Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Рубежный контроль №2

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:

студент группы ИУ5-35Б

Емельянова Т.И.

2021

Задание РК1

Вариант Д-5

Вариант Д.

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их отделов.

2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате (отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений).

3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А», и список работающих в них сотрудников.

Вариант предметной области 5.

Музыкант-Оркестр

В соответствии с предметной областью, задание было немного изменено:

1. «Оркестр» и «Музыкант» связаны соотношением один-ко-многим. Выведите список всех музыкантов, у которых фамилия заканчивается на «ич», и названия их оркестров.

2. «Оркестр» и «Музыкант» связаны соотношением один-ко-многим. Выведите список оркестров со средней зарплатой музыкантов в каждом отделе, отсортированный по средней зарплате (*отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений*).
3. «Оркестр» и «Музыкант» связаны соотношением многие-ко-многим. Выведите список всех оркестров, у которых название начинается с буквы «С», и список состоящих в них музыкантов.

Задание РК2

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).

main.py

```
from operator import itemgetter


class Musician:
    def __init__(self, element_id, s_name, wage, orchestra_id):
        self.id = element_id
        self.s_name = s_name
        self.wage = wage
        self.orchestra_id = orchestra_id


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


class MusicianOrchestra:
    def __init__(self, orchestra_id, musician_id):
        self.orchestra_id = orchestra_id
        self.musician_id = musician_id


musicians = [
    Musician(1, "Милевич", 50000, 1),
    Musician(2, "Симонович", 30000, 1),
    Musician(3, "Селиванова", 60000, 2),
    Musician(4, "Майоров", 70000, 2),
    Musician(5, "Кириевич", 80000, 3)
]

orchestras = [
    Orchestra(1, "Симфонический"),
    Orchestra(2, "Духовой"),
    Orchestra(3, "Струнный"),
    Orchestra(11, "Военный"),
    Orchestra(22, "Джазовый"),
    Orchestra(33, "Симфонический новый")
]
```

```
mus_orchestras = [
    MusicianOrchestra(1, 1),
    MusicianOrchestra(1, 2),
    MusicianOrchestra(2, 3),
    MusicianOrchestra(2, 4),
    MusicianOrchestra(3, 5),
    MusicianOrchestra(11, 1),
    MusicianOrchestra(11, 2),
    MusicianOrchestra(22, 3),
    MusicianOrchestra(22, 4),
    MusicianOrchestra(33, 5)
]


def part_one(one_to_many):
    res_11 = list(filter(lambda x: x[0].endswith("ич"), one_to_many))
    return res_11


def part_two(one_to_many):
    res_12_unsorted = []
    for o in orchestras:
        mus = list(filter(lambda i: i[2] == o.name, one_to_many))
        if len(mus) > 0:
            wages = [wage for _, wage, _ in mus]
            wage_ave = round(sum(wages) / len(wages), 2)
            res_12_unsorted.append((o.name, wage_ave))
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    return res_12


def part_three(many_to_many):
    res_13 = {}
    for o in orchestras:
        if o.name.startswith("С"):
            mus = list(filter(lambda i: i[2] == o.name, many_to_many))
            mus_s_names = [s_name for s_name, _, _ in mus]
            res_13[o.name] = mus_s_names
    return res_13


def main():
    one_to_many = [(m.s_name, m.wage, o.name)
                   for o in orchestras
                   for m in musicians
                   if m.orchestra_id == o.id]
    many_to_many_temp = [(o.name, mo.orchestra_id, mo.musician_id)
                         for o in orchestras
                         for mo in mus_orchestras
                         if o.id == mo.orchestra_id]
    many_to_many = [(m.s_name, m.wage, o_name)
                    for o_name, o_id, m_id in many_to_many_temp
                    for m in musicians if m.id == m_id]

    print('Задание Д1')
    print(part_one(one_to_many))

    print('\nЗадание Д2')
    print(part_two(one_to_many))

    print('\nЗадание Д3')
    print(part_three(many_to_many))


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

tests.py

```
import unittest
from main import Musician, Orchestra, MusicianOrchestra, part_one, part_two,
part_three


class Parts(unittest.TestCase):
    def setUp(self):
        self.musicians = [
            Musician(1, "Милевич", 50000, 1),
            Musician(2, "Симонович", 30000, 1),
            Musician(3, "Селиванова", 60000, 2),
            Musician(4, "Майоров", 70000, 2),
            Musician(5, "Кириевич", 80000, 3)
        ]
        self.orchestras = [
            Orchestra(1, "Симфонический"),
            Orchestra(2, "Духовой"),
            Orchestra(3, "Струнный"),
            Orchestra(11, "Военный"),
            Orchestra(22, "Джазовый"),
            Orchestra(33, "Симфонический новый")
        ]
        self.mus_orchestras = [
            MusicianOrchestra(1, 1),
            MusicianOrchestra(1, 2),
            MusicianOrchestra(2, 3),
            MusicianOrchestra(2, 4),
            MusicianOrchestra(3, 5),
            MusicianOrchestra(11, 1),
            MusicianOrchestra(11, 2),
            MusicianOrchestra(22, 3),
            MusicianOrchestra(22, 4),
            MusicianOrchestra(33, 5)
        ]
        self.one_to_many = [(m.s_name, m.wage, o.name)
                            for o in self.orchestras
                            for m in self.musicians
                            if m.orchestra_id == o.id]
        self.many_to_many_temp = [(o.name, mo.orchestra_id, mo.musician_id)
                                   for o in self.orchestras
                                   for mo in self.mus_orchestras
                                   if o.id == mo.orchestra_id]
        self.many_to_many = [(m.s_name, m.wage, o_name)
                             for o_name, o_id, m_id in self.many_to_many_temp
                             for m in self.musicians if m.id == m_id]

    def test_part_one(self):
        answer1 = [('Милевич', 50000, 'Симфонический'),
                   ('Симонович', 30000, 'Симфонический'),
                   ('Кириевич', 80000, 'Струнный')]
        self.assertEqual(part_one(self.one_to_many), answer1)

    def test_part_two(self):
        answer2 = [('Струнный', 80000.0), ('Духовой', 65000.0),
('Симфонический', 40000.0)]
        self.assertEqual(part_two(self.one_to_many), answer2)

    def test_part_three(self):
        answer3 = {'Симфонический': ['Милевич', 'Симонович'],
                   'Струнный': ['Кириевич'],
                   'Симфонический новый': ['Кириевич']}
        self.assertEqual(part_three(self.many_to_many), answer3)
```

```python
if __name__ == "__main__":
    unittest.main()
```

Результат выполнения main.py.

```
Задание Д1
[('Милевич', 50000, 'Симфонический'), ('Симонович', 30000, 'Симфонический'), ('Кириевич', 80000, 'Струнный')]

Задание Д2
[('Струнный', 80000.0), ('Духовой', 65000.0), ('Симфонический', 40000.0)]

Задание Д3
{'Симфонический': ['Милевич', 'Симонович'], 'Струнный': ['Кириевич'], 'Симфонический новый': ['Кириевич']}
```

Результат выполнения main.py в виде текста:

Задание Д1

[('Милевич', 50000, 'Симфонический'), ('Симонович', 30000, 'Симфонический'), ('Кириевич', 80000, 'Струнный')]

Задание Д2

[('Струнный', 80000.0), ('Духовой', 65000.0), ('Симфонический', 40000.0)]

Задание Д3

{'Симфонический': ['Милевич', 'Симонович'], 'Струнный': ['Кириевич'], 'Симфонический новый': ['Кириевич']}

Результат выполнения tests.py.

```
Ran 3 tests in 0.001s

Launching unittests with arguments python -m unittest C:/Users/Tatyana/PycharmProjects/RK2/tests.py in C:\Users\Tatyana\PycharmProjects\RK2
OK


Process finished with exit code 0
```