Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»
Отчет по рубежному контролю №2

Выполнил:
студент группы ИУ5-31Б
Коваленко
Алексей Викторович

Подпись:_____

Дата:_____

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич

Подпись:_____

Дата:_____

Москва, 2021 г.

Вариант предметной области -9. («Операционная система» и «Компьютер»)
Вариант запросов – А.

Описание задания

Задание:

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

Папка patterns

Файл main.py

```
from operator import itemgetter

#Сотрудник
class OC:
    def __init__(self, id, name, cost, pc_id):
        self.id = id
        self.name = name
        self.cost = cost
        self.pc_id = pc_id

#Отдел
class PC:
    def __init__(self, id, name):
        self.id = id
        self.name = name


class OCPC:
    def __init__(self, PC_id, OC_id):
        self.PC_id = PC_id
        self.OC_id = OC_id


PCs = [
    PC(1, 'Рабочий ноутбук'),
    PC(2, 'Домашний компьютер'),
    PC(3, 'Рабочий компьютер'),

    PC(11, 'Другой рабочий ноутбук'),
    PC(22, 'Другой домашний компьютер'),
    PC(33, 'Другой рабочий компьютер')
]

OCs = [
    OC(1, 'Windows 10', 7000,1),
    OC(2, 'Windows 11', 11000,2),
    OC(3, 'Arch Linux', 500,3),
    OC(4, 'Windows 10', 10000,3),
]

OC_PC = [
    OCPC(1,1),
    OCPC(2,2),
    OCPC(3,3),
    OCPC(3,4),

    OCPC(11,1),
    OCPC(22,2),
    OCPC(33,3),
    OCPC(33,4)
]

def create_one_no_many(PCs,OCs):
    one_to_many = [(oc.name, oc.cost, pc.name)
                   for pc in PCs
                   for oc in OCs
                   if oc.pc_id == pc.id]
```

```
    return one_to_many

def create_many_to_many(PCs,OCs, OC_PC):
    many_to_many_temp = [(pc.name, op.PC_id, op.OC_id)
                         for pc in PCs
                         for op in OC_PC
                         if pc.id == op.PC_id]
    many_to_many = [(oc.name, oc.cost, pc_name)
                    for pc_name, pc_id, oc_id in many_to_many_temp
                    for oc in OCs if oc.id==oc_id]
    return many_to_many

def task_one(otm):
    return sorted(otm, key=itemgetter(2))

def task_two(PCs,otm):
    res_12_unsorted = []
    for pc in PCs:
        pc_ocs = list(filter(lambda i: i[2]==pc.name, otm))
        if len(pc_ocs) > 0:
            PC_cost = [cost for a, cost, a in pc_ocs]
            PC_cost_sum = sum(PC_cost)
            res_12_unsorted.append((pc.name, PC_cost_sum))

    return sorted(res_12_unsorted, key=itemgetter(1), reverse=True)

def task_three(PCs,mtm):
    res_13 = {}
    for pc in PCs:
        if 'компьютер' in pc.name:
            op = list(filter(lambda i: i[2]==pc.name, mtm))
            op_names = [x for x,a,a in op]
            res_13[pc.name] = op_names

    return res_13


def main():
    one_to_many=create_one_no_many(PCs,OCs)
    many_to_many=create_many_to_many(PCs,OCs,OC_PC)
    print('Задание А1')
    print(task_one(one_to_many))
    print('\nЗадание А2')
    print(task_two(PCs,one_to_many))
    print('\nЗадание А3')
    print(task_three(PCs,many_to_many))


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

Файл tests.py

```
from main import *

PCs = [
    PC(1, 'Рабочий ноутбук'),
    PC(2, 'Домашний компьютер'),
    PC(3, 'Рабочий компьютер'),

    PC(11, 'Другой рабочий ноутбук'),
    PC(22, 'Другой домашний компьютер'),
    PC(33, 'Другой рабочий компьютер')
```

```
]

OCs = [
    OC(1, 'Windows 10', 7000,1),
    OC(2, 'Windows 11', 11000,2),
    OC(3, 'Arch Linux', 500,3),
    OC(4, 'Windows 10', 10000,3),
]


OC_PC = [
    OCPC(1,1),
    OCPC(2,2),
    OCPC(3,3),
    OCPC(3,4),

    OCPC(11,1),
    OCPC(22,2),
    OCPC(33,3),
    OCPC(33,4)
]

otm = create_one_no_many(PCs,OCs)
mtm=create_many_to_many(PCs,OCs,OC_PC)
def test_one():
    result = [('Windows 11', 11000, 'Домашний компьютер'), ('Arch Linux',
500, 'Рабочий компьютер'),\
              ('Windows 10', 10000, 'Рабочий компьютер'), ('Windows 10', 7000,
'Рабочий ноутбук')]
    assert task_one(otm) == result

def test_two():
    result = [('Домашний компьютер', 11000), ('Рабочий компьютер', 10500),
('Рабочий ноутбук', 7000)]
    assert task_two(PCs,otm) == result

def test_three():
    result = {'Домашний компьютер': ['Windows 11'], 'Рабочий компьютер':
['Arch Linux', 'Windows 10']\
           , 'Другой домашний компьютер': ['Windows 11'], 'Другой рабочий
компьютер': ['Arch Linux', 'Windows 10']}
    assert task_three(PCs,mtm) == result
```

Результат выполнения программы

```
============================= test session starts ==============================
collecting ... collected 3 items

tests.py::test_one PASSED                                                [ 33%]
tests.py::test_two PASSED                                                [ 66%]
tests.py::test_three PASSED                                              [100%]

============================== 3 passed in 0.03s ===============================

Process finished with exit code 0
```