**Московский государственный технический** **университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Парадигмы и конструкции языков программирования»

Отчет по РК№2

Вариант запросов: Г

Вариант предметной области: 14

Выполнил: Проверил:

студент группы ИУ5-34Б преподаватель каф. ИУ5

Сергеев Максим Гапанюк Ю.Е. Подпись и дата: Подпись и дата:

Москва, 2023 г.

**Задача:**

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:**

# Файл main.py  
class Disk:  
 def \_\_init\_\_(self, id, name, publisher, release\_date, lib\_id):  
 self.id = id,  
 self.name = name,  
 self.publisher = publisher,  
 self.release\_date = release\_date  
 self.lib\_id = lib\_id  
  
 def \_\_str\_\_(self):  
 return f"{self.name[0]} by {self.publisher[0]}({self.release\_date})"  
  
  
class Library:  
 def \_\_init\_\_(self, id, owner):  
 self.id = id,  
 self.owner = owner  
  
 def \_\_str\_\_(self):  
 return f"{self.owner}"  
  
  
class DiskLibrary:  
 def \_\_init\_\_(self, disk\_id, lib\_id):  
 self.disk\_id = disk\_id  
 self.lib\_id = lib\_id  
  
 def \_\_str\_\_(self):  
 return f"{self.disk\_id[0]} - {self.lib\_id[0]}"  
  
  
disks = [Disk(1, 'Alphaville - Jerusalem', 'WEA', 1986, 1),  
 Disk(2, 'Orden Ogan - Final Days', 'AFM', 2021, 1),  
 Disk(3, 'George Orwell - 1984', 'Verlag', 2007, 3),  
 Disk(4, 'Read dead redemption 2', 'Rockstar', 2018, 2),  
 Disk(5, 'God of War: Ragnarok', 'Sony', 2022, 2)  
 ]  
  
libs = [Library(1, 'Max Sergeev'),  
 Library(2, 'Name Nickname'),  
 Library(3, 'MGTU')  
 ]  
  
disk\_lib = [DiskLibrary(1, 1),  
 DiskLibrary(2, 1),  
 DiskLibrary(1, 2),  
 DiskLibrary(4, 2),  
 DiskLibrary(3, 2),  
 DiskLibrary(3, 3),  
 DiskLibrary(2, 2),  
 DiskLibrary(5, 2)  
 ]  
  
  
def main():  
 otm = [(i, j) for i in disks for j in libs if i.lib\_id == j.id[0]]  
 otm.sort(key=lambda x: x[1].id)  
 mtm\_tmp = [(i.owner, j.disk\_id) for i in libs for j in disk\_lib if i.id[0] == j.lib\_id]  
 mtm = [(i[0], str(j)) for i in mtm\_tmp for j in disks if i[1] == j.id[0]]  
 print('-----------------Task 1:-----------------')  
 print('...............Their CDs:...............')  
 ans = []  
 for i in otm:  
 if i[1].owner[0] == 'M':  
 ans.append(i[1].owner)  
 print(i[1], 'owns', i[0])  
 print('.......................................')  
 ans1 = sorted(list(set(ans)))  
 print('Owners name start with m:', ans1)  
  
 print('-----------------Task 2:-----------------')  
 dic = {}  
 for i in otm:  
 if i[1].owner in dic:  
 dic[i[1].owner] = min(dic[i[1].owner], i[0].release\_date)  
 else:  
 dic[i[1].owner] = i[0].release\_date  
 sorted\_dic = sorted(dic.items(), key=lambda x: x[1])  
 print(sorted\_dic)  
  
 print('-----------------Task 3:-----------------')  
 ans3 = sorted(mtm, key=lambda x: x[0])  
 print(ans3)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

# Файл tests.py  
  
import unittest  
from main import \*  
  
  
class Test(unittest.TestCase):  
 disks = [Disk(1, 'Alphaville - Jerusalem', 'WEA', 1986, 1),  
 Disk(2, 'Orden Ogan - Final Days', 'AFM', 2021, 1),  
 Disk(3, 'George Orwell - 1984', 'Verlag', 2007, 3),  
 Disk(4, 'Read dead redemption 2', 'Rockstar', 2018, 2),  
 Disk(5, 'God of War: Ragnarok', 'Sony', 2022, 2)  
 ]  
  
 libs = [Library(1, 'Max Sergeev'),  
 Library(2, 'Name Nickname'),  
 Library(3, 'MGTU')  
 ]  
  
 disk\_lib = [DiskLibrary(1, 1),  
 DiskLibrary(2, 1),  
 DiskLibrary(1, 2),  
 DiskLibrary(4, 2),  
 DiskLibrary(3, 2),  
 DiskLibrary(3, 3),  
 DiskLibrary(2, 2),  
 DiskLibrary(5, 2)  
 ]  
  
 def test\_1(self):  
 otm = [(i, j) for i in disks for j in libs if i.lib\_id == j.id[0]]  
 otm.sort(key=lambda x: x[1].id)  
 ans = []  
 for i in otm:  
 if i[1].owner[0] == 'M':  
 ans.append(i[1].owner)  
 ans1 = sorted(list(set(ans)))  
 self.assertEqual(ans1, ['MGTU', 'Max Sergeev'])  
  
 def test\_2(self):  
 otm = [(i, j) for i in disks for j in libs if i.lib\_id == j.id[0]]  
 otm.sort(key=lambda x: x[1].id)  
 dic = {}  
 for i in otm:  
 if i[1].owner in dic:  
 dic[i[1].owner] = min(dic[i[1].owner], i[0].release\_date)  
 else:  
 dic[i[1].owner] = i[0].release\_date  
 sorted\_dic = sorted(dic.items(), key=lambda x: x[1])  
 self.assertEqual(sorted\_dic, [('Max Sergeev', 1986), ('MGTU', 2007), ('Name Nickname', 2018)])  
  
 def test\_3(self):  
 mtm\_tmp = [(i.owner, j.disk\_id) for i in libs for j in disk\_lib if i.id[0] == j.lib\_id]  
 mtm = [(i[0], str(j)) for i in mtm\_tmp for j in disks if i[1] == j.id[0]]  
 ans3 = sorted(mtm, key=lambda x: x[0])  
 self.assertEqual(ans3, [('MGTU', 'George Orwell - 1984 by Verlag(2007)'), ('Max Sergeev', 'Alphaville - Jerusalem by WEA(1986)'), ('Max Sergeev', 'Orden Ogan - Final Days by AFM(2021)'), ('Name Nickname', 'Alphaville - Jerusalem by WEA(1986)'), ('Name Nickname', 'Read dead redemption 2 by Rockstar(2018)'), ('Name Nickname', 'George Orwell - 1984 by Verlag(2007)'), ('Name Nickname', 'Orden Ogan - Final Days by AFM(2021)'), ('Name Nickname', 'God of War: Ragnarok by Sony(2022)')])

**Результат работы программы:**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание