**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»**

Рубежная контрольная 2

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:

студент группы ИУ5-35Б

Акулова А.А.

Преподаватель:

Ю.Е. Гапанюк

2021 г.

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).

**Вариант Д.(Задание из рубежного контроля 1)**

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 предметной области 1

Студент-группа

Текст программы

1. Файл main.py

import pytest  
  
  
from for\_me\_dz import zadanie\_1, zadanie\_2, zadanie\_3  
  
  
def test\_zadanie\_1():  
 zadanie\_1\_needed\_result = [[('Александров', 1, 'ИУ5-35')], [('Беленьков', 4, 'Аэрокосмическая группа #1')],  
 [('Норков', 5, 'Группа по созданию черного ящика "ГАС Контур"')]]  
 assert zadanie\_1() == zadanie\_1\_needed\_result  
  
def test\_zadanie\_2():  
 zadanie\_2\_needed\_result = [('Группа по созданию черного ящика "ГАС Контур"', 6.0),  
 ('Аэрокосмическая группа #1', 4.0), ('ИУ5-35', 0.5)]  
 assert zadanie\_2() == zadanie\_2\_needed\_result  
  
  
def test\_zadanie\_3():  
 zadanie\_3\_needed\_result = {'Аэрокосмическая группа #1': ['Беленьков'],  
 'Амбициозная группа по разработке АРМ ЛПР': ['Ким', 'Норков']}  
 assert zadanie\_3() == zadanie\_3\_needed\_result  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 pytest.main()

1. Файл for\_me\_dz.py

*# используется для сортировки*from operator import itemgetter  
  
  
class Student:  
 *"""Студент"""* def \_\_init\_\_(self, id, fio, debt\_quantity, group\_id):  
 self.id = id  
 self.fio = fio  
 self.debt\_quantity = debt\_quantity  
 self.group\_id = group\_id  
  
  
class Group:  
 *"""Группа"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
class StudentGroup:  
 *"""  
 'Студенты группы' для реализации  
 связи многие-ко-многим  
 """* def \_\_init\_\_(self, group\_id, student\_id):  
 self.student\_id = student\_id  
 self.group\_id = group\_id  
  
  
class InfoFromTables:  
 def \_\_init\_\_(self):  
 *# Студенты* self.students = [  
 Student(1, 'Акулова', 0, 1),  
 Student(2, 'Александров', 1, 1),  
 Student(3, 'Беленьков', 4, 2),  
 Student(4, 'Ким', 7, 3),  
 Student(5, 'Норков', 5, 3),  
 ]  
  
 *# Группы* self.groups = [  
 Group(1, 'ИУ5-35'),  
 Group(2, 'Аэрокосмическая группа #1'),  
 Group(3, 'Группа по созданию черного ящика "ГАС Контур"'),  
  
 Group(11, 'ИУ5-31'),  
 Group(22, 'aэрокосмическая группа #2'),  
 Group(33, 'Амбициозная группа по разработке АРМ ЛПР'),  
 ]  
  
 self.students\_groups = [  
 StudentGroup(1, 1),  
 StudentGroup(1, 2),  
 StudentGroup(2, 3),  
 StudentGroup(3, 4),  
 StudentGroup(3, 5),  
  
 StudentGroup(11, 1),  
 StudentGroup(11, 2),  
 StudentGroup(22, 3),  
 StudentGroup(33, 4),  
 StudentGroup(33, 5),  
 ]  
  
 *# Соединение данных один-ко-многим* self.one\_to\_many = [(s.fio, s.debt\_quantity, g.name)  
 for g in self.groups  
 for s in self.students  
 if s.group\_id == g.id]  
  
 *# Соединение данных многие-ко-многим* self.many\_to\_many\_temp = [(g.name, sg.group\_id, sg.student\_id)  
 for g in self.groups  
 for sg in self.students\_groups  
 if g.id == sg.group\_id]  
  
 self.many\_to\_many = [(s.fio, s.debt\_quantity, group\_name)  
 for group\_name, group\_id, student\_id in self.many\_to\_many\_temp  
 for s in self.students if s.id == student\_id]  
  
  
def zadanie\_1():  
 print('Задание А1')  
 info = InfoFromTables()  
 res\_11 = []  
 for s in info.students:  
 *# Перебираем всех студентов* if s.fio.endswith('ов'):  
 *# Список студетов группы* g\_students = list(filter(lambda i: i[0] == s.fio, info.one\_to\_many))  
 res\_11.append(g\_students)  
 print(res\_11)  
 return res\_11  
  
  
def zadanie\_2():  
 print('\nЗадание А2')  
 info = InfoFromTables()  
 res\_12\_unsorted = []  
 *# Перебираем все отделы* for g in info.groups:  
 *# Список студентов группы* g\_students = list(filter(lambda i: i[2] == g.name, info.one\_to\_many))  
 *# Если группа не пустая* if len(g\_students) > 0:  
 *# Долги студентов группы* g\_debts = [debt for \_, debt, \_ in g\_students]  
 *# Средний долг студентов группы* g\_debts\_avg = sum(g\_debts) / len(g\_debts)  
 res\_12\_unsorted.append((g.name, g\_debts\_avg))  
  
 *# Сортировка по среднему долгу* res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)  
 print(res\_12)  
 return res\_12  
  
  
def zadanie\_3():  
 print('\nЗадание А3')  
 info = InfoFromTables()  
 res\_13 = {}  
 *# Перебираем все группы* for g in info.groups:  
 if g.name.startswith('А'): *# регистрозависимая операция  
 # Список студентов группы* g\_students = list(filter(lambda i: i[2] == g.name, info.many\_to\_many))  
 *# Только ФИО студентов* g\_students\_names = [fio for fio, \_, \_ in g\_students]  
 *# Добавляем результат в словарь  
 # ключ - группа, значение - список фамилий* res\_13[g.name] = g\_students\_names  
  
 print(res\_13)  
 return res\_13

Результаты программы

