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

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Рубежный контроль №2

Выполнил:
студент группы ИУ5-34Б
Данилин Максим
Подпись и дата:

Проверил:
Гапанюк Ю. Е.
Подпись и дата:

Москва, 2021 г.

## Постановка задачи

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).

# Текст программы

## Файл rk1.py

```
from operator import itemgetter

class House:

    def __init__(self, id, number, floor, street_id):
        self.id = id
        self.number = number
        self.floor = floor
        self.street_id = street_id


class Street:

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


class HouseStreet:

    def __init__(self, street_id, house_id):
        self.street_id = street_id
        self.house_id = house_id


# Улицы
streets = [
    Street(1, 'Академическая'),
    Street(2, 'Арбат'),
    Street(3, 'Тверская'),

    Street(11, 'Спасская'),
    Street(22, 'Коптевская'),
    Street(33, 'Михалковская'),
]

# Дома
houses = [
    House(1, 26, 5, 3),
    House(2, 71, 21, 2),
    House(3, 23, 16, 3),
    House(4, 7, 9, 1),
    House(5, 45, 11, 1),
]

houses_streets = [
    HouseStreet(1, 1),
    HouseStreet(1, 2),
    HouseStreet(3, 3),
    HouseStreet(2, 3),
    HouseStreet(3, 5),

    HouseStreet(22, 1),
    HouseStreet(11, 2),
    HouseStreet(33, 3),
    HouseStreet(33, 4),
    HouseStreet(22, 5),
```

```
]

def task_1(streets, one_to_many):
    str = [s.name for s in streets if s.name[0] == 'A']
    if len(str) > 0:
        res_11 = [(s, list(number for number, _, n in one_to_many if n == s))
for s in str]
    return res_11

def task_2(streets, one_to_many):
    res_12_unsorted = []
    # Перебираем все улицы
    for s in streets:
        # Список домов на улице
        s_houses = list(filter(lambda i: i[2] == s.name, one_to_many))
        # Список количества этажей
        s_floor = list(f[1] for f in s_houses if f[2] == s.name)
        # Если улица не пустая
        if len(s_houses) > 0:
            # максимальный этаж
            max_floor = max(s_floor)
            res_12_unsorted.append((s.name, max_floor))

    # Сортировка по максимальному количеству этажей
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    return res_12

def task_3(many_to_many):
    res_13 = []
    for house, _, street in many_to_many:
        res_13.append((street, house))
    res_13 = sorted(res_13, key=itemgetter(0))
    return res_13

def main():

    # один-ко-многим
    one_to_many = [(h.number, h.floor, s.name)
                   for s in streets
                   for h in houses
                   if h.street_id == s.id]

    # многие-ко-многим
    many_to_many_temp = [(s.name, hs.street_id, hs.house_id)
                         for s in streets
                         for hs in houses_streets
                         if s.id == hs.street_id]

    many_to_many = [(h.number, h.floor, street_name)
                    for street_name, street_id, house_id in many_to_many_temp
                    for h in houses if h.id == house_id]

    print('Задание Г1', '\n', task_1(streets, one_to_many))

    print('\nЗадание Г2', '\n', task_2(streets, one_to_many))

    print('\nЗадание Г3', '\n', task_3(many_to_many))

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

## Файл test_rk1.py

```
import unittest
from rk1 import task_1, task_2, task_3, streets, houses, houses_streets

one_to_many = [(h.number, h.floor, s.name)
                   for s in streets
                   for h in houses
                   if h.street_id == s.id]

many_to_many_temp = [(s.name, hs.street_id, hs.house_id)
                         for s in streets
                         for hs in houses_streets
                         if s.id == hs.street_id]

many_to_many = [(h.number, h.floor, street_name)
                    for street_name, street_id, house_id in many_to_many_temp
                    for h in houses if h.id == house_id]

class Test(unittest.TestCase):

    def test1(self):
        self.assertEqual(task_1(streets, one_to_many), [('Академическая', [7,
45]), ('Арбат', [71])])

    def test2(self):
        self.assertEqual(task_2(streets, one_to_many), [('Арбат', 21),
('Тверская', 16), ('Академическая', 11)])

    def test3(self):
        self.assertEqual(task_3(many_to_many), [('Академическая', 26),
('Академическая', 71), ('Арбат', 23),
                                                ('Коптевская', 26),
('Коптевская', 45), ('Михалковская', 23),
                                                ('Михалковская', 7),
('Спасская', 71), ('Тверская', 23),
                                                ('Тверская', 45)])

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

# Результат выполнения

```
Launching unittests with arguments python -m unittest /Users/choza_max/PycharmProjects/rk2/test_rk1.py in
 /Users/choza_max/PycharmProjects/rk2



Ran 3 tests in 0.002s

OK
```