Московский государственный технический университет им. Н.Э. Баумана

Факультет информатика и системы управления
Кафедра системы обработки информации и управления

Курс «Парадигмы и конструкции языков программирования»
Отчет по рубежному контролю №2
Вариант Б17

Выполнил:
студент группы ИУ5-32Б:
Шпакова Д. Л.
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Москва, 2025 г.

Постановка задачи

Рубежный контроль представляет собой разработку тестов на языке Python.

роведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.
ля текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

Текст программы

`main.py`

```
import unittest
from operator import itemgetter

class Conductor:
    """Дирижер"""
    def __init__(self, id, fio, salary, orchestra_id):
        self.id = id
        self.fio = fio
        self.salary = salary
        self.orchestra_id = orchestra_id

class Orchestra:
    """Оркестр"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

class ConductorOrchestra:
    """Связь многие-ко-многим"""
    def __init__(self, conductor_id, orchestra_id):
        self.conductor_id = conductor_id
        self.orchestra_id = orchestra_id

def solve_b1(conductors, orchestras):
    """Запрос Б1: Список связанных дирижеров и оркестров, отсортированный по
ФИО"""
    one_to_many = [(c.fio, c.salary, o.name)
                   for o in orchestras
                   for c in conductors
                   if c.orchestra_id == o.id]
    return sorted(one_to_many, key=itemgetter(0))

def solve_b2(conductors, orchestras):
    """Запрос Б2: Список оркестров с количеством дирижеров, отсортированный по
количеству"""
    one_to_many = [(c.fio, c.salary, o.name)
                   for o in orchestras
                   for c in conductors
                   if c.orchestra_id == o.id]
    res_unsorted = []
    for o in orchestras:
        o_conds = list(filter(lambda i: i[2] == o.name, one_to_many))
        if len(o_conds) > 0:
            res_unsorted.append((o.name, len(o_conds)))
    return sorted(res_unsorted, key=itemgetter(1), reverse=True)

def solve_b3(conductors, orchestras, conductors_orchestras):
```

```python
    """Запрос Б3: Дирижеры с фамилией на «ов» и их оркестры (многие-ко-многим)"""
    many_to_many_temp = [(o.name, co.orchestra_id, co.conductor_id)
                         for o in orchestras
                         for co in conductors_orchestras
                         if o.id == co.orchestra_id]

    many_to_many = [(c.fio, orch_name)
                    for orch_name, orch_id, cond_id in many_to_many_temp
                    for c in conductors if c.id == cond_id]

    return [(fio, orch_name) for fio, orch_name in many_to_many if
fio.endswith('ов')]

if __name__ == '__main__':
    # 1. Подготовка тестовых данных (из вашего отчета)
    orchestras = [
        Orchestra(1, 'Венский филармонический'),
        Orchestra(2, 'Берлинский филармонический'),
        Orchestra(3, 'Лондонский симфонический'),
        Orchestra(11, 'Нью-Йоркский филармонический'),
        Orchestra(22, 'Бостонский симфонический')
    ]

    conductors = [
        Conductor(1, 'Караян', 100000, 2),
        Conductor(2, 'Бернстайн', 120000, 1),
        Conductor(3, 'Мути', 90000, 1),
        Conductor(4, 'Петров', 80000, 3),
        Conductor(5, 'Иванов', 75000, 3)
    ]

    conductors_orchestras = [
        ConductorOrchestra(1, 1),
        ConductorOrchestra(1, 2),
        ConductorOrchestra(2, 11),
        ConductorOrchestra(3, 1),
        ConductorOrchestra(4, 3),
        ConductorOrchestra(4, 22),
        ConductorOrchestra(5, 3),
        ConductorOrchestra(5, 11)
    ]

    # 2. Выполнение и вывод запросов
    print('Задание Б1')
    print(solve_b1(conductors, orchestras))

    print('\nЗадание Б2')
    print(solve_b2(conductors, orchestras))

    print('\nЗадание Б3')
    print(solve_b3(conductors, orchestras, conductors_orchestras))
```

`tests.py`

```
import unittest
from main import Conductor, Orchestra, ConductorOrchestra, solve_b1, solve_b2,
solve_b3

class TestOrchestraLogic(unittest.TestCase):

    def setUp(self):
        """Подготовка тестовых данных"""
        self.orchestras = [
            Orchestra(1, 'Венский филармонический'),
            Orchestra(2, 'Берлинский филармонический'),
            Orchestra(3, 'Лондонский симфонический')
        ]
        self.conductors = [
            Conductor(1, 'Караян', 100000, 2),
            Conductor(2, 'Бернстайн', 120000, 1),
            Conductor(3, 'Петров', 80000, 3),
            Conductor(4, 'Иванов', 75000, 3)
        ]
        self.links = [
            ConductorOrchestra(1, 1),
            ConductorOrchestra(1, 2),
            ConductorOrchestra(3, 3)
        ]

    def test_solve_b1_sorting(self):
        """Тест 1: Проверка сортировки по ФИО в запросе Б1"""
        result = solve_b1(self.conductors, self.orchestras)
        # Проверяем, что первый по алфавиту — Бернстайн
        self.assertEqual(result[0][0], 'Бернстайн')
        # Проверяем, что последний — Петров
        self.assertEqual(result[-1][0], 'Петров')

    def test_solve_b2_counts(self):
        """Тест 2: Проверка подсчета дирижеров в оркестрах в запросе Б2"""
        result = solve_b2(self.conductors, self.orchestras)
        # В Лондонском (id 3) два дирижера (Петров, Иванов)
        # Он должен быть первым в списке из-за reverse=True
        self.assertEqual(result[0], ('Лондонский симфонический', 2))

    def test_solve_b3_filter(self):
        """Тест 3: Проверка фильтрации фамилий на 'ов' в запросе Б3"""
        result = solve_b3(self.conductors, self.orchestras, self.links)
        # Из списка links и conductors на 'ов' заканчиваются только Петров и
Иванов
        # Караян и Бернстайн должны быть отфильтрованы
        fios = [item[0] for item in result]
        for name in fios:
            self.assertTrue(name.endswith('ов'))
```

```
        self.assertIn('Петров', fios)
        self.assertNotIn('Караян', fios)

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

Скриншот работы приложения

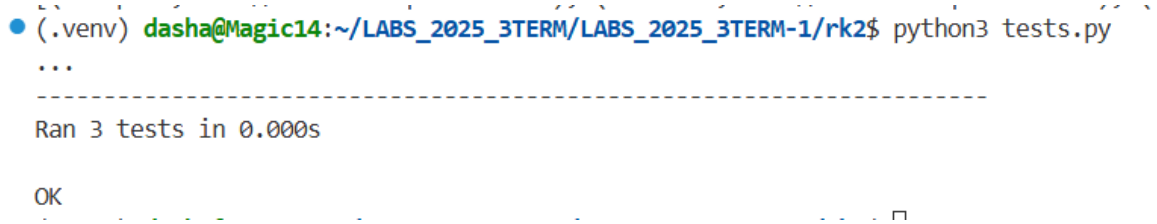

Рис. 1 Вывод программы