ИУ5-32Б,

Старкин Александр

Рубежный контроль 2

Условия:

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).

Main.py

```
from operator import itemgetter


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


class Detail:
    def __init__(self, id, name, GOST_id,creator_id):
        self.id = id
        self.name = name
        self.GOST_id = GOST_id
        self.creator_id = creator_id


class CreatorDetail:
    def __init__(self, creator_id, detail_id):
        self.creator_id = creator_id
        self.detail_id = detail_id


Creators = [
    Creator(1, "BMSTU_RK1"),
    Creator(2, "BMSTU_IU3"),
    Creator(3, "BMSTU_L1"),
]

Details = [
    Detail(1, "Soska", 1999_43,1),
    Detail(2, "Stul", 1945_21,2),
    Detail(3, "Stol", 1000_61,3),
    Detail(4, "Podik", 1943_38, 3),
    Detail(5, "Girya", 1942_47, 1),
    Detail(6, "Dver", 1999_42, 1)
```

```
]

Creator_to_detail = [
    CreatorDetail(1, 1),
    CreatorDetail(2, 2),
    CreatorDetail(3, 3),
    CreatorDetail(3, 4),
    CreatorDetail(1, 5),
]


def first_task(details_list):
    res_1 = sorted(details_list, key=itemgetter(2))
    return res_1


def second_task(details_list):
    res_2 = []
    temp_dict = dict()
    for i in details_list:
        if i[2] in temp_dict:
            temp_dict[i[2]] += 1
        else:
            temp_dict[i[2]] = 1
    for i in temp_dict.keys():
        res_2.append((i, temp_dict[i]))

    res_2.sort(key=itemgetter(1), reverse=True)
    return res_2


def third_task(details_list, end_ch):
    res_3 = [(i[0], i[2]) for i in details_list if str(i[1]).endswith(end_ch)]
    return res_3


def main():
    one_to_many = [(Detail.name, Detail.GOST_id, Creator.name)
                   for Creator in Creators
                   for Detail in Details
                   if Detail.creator_id == Creator.id]

    many_to_many_temp = [(Creator.name, connection.creator_id, connection.detail_id)
                         for Creator in Creators
                         for connection in Creator_to_detail
                         if connection.creator_id == Creator.id]

    many_to_many = [(Detail.name, Detail.GOST_id, Creator_name)
                    for Creator_name, Creator_id, detail_id in many_to_many_temp
                    for Detail in Details if Detail.id == detail_id]
```

```
    print('Задание Б1')
    print(first_task(one_to_many))

    print("\nЗадание Б2")
    print(second_task(one_to_many))

    print("\nЗадание Б3")
    print(third_task(many_to_many, '1'))


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

unit-tests.py

```
import main
from operator import itemgetter
import unittest

class TestMainMethods(unittest.TestCase):
    def test_first_task_method(self):
        test_list = [('first', 1999_45, "BMSTU_RK1"), ('forth', 1924_24, "BMSTU_IU3"), ('second',
2024_352,"BMSTU_L3")]
        result = main.first_task(test_list)
        reference = sorted(test_list, key=itemgetter(2))
        self.assertEqual(result, reference)

    def test_second_task_method(self):
        test_list = [("Pen", 1999_43, "John" ),
                     ("Car", 1945_21, "Alex"),
                     ("Stol", 1000_61, "Alex"),
                     ("Laptop", 1943_38, "Alex"),
                     ("Carry", 1942_47, "John"),
                     ("Lamp", 1999_42, "Jean")]
        result = main.second_task(test_list)
        reference = [('Alex', 3), ('John', 2), ('Jean', 1)]
        self.assertEqual(result, reference)

    def test_third_method(self):
        test_list = [("Soska", 1999_43,1),
                     ("Stul", 1945_214,2),
                     ("Stol", 1000_61,3),
                     ("Podik", 1943_38, 3),
                     ("Girya", 1942_474, 1),
                     ("Dver", 1999_4245, 1)]
        result = main.third_task(test_list, '4')
        reference = [('Stul', 2), ('Girya', 1)]
        self.assertEqual(result, reference)
```

Результат запуска:

```
PS D:\study\BMSTU\paradigms_structures_of_pl\PIKYAPA_PK1> python -m unittest unit-tests.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
```