

## РК2 Воробьев Егор ИУ5-32Б Вариант № 30/A

Листинг кода RK2\_main.py:

```
from operator import itemgetter

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

class University:
    def __init__(self, id, name, students_count, faculty_id):
        self.id = id
        self.name = name
        self.students_count = students_count
        self.faculty_id = faculty_id

class FacultyUniversity:
    def __init__(self, faculty_id, university_id):
        self.faculty_id = faculty_id
        self.university_id = university_id

def get_data():
    faculties = [
        Faculty(1, 'Факультет информатики'),
        Faculty(2, 'Экономический факультет'),
        Faculty(3, 'Юридический факультет'),
    ]
    universities = [
        University(1, 'МГУ', 1500, 1),
        University(2, 'МГИМО', 1200, 2),
        University(3, 'МФТИ', 800, 1),
    ]
    faculties_universities = [
        FacultyUniversity(1, 1),
        FacultyUniversity(1, 3),
        FacultyUniversity(2, 2),
        FacultyUniversity(1, 2),
    ]
    return faculties, universities, faculties_universities

def prepare_one_to_many(faculties, universities):
    return [(u.name, u.students_count, f.name)
            for f in faculties
            for u in universities
            if u.faculty_id == f.id]

def prepare_many_to_many(faculties, universities, faculties_universities):
    many_to_many_temp = [(f.name, fu.faculty_id, fu.university_id)
                         for f in faculties
                         for fu in faculties_universities
                         if f.id == fu.faculty_id]

    return [(u.name, u.students_count, fac_name)
```

```

        for fac_name, fac_id, uni_id in many_to_many_temp
        for u in universities if u.id == uni_id]

def query_a1(one_to_many):
    """Список связанных университетов и факультетов, отсортированный по количеству
    студентов"""
    return sorted(one_to_many, key=itemgetter(1))

def query_a2(faculties, one_to_many):
    """Список факультетов с суммарным количеством студентов, отсортированный по
    убыванию"""
    res = []
    for f in faculties:
        f_universities = list(filter(lambda i: i[2] == f.name, one_to_many))
        if len(f_universities) > 0:
            f_students_sum = sum([students for _, students, _ in f_universities])
            res.append((f.name, f_students_sum))
    return sorted(res, key=itemgetter(1), reverse=True)

def query_a3(faculties, many_to_many):
    """Словарь факультетов (содержащих 'факультет') и их университетов"""
    res = {}
    for f in faculties:
        if 'факультет' in f.name.lower():
            f_universities = list(filter(lambda i: i[2] == f.name, many_to_many))
            f_universities_names = [name for name, _, _ in f_universities]
            res[f.name] = f_universities_names
    return res

if __name__ == '__main__':
    facs, unis, facs_unis = get_data()
    o2m = prepare_one_to_many(facs, unis)
    m2m = prepare_many_to_many(facs, unis, facs_unis)

    print('A1:', query_a1(o2m))
    print('A2:', query_a2(facs, o2m))
    print('A3:', query_a3(facs, m2m))

```

Вывод программы:

A1: [('МФТИ', 800, 'Факультет информатики'), ('МГИМО', 1200, 'Экономический факультет'), ('МГУ', 1500, 'Факультет информатики')]

A2: [('Факультет информатики', 2300), ('Экономический факультет', 1200)]

A3: {'Факультет информатики': ['МГУ', 'МФТИ', 'МГИМО'], 'Экономический факультет': ['МГИМО'], 'Юридический факультет': []}

Листинг кода RK2\_test.py:

```

import unittest
from RK2_main import (Faculty, University, query_a1, query_a2, query_a3,

```

```

        prepare_one_to_many, prepare_many_to_many)

class TestRK2(unittest.TestCase):

    def setUp(self):
        self.facs = [
            Faculty(1, 'Технический факультет'),
            Faculty(2, 'Другой отдел')
        ]
        self.unis = [
            University(1, 'Университет 1', 100, 1),
            University(2, 'Университет 2', 50, 1)
        ]
        self.o2m = prepare_one_to_many(self.facs, self.unis)

    def test_query_a1_sorting(self):
        """Тест сортировки по количеству студентов (A1)"""
        result = query_a1(self.o2m)
        self.assertEqual(result[0][1], 50)
        self.assertEqual(result[1][1], 100)

    def test_query_a2_sum(self):
        """Тест суммирования студентов по факультетам (A2)"""
        result = query_a2(self.facs, self.o2m)
        self.assertEqual(result[0], ('Технический факультет', 150))

    def test_query_a3_filter(self):
        """Тест фильтрации по слову 'факультет' (A3)"""
        m2m = [('Университет 1', 100, 'Технический факультет'),
                ('Университет 2', 50, 'Другой отдел')]

        result = query_a3(self.facs, m2m)
        self.assertIn('Технический факультет', result)
        self.assertNotIn('Другой отдел', result)
        self.assertEqual(len(result), 1)

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

```

Вывод программы:

...

Ran 3 tests in 0.000s

OK