

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

Факультет информатика и системы управления
Кафедра системы обработки информации и управления

Курс «Парадигмы и конструкции языков программирования»
Отчет по рубежному контролю №2
Вариант 28

Выполнил:
студент группы ИУ5-32Б:
Багандов М. А.
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.
Подпись и дата:

Постановка задачи

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

rk2.py:

```
import sys

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

class StudentGroup:
    def __init__(self, id, name, students_count, department_id):
        self.id = id
        self.name = name
        self.students_count = students_count
        self.department_id = department_id

class GroupDepartment:
    def __init__(self, group_id, department_id):
        self.group_id = group_id
        self.department_id = department_id

def get_group_list(groups, departments):

    one_to_many = [(g.name, g.students_count, d.name)
                    for d in departments
                    for g in groups
                    if g.department_id == d.id]

    return sorted(one_to_many, key=lambda x: x[2])

def get_students_count(groups, departments):
    dep_total = []
    for d in departments:
        count = sum(g.students_count for g in groups if g.department_id == d.id)
        dep_total.append((d.name, count))

    return sorted(dep_total, key=lambda x: x[1], reverse=True)

def get_groups_with_keyword(groups, departments, group_departments, keyword):
    #many-to-many
    many_to_many_temp = [(d.name, gd.department_id, gd.group_id)
                          for d in departments
                          for gd in group_departments
                          if d.id == gd.department_id]

    many_to_many = [(g.name, d_name)
                    for d_name, dep_id, grp_id in many_to_many_temp
                    for g in groups if g.id == grp_id]

    result = {}
    for d in departments:
```

```

        if keyword.lower() in d.name.lower():
            related_groups = [g_name for g_name, d_name in many_to_many if d_name ==
d.name]
            result[d.name] = related_groups

    return result

def main():
    departments = [
        Department(1, "Кафедра информатики"),
        Department(2, "Кафедра математики"),
        Department(3, "Кафедра физики"),
    ]

    groups = [
        StudentGroup(1, "Группа ИНФ-101", 25, 1),
        StudentGroup(2, "Группа ИНФ-102", 30, 1),
        StudentGroup(3, "Группа МАТ-201", 28, 2),
        StudentGroup(4, "Группа ФИЗ-301", 22, 3),
        StudentGroup(5, "Группа ФИЗ-302", 26, 3),
    ]

    group_departments = [
        GroupDepartment(1, 1),
        GroupDepartment(2, 1),
        GroupDepartment(3, 2),
        GroupDepartment(4, 3),
        GroupDepartment(5, 3),
        GroupDepartment(2, 3),
    ]

    print("\n1) Список групп:")
    print(get_group_list(groups, departments))

    print("\n2) Количество студентов:")
    print(get_students_count(groups, departments))

    print("\n3) Кафедры 'кафедра':")
    print(get_groups_with_keyword(groups, departments, group_departments, "кафедра"))

if __name__ == '__main__':
    main()

```

rk2_tests.py:

```
import unittest

from rk2 import Department, StudentGroup, GroupDepartment, get_group_list,
get_students_count, get_groups_with_keyword

class TestRK2(unittest.TestCase):
    def setUp(self):
        """Инициализация данных перед каждым тестом"""
        self.departments = [
            Department(1, "Кафедра А"),
            Department(2, "Кафедра Б"),
            Department(3, "Центр В"),
        ]

        self.groups = [
            StudentGroup(1, "Гр-А1", 10, 1),
            StudentGroup(2, "Гр-А2", 20, 1),
            StudentGroup(3, "Гр-Б1", 15, 2),
            StudentGroup(4, "Гр-Б1", 5, 3),
        ]

        self.group_departments = [
            GroupDepartment(1, 1),
            GroupDepartment(2, 1),
            GroupDepartment(3, 2),
            GroupDepartment(4, 3),
        ]

    def test_1_one_to_many(self):
        result = get_group_list(self.groups, self.departments)

        self.assertEqual(result[0][2], "Кафедра А")
        self.assertEqual(len(result), 4)
        self.assertEqual(result[0], ("Гр-А1", 10, "Кафедра А"))

    def test_2_aggregation(self):
        result = get_students_count(self.groups, self.departments)
        expected = [
            ("Кафедра А", 30),
            ("Кафедра Б", 15),
            ("Центр В", 5)
        ]

        self.assertEqual(result, expected)
        self.assertEqual(result[0][1], 30)

    def test_3_many_to_many(self):
        result = get_groups_with_keyword(self.groups, self.departments,
self.group_departments, "кафедра")
        self.assertNotIn("Центр В", result)
        self.assertIn("Кафедра А", result)
```

```
self.assertIn("Гр-А1", result["Кафедра А"])
self.assertIn("Гр-А2", result["Кафедра А"])

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

Скриншот работы приложения

```
rk2_tests.py rk2.py
> python -m unittest rk2_tests.py
...
-----
Ran 3 tests in 0.000s

OK
> cd ..
> git add .
```

Рисунок 1. Вывод результатов программы