Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчет по рубежному контролю №2

Выполнил:
студент группы ИУ5-31Б
Койбаев Тамерлан
Подпись и дата:

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

Москва, 2022 г.

Условия рубежного контроля №2 по курсу БКИТ

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.

Измененный код РК №1:

```
# используется для сортировки
from operator import itemgetter


class Musician:
    """Музыкант"""

    def __init__(self, id, name, age, orch_id):
        self.id = id
        self.name = name
        self.age = age
        self.orch_id = orch_id


class Orchestra:
    """Оркестр"""

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


class MusOrch:
    """
    'Музыкант в оркестре' для реализации
    связи многие-ко-многим
    """

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


# Оркестры
orch = [
    Orchestra(1, 'оркестр 1'),
    Orchestra(2, 'оркестр 2'),
    Orchestra(3, 'оркестр 3'),
]

# Музыканты
mus = [
    Musician(1, 'Холмогоров', 25, 1),
    Musician(2, 'Пчелкин', 31, 1),
    Musician(3, 'Белый', 19, 2),
    Musician(4, 'Филатов', 44, 2),
    Musician(5, 'Лапшин', 50, 3),
    Musician(6, 'Шмидт', 22, 4)
```

```
]

mus_orch = [
    MusOrch(1, 1),
    MusOrch(1, 2),
    MusOrch(2, 3),
    MusOrch(2, 4),
    MusOrch(3, 5),
    MusOrch(3, 6)
]

def  one_to_many(orch,mus):
      return [(c.name, c.age, r.name)
                  for r in orch
                  for c in mus
                  if c.orch_id == r.id]

def many_to_many(orch,mus):
    many_to_many_temp = [(r.name, cr.orch_id, cr.mus_id)
                           for r in orch
                           for cr in mus_orch
                           if r.id == cr.orch_id]
    return [(c.name, c.age, orch_name)
                   for orch_name, orch_id, mus_id in many_to_many_temp
                   for c in mus if c.id == mus_id]

def A1(orch,mus) ->list:
    res_11 = sorted(one_to_many(orch,mus), key=itemgetter(2))
    return res_11

def A2(orch,mus) ->list:
    res_12_unsorted = []
    # Перебираем все оркестры
    for r in orch:
        # Список музыкантов оркестра
        r_orch = list(filter(lambda i: i[2] == r.name,
one_to_many(orch,mus)))
        # Если музыкантов > 0
        if len(r_orch) > 0:
            # Возраст музыкантов в оркестре
            r_age = [age for _, age, _ in r_orch]
            # Суммарный возраст
            r_age_sum = sum(r_age)
            res_12_unsorted.append((r.name, r_age_sum))

    # Сортировка по суммарному возрасту
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    return res_12
def A3(orch,mus,str) ->list:
    res_13 = {}
    # Перебираем все оркестры
    for r in orch:
        if str not in r.name:
            # Список музыкантов оркестра
            r_orch = list(filter(lambda i: i[2] == r.name,
many_to_many(orch,mus)))
            # Только имена
            r_orch_names = [x for x, _, _ in r_orch]
            # Добавляем результат в словарь
            # ключ - оркестр, значение - имена
            res_13[r.name] = r_orch_names

    return(res_13)
```

```
if __name__ == '__main__':
    print(A1(orch,mus))
    print(A2(orch, mus))
    print(A3(orch, mus))
```

Тестирование:

```
import unittest
from rk1re import Musician, Orchestra, MusOrch, A1, A2, A3


class test(unittest.TestCase):

    def setUp(self):
        self.orch = [
    Orchestra(1, 'оркестр 1'),
    Orchestra(2, 'оркестр 2'),
    Orchestra(3, 'оркестр 3'),
]
        self.mus = [
    Musician(1, 'Холмогоров', 25, 1),
    Musician(2, 'Пчелкин', 31, 1),
    Musician(3, 'Белый', 19, 2),
    Musician(4, 'Филатов', 44, 2),
    Musician(5, 'Лапшин', 50, 3),
    Musician(6, 'Шмидт', 22, 4)
]
        self.mus_orch = [
    MusOrch(1, 1),
    MusOrch(1, 2),
    MusOrch(2, 3),
    MusOrch(2, 4),
    MusOrch(3, 5),
    MusOrch(3, 6)
]

    def test_A1(self):
        expected_result = [('Холмогоров', 25, 'оркестр 1'),
                           ('Пчелкин', 31, 'оркестр 1'),
                           ('Белый', 19, 'оркестр 2'),
                           ('Филатов', 44, 'оркестр 2'),
                           ('Лапшин', 50, 'оркестр 3')]

        result = A1(self.orch, self.mus)
        self.assertEqual(result, expected_result)

    def test_A2(self):
        expected_result = [('оркестр 2', 63),
                           ('оркестр 1', 56),
                           ('оркестр 3', 50)]
        result = A2(self.orch, self.mus)
        self.assertEqual(result, expected_result)

    def test_A3(self):
        expected_result = {'оркестр 2': ['Белый', 'Филатов'], 'оркестр 3':
['Лапшин', 'Шмидт']}
        result = A3(self.orch, self.mus,'1')
        self.assertEqual(result, expected_result)


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

Результаты тестирования:

```
Testing started at 18:43 ...
Launching pytest with arguments C:\Users\Тамерлан\PycharmProjects\rk2\rk2.py --no-h

============================= test session starts =============================
collecting ... collected 3 items

rk2.py::test::test_A1 PASSED                                            [ 33%]
rk2.py::test::test_A2 PASSED                                            [ 66%]
rk2.py::test::test_A3 PASSED                                            [100%]

============================== 3 passed in 0.02s ==============================
```