```
import unittest
from main import StudentGroup, Department, StudentsInDepartment


class TestStudentGroup(unittest.TestCase):
    def setUp(self):
        self.student_groups = [
            StudentGroup(1, '1 группа', [1, 2]),
            StudentGroup(2, '2 группа', [2, 3]),
            StudentGroup(3, '3 группа', [1, 3])
        ]

        self.departments = [
            Department(1, 'Кафедра математики', [1, 3]),
            Department(2, 'Кафедра физики', [1]),
            Department(3, 'Кафедра физического воспитания', [2])
        ]

        self.students_in_departments = [
            StudentsInDepartment(1, 1, 90),
            StudentsInDepartment(2, 2, 85),
            StudentsInDepartment(3, 1, 92),
            StudentsInDepartment(4, 3, 88),
            StudentsInDepartment(5, 3, 94)
        ]

    def test_filter_departments(self):
        filtered_departments = [(department.department_name, [group.group_name for
group in self.student_groups if group.group_id in department.student_ids]) for
department in self.departments if 'кафедра' in department.department_name.lower()]
        expected_result = [("Кафедра математики", ['1 группа', '3 группа']), ("Кафедра
физики", ['1 группа']), ("Кафедра физического воспитания", ['2 группа'])]
        self.assertEqual(filtered_departments, expected_result)

    def test_calculate_average_grades(self):
        average_grades = {}
        for department in self.departments:
            grades = [student.grade for student in self.students_in_departments if
student.department_id == department.department_id]
            if grades:
                average_grade = round(sum(grades) / len(grades), 2)
                average_grades[department.department_name] = average_grade
        expected_result = {'Кафедра математики': 91.0, 'Кафедра физики': 85.0,
'Кафедра физического воспитания': 91.0}
        self.assertEqual(average_grades, expected_result)

    def test_group_departments(self):
        group_names = [group.group_name for group in self.student_groups if
group.group_name[0].isdigit() and group.group_name[0] == '1']
        group_departments = {group.group_name: sorted([department.department_name for
department in self.departments if department.department_id in group.department_ids])
for group in self.student_groups if group.group_name in group_names}
        expected_result = {'1 группа': ['Кафедра физики', 'Кафедра математики']}
        # Пересчитаем количество элементов, игнорируя порядок
        self.assertCountEqual(group_departments.keys(), expected_result.keys())
        for key in group_departments.keys():
            self.assertCountEqual(group_departments[key], expected_result[key])

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

Результат выполнения программы:

```
Ran 3 tests in 0.001s

OK

Process finished with exit code 0
```