Московский государственный технический университет им. Н.Э. Баумана

Факультет «Радиотехнический»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»

Отчёт по рубежному контролю №2
Вариант №15

Выполнил:
студент группы РТ5-31Б
Мицкевич Владислав
Подпись и дата:

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

Москва, 2021 г

Описание задания

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

```
class File:
    def __init__(self, id_file, name_file, size_file, file_extension, id_directory):
        self.name_file = name_file
        self.size_file = size_file
        self.file_extension = file_extension
        self.id_file = id_file
        self.id_directory = id_directory
class Directory:
    def __init__(self, name_directory, size_directory, id_directory):
        self.name_directory = name_directory
        self.size_directory = size_directory
        self.id_directory = id_directory
    def __repr__(self):
        return f'{self.size_directory} {self.name_directory} {self.size_directory}'

class DirFile:
    def __init__(self, id_directory, id_file):
        self.id_directory = id_directory
        self.id_file = id_file

files = [File(7, "Lab1", 101, "docx", 1), File(5, "Lab2", 150, "docx", 1), File(120,
"Lab3", 98, "docx", 1),
        File(4, "Math", 700, "pdf", 3), File(27, "Python_Book", 850, "pdf", 3),
File(2, "It", 1500, "djvu", 3),
        File(145, "Tester", 160, "py", 7)]

directories = [Directory("Labs", 340, 1), Directory("Film", 0, 45),
Directory("BookForLabs", 3050, 3), Directory("Pycharm_programmers", 160, 7)]



one_to_many = [(x,[y for y in files if y.id_directory == x.id_directory]) for x in
directories]
many_to_many = [DirFile(1, 7), DirFile(1, 5), DirFile(1, 120), DirFile(1, 5416),
DirFile(3, 4), DirFile(7, 145), DirFile(3, 27)]


def test1():
#«Отдел» и «Сотрудник» связаны соотношением один - ко - многим.Выведите список всех
```

```
#отделов, у которых в названии присутствует слово «отдел», и список работающих в них сотрудников.

    a = str()
    for dir, fil in one_to_many:
        if dir.name_directory.find('Labs') != -1:
            a += dir.name_directory + ": "
            for x in fil:
                a += x.name_file + " "
            a += "\n"
    return a

def test2():
#«Отдел» и «Сотрудник» связаны соотношением один - ко - многим.Выведите список отделов со средней зарплатой
#сотрудников в каждом отделе, отсортированный по средней зарплате.Средняя зарплата должна быть округлена до 2 знака после запятой
    def func(lst):
        return lst[1]
    lst = list()
    for dir, fil in one_to_many:
        sr = 0
        sum = 0
        if len(fil) != 0:
            for x in fil:
                sum += x.size_file
            sr = round(sum / len(fil), 2)
            lst.append([dir.name_directory, sr])
        else:
            lst.append([dir.name_directory, 0])
    a = sorted(lst, key=func)
    return(a)


def test3():
#«Отдел» и «Сотрудник» связаны соотношением многие - ко - многим.Выведите список всех сотрудников, у
#которых фамилия начинается с буквы «А», и названия их отделов.
    a = ''
    #Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех сотрудников,
    # у которых фамилия начинается с буквы «А», и названия их отделов.
    for x in many_to_many:
        for y in files:
            if x.id_file == y.id_file:
                if y.name_file[0] == 'L':
                    a += y.name_file + "\t"
                    for z in directories:
                        if z.id_directory == x.id_directory:
                            a += z.name_directory + "\n"
    return a

def main():
    print(test1())
```

```
    print(test2())
    print(test3())

main()
```

test.py

```
import unittest
import PK2
class test(unittest.TestCase):
    def test_tasks(self):
        self.assertEqual(PK2.test1(),'Labs: Lab1 Lab2 Lab3 \nBookForLabs: Math
Python_Book It \n')
        self.assertEqual(PK2.test2(), [['Film', 0], ['Labs', 116.33],
['Pycharm_programmers', 160.0], ['BookForLabs', 1016.67]])
        self.assertEqual(PK2.test3(),'Lab1\tLabs\nLab2\tLabs\nLab3\tLabs\n')
if __name__ == "__main__":
    unittest.main()
```

Пример работы программы

```
PS C:\Users\vlad2\OneDrive\Рабочий стол\лабы> cd .\RK2\
PS C:\Users\vlad2\OneDrive\Рабочий стол\лабы\RK2> python .\test.py
Labs: Lab1 Lab2 Lab3
BookForLabs: Math Python_Book It

[['Film', 0], ['Labs', 116.33], ['Pycharm_programmers', 160.0], ['BookForLabs', 1016.67]]
Lab1    Labs
Lab2    Labs
Lab3    Labs

.
----------------------------------------------------------------------
Ran 1 test in 0.000s

OK
PS C:\Users\vlad2\OneDrive\Рабочий стол\лабы\RK2>
```