3, 2),
]


def computers_with_a_display(display_classes, computers, computer_displays):
    return [
        (d.name, [c.name for c in computers if
                  c.comp_id in [cd.comp_id for cd in computer_displays if
cd.display_id == d.display_id]])
        for d in display_classes if d.name.startswith('A')
    ]
```

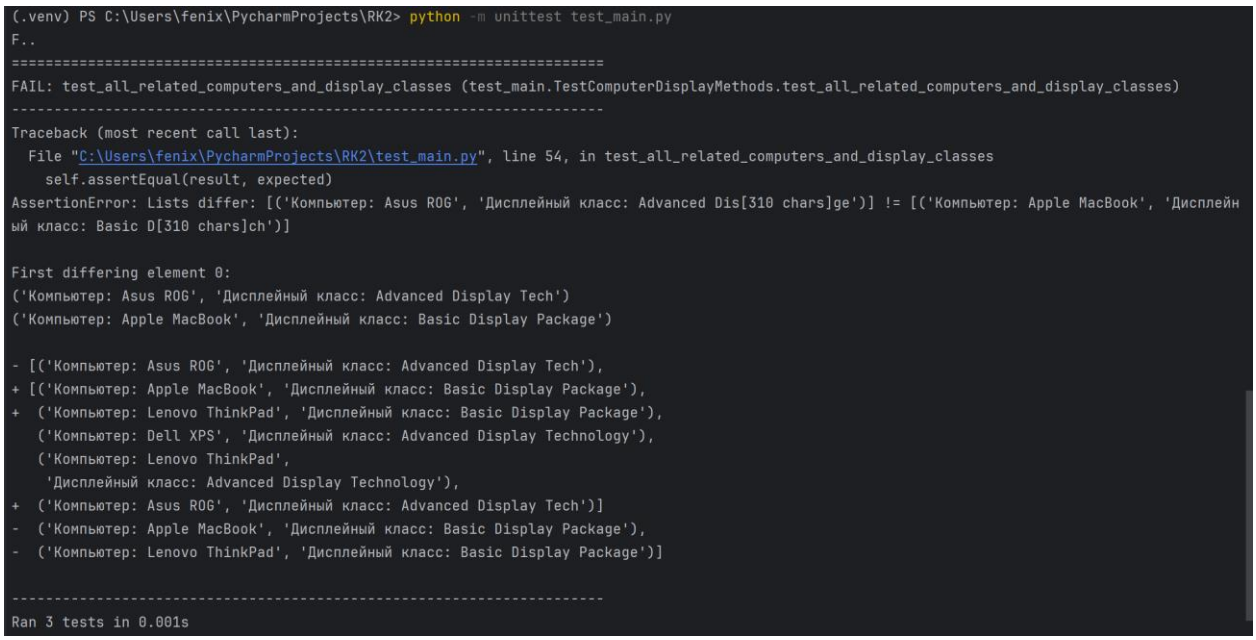
```
def display_classes_with_max_price(display_classes, computers,
computer_displays):
    max_price_by_display = {
        d.display_id: max(
            [c.price for c in computers if
             c.comp_id in [cd.comp_id for cd in computer_displays if
cd.display_id == d.display_id]],
            default=0
        )
        for d in display_classes
    }
    sorted_by_max_price = sorted(max_price_by_display.items(), key=lambda x:
x[1], reverse=True)
    return [
        (next((d.name for d in display_classes if d.display_id ==
display_id), "Unknown"), price)
        for display_id, price in sorted_by_max_price
    ]


def all_related_computers_and_display_classes(display_classes, computers,
computer_displays):
    related_data = [
        (f"Компьютер: {c.name}", f"Дисплейный класс: {d.name}")
        for cd in computer_displays
        for c in computers
        for d in display_classes
        if cd.comp_id == c.comp_id and cd.display_id == d.display_id
    ]
    related_data.sort(key=lambda x: x[1])
    return related_data


if __name__ == "__main__":
    print("Компьютеры с дисплейным классом, начинающимся на 'A':")
    print(computers_with_a_display(display_classes, computers,
computer_displays))

    print("\nДисплейные классы с максимальной ценой компьютера:")
    print(display_classes_with_max_price(display_classes, computers,
computer_displays))

    print("\nВсе связанные компьютеры и дисплейные классы:")
    print(all_related_computers_and_display_classes(display_classes,
computers, computer_displays))
```

test_main.py

```
# test_main.py

import unittest
from main import Computer, DisplayClass, ComputerDisplay,
computers_with_a_display, display_classes_with_max_price,
all_related_computers_and_display_classes
```

```
class TestComputerDisplayMethods(unittest.TestCase):

    def setUp(self):
        self.computers = [
            Computer(1, 'Dell XPS', 1200),
            Computer(2, 'Apple MacBook', 2400),
            Computer(3, 'Lenovo ThinkPad', 900),
            Computer(4, 'Asus ROG', 1500),
        ]
        self.display_classes = [
            DisplayClass(1, 'Advanced Display Technology'),
            DisplayClass(2, 'Basic Display Package'),
            DisplayClass(3, 'Advanced Display Tech'),
        ]
        self.computer_displays = [
            ComputerDisplay(1, 1),
            ComputerDisplay(2, 2),
            ComputerDisplay(3, 1),
            ComputerDisplay(4, 3),
            ComputerDisplay(3, 2),
        ]

    def test_computers_with_a_display(self):
        result = computers_with_a_display(self.display_classes,
self.computers, self.computer_displays)
        expected = [
            ('Advanced Display Technology', ['Dell XPS', 'Lenovo ThinkPad']),
            ('Advanced Display Tech', ['Asus ROG'])
        ]
        self.assertEqual(result, expected)

    def test_display_classes_with_max_price(self):
        result = display_classes_with_max_price(self.display_classes,
self.computers, self.computer_displays)
        expected = [
            ('Basic Display Package', 2400),
            ('Advanced Display Tech', 1500),
            ('Advanced Display Technology', 1200)
        ]
        self.assertEqual(result, expected)

    def test_all_related_computers_and_display_classes(self):
        result =
all_related_computers_and_display_classes(self.display_classes,
self.computers, self.computer_displays)
        expected = [
            ('Компьютер: Apple MacBook', 'Дисплейный класс: Basic Display
Package'),
            ('Компьютер: Lenovo ThinkPad', 'Дисплейный класс: Basic Display
Package'),
            ('Компьютер: Dell XPS', 'Дисплейный класс: Advanced Display
Technology'),
            ('Компьютер: Lenovo ThinkPad', 'Дисплейный класс: Advanced
Display Technology'),
            ('Компьютер: Asus ROG', 'Дисплейный класс: Advanced Display
Tech')
        ]
        self.assertEqual(result, expected)

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

Результат выполнения программы



```
(.venv) PS C:\Users\fenix\PycharmProjects\RK2> python -m unittest test_main.py
F..
======================================================================
FAIL: test_all_related_computers_and_display_classes (test_main.TestComputerDisplayMethods.test_all_related_computers_and_display_classes)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "C:\Users\fenix\PycharmProjects\RK2\test_main.py", line 54, in test_all_related_computers_and_display_classes
    self.assertEqual(result, expected)
AssertionError: Lists differ: [('Компьютер: Asus ROG', 'Дисплейный класс: Advanced Dis[310 chars]ge')] != [('Компьютер: Apple MacBook', 'Дисплейн
ый класс: Basic D[310 chars]ch')]

First differing element 0:
('Компьютер: Asus ROG', 'Дисплейный класс: Advanced Display Tech')
('Компьютер: Apple MacBook', 'Дисплейный класс: Basic Display Package')

- [('Компьютер: Asus ROG', 'Дисплейный класс: Advanced Display Tech'),
+ [('Компьютер: Apple MacBook', 'Дисплейный класс: Basic Display Package'),
+  ('Компьютер: Lenovo ThinkPad', 'Дисплейный класс: Basic Display Package'),
   ('Компьютер: Dell XPS', 'Дисплейный класс: Advanced Display Technology'),
   ('Компьютер: Lenovo ThinkPad',
    'Дисплейный класс: Advanced Display Technology'),
+  ('Компьютер: Asus ROG', 'Дисплейный класс: Advanced Display Tech')]
-  ('Компьютер: Apple MacBook', 'Дисплейный класс: Basic Display Package'),
-  ('Компьютер: Lenovo ThinkPad', 'Дисплейный класс: Basic Display Package')]

----------------------------------------------------------------------
Ran 3 tests in 0.001s
```