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 «ГУИМЦ»

Кафедра ИУ5 «Системы обработки информации и управления»

Дисциплина «Базовые компоненты ИТ»

ОТЧЕТ

Рубежный контроль №2

Студент: Фруктин А.Е., группа РТ5-31Б

Преподаватель: Гапанюк Ю.Е.

2022г.

Описание задания:

Вариант Е, вариант предметной области №29.

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).

Листинг программы:

testrk2.py

```
import unittest
from rk2 import *
expect_result_for_e1=[('Факультет: Информатика, искусственный интеллект и системы управления'), ('Факультет: Инженерный бизнес и менеджмент'), ('Факультет: Энергомашиностроение'), ('Факультет: Информатика, искусственный интеллект и системы управления(другие кафедры)'), ('Факультет: Инженерный бизнес и менеджмент(другие кафедры)'), ('Факультет: Энергомашиностроение(другие кафедры)')]
expect_result_for_e2=[('Факультет: Инженерный бизнес и менеджмент', 16.5), ('Факультет: Информатика, искусственный интеллект и системы управления', 13.5), ('Факультет: Энергомашиностроение', 10)]
class RK_test(unittest.TestCase):
    def test_e1(self):
        self.assertEqual(e1(Facultets,Kafedres),expect_result_for_e1)
    def test_e2(self):
        self.assertEqual(e2(Facultets,Kafedres),expect_result_for_e2)




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

rk2.py

```
# Вариант 29 Фруктин А.Е. РТ5-31Б

from operator import itemgetter
import statistics

class Kafedra:
    """Кафедра"""

    def __init__(self, id, name, count_sotrud, fac_id):
        self.id = id
        self.name = name
        self.sotr = count_sotrud
        self.fac_id = fac_id


class Facultet:
    """Факультет"""

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


class KafFac:
    """
      'Кафедры факультетов' для реализации
      связи многие-ко-многим
      """

    def __init__(self, id_kaf, id_fac):
        self.kaf_id = id_kaf
        self.fac_id = id_fac
```

```
#Кафедры
Kafedres = [
    Kafedra(1,'ИУ1-Системы автоматического управления',12, 1),
    Kafedra(2,'ИУ2 - Приборы и системы ориентации, стабилизации и навигации',15, 1),
    Kafedra(3,'Э9- Экология и промышленная безопасность',10, 3),
    Kafedra(4,'ИБМ5 - Финансы',11, 2),
    Kafedra(5,'ИБМ6 - Предпринимательство и внешнеэкономическая деятельность',22, 2),

]
Facultets = [
    Facultet(1,"Факультет: Информатика, искусственный интеллект и системы управления"),
    Facultet(2, "Факультет: Инженерный бизнес и менеджмент"),
    Facultet(3, "Факультет: Энергомашиностроение"),
    Facultet(11,"Факультет: Информатика, искусственный интеллект и системы управления(другие кафедры)"),
    Facultet(22, "Факультет: Инженерный бизнес и менеджмент(другие кафедры)"),
    Facultet(33, "Факультет: Энергомашиностроение(другие кафедры)"),
]

Kaf_Fac = [
    KafFac(1,1),
    KafFac(2,1),
    KafFac(3,3),
    KafFac(4,2),
    KafFac(5,2),
    KafFac(1,11),
    KafFac(2,11),
    KafFac(3,33),
    KafFac(4,22),
    KafFac(5,22),




]
# Соединение данных один-ко-многим
def one_to_many(Facultets,Kafedres):
    return [(kaf.name, kaf.sotr, fac.name)
            for fac in Facultets
            for kaf in Kafedres
            if kaf.fac_id == fac.id]

def e1(Facultets, Kafedres):
        print('Задание E1')
        res_1 = {}
        for fac in Facultets:
            if 'Факультет' in fac.name:
                fac_kafs = list(filter(lambda i: i[2] == fac.name, one_to_many(Facultets,Kafedres)))
                fac_kafs_names = [x for x, _, _ in fac_kafs]
                res_1[fac.name] = fac_kafs_names

        print(list(res_1))
        return list(res_1)
def e2(Facultets, Kafedres):
```

```
    print('Задание E2')
    res_2_unsorted = []
    for fac in Facultets:
        fac_kaf = list(filter(lambda i: i[2] == fac.name, one_to_many(Facultets,Kafedres)))
        if len(fac_kaf) > 0:

            kaf_count = [sotr for _, sotr, _ in fac_kaf]
            fac_sotr_mean = statistics.mean(kaf_count)
            res_2_unsorted.append((fac.name, fac_sotr_mean))

    res_2 = sorted(res_2_unsorted, key=itemgetter(1), reverse=True)
    print(list(res_2))
    return list(res_2)



if __name__ == '__main__':
    e1(Facultets, Kafedres)
    e2(Facultets,Kafedres)
```

Результат выполнения программы:

```
Задание E1
['Факультет: Информатика, искусственный интеллект и системы управления', 'Факультет: Инженерный бизнес и менеджмент', 'Факультет: Энергомашиностроение', 'Факультет:
    Информатика, искусственный интеллект и системы управления(другие кафедры)', 'Факультет: Инженерный бизнес и менеджмент(другие кафедры)', 'Факультет: Энергомашиностроение(
    другие кафедры)']
.Задание E2
[('Факультет: Инженерный бизнес и менеджмент', 16.5), ('Факультет: Информатика, искусственный интеллект и системы управления', 13.5), ('Факультет: Энергомашиностроение', 10)]
.
----------------------------------------------------------------------
Ran 2 tests in 0.001s

OK
```