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

Факультет «Радиотехнический»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчёт по рубежному контролю №2

Вариант №1

|  |  |
| --- | --- |
| Выполнил: | Проверил: |
| студент группы РТ5-31Б | преподаватель каф. ИУ5 |
| Агеев Алексей | Гапанюк Ю.Е. |
| Подпись и дата: | Подпись и дата: |

Москва, 2021 г

# Описание задания

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

**tut.py**

*# используется для сортировки*

from operator import itemgetter

class Group:

    """

    Группа - id, название.

    """

    def \_\_init\_\_(self, id, name):

        self.id = id

        self.name = name

class Student:

    """

    Студент - id, фамилия, id группы, кол-во статей.

    """

    def \_\_init\_\_(self, id, surname, group\_id, article\_num):

        self.id = id

        self.surname = surname

        self.group\_id = group\_id

        self.article\_num = article\_num

class StudentsOfGroup:

    """

    Студенты групы - id студента, id группы

    """

    def \_\_init\_\_(self, student\_id, group\_id):

        self.student\_id = student\_id

        self.group\_id = group\_id

*#Создаем фамилии и названия групп*

surname = ["Агеев", "Петров", "Иванов", "Гонт", "Гур", "Блур"]

group = ["Группа ИУ", "Группа РК", "РТ"]

*#Создаем массивы объектов студент и групп*

students = [Student(i, surname[i], (i + 1) % len(group), i % 4) for i in range(len(surname))]

groups = [Group(i, group[i]) for i in range(len(group))]

*#Создаём массив объектов типа студенты группы(многие ко многим)*

Students\_Of\_Group = [StudentsOfGroup(students[i].id, groups[i%len(group)].id) for i in range(len(surname))]

one\_to\_many = [(stud.surname, stud.article\_num, grp.name)

                    for grp in groups

                    for stud in students

                    if stud.group\_id == grp.id]

many\_to\_many = [(stud.surname, stud.article\_num, grp.name)

                    for stofgr in Students\_Of\_Group

                    for stud in students

                    for grp in groups

                    if(stud.id == stofgr.student\_id and grp.id == stofgr.group\_id)]

def task1():

    """

    1 задание

    """

    print("1 задание")

    res = []

    buff = []

    for \_,\_,grp in one\_to\_many:

        if grp.find("Группа") != -1 and grp not in buff:

            buff.append(grp)

            res.append((grp, list(name for name,\_,grp\_temp in one\_to\_many if grp\_temp == grp)))

    return res

def task2():

    """

    2 задание

    """

    print("2 задание")

*#массив среднего количества статей*

    index = 0

    count = 0

    middle\_num = 0

    answer2 = []

    gr = ""

    for name, num, grp in one\_to\_many:

        if (gr != grp):

            if (count != 0):

                answer2.append((gr, (round(middle\_num/count, 2))))

            gr = grp

            middle\_num = 0

            count = 0

        if (gr == grp):

            middle\_num += num

            count += 1

    answer2.append((gr, (round(middle\_num/count, 2)))) *# так как последний append не будет произведен*

    answer2 = sorted(answer2, key=itemgetter(1))

    return answer2

def task3():

    """

    3 задание

    """

    print("3 задание")

    answer3 = []

    for surname, num, grp in many\_to\_many:

*# если фамилия начинается с "А"*

        if (surname[0] == "А"):

            answer3.append((surname, grp))

    return answer3

**test\_tut.py**

import unittest

import tut

class Test\_tut(unittest.TestCase):

    def test\_tasks(self):

        self.assertEqual(tut.task1(), [('Группа ИУ', ['Иванов', 'Блур']), ('Группа РК', ['Агеев', 'Гонт'])])

        self.assertEqual(tut.task2(), [('РТ', 0.5), ('Группа ИУ', 1.5), ('Группа РК', 1.5)])

        self.assertEqual(tut.task3(),[('Агеев', 'Группа ИУ')])

if \_\_name\_\_ == "\_\_main\_\_":

    unittest.main()

# Пример работы программы

