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
ИУ5Ц-52Б Дзауров И.А.
_____, “___” _____ 2022 г.

Проверил:
преподаватель кафедры
ИУ5 - Гапанюк Ю.Е.
_____, “___” _____ 2022 г.

Москва, 2022 г.

Описание задания

Вариант предметной области – 26 [«Студенческая группа» - «Учебный курс»].
Вариант запросов – Б.

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

Основной файл – main.py:

```
# используется для сортировки
from operator import itemgetter


class Group:
    """Студенческая группа"""

    def __init__(self, id, name, studentsCount, course_id):
        self.id = id
        self.name = name
        self.studentsCount = studentsCount  # количество студентов
        self.course_id = course_id


class Course:
    """Учебный курс"""

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


class GroupCourse:
    """
    'Учебный курс в студенческой группе' для реализации
    связи многие-ко-многим
    """

    def __init__(self, group_id, course_id):
        self.group_id = group_id
        self.course_id = course_id


# Учебные курсы
courses = [
    Course(1, 'БКИТ'),
    Course(2, 'Английский язык'),
    Course(3, 'Теория вероятности'),
    Course(4, 'Экология')
]

# Студенческие группы
groups = [
    Group(1, 'ИУ5-32Б', 18, 1),
    Group(2, 'ИУ5-31Б', 20, 1),
    Group(3, 'ИУ5-33Б', 22, 1),
    Group(4, 'ИУ5-34Б', 21, 1),
    Group(5, 'Э9-31Б', 16, 4),
    Group(6, 'Э9-22Б', 18, 4),
    Group(7, 'Э9-33Б', 20, 4),
]
```

```python
groups_courses = [
    GroupCourse(1, 1),
    GroupCourse(1, 2),
    GroupCourse(1, 3),

    GroupCourse(1, 4),
    GroupCourse(2, 1),
    GroupCourse(2, 2),

    GroupCourse(2, 3),
    GroupCourse(2, 4),
    GroupCourse(3, 1),

    GroupCourse(3, 2),
    GroupCourse(3, 3),
    GroupCourse(3, 4),

    GroupCourse(4, 1),
    GroupCourse(4, 2),
    GroupCourse(4, 3),

    GroupCourse(4, 4),
    GroupCourse(5, 2),
    GroupCourse(5, 3),

    GroupCourse(5, 4),
    GroupCourse(6, 2),
    GroupCourse(6, 3),

    GroupCourse(6, 4),
    GroupCourse(7, 2),
    GroupCourse(7, 3),
    GroupCourse(7, 4),
]


def sortByGroup(table):
    return sorted(table, key=itemgetter(0))


def sortByCount(table, courses):
    result = []
    # Перебираем все учебные курсы
    for c in courses:
        # Список учебных курсов
        c_groups = list(filter(lambda i: i[2] == c.name, table))
        # Если учебный курс не пустой
        if len(c_groups) > 0:
            # Количество студенческих групп
            c_lens = [len for _, len, _ in c_groups]
            result.append((c.name, len(c_lens)))

    # Сортировка по количеству студенческих групп
    return sorted(result, key=itemgetter(1), reverse=False)


def endsWith(table, groups):
```

```
    result = {}
    # Перебираем все учебные курсы
    for g in groups:
        if g.name.endswith('31Б'):
            # Список учебных курсов у студенческих групп
            g_courses = list(filter(lambda i: i[0] == g.name, table))
            # Только название студенческой группы
            g_courses_name = [x[2] for x in g_courses]
            # ключ - студенческая группа, значение - список учебных курсов
            result[g.name] = g_courses_name

    return result


def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(g.name, g.studentsCount, c.name)
                   for c in courses
                   for g in groups
                   if g.course_id == c.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(c.name, gc.course_id, gc.group_id)
                         for c in courses
                         for gc in groups_courses
                         if c.id == gc.course_id]

    many_to_many = [(g.name, g.studentsCount, course_name)
                    for course_name, course_id, group_id in many_to_many_temp
                    for g in groups if g.id == group_id]

    print('Задание Б1')
    print(sortByGroup(one_to_many))

    print('\nЗадание Б2')
    print(sortByCount(one_to_many, courses))

    print('\nЗадание Б3')
    print(endsWith(many_to_many, groups))


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

Файл с тестами – TDDTests.py:

```
from main import Group, Course, GroupCourse, sortByGroup, sortByCount, endsWith
import unittest


class Tests(unittest.TestCase):
    def setUp(self):
        # Студенческие группы
        self.groups = [
            Group(1, 'ИУ5-32Б', 18, 1),
            Group(2, 'ИУ5-31Б', 20, 1),
            Group(3, 'ИУ5-33Б', 22, 1),
            Group(4, 'ИУ5-34Б', 21, 1),
            Group(5, 'Э9-31Б', 16, 4),
            Group(6, 'Э9-22Б', 18, 4),
            Group(7, 'Э9-33Б', 20, 4),
        ]

        # Учебные курсы
        self.courses = [
            Course(1, 'БКИТ'),
            Course(2, 'Английский язык'),
            Course(3, 'Теория вероятности'),
            Course(4, 'Экология')
        ]

        self.groups_courses = [
            GroupCourse(1, 1),
            GroupCourse(1, 2),
            GroupCourse(1, 3),
            GroupCourse(1, 4),
            GroupCourse(2, 1),
            GroupCourse(2, 2),
            GroupCourse(2, 3),
            GroupCourse(2, 4),
            GroupCourse(3, 1),
            GroupCourse(3, 2),
            GroupCourse(3, 3),
            GroupCourse(3, 4),
            GroupCourse(4, 1),
            GroupCourse(4, 2),
            GroupCourse(4, 3),
            GroupCourse(4, 4),
            GroupCourse(5, 2),
            GroupCourse(5, 3),
            GroupCourse(5, 4),
            GroupCourse(6, 2),
            GroupCourse(6, 3),
            GroupCourse(6, 4),
            GroupCourse(7, 2),
            GroupCourse(7, 3),
            GroupCourse(7, 4),
        ]
        # Соединение данных один-ко-многим
        self.one_to_many = [(g.name, g.studentsCount, c.name)
```

```
                    for c in self.courses
                    for g in self.groups
                    if g.course_id == c.id]

        # Соединение данных многие-ко-многим
        self.many_to_many_temp = [(c.name, gc.course_id, gc.group_id)
                        for c in self.courses
                        for gc in self.groups_courses
                        if c.id == gc.course_id]

        self.many_to_many = [(g.name, g.studentsCount, course_name)
            for course_name, course_id, group_id in self.many_to_many_temp
            for g in self.groups if g.id==group_id]

    def testSortByGroup(self):
        result = sortByGroup(self.one_to_many)
        desired_result = [('ИУ5-31Б', 20, 'БКИТ'), ('ИУ5-32Б', 18, 'БКИТ'),
('ИУ5-33Б', 22, 'БКИТ'), ('ИУ5-34Б', 21, 'БКИТ'), ('Э9-22Б', 18, 'Экология'),
('Э9-31Б', 16, 'Экология'), ('Э9-33Б', 20, 'Экология')]
        self.assertEqual(result, desired_result)

    def testSortByCount(self):
        result = sortByCount(self.one_to_many, self.courses)
        desired_result = [('Экология', 3), ('БКИТ', 4)]
        self.assertEqual(result, desired_result)

    def testEndsWith(self):
        result = endsWith(self.many_to_many, self.groups)
        desired_result = {'ИУ5-31Б': ['БКИТ', 'Английский язык', 'Теория
вероятности', 'Экология'], 'Э9-31Б': ['Английский язык', 'Теория вероятности',
'Экология']}
        self.assertEqual(result, desired_result)
```

Экранные формы с примерами выполнения программы

Результаты выполнения:

```
Задание Б1
[('ИУ5-31Б', 20, 'БКИТ'), ('ИУ5-32Б', 18, 'БКИТ'), ('ИУ5-33Б', 22,
'БКИТ'), ('ИУ5-34Б', 21, 'БКИТ'), ('Э9-22Б', 18, 'Экология'), ('Э9-
31Б', 16, 'Экология'), ('Э9-33Б', 20, 'Экология')]

Задание Б2
[('Экология', 3), ('БКИТ', 4)]

Задание Б3
{'ИУ5-31Б': ['БКИТ', 'Английский язык', 'Теория вероятности',
'Экология'], 'Э9-31Б': ['Английский язык', 'Теория вероятности',
'Экология']}
```