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 Лаврик Д.Д.

Проверил:
Гапанюк Ю.Е.

2021 г.

Задание рубежного контроля №2:
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).

Задание рубежного контроля №1:
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 10: Браузер, Компьютер.

Текст программы:

```
import unittest
from operator import itemgetter


class Browser:

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


class Computer:

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


class ComputerBrowser:

    def __init__(self, browsers_id, computers_id):
        self.browsers_id = browsers_id
        self.computers_id = computers_id


computers = [
    Computer(1, 'Апл'),
    Computer(2, 'Делл'),
    Computer(3, 'Асус'),
    Computer(4, 'Сяоми'),
```

```
]

browsers = [
    Browser(1, 'Опера', 87, 4),
    Browser(2, 'Сафари', 345, 3),
    Browser(3, 'Хром', 189, 4),
    Browser(4, 'Эдж', 955, 1),
    Browser(5, 'Яху', 286, 2),
    Browser(6, 'Мозилла', 721, 1)
]

computers_browsers = [
    ComputerBrowser(4, 1),
    ComputerBrowser(2, 3),
    ComputerBrowser(3, 3),
    ComputerBrowser(6, 4),
    ComputerBrowser(1, 2),
    ComputerBrowser(5, 2),
    ComputerBrowser(2, 1),
    ComputerBrowser(4, 2),
    ComputerBrowser(4, 3),
    ComputerBrowser(4, 4),
    ComputerBrowser(5, 1),
    ComputerBrowser(6, 2)
]


def fun1(one_to_many1):
    print('Задание Д1')
    res_1 = []
    for name, size, computers_name in one_to_many1:
        if 'a' in name[len(name) - 1]:
            res_1.append((name, computers_name))
    return res_1


def fun2(one_to_many1):
    print('Задание Д2')
    temp = []
    for c in computers:
        c_models = list(filter(lambda i: i[2] == c.model, one_to_many1))
        if len(c_models) > 0:
            c_size = [size for _, size, _ in c_models]
            sum_size = sum(c_size)
            count_browsers = len(c_size)
            average_size = sum_size / count_browsers
            temp.append((c.model, average_size))
    res_2 = sorted(temp, key=itemgetter(1))
    return res_2


def fun3(many_to_many1):
    print('Задание Д3')
    res_3 = []
    for name, size, computers_name in many_to_many1:
        if 'A' in computers_name[0]:
            res_3.append((name, computers_name))
    return res_3


class Test(unittest.TestCase):
    def test_task1(self):

        test_computers = Computer(1, 'Апл')
        test_browsers = Browser(1, 'Опера', 234, 2)
        test_one_to_many = [(test_browsers.name, test_browsers.size,
test_computers.model)]
```

```
        ans = fun1(test_one_to_many)

        self.assertTrue(ans, [('Опера', 'Апл')])

    def test_task2(self):

        test_computers = Computer(1, 'Acyc')
        test_browsers = Browser(6, 'Мозилла', 721, 1)
        test_one_to_many = [(test_browsers.name, test_browsers.size,
test_computers.model)]
        ans = fun2(test_one_to_many)

        self.assertTrue(ans, [('Мозилла', 721)])

    def test_task3(self):

        test_computers = Computer(1, 'Acyc')
        test_browsers = Browser(6, 'Мозилла', 721, 1)
        test_many_to_many = [(test_browsers.name, test_browsers.size,
test_computers.model)]
        ans = fun3(test_many_to_many)

        self.assertTrue(ans, {'Мозилла', 'Acyc'})


def main():
    # Соединение данных один-ко-многим
    one_to_many = [(b.name, b.size, c.model)
                   for c in computers
                   for b in browsers
                   if b.comp == c.id]

    many_to_many_temp = [(c.model, cb.computers_id, cb.browsers_id)
                         for c in computers
                         for cb in computers_browsers
                         if c.id == cb.computers_id]

    many_to_many = [(b.name, b.size, computers_name)
                    for computers_name, computers_id, browsers_id in
many_to_many_temp
                    for b in browsers if b.id == browsers_id]


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

Результаты выполнения программы:

```
Launching unittests with arguments python -m unittest C:/Users/igrun/PycharmProjects/PK/main.py in C:\Users\igrun\PycharmProjects\PK



Ran 3 tests in 0.007s

OK
Задание Д1
Задание Д2
Задание Д3
```