**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

13 вариант

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-32Б: |  | преподаватель каф. ИУ5 |
| Лапотко Ирина Евгеньевна |  | Гапанюк Ю.Е. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2022 г.

**Постановка задачи:**

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.py**

from operator import itemgetter

class Book :

    def \_\_init\_\_(self, ID, name, num, rack\_ID):

        self.ID = ID

        self.name = name

        self.num = num

        self.rack\_ID = rack\_ID

class Library:

    def \_\_init\_\_(self, ID, rack):

        self.ID = ID

        self.rack = rack

class BookLibrary:

    def \_\_init\_\_(self, rack\_ID, book\_ID):

        self.book\_ID = book\_ID

        self.rack\_ID = rack\_ID

Books = [

    Book(1, 'Анна Каренина', 4, 1),

    Book(2, 'Война и Мир', 10, 1),

    Book(3, 'Мертвые души', 3, 2),

    Book(4, 'Преступление и наказание', 3, 2),

    Book(5, 'Унесенные Ветром', 5, 3),

    Book(6, 'Горе от ума', 6, 4)

]

Lib = [

    Library(1, ' Л. Н. Толстой'),

    Library(2, 'Русская проза'),

    Library(3, 'Зарубежная проза'),

    Library(4, 'Пьеса')

]

books\_lib = [

    BookLibrary(1,1),

    BookLibrary(1,2),

    BookLibrary(2,3),

    BookLibrary(2,4),

    BookLibrary(3,5),

    BookLibrary(4,6)]

def sort\_name(mas):

    return sorted(mas,key = itemgetter(0))

def book\_in\_lib(mas,Lib2):

    res2 = []

    for l in Lib2:

        books\_rack = list(filter(lambda i: i[2] == l.rack, mas))

        rack\_num = len(books\_rack)

        res2.append((l.rack, rack\_num))

    return res2

def run\_book(mas, Lib3):

    res3 = []

    for l in Lib3:

        books\_rack = list(filter(lambda i: i[2] == l.rack, mas))

        for bn in books\_rack:

            if 'ир' in bn[0]:

                res3.append((l.rack, bn[0]))

    return res3

def main():

    one\_to\_many = [(b.name, b.num, l.rack)

    for b in Books

    for l in Lib

    if b.rack\_ID == l.ID]

    many\_to\_many\_temp = [(l.rack, bl.rack\_ID, bl.book\_ID)

                    for l in Lib

                    for bl in books\_lib

                    if l.ID == bl.rack\_ID]

    many\_to\_many = [(b.name, b.num, lib\_rack)

                    for lib\_rack, rack\_ID, book\_ID in many\_to\_many\_temp

                    for b in Books if b.ID == book\_ID]

    #Сортировка по названию книги

    print("Задание Б1")

    #res1 = sorted(one\_to\_many, key = itemgetter(0))

    res1 = sort\_name(one\_to\_many)

    print(res1)

    #Какие книги есть на стеллажах

    print("\nЗадание Б2")

    res2 = book\_in\_lib(one\_to\_many, Lib)

    print(res2)

    #Нахождение книги по окончанию

    print("\nЗадание Б3")

    res3 = run\_book(many\_to\_many, Lib)

    print(res3)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**testsRK2.py**

import unittest

from RK2 import run\_book, sort\_name, book\_in\_lib, Book, Library, BookLibrary

class test\_RK2(unittest.TestCase):

    def setUp(self):

        self.Books = [

            Book(1, 'Анна Каренина', 4, 1),

            Book(2, 'Война и Мир', 10, 1),

            Book(3, 'Мертвые души', 3, 2),

            Book(4, 'Преступление и наказание', 3, 2),

            Book(5, 'Унесенные Ветром', 5, 3),

            Book(6, 'Горе от ума', 6, 4)]

        self.Lib = [

            Library(1, ' Л. Н. Толстой'),

            Library(2, 'Русская проза'),

            Library(3, 'Зарубежная проза'),

            Library(4, 'Пьеса')]

        self.books\_lib = [

            BookLibrary(1,1),

            BookLibrary(1,2),

            BookLibrary(2,3),

            BookLibrary(2,4),

            BookLibrary(3,5),

            BookLibrary(4,6)]

        self.one\_to\_many = [(b.name, b.num, l.rack)

            for b in self.Books

            for l in self.Lib

            if b.rack\_ID == l.ID]

        self.many\_to\_many\_temp = [(l.rack, bl.rack\_ID, bl.book\_ID)

            for l in self.Lib

            for bl in self.books\_lib

            if l.ID == bl.rack\_ID]

        self.many\_to\_many = [(b.name, b.num, lib\_rack)

            for lib\_rack, rack\_ID, book\_ID in self.many\_to\_many\_temp

            for b in self.Books if b.ID == book\_ID]

    def testSortname(self):

        res = sort\_name(self.one\_to\_many)

        res2 = [('Анна Каренина', 4, ' Л. Н. Толстой'),

        ('Война и Мир', 10, ' Л. Н. Толстой'),

        ('Горе от ума', 6, 'Пьеса'),

         ('Мертвые души', 3, 'Русская проза'),

         ('Преступление и наказание', 3, 'Русская проза'),

         ('Унесенные Ветром', 5, 'Зарубежная проза')]

        self.assertEqual(res,res2)

    def test\_book\_in\_lib(self):

        res = book\_in\_lib(self.one\_to\_many, self.Lib)

        res2 = [(' Л. Н. Толстой', 2), ('Русская проза', 2), ('Зарубежная проза', 1), ('Пьеса', 1)]

        self.assertEqual(res,res2)

    def test\_run\_book(self):

        res = run\_book(self.many\_to\_many, self.Lib)

        res2 = [(' Л. Н. Толстой', 'Война и Мир')]

        self.assertEqual(res,res2)

if \_\_name\_\_ == '\_\_main\_\_':

    unittest.main()

**Экранные формы с примерами выполнения программы**

