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 ИУ5-33Б
Ахтамбаев Л.Н.

Проверил:
Гапанюк Ю.Е.

2021 г.

Полученное задание:

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).

Текст программы:

Файл main.py

```
# используется для сортировки
from operator import itemgetter


class Stu:
    """Школьник"""

    def __init__(self, id, fio, avg, cls_id):
        self.id = id
        self.fio = fio
        self.avg = avg  # Средний балл
        self.cls_id = cls_id


class Cls:
    """Класс"""

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


class ClsStu:
    """
    'Школьники класса' для реализации
    связи многие-ко-многим
    """

    def __init__(self, cls_id, stu_id):
        self.cls_id = cls_id
        self.stu_id = stu_id


# Классы
classes = [
    Cls(1, '11А'),
    Cls(2, '11Б'),
    Cls(3, '11В'),
    Cls(4, '11Г'),

    Cls(5, '11Д'),
    Cls(6, '11Е'),

]

# Сотрудники
students = [
    Stu(1, 'Ахтамбаев', 4.9, 1),
    Stu(2, 'Абрамов', 4.7, 1),
```

```
    Stu(3, 'Зорькин', 5.0, 1),
    Stu(4, 'Некрасов', 4.8, 1),
    Stu(5, 'Семенов', 3.5, 2),
    Stu(6, 'Ефремов', 4.2, 2),
    Stu(7, 'Стебунов', 3.7, 3),
    Stu(8, 'Требуков', 4.7, 4),
    Stu(9, 'Носкин', 3.9, 4),

]
# Классы и студенты, для связи многие-ко-многим
classes_students = [
    ClsStu(1, 1),
    ClsStu(1, 2),
    ClsStu(1, 3),
    ClsStu(1, 4),
    ClsStu(2, 5),
    ClsStu(2, 6),
    ClsStu(3, 7),
    ClsStu(4, 8),

    ClsStu(5, 1),
    ClsStu(5, 2),
    ClsStu(6, 3),
    ClsStu(6, 4),
    ClsStu(6, 5),

]


def first_task(one_to_many):
    task1 = []
    for fio, avg, name in one_to_many:
        if fio[0] == "A":
            task1.append((fio, name))
    return task1


def second_task(one_to_many):
    task2_uns = []
    for c in classes:
        # все школьники класса
        s_cls = list(filter(lambda i: i[2] == c.name, one_to_many))
        if len(s_cls) > 0:
            c_avg = [avg for _, avg, _ in s_cls]
            c_minAvg = min(c_avg)
            task2_uns.append((c.name, c_minAvg))
    task2 = sorted(task2_uns, key=itemgetter(1))
    return task2


def third_task(many_to_many):
    task3_uns = []
    for fio, avg, name in many_to_many:
        task3_uns.append((fio, name))

    task3 = list(sorted(task3_uns, key=itemgetter(0)))
    return task3


def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(s.fio, s.avg, c.name)
```

```
                    for c in classes
                    for s in students
                    if s.cls_id == c.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(c.name, cs.cls_id, cs.stu_id)
                         for c in classes
                         for cs in classes_students
                         if c.id == cs.cls_id]

    many_to_many = [(s.fio, s.avg, cls_name)
                    for cls_name, cls_id, stu_id in many_to_many_temp
                    for s in students if s.id == stu_id]

    print('\nЗадание 1 \n', first_task(one_to_many))
    print('\nЗадание 2 \n', second_task(one_to_many))
    print('\nЗадание 3 \n', third_task(many_to_many))


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

Файл test.py

```
import unittest
import sys, os

sys.path.append(os.getcwd())
from main import *


class TddTest(unittest.TestCase):
    def test_first_task(self):
        one_to_many = [(s.fio, s.avg, c.name)
                       for c in classes
                       for s in students
                       if s.cls_id == c.id]
        self.assertEqual(first_task(one_to_many), [('Ахтамбаев', '11А'),
('Абрамов', '11А')])

    def test_2(self):
        one_to_many = [(s.fio, s.avg, c.name)
                       for c in classes
                       for s in students
                       if s.cls_id == c.id]
        self.assertEqual(second_task(one_to_many), [('11Б', 3.5), ('11В',
3.7), ('11Г', 3.9), ('11А', 4.7)])

    def test_3(self):
        many_to_many_temp = [(c.name, cs.cls_id, cs.stu_id)
                             for c in classes
                             for cs in classes_students
                             if c.id == cs.cls_id]
        many_to_many = [(s.fio, s.avg, cls_name)
                        for cls_name, cls_id, stu_id in many_to_many_temp
                        for s in students if s.id == stu_id]
        self.assertEqual(third_task(many_to_many),
                         [('Абрамов', '11А'), ('Абрамов', '11Д'),
('Ахтамбаев', '11А'), ('Ахтамбаев', '11Д'),
                          ('Ефремов', '11Б'), ('Зорькин', '11А'), ('Зорькин',
'11Е'), ('Некрасов', '11А'),
                          ('Некрасов', '11Е'), ('Семенов', '11Б'),
```

```
('Семенов', '11Е'), ('Стебунов', '11В'),
                           ('Требуков', '11Г')])
```

Результат выполнения:

```
C:\Users\hoppl\AppData\Local\Programs\Python\Python39\python.
Testing started at 19:39 ...


Ran 3 tests in 0.003s
Launching unittests with arguments python -m unittest test.Td


OK

Process finished with exit code 0
```