



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 _____ Системы обработки информации и управления

**Отчет по рубежному контролю № 2 по курсу
базовые компоненты интернет-технологий**

Вариант 28А

ИСПОЛНИТЕЛЬ:

Студент группы ИУ5Ц-51Б
Печуркин Д.С.

(подпись)

Преподаватель
Гапанюк Ю.Е.

"__" _____ 2022 г.

Москва, МГТУ - 2022

СОДЕРЖАНИЕ ОТЧЕТА

Задание	3
Листинг программы	4
class_group.py (Основной файл).....	4
test_stud_group.py (Файл модульного теста)	7
Результат работы программы	9
class_group.py (Основной файл)	9
PyCharm	9
cmd (Командная строка).....	9
test_stud_group.py (Файл модульного теста)	10
PyCharm	10
cmd (Командная строка).....	10

Задание

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).

Листинг программы class_group.py (Основной файл)

```
# Вариант 28 А Печуркин Д.С. ИУ5Ц-51Б РК2

from operator import itemgetter

class Stud_Group:
    """Студенческая группа"""

    def __init__(self, id, name, count_students, dep_id):
        self.id = id
        self.name = name
        self.students = count_students
        self.dep_id = dep_id

class Departament:
    """Кафедра"""

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

class Merge:
    """
    Студенческие группы кафедр
    для реализации связи многие-ко-многим
    """

    def __init__(self, dep_id, stud_id):
        self.dep_id = dep_id
        self.stud_id = stud_id

Stud_Groups = [
    Stud_Group(1, 'ИУ5Ц-51Б', 3, 1),
    Stud_Group(2, 'РК6-30', 10, 4),
    Stud_Group(3, 'ИБМ2-61', 6, 5),
    Stud_Group(4, 'ИБМ2-62', 6, 5),
    Stud_Group(5, 'МТ8-31', 8, 3),
    Stud_Group(6, 'ИУ3-31', 11, 2),
    Stud_Group(7, 'ИУ3-32', 11, 2),
    Stud_Group(8, 'ИУ5Ц-52Б', 5, 1),
    Stud_Group(9, 'ИУ5Ц-53Б', 7, 1),
]

Departaments = [
    Departament(1, 'ИУ5 - Системы обработки информации и управления'),
    Departament(2, 'ИУ3 - Информационные системы и телекоммуникации'),
    Departament(3, 'МТ8 - Материаловедение'),
    Departament(4, 'РК6 - Системы автоматизированного проектирования', ),
    Departament(5, 'ИБМ2 - Экономика и организация производства'),
]

Merges = [
    Merge(1, 1),
    Merge(1, 8),
    Merge(1, 9),
    Merge(3, 5),
    Merge(5, 3),
    Merge(5, 4),
    Merge(2, 6),
    Merge(2, 7),
]
```

```

def exercise_G1(Stud_Groups, Departaments):
    # Соединение данных один-ко-многим
    one_to_many = [(group.name, group.students, dep.name)
                    for dep in Departaments
                    for group in Stud_Groups
                    if dep.id == group.dep_id]
    res_0 = sorted(one_to_many, key=itemgetter(2))
    return res_0

def exercise_G2(Departaments, one_to_many):
    res_1 = []

    for dep in Departaments:
        # Список студентов кафедры
        deps = list(filter(lambda i: i[2] == dep.name, one_to_many))
        # Если кафедра не пустая
        if len(deps) > 0:
            # Кол-во студентов и суммарно студентов на кафедре
            stud_sum = sum([count for _, count, _ in deps])
            res_1.append((dep.name, stud_sum))

    # Сортировка по количеству по убыванию
    res_1 = sorted(res_1, key=itemgetter(1), reverse=True)
    return res_1

def exercise_G3(Stud_Groups, Departaments, Merges):
    res_2 = {}

    many_to_many_temp = [(dep.name, merge.dep_id, merge.stud_id)
                          for dep in Departaments
                          for merge in Merges
                          if dep.id == merge.dep_id]

    many_to_many = [(stud.name, stud.students, name)
                    for name, dep_id, stud_id in many_to_many_temp
                    for stud in Stud_Groups if stud.id == stud_id]

    # Перебираем все кафедры
    for dep in Departaments:
        if 'ИУ' in dep.name:
            # Список студентов кафедры
            list_studs = list(filter(lambda x: x[2] == dep.name, many_to_many))
            # Только названия кафедр
            list_deps = [name for name, _, _ in list_studs]
            # Добавляем результат в словарь
            # ключ - кафедра, значение - список групп
            res_2[dep.name] = list_deps

    return res_2

def main():
    # print('Вариант 28А Печуркин Д.С. ИУ5Ц-51Б\n')
    one_to_many = exercise_G1(Stud_Groups, Departaments)
    print("Задание 1")
    print(one_to_many)

    res_1 = exercise_G2(Departaments, one_to_many)
    print("Задание 2")
    print(res_1)

    res_2 = exercise_G3(Stud_Groups, Departaments, Merges)
    print("Задание 3")
    print(res_2)

if __name__ == '__main__':

```

```
main()
```

test_stud_group.py (Файл модульного теста)

```
import unittest

from class_group import Stud_Group, Merge, Departament, exercise_G1, exercise_G2, exercise_G3

Stud_Groups = [
    Stud_Group(1, 'ИУ5Ц-51Б', 3, 1),
    Stud_Group(2, 'РК6-30', 10, 4),
    Stud_Group(3, 'ИБМ2-61', 6, 5),
    Stud_Group(4, 'ИБМ2-62', 6, 5),
    Stud_Group(5, 'МТ8-31', 8, 3),
    Stud_Group(6, 'ИУ3-31', 11, 2),
    Stud_Group(7, 'ИУ3-32', 11, 2),
    Stud_Group(8, 'ИУ5Ц-52Б', 5, 1),
    Stud_Group(9, 'ИУ5Ц-53Б', 7, 1),
]

Departaments = [
    Departament(1, 'ИУ5 - Системы обработки информации и управления'),
    Departament(2, 'ИУ3 - Информационные системы и телекоммуникации'),
    Departament(3, 'МТ8 - Материаловедение'),
    Departament(4, 'РК6 - Системы автоматизированного проектирования', ),
    Departament(5, 'ИБМ2 - Экономика и организация производства'),
]

Merges = [
    Merge(1, 1),
    Merge(1, 8),
    Merge(1, 9),
    Merge(3, 5),
    Merge(5, 3),
    Merge(5, 4),
    Merge(2, 6),
    Merge(2, 7),
]

one_to_many = exercise_G1(Stud_Groups, Departaments)

class test_Departament(unittest.TestCase):
    def test_class_Stud_Group_meaning(self):
        test_class_Stud_Group = Stud_Group(5, 'МТ8-31', 8, 3)
        self.assertEqual(test_class_Stud_Group.id, 5)
        self.assertEqual(test_class_Stud_Group.name, 'МТ8-31')
        self.assertEqual(test_class_Stud_Group.students, 8)
        self.assertEqual(test_class_Stud_Group.dep_id, 3)

    def test_class_Departament_zero_parameters(self):
        with self.assertRaises(TypeError) as context:
            Departament()
        self.assertEqual(
            "Departament.__init__() missing 2 required positional arguments: 'id' and 'name'",
            str(context.exception)
        )

    def test_class_Departament_zero_meaning(self):
        test_class_Departament = Departament(None, None)
        self.assertEqual(test_class_Departament.id, None)
        self.assertEqual(test_class_Departament.name, None)

    def test_class_Departament_meaning(self):
        test_class_Departament = Departament(2, 'ИБМ2 - Экономика и организация
```

```

производства')
    self.assertEqual(test_class_Departament.id, 2)
    self.assertEqual(test_class_Departament.name, 'ИБМ2 - Экономика и
организация производства')

    def test_class_Merge_meaning(self):
        test_class_sections_documents = Merge(5, 7)
        self.assertEqual(test_class_sections_documents.dep_id, 5)
        self.assertEqual(test_class_sections_documents.stud_id, 7)

# Тестирование задания 1
def test_task_1(self):
    self.assertEqual(exercise_G1(Stud_Groups, Departaments),
        [('ИБМ2-61', 6, 'ИБМ2 - Экономика и организация
производства'),
        ('ИБМ2-62', 6, 'ИБМ2 - Экономика и организация
производства'),
        ('ИУ3-31', 11, 'ИУ3 - Информационные системы и
телекоммуникации'),
        ('ИУ3-32', 11, 'ИУ3 - Информационные системы и
телекоммуникации'),
        ('ИУ5Ц-51Б', 3, 'ИУ5 - Системы обработки информации и
управления'),
        ('ИУ5Ц-52Б', 5, 'ИУ5 - Системы обработки информации и
управления'),
        ('ИУ5Ц-53Б', 7, 'ИУ5 - Системы обработки информации и
управления'),
        ('МТ8-31', 8, 'МТ8 - Материаловедение'),
        ('РК6-30', 10, 'РК6 - Системы автоматизированного
проектирования')])

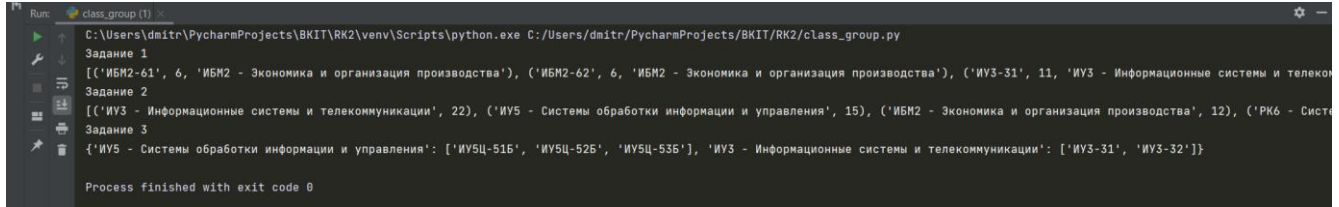
# Тестирование задания 2
def test_task_2(self):
    self.assertEqual(exercise_G2(Departaments, one_to_many),
        [('ИУ3 - Информационные системы и телекоммуникации', 22),
        ('ИУ5 - Системы обработки информации и управления', 15),
        ('ИБМ2 - Экономика и организация производства', 12),
        ('РК6 - Системы автоматизированного проектирования', 10),
        ('МТ8 - Материаловедение', 8)])

# Тестирование задания 3
def test_task_3(self):
    self.assertEqual(exercise_G3(Stud_Groups, Departaments, Merges),
        {'ИУ5 - Системы обработки информации и управления':
['ИУ5Ц-51Б', 'ИУ5Ц-52Б', 'ИУ5Ц-53Б'],
        'ИУ3 - Информационные системы и телекоммуникации':
['ИУ3-31', 'ИУ3-32']})

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

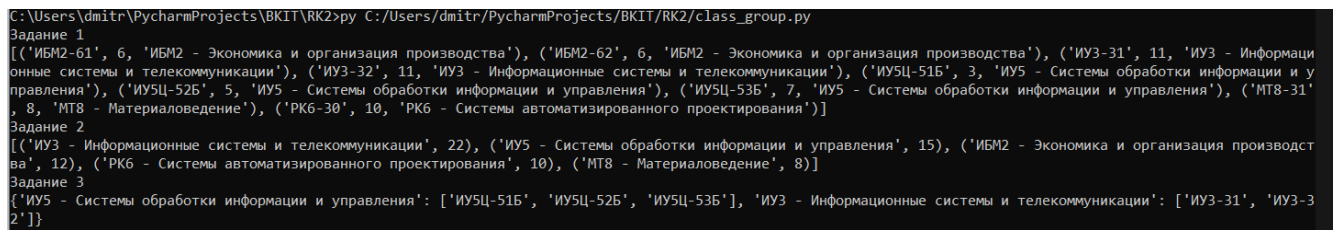
```


Результат работы программы class_group.py (Основной файл) PyCharm



```
Run: class_group (1)
C:\Users\dmityr\PycharmProjects\BKIT\RK2\venv\Scripts\python.exe C:/Users/dmityr/PycharmProjects/BKIT/RK2/class_group.py
Задание 1
[('ИБМ2-61', 6, 'ИБМ2 - Экономика и организация производства'), ('ИБМ2-62', 6, 'ИБМ2 - Экономика и организация производства'), ('ИУ3-31', 11, 'ИУ3 - Информационные системы и телекоммуникации'), ('ИУ3-32', 11, 'ИУ3 - Информационные системы и телекоммуникации'), ('ИУ5Ц-51Б', 3, 'ИУ5 - Системы обработки информации и управления'), ('ИУ5Ц-52Б', 5, 'ИУ5 - Системы обработки информации и управления'), ('ИУ5Ц-53Б', 7, 'ИУ5 - Системы обработки информации и управления'), ('МТ8-31', 8, 'МТ8 - Материаловедение'), ('РК6-30', 10, 'РК6 - Системы автоматизированного проектирования')]
Задание 2
[('ИУ3 - Информационные системы и телекоммуникации', 22), ('ИУ5 - Системы обработки информации и управления', 15), ('ИБМ2 - Экономика и организация производства', 12), ('РК6 - Системы автоматизированного проектирования', 10), ('МТ8 - Материаловедение', 8)]
Задание 3
{'ИУ5 - Системы обработки информации и управления': ['ИУ5Ц-51Б', 'ИУ5Ц-52Б', 'ИУ5Ц-53Б'], 'ИУ3 - Информационные системы и телекоммуникации': ['ИУ3-31', 'ИУ3-32']}
Process finished with exit code 0
```

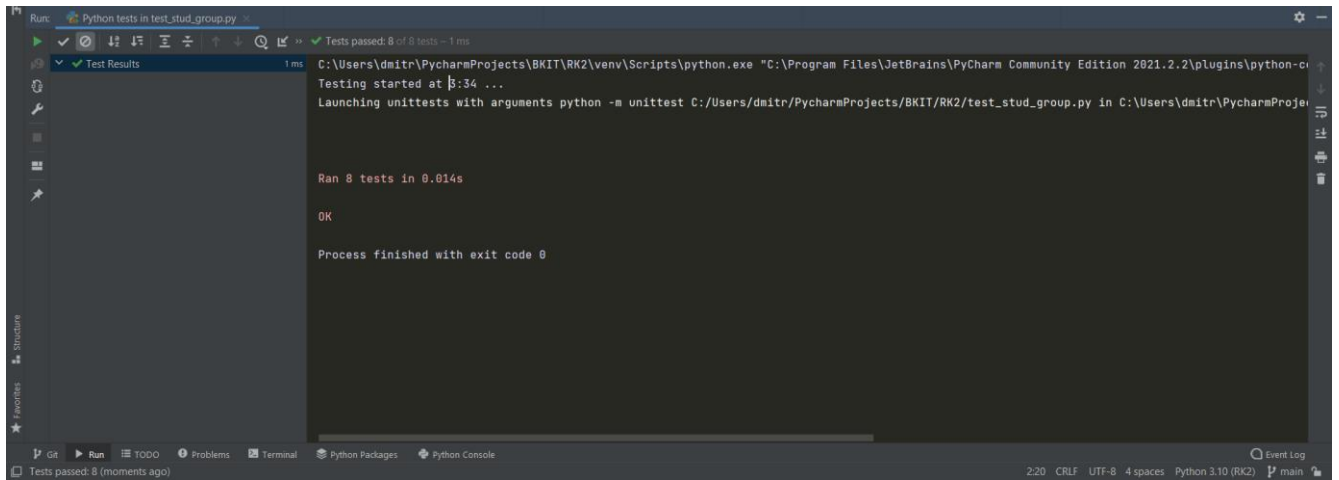
cmd (Командная строка)



```
C:\Users\dmityr\PycharmProjects\BKIT\RK2>py C:/Users/dmityr/PycharmProjects/BKIT/RK2/class_group.py
Задание 1
[('ИБМ2-61', 6, 'ИБМ2 - Экономика и организация производства'), ('ИБМ2-62', 6, 'ИБМ2 - Экономика и организация производства'), ('ИУ3-31', 11, 'ИУ3 - Информационные системы и телекоммуникации'), ('ИУ3-32', 11, 'ИУ3 - Информационные системы и телекоммуникации'), ('ИУ5Ц-51Б', 3, 'ИУ5 - Системы обработки информации и управления'), ('ИУ5Ц-52Б', 5, 'ИУ5 - Системы обработки информации и управления'), ('ИУ5Ц-53Б', 7, 'ИУ5 - Системы обработки информации и управления'), ('МТ8-31', 8, 'МТ8 - Материаловедение'), ('РК6-30', 10, 'РК6 - Системы автоматизированного проектирования')]
Задание 2
[('ИУ3 - Информационные системы и телекоммуникации', 22), ('ИУ5 - Системы обработки информации и управления', 15), ('ИБМ2 - Экономика и организация производства', 12), ('РК6 - Системы автоматизированного проектирования', 10), ('МТ8 - Материаловедение', 8)]
Задание 3
{'ИУ5 - Системы обработки информации и управления': ['ИУ5Ц-51Б', 'ИУ5Ц-52Б', 'ИУ5Ц-53Б'], 'ИУ3 - Информационные системы и телекоммуникации': ['ИУ3-31', 'ИУ3-32']}

```

test_stud_group.py (Файл модульного теста) PyCharm



cmd (Командная строка)

