

Защищено:  
Гапанюк Ю.Е.

"\_\_" \_\_\_\_\_ 2025 г.

Демонстрация:  
Константинов А.А.

"\_\_" \_\_\_\_\_ 2025 г.

**Отчет по рубежному контролю № 2 по курсу  
Парадигмы и конструкции языков программирования  
Вариант №26**

5

(количество листов)

студент группы ИУ5Ц-53Б

Константинов А.А.

\_\_\_\_\_  
(подпись)

"\_\_" \_\_\_\_\_ 2025 г.

## 1. Описание задания

Вариант предметной области:

26	Студенческая группа	Учебный курс
----	---------------------	--------------

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).

## 2. Листинг программы

```
class StudyCourse:
    def __init__(self, course_id, name):
        self.id = course_id
        self.name = name

class StudentGroup:
    def __init__(self, group_id, name, student_count, course_id):
        self.id = group_id
        self.name = name
        self.student_count = student_count
        self.course_id = course_id

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

#рефакторинг

def get_groups_ending_with_01(groups, courses):
    """Возвращает группы с названием '-01' и их курсы"""
    course_by_id = {c.id: c.name for c in courses}
    return [
        (group.name, course_by_id.get(group.course_id))
        for group in groups
        if group.name.endswith("-01")
    ]

def get_course_statistics(groups, courses):
    """Возвращает статистику по курсам"""
    result = []
    for course in courses:
        related_groups = [g for g in groups if g.course_id == course.id]
        if related_groups:
            total = sum(g.student_count for g in related_groups)
            avg = total / len(related_groups)
            result.append((course.name, avg, total, len(related_groups)))
    return sorted(result, key=lambda x: x[1])

def get_math_courses_with_groups(courses, groups, relations):
    """Возвращает курсы 'Математика*' и связанные группы (многие-ко-многим)"""
    math_courses = [c for c in courses if c.name.startswith("Математика")]
    groups_by_id = {g.id: g for g in groups}

    result = {}
    for course in math_courses:
```

```

        related_group_ids = [
            rel.group_id for rel in relations if rel.course_id == course.id
        ]
        result[course.name] = [
            groups_by_id[g_id].name for g_id in related_group_ids
        ]
    return result

#Данные

courses = [
    StudyCourse(1, "Математика для инженеров"),
    StudyCourse(2, "Физика"),
    StudyCourse(3, "Математический анализ"),
    StudyCourse(4, "Программирование"),
    StudyCourse(5, "Математика для физиков")
]

groups = [
    StudentGroup(1, "ИУ-101", 25, 1),
    StudentGroup(2, "ФИ-201", 30, 2),
    StudentGroup(3, "ИУ-301", 20, 4),
    StudentGroup(4, "МТ-102", 28, 1),
    StudentGroup(5, "ИБМ-202", 32, 2),
    StudentGroup(6, "МТ-103-01", 22, 3),
    StudentGroup(7, "РК-302-01", 18, 4),
    StudentGroup(8, "ИБМ-203-01", 26, 5)
]

group_courses = [
    GroupCourse(1, 1),
    GroupCourse(2, 2),
    GroupCourse(3, 4),
    GroupCourse(4, 1),
    GroupCourse(5, 2),
    GroupCourse(6, 3),
    GroupCourse(7, 4),
    GroupCourse(8, 5),
    GroupCourse(1, 3),
    GroupCourse(2, 5),
    GroupCourse(3, 1)
]

#Тесты (TDD)

if __name__ == "__main__":
    import unittest

    class TestStudySystem(unittest.TestCase):

        def test_groups_ending_with_01(self):
            result = get_groups_ending_with_01(groups, courses)
            self.assertIn("МТ-103-01", "Математический анализ", result)
            self.assertIn("РК-302-01", "Программирование", result)

        def test_course_statistics(self):
            stats = get_course_statistics(groups, courses)
            course_names = [s[0] for s in stats]
            self.assertIn("Математика для инженеров", course_names)

```

```
def test_math_courses_many_to_many(self):
    result = get_math_courses_with_groups(courses, groups, group_courses)
    self.assertIn("Математика для инженеров", result)
    self.assertTrue(len(result["Математика для инженеров"]) > 0)

unittest.main()
```

### 3. Результаты работы программы

```
PS C:\Users\anton\OneDrive\Рабочий стол\Учеба 3курс\ПиКЯП\РК2> & C:\Us  
чий стол\Учеба 3курс\ПиКЯП\РК2\RK2.py"
```

```
...
```

```
-----  
Ran 3 tests in 0.000s
```

```
OK
```

```
PS C:\Users\anton\OneDrive\Рабочий стол\Учеба 3курс\ПиКЯП\РК2> █
```