Мироненков Арсений
ИУ5-36Б

РК2

1)Рефакторинг текста программы

Текст программы:

```
from operator import itemgetter
class MusicalPiece:
    """Музыкальное произведение"""
    def __init__(self, piece_id, title, composer, orchestra_id):
        self.piece_id = piece_id
        self.title = title
        self.composer = composer
        self.orchestra_id = orchestra_id
class Orchestra:
    """Оркестр"""
    def __init__(self, orchestra_id, name):
        self.orchestra_id = orchestra_id
        self.name = name
class Performers:
    """Исполнители произведения"""
    def __init__(self, piece_id, orchestra_id, performers_count):
        self.piece_id = piece_id
        self.orchestra_id = orchestra_id
        self.performers_count = performers_count
# Функции для обработки данных
def get_pieces_with_title_ending_a(musical_pieces, orchestras):
    """Возвращает список произведений с заг
лавием, заканчивающимся на 'а'."""
    return sorted(
        [
            (piece.title, next(
                (orch.name for orch in orchestras if orch.orchestra_id ==
piece.orchestra_id), None
            ))
            for piece in musical_pieces if piece.title.endswith('а')
        ],
        key=itemgetter(1)
    )
def get_orchestra_avg_performers(performers, orchestras):
    """Возвращает список оркестров и их сред
ней численности исполнителей."""
    orchestra_performers_count = {}
    for performer in performers:
        orchestra_performers_count.setdefault(performer.orchestra_id,
```

```
[]).append(performer.performers_count)
    return sorted(
        {
            orch.name: sum(counts) / len(counts)
            for orch in orchestras
            if (counts := orchestra_performers_count.get(orch.orchestra_id))
        }.items(),
        key=itemgetter(1), reverse=True
    )
def get_orchestras_starting_with_G(orchestras, musical_pieces):
    """Возвращает список оркестров, начинающихся на 'Г', и исполняемых ими произведений."""
    return {
        orch.name: [piece.title for piece in musical_pieces if piece.orchestra_id == orch.orchestra_id]
        for orch in orchestras if orch.name.startswith("Г")
    }
def main():
    orchestras = [
        Orchestra(1, "Филармонический оркестр"),
        Orchestra(2, "Государственный оркестр"),
        Orchestra(3, "Молодежный оркестр"),
        Orchestra(4, "Главный оркестр Москвы"),
    ]
    musical_pieces = [
        MusicalPiece(1, "Симфония №5", "Бетховен", 1),
        MusicalPiece(2, "Концерт для фортепиано", "Григ", 1),
        MusicalPiece(3, "Симфония №9", "Дворжак", 2),
        MusicalPiece(4, "Рапсодия на тему Паганини", "Рахманинов", 1),
        MusicalPiece(5, "Импровизация", "Шостакович", 3),
        MusicalPiece(6, "Императорский марш для оркестра", "Рихард Вагнер", 4),
    ]
    performers = [
        Performers(1, 1, 25),
        Performers(2, 1, 20),
        Performers(3, 2, 30),
        Performers(4, 1, 22),
        Performers(5, 3, 15),
        Performers(6, 4, 45),
```

```
    ]
    print('Задание 1:', get_pieces_with_title_ending_a(musical_pieces,
orchestras))
    print('Задание 2:', get_orchestra_avg_performers(performers,
orchestras))
    print('Задание 3:', get_orchestras_starting_with_G(orchestras,
musical_pieces))
if __name__ == '__main__':
    main()
```

Результат:

```
PS C:\Users\DNS\PycharmProjects\RK2> python test.py
Задание 1: [('Императорский марш для оркестра', 'Главный оркестр Москвы')]
Задание 2: [('Главный оркестр Москвы', 45.0), ('Государственный оркестр', 30.0), ('Филармонический оркестр', 22.333333333333332), ('Молодежный оркестр', 15.0)]
Задание 3: {'Государственный оркестр': ['Симфония №9'], 'Главный оркестр Москвы': ['Императорский марш для оркестра']}
```

2) Модульные тесты с применением TDD - фреймворка (3 теста)

```
from operator import itemgetter
import unittest
class MusicalPiece:
    """Музыкальное произведение"""
    def __init__(self, piece_id, title, composer, orchestra_id):
        self.piece_id = piece_id
        self.title = title
        self.composer = composer
        self.orchestra_id = orchestra_id
class Orchestra:
    """Оркестр"""
    def __init__(self, orchestra_id, name):
        self.orchestra_id = orchestra_id
        self.name = name
class Performers:
    """Исполнители произведения"""
    def __init__(self, piece_id, orchestra_id, performers_count):
        self.piece_id = piece_id
        self.orchestra_id = orchestra_id
        self.performers_count = performers_count
# Functions for data processing
def get_pieces_with_title_ending_a(musical_pieces, orchestras):
    """Returns a list of pieces with titles ending in 'a'."""
    return sorted(
        [
            (piece.title, next(
                (orch.name for orch in orchestras if orch.orchestra_id ==
piece.orchestra_id), None
            ))
```

```
            for piece in musical_pieces if piece.title.endswith('а')
        ],
        key=itemgetter(1)
    )
def get_orchestra_avg_performers(performers, orchestras):
    """Returns a list of orchestras with their average performer count."""
    orchestra_performers_count = {}
    for performer in performers:
        orchestra_performers_count.setdefault(performer.orchestra_id,
[]).append(performer.performers_count)
    return sorted(
        {
            orch.name: sum(counts) / len(counts)
            for orch in orchestras
            if (counts := orchestra_performers_count.get(orch.orchestra_id))
        }.items(),
        key=itemgetter(1), reverse=True
    )
def get_orchestras_starting_with_G(orchestras, musical_pieces):
    """Returns orchestras starting with 'Г' and their performed pieces."""
    return {
        orch.name: [piece.title for piece in musical_pieces if
piece.orchestra_id == orch.orchestra_id]
        for orch in orchestras if orch.name.startswith("Г")
    }
class TestMusicFunctions(unittest.TestCase):
    def setUp(self):
        self.orchestras = [
            Orchestra(1, "Филармонический оркестр"),
            Orchestra(2, "Государственный оркестр"),
            Orchestra(3, "Молодежный оркестр"),
            Orchestra(4, "Главный оркестр Москвы"),
        ]
        self.musical_pieces = [
            MusicalPiece(1, "Симфония №5", "Бетховен", 1),
            MusicalPiece(2, "Концерт для фортепиано", "
Григ", 1),
            MusicalPiece(3, "Симфония №9", "Дворжак", 2),
            MusicalPiece(4, "Рапсодия на тему Паганини
", "Рахманинов", 1),
            MusicalPiece(5, "Импровизация", "Шостакови
ч", 3),
            MusicalPiece(6, "Императорский марш для ор
кестра", "Рихард Вагнер", 4),
```

```
        ]
        self.performers = [
            Performers(1, 1, 25),
            Performers(2, 1, 20),
            Performers(3, 2, 30),
            Performers(4, 1, 22),
            Performers(5, 3, 15),
            Performers(6, 4, 45),
        ]
    def test_get_pieces_with_title_ending_a(self):
        expected = [
            ('Императорский марш для оркестра',
'Главный оркестр Москвы'),
        ]
        result = get_pieces_with_title_ending_a(self.musical_pieces,
self.orchestras)
        self.assertEqual(result, expected)
    def test_get_orchestra_avg_performers(self):
        expected = [
            ('Главный оркестр Москвы', 45.0),
            ('Государственный оркестр', 30.0),
            ('Филармонический оркестр',
22.333333333333332),
            ('Молодежный оркестр', 15.0),
        ]
        result = get_orchestra_avg_performers(self.performers,
self.orchestras)
        self.assertEqual(result, expected)
    def test_get_orchestras_starting_with_G(self):
        expected = {
            'Государственный оркестр': ['Симфон
ия №9'],
            'Главный оркестр Москвы': ['Императо
рский марш для оркестра']
        }
        result = get_orchestras_starting_with_G(self.orchestras,
self.musical_pieces)
        self.assertEqual(result, expected)
def main():
    orchestras = [
        Orchestra(1, "Филармонический оркестр"),
        Orchestra(2, "Государственный оркестр"),
        Orchestra(3, "Молодежный оркестр"),
        Orchestra(4, "Главный оркестр Москвы"),
```

```
    ]
    musical_pieces = [
        MusicalPiece(1, "Симфония №5", "Бетховен", 1),
        MusicalPiece(2, "Концерт для фортепиано", "Григ", 1),
        MusicalPiece(3, "Симфония №9", "Дворжак", 2),
        MusicalPiece(4, "Рапсодия на тему Паганини", "Рахманинов", 1),
        MusicalPiece(5, "Импровизация", "Шостакович", 3),
        MusicalPiece(6, "Императорский марш для оркестра", "Рихард Вагнер", 4),
    ]
    performers = [
        Performers(1, 1, 25),
        Performers(2, 1, 20),
        Performers(3, 2, 30),
        Performers(4, 1, 22),
        Performers(5, 3, 15),
        Performers(6, 4, 45),
    ]
    print('Задание 1:', get_pieces_with_title_ending_a(musical_pieces, orchestras))
    print('Задание 2:', get_orchestra_avg_performers(performers, orchestras))
    print('Задание 3:', get_orchestras_starting_with_G(orchestras, musical_pieces))
if __name__ == '__main__':
    main()
    unittest.main()
```

Результат:

```
PS C:\Users\DNS\PycharmProjects\RK2> python main.py
Задание 1: [('Императорский марш для оркестра', 'Главный оркестр Москвы')]
Задание 2: [('Главный оркестр Москвы', 45.0), ('Государственный оркестр', 30.0), ('Филармонический оркестр', 22.333333333333332), ('Молодежный оркестр', 15.0)]
Задание 3: {'Государственный оркестр': ['Симфония №9'], 'Главный оркестр Москвы': ['Императорский марш для оркестра']}
...
----------------------------------------------------------------------
Ran 3 tests in 0.000s

OK
```