Московский государственный технический университет им. Н.Э. Баумана

Факультет ИУ
Кафедра ИУ5

Курс «Основы информатики»
Отчет по Рубежному контролю №2

Выполнил студент группы ИУ5-33Б:
Бакушев И.О.
Подпись и дата:

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

Москва, 2024 г

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

```
import unittest
from operator import itemgetter

class Faculty:

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


class University:

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


class FacultyUniversity:

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



faculties = [
    Faculty(1, 'Факультет информационных технологий', 150),
    Faculty(2, 'Факультет экономики', 120),
    Faculty(3, 'Факультет гуманитарных наук', 80),
    Faculty(4, 'Факультет права', 100),
]


universities = [
    University(1, 'Московский государственный университет'),
    University(2, 'Санкт-Петербургский государственный
университет'),
    University(3, 'Казанский федеральный университет'),
    University(4, 'Сибирский федеральный университет'),
    University(5, 'МГТУ им. Баумана'),
]

decans = [
    ('Иванов', 1),
```

```
    ('Петров', 2),
    ('Сидоров', 3),
    ('Кузнецов', 4),
    ('Смирнов', 1),
]

faculties_universities = [
    FacultyUniversity(1, 1),
    FacultyUniversity(2, 2),
    FacultyUniversity(1, 3),
    FacultyUniversity(2, 4),
    FacultyUniversity(3, 1),
    FacultyUniversity(4, 2),
    FacultyUniversity(3, 4),
    FacultyUniversity(5, 1),
    FacultyUniversity(5, 2),
]


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

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

def calculate_university_employees():
    res_12_unsorted = []
    for u in universities:
        u_faculties = list(filter(lambda i: i[1] == u.name, get_one_to_many()))
        if len(u_faculties) > 0:
            total_employees = sum(f.num_employees for f in faculties if f.name in [x[0] for x in
u_faculties])
            res_12_unsorted.append((u.name, total_employees))
    return sorted(res_12_unsorted, key=itemgetter(1))

def find_decans_universities():
    res_13_new = []
    for d in decans:
        if d[0].endswith('ов'):
            for u_name, university_id, faculty_id in [(u.name, fu.university_id, fu.faculty_id)
                                                      for u in universities
                                                      for fu in faculties_universities
                                                      if u.id == fu.university_id]:
                if d[1] == faculty_id:
                    res_13_new.append((d[0], u_name))
                    break
    return res_13_new


class TestUniversity(unittest.TestCase):

    def test_b1_sorted_one_to_many(self):
        expected = sorted(get_one_to_many(), key=itemgetter(0))
```

```
        self.assertEqual(len(expected), 4)
        self.assertEqual(expected[0][0], 'Факультет гуманитарных наук')
        self.assertEqual(expected[1][0], 'Факультет информационных
технологий')
        self.assertEqual(expected[2][0], 'Факультет права')
        self.assertEqual(expected[3][0], 'Факультет экономики')


    def test_b2_university_employees(self):
        expected = calculate_university_employees()
        self.assertEqual(len(expected), 4)
        self.assertEqual(expected[0][0], 'Казанский федеральный
университет')
        self.assertEqual(expected[0][1], 80)
        self.assertEqual(expected[1][0], 'Сибирский федеральный
университет')
        self.assertEqual(expected[1][1], 100)


    def test_b3_decans_universities(self):
        expected = find_decans_universities()
        self.assertEqual(len(expected), 5)
        self.assertIn(('Смирнов', 'Московский государственный
университет'), expected)
        self.assertIn(('Иванов', 'Московский государственный
университет'), expected)
        self.assertIn(('Кузнецов', 'Санкт-Петербургский
государственный университет'), expected)
if __name__ == '__main__':
    unittest.main()
```

Анализ результатов

```
C:\Users\encourage\PycharmProjects\pythonProject\venv\Scripts\python.ex
...
----------------------------------------------------------------------
Ran 3 tests in 0.002s

OK
```