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 «Системы обработки информации и управления»

Рубежный контроль No2
по дисциплине «Базовые компоненты интернет-технологий»

Выполнил: студент группы ИУ5-35Б Ханунов Г.И

Файл Claas.py

```
class Composition:
    def __init__(self, id, name, num):
        self.id = id
        self.name = name
        self.num = num
class orchest:
    def __init__(self, id,name):
        self.id = id
        self.name = name # название

class orchestComposition:
    def __init__(self, orc_id, comp_id,):
        self.orc_id = orc_id
        self.comp_id = comp_id
```

Файл main.py

```
from operator import itemgetter
import unittest
from claas import  orchest,orchestComposition, Composition

# Оркестр
orc = [
```

```
    orchest(1, 'академический'),
    orchest(2, 'симфонический'),
    orchest(3, 'необыкновенный'),
    orchest(4, 'восхитительный'),
    orchest(5, 'аутентичный'),
]
 #среднее количество муз композиций отсортированных по возрастанию
 # Музыкальные композиции
comp = [
Composition(1, 'первая', 2,),
Composition(2, 'вторая', 3,),
Composition(3, 'третов', 4,),
 Composition(4, 'четвертая', 1,),
 Composition(5, 'пятая', 5,),
 Composition(6, 'шесстов', 2,),
 Composition(7, 'симфония раз', 2,),
Composition(8, 'симфония дов', 4,),
 Composition(9, 'симфония три', 1,)
]

orchest_composition = [

 orchestComposition(1, 2),
orchestComposition(2, 3),
orchestComposition(3, 4),
orchestComposition(4, 1),
orchestComposition(5, 5),
orchestComposition(6, 2),
orchestComposition(7, 2),
orchestComposition(8, 4),
orchestComposition(3, 2),
orchestComposition(5, 3),
orchestComposition(2, 5)
]
#def main():
        # Соединение данных один-ко-многим
class RK:
    def __init__(self, orchest, orchestComposition, Composition):
        self.one_to_many = [(c.name, c.num, o.name)
            for o in orc
            for c in comp
            if c.num == o.id]
                # Соединение данных многие-ко-многим
        self.many_to_many_temp = [(o.name, oc.orc_id, oc.comp_id)
                for o in orc
                for oc in orchest_composition
                if o.id == oc.orc_id]

        self.many_to_many = [(c.name, c.num, orc_name)
                for orc_name, orc_id, comp_id in
        self.many_to_many_temp
                for c in comp if c.id == comp_id]
    def N1(self):
```

```
        print('Задание B1')
        b1= []
        for name, num, orc_name in self.one_to_many:
            buf=len(name)
            if 'в' in name[buf-1] and 'о' in name[buf-2]:
                b1.append((name,orc_name))
        print(b1)
            #среднее количество муз композиций отсортированных по возрастанию
            #(не совсем понятно как реализовать)
    def N2(self):
            print('Задание B2')
            buff = []
            for o in orc:

                        o_names = list(filter(lambda i: i[2] == o.name, self.one_to_many))
                        if len(o_names) > 0:
                            o_num = [num for _, num, _ in o_names]
                            min_num = min(o_num)
                            buff.append((o.name, min_num,))
            b2 = sorted(buff, key=itemgetter(1))
            print(b2)
    def N3(self):
            print('Задание B3')
            buff = []
            for name, num, orc_name in self.many_to_many:
                        if 'a' in orc_name[0]:
                            buff1=orc_name
                            buff.append((name,buff1))
            b3= list(sorted(buff, key=itemgetter(0)))
            print(b3)
if __name__ == '__main__':
    rk=RK(orchest, orchestComposition, Composition)
    rk.N1()
    rk.N2()
    rk.N3()
```

Файл testfile.py

```
import unittest
from main import RK
from claas import orchest,orchestComposition, Composition




res_1 = [('шесстов', 'симфонический'), ('третов', 'восхитительный'), ('симфония дов', 'восхитительный')]



res_2 = [('академический', 1), ('симфонический', 2), ('необыкновенный', 3), ('восхитительный', 4), ('аутентичный', 5)]
```

```
res_3 = [('вторая', 'академический'), ('пятая', 'аутентичный'), ('третов', 'аутентичный')]




class MyTestCase(unittest.TestCase):
    def test_n1(self):
        rk =RK(orchest,orchestComposition, Composition)
        self.assertEqual(res_1, rk.N1())
    def test_n2(self):
        rk = RK(orchest,orchestComposition, Composition)
        self.assertEqual(res_2, rk.N2())
    def test_n3(self):
        rk = RK(orchest,orchestComposition, Composition)
        self.assertEqual(res_3, rk.N2())
if __name__ == '__main__':
    unittest.main()
```

Вывод:

Файл main.py

```
study@Norma rk2BKIT % cd /Users/study/Desktop/rk2BKIT ; /usr/bin/env /usr/local/bin/python3 /Users/study/.vscode/extensio
ns/ms-python.python-2021.12.1559732655/pythonFiles/lib/python/debugpy/launcher 60509 -- /Users/study/Desktop/rk2BKIT/main.
py
Задание B1
[('шесстов', 'симфонический'), ('третов', 'восхитительный'), ('симфония дов', 'восхитительный')]
Задание B2
[('академический', 1), ('симфонический', 2), ('необыкновенный', 3), ('восхитительный', 4), ('аутентичный', 5)]
Задание B3
[('вторая', 'академический'), ('пятая', 'аутентичный'), ('третов', 'аутентичный')]
study@Norma rk2BKIT %
```

Файл testfile.py

```
Задание B1
[('шесстов', 'симфонический'), ('третов', 'восхитительный'), ('симфония дов', 'восхитительный')]
.Задание B2
[('академический', 1), ('симфонический', 2), ('необыкновенный', 3), ('восхитительный', 4), ('аутентичный', 5)]
.Задание B2
[('академический', 1), ('симфонический', 2), ('необыкновенный', 3), ('восхитительный', 4), ('аутентичный', 5)]
.
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
study@Norma rk2BKIT %
```