

Московский государственный технический университет

МГТУ им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет технологий»

Отчет по Рубежному контролю №2

Выполнил:

студент группы ИУ5-34Б

Григорян А.А.

Москва, 2021г.

Постановка задачи:

Рубежный контроль представляет собой разработку тестов на языке Python.

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

Текст программы:

main.py

```
import operator

class Mus:
    def __init__(self, id, fio, sal, orch_id):
        self.id = id
        self.fio = fio
        self.sal = sal
        self.orch_id = orch_id

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

class Mus_Orch:
    def __init__(self, mus_id, orch_id):
        self.mus_id = mus_id
        self.orch_id = orch_id

orch = [
    Orch(1, "Московский симфонический оркестр"),
    Orch(2, "Симфонический оркестр Большого театра"),
    Orch(3, "Оркестр имени П.И.Чайковского"),
    Orch(4, "Государственный оркестр России"),
    Orch(5, "Академический оркестр Московской филармонии"),
]

mus = [
    Mus(1, "Иванов", 67000, 5),
    Mus(2, "Дмитриев", 55000, 1),
    Mus(3, "Васильев", 89000, 3),
    Mus(4, "Герасимов", 64000, 4),
    Mus(5, "Зайцев", 95000, 4),
]

mus_orch = [
    Mus_Orch(1, 5),
    Mus_Orch(2, 1),
    Mus_Orch(3, 3),
    Mus_Orch(4, 4),
    Mus_Orch(5, 4)
]
```

```

# соединение данных один-ко-многим
one_to_many = [(o.name, m.fio, m.sal)
                for o in orch
                for m in mus
                if m.orc_id == o.id
                ]

# соединение данных многие-ко-многим
many_to_many_temp = [(o.name, m_o.orc_id, m_o.mus_id)
                      for o in orch
                      for m_o in mus_orch
                      if o.id == m_o.orc_id]

many_to_many = [(m.fio, m.sal, orch_name)
                 for orch_name, orch_id, mus_id in many_to_many_temp
                 for m in mus if m.id == mus_id]

def func_task1():
    res1 = []
    for o in orch:
        if o.name[0] == 'A':
            # список музыкантов оркестра
            orch_mus = list(filter(lambda i: i[0] == o.name, one_to_many))
            if len(orch_mus) > 0:
                # имена музыкантов оркестра
                names_mus = [name for _, name, _ in orch_mus]
                res1.append((o.name, names_mus))
    """
    print("Задание Г1: ")
    print(res1)
    """
    return res1

def func_task2():
    # список оркестров, названия которых начинаются с буквы 'A'
    res2_unsorted = []
    # зарплаты
    sal = []
    for o in orch:
        # список музыкантов оркестра
        orch_mus = list(filter(lambda i: i[0] == o.name, one_to_many))
        if len(orch_mus) > 0:
            # зарплаты музыкантов оркестра
            sal = [sal for _, sal in orch_mus]
            # максимальная зарплата
            max_sal = max(sal)
            res2_unsorted.append((o.name, max_sal))
    res2_sorted = sorted(res2_unsorted, key=operator.itemgetter(1), reverse=True)
    """
    print("Задание Г2: ")
    print(res2_sorted)
    """
    return res2_sorted

def func_task3():
    res3_unsorted = []
    for o in orch:
        # список музыкантов оркестра

```

```

orch_mus = list(filter(lambda i: i[2] == o.name, many_to_many))
if len(orch_mus) > 0:
    fio_mus_unsorted = [fio for fio, _, _ in orch_mus]
    # отсортированный список фамилий музыкантов
    fio_mus_sorted = sorted(fio_mus_unsorted, key=operator.itemgetter(0))
    res3_unsorted.append((o.name, fio_mus_sorted))
res3_sorted = sorted(res3_unsorted, key=operator.itemgetter(0), reverse=True)
"""
print("Задание Г3: ")
print(res3_sorted)
"""
return res3_sorted

```

rk2.py

```

import unittest
from main import *

class TestClass1(unittest.TestCase):
    def test_function1(self):
        expected_result = [('Академический оркестр Московской филармонии', ['Иванов'])]
        self.assertEqual(func_task1(), expected_result)

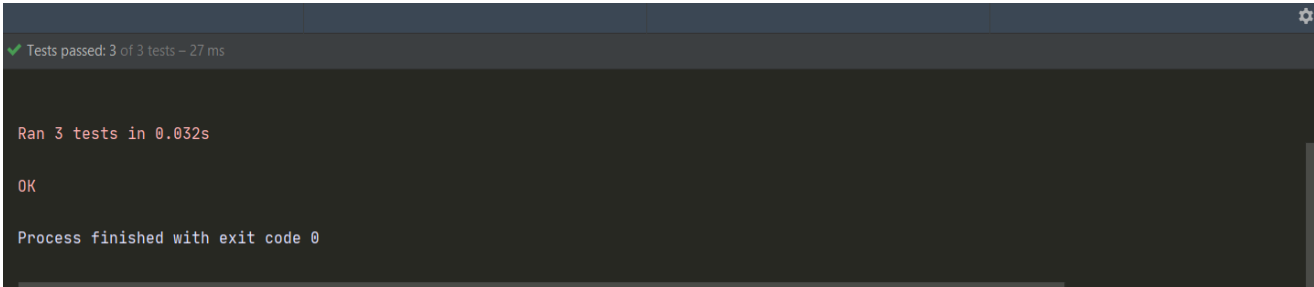
class TestClass2(unittest.TestCase):
    def test_function(self):
        expected_result = [('Государственный оркестр России', 95000), ('Оркестр имени П.И.Чайковского', 89000), ('Академический оркестр Московской филармонии', 67000), ('Московский симфонический оркестр', 55000)]
        self.assertEqual(func_task2(), expected_result)

class TestCase3(unittest.TestCase):
    def test_function(self):
        expected_result = [('Оркестр имени П.И.Чайковского', ['Васильев']), ('Московский симфонический оркестр', ['Дмитриев']), ('Государственный оркестр России', ['Герасимов', 'Зайцев']), ('Академический оркестр Московской филармонии', ['Иванов'])]
        self.assertEqual(func_task3(), expected_result)

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

```

Результат выполнения:



```

Tests passed: 3 of 3 tests - 27 ms

Ran 3 tests in 0.032s

OK

Process finished with exit code 0

```