Студент: Поляков Леонид

Группа: ИУ5-32Б

Рубежный контроль №2

Вариант 18

Условие:

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).

Решение:

main.py

```
from operator import itemgetter


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


class MusicalComposition:
    def __init__(self, id, name, author, orchestra_id):
        self.id = id
        self.name = name
        self.author = author
        self.orchestra_id = orchestra_id


class MusicalCompositionOrchestra:
    def __init__(self, orchestra_id, composition_id):
        self.orchestra_id = orchestra_id
        self.composition_id = composition_id


orchestras = [
    Orchestra(1, "Leningrad Philharmonic"),
    Orchestra(2, "Benzogang Orchestra"),
    Orchestra(3, "Budapest Festival Orchestra"),
]
```

```
musicalCompositions = [
    MusicalComposition(1, "Morning", "Edward Grieg", 1),
    MusicalComposition(2, "Moonlight Sonata", "Ludwig van Beethoven", 2),
    MusicalComposition(3, "Saber Dance", "Aram Khachaturian", 3),
    MusicalComposition(4, "Summer Storm", "Antonio Vivaldi ", 3),
    MusicalComposition(5, "Fly", "Ludovico Einaudi", 1),
    MusicalComposition(6, "I'm swimming in the river", "Ivan Dremin", 1)
]

composition_to_orchestra = [
    MusicalCompositionOrchestra(1, 1),
    MusicalCompositionOrchestra(2, 2),
    MusicalCompositionOrchestra(3, 3),
    MusicalCompositionOrchestra(3, 4),
    MusicalCompositionOrchestra(1, 5),
    MusicalCompositionOrchestra(1, 6),

]


def first_task(composition_list):
    return sorted(composition_list, key=itemgetter(0))


def second_task(composition_list):
    res = []
    temp = dict()
    for i in composition_list:
        if i[2] in temp:
            temp[i[2]] += 1
        else:
            temp[i[2]] = 1
    for i in temp.keys():
        res.append((i, temp[i]))

    res.sort(key=itemgetter(1), reverse=True)
    return res


def third_task(composition_list, suffix):
    res = [(i[0], i[2]) for i in composition_list if str(i[1]).endswith(suffix)]
    return res


def main():
    one_to_many = [(composition.name, composition.author, orchestra.name)
                   for orchestra in orchestras
                   for composition in musicalCompositions
                   if composition.orchestra_id == orchestra.id]
```

```
    many_to_many_temp = [(orchestra.name, relation.orchestra_id,
relation.composition_id)
                         for orchestra in orchestras
                         for relation in composition_to_orchestra
                         if relation.orchestra_id == orchestra.id]

    many_to_many = [(composition.name, composition.author, orchestra_name)
                    for orchestra_name, orchestra_id, composition_id in
many_to_many_temp
                    for composition in musicalCompositions if composition.id ==
composition_id]

    print("\nSelection Б1:")
    print(first_task(one_to_many))
    print("\n")

    print("\nSelection Б2")
    print(second_task(one_to_many))
    print("\n")


    print("\nSelection Б3")
    print(third_task(many_to_many, 'n'))
    print("\n")



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

main_test.py

```
import main
from operator import itemgetter
import unittest


class TestMainMethods(unittest.TestCase):
    def test_first_task(self):
        test_data = [
            ('c', 'Ludovico Einaudi', 'Leningrad Philharmonic'),
            ("f", 'Ivan Dremin', 'Leningrad Philharmonic'),
            ('b', 'Ludwig van Beethoven', 'Benzogang Orchestra'),
            ('d', 'Edward Grieg', 'Leningrad Philharmonic'),
            ('e', 'Aram Khachaturian', 'Budapest Festival Orchestra'),
            ('a', 'Antonio Vivaldi ', 'Budapest Festival Orchestra')
        ]
        result = main.first_task(test_data)
        expected = sorted(test_data, key=itemgetter(0))
        self.assertEqual(expected, result)
```

```
    def test_second_task(self):
        test_data = [
            ('Fly', 'Ludovico Einaudi', 'Leningrad Philharmonic'),
            ("I'm swimming in the river", 'Ivan Dremin', 'Leningrad
Philharmonic'),
            ('Moonlight Sonata', 'Ludwig van Beethoven', 'Benzogang Orchestra'),
            ('Morning', 'Edward Grieg', 'Leningrad Philharmonic'),
            ('Saber Dance', 'Aram Khachaturian', 'Budapest Festival Orchestra'),
            ('Summer Storm', 'Antonio Vivaldi ', 'Budapest Festival Orchestra')
        ]
        result = main.second_task(test_data)
        expected = [('Leningrad Philharmonic', 3), ('Budapest Festival
Orchestra', 2), ('Benzogang Orchestra', 1)]
        self.assertEqual(expected, result)

    def test_third_task(self):
        test_data = [
            ('Fly', 'Ludovico Einaudi', 'Leningrad Philharmonic'),
            ("I'm swimming in the river", 'Ivan Dremin', 'Leningrad
Philharmonic'),
            ("I'm crazy", 'Ivan Dremin', 'Unknown Orchestra'),
            ('Moonlight Sonata', 'Ludwig van Beethoven', 'Benzogang Orchestra'),
            ('Morning', 'Edward Grieg', 'Leningrad Philharmonic'),
            ('Saber Dance', 'Aram Khachaturian', 'Budapest Festival Orchestra'),
            ('Summer Storm', 'Antonio Vivaldi ', 'Budapest Festival Orchestra')
        ]
        result = main.third_task(test_data, 'n')
        expected = [
            ("I'm swimming in the river", 'Leningrad Philharmonic'),
            ("I'm crazy", 'Unknown Orchestra'),
            ('Moonlight Sonata', 'Benzogang Orchestra'),
            ('Saber Dance', 'Budapest Festival Orchestra')
        ]
        self.assertEqual(expected, result)
```

Результат:

```
PS D:\BMSTU\Study\Course 2\PCPL\RC1-2> python -m unittest -v .\main_test.py
test_first_task (main_test.TestMainMethods.test_first_task) ... ok
test_second_task (main_test.TestMainMethods.test_second_task) ... ok
test_third_task (main_test.TestMainMethods.test_third_task) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.002s

OK
PS D:\BMSTU\Study\Course 2\PCPL\RC1-2>
```