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 РК №2

Выполнил:
студент группы ИУ5-34Б:
Такташова Дарья Юрьевна
Подпись и дата:

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

Москва, 2022 г.

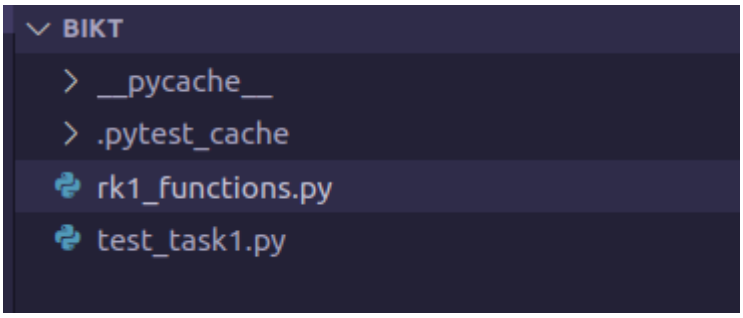
Задание:

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).

Код программы:



rk1_functions.py:

```python
'''
Вариант 19
Деталь Производитель
Вариант Г.
1) «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А»,
 и список работающих в них сотрудников.
2) «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов с максимальной зарплатой сотрудников в каждом отделе,
 отсортированный по максимальной зарплате.
3) «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех связанных сотрудников и отделов, отсортированный по отделам,
сортировка по сотрудникам произвольная.
'''
from heapq import merge
from operator import itemgetter

class Detail:
    """Деталь"""
    def __init__(self, id, name, man_id, price):
        self.id = id
        self.name = name
        self.man_id = man_id
        self.price = price

class Manufacturer:
    """Производитель"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

class DetMan:
    """
    'Детали производителя' для реализации
    связи многие-ко-многим
    """
    def __init__(self, det_id, man_id):
        self.det_id = det_id
        self.man_id = man_id

def task1 (man, det):
    one_to_many = [(d.name, m.name, d.price)
        for m in man
        for d in det
        if d.man_id == m.id]

    res_1 = {}
    for i in one_to_many:
        if i[1][0] == 'A':
            if res_1.get(i[1]) == None:
                res_1[i[1]] = [i[0]]
            else:
                res_1[i[1]].append(i[0])
    return res_1
```

```python
def task2 (man, det):
    res_2 = []
    for m in man:
        det_tmp = list((m.name, d.price) for d in det if d.man_id == m.id)
        if len(det_tmp) > 0:
            res_2. append(max(det_tmp, key = lambda i: i[1]))
    tmp = sorted(res_2, key = itemgetter(1), reverse = True)
    return tmp

def task3 (man, det, men_dets):
    many_to_many_temp = [(m.name, md.det_id, md.man_id)
        for m in man
        for md in men_dets
        if m.id == md.man_id]

    many_to_many = [(man_name, d.name, d.price)
        for man_name, det_id, _ in many_to_many_temp
        for d in det if d.id == det_id]

    return sorted(many_to_many, key = lambda i: i[0])
```

test_task1.py:

```python
import pytest
from rk1_functions import Manufacturer, Detail, DetMan, task1, task2, task3

@pytest.fixture
def man():

    man = [
        Manufacturer(1, "Гидросталь"),
        Manufacturer(2, "Росталь"),
        Manufacturer(3, "Эргон"),
        Manufacturer(4, "Авток"),
        Manufacturer(5, "Арстат")
    ]

    return man

@pytest.fixture
def det():

    det = [
    Detail(1, "штуцер", 1, 100),
    Detail(2, "Винт", 2, 50),
    Detail(3, "Втулка", 4, 70),
    Detail(4, "Шпонка", 3, 110),
    Detail(5, "Пружина", 4, 45),
    Detail(6, "Шпонка2", 5, 40)
    ]

    return det

@pytest.fixture
def men_dets():
    men_dets = [
    DetMan(1,1),
    DetMan(2,2),
    DetMan(3,3),
    DetMan(3,4),
    DetMan(3,5),
    DetMan(4,1),
    DetMan(5,2),
    DetMan(5,3),
    DetMan(4,4),
    DetMan(6,5),
    ]
    return men_dets


def test_task1(man, det):
    assert task1(man, det) == {'Авток': ['Втулка', 'Пружина'], 'Арстат': ['Шпонка2']}

def test_task2(man, det):
    assert task2(man, det) == [('Эргон', 110), ('Гидросталь', 100), ('Авток', 70), ('Росталь', 50), ('Арстат', 40)]

def test_task3(man, det, men_dets):
    assert task3(man, det, men_dets) == [('Авток', 'Втулка', 70), ('Авток', 'Шпонка', 110), ('Арстат', 'Втулка', 70), ('Арстат', 'Шпонка2', 40),
    ('Гидросталь', 'штуцер', 100), ('Гидросталь', 'Шпонка', 110), ('Росталь', 'Винт', 50), ('Росталь', 'Пружина', 45), ('Эргон', 'Втулка', 70),
    ('Эргон', 'Пружина', 45)]
```

Результаты выполнения программы:

```
dasha@dasha-BOHB-WAX9:~/Desktop/BIKT/BIKT$ pytest test_task1.py
============================= test session starts ==============================
platform linux -- Python 3.10.6, pytest-7.2.0, pluggy-1.0.0
rootdir: /home/dasha/Desktop/BIKT/BIKT
plugins: bdd-6.1.0, anyio-3.6.2
collected 3 items

test_task1.py ...                                                        [100%]
```