МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Московский государственный технический университет

им. Н.Э. Баумана

(МГТУ им. Н.Э. Баумана)

Кафедра «Системы обработки информации и управления» (ИУ5)

Отчёт по рубежному контролю № 2

По курсу: «Базовые компоненты интернет-технологий»

Выполнил: Никулин Данила Дмитриевич

студент группы ИУ5-31Б.

Проверил: Гапанюк Юрий Евгеньевич

Преподаватель кафедры ИУ5

г. 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).

Текст программы:

rk1.py

```
from operator import itemgetter

class Book:
    def __init__(self, id, title, number, library_id):
        self.id = id
        self.title = title
        self.number = number
        self.library_id = library_id

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

class BookLibrary:
    def __init__(self, library_id, book_id):
        self.library_id = library_id
        self.book_id = book_id

#Библиотеки
libraries = [
    Library(1, 'Российская государственная библиотека'),
    Library(2, 'Библиотека имени Достоевского'),
    Library(3, 'Библиотека имени Некрасова'),
    Library(4, 'Библиотека-читальня имени И.С. Тургенева'),
    Library(5, 'Центральная библиотека имени Добролюбова'),
    Library(6, 'Российская национальная библиотека'),
]

#Книги
books = [
    Book(1, 'Горе от ума', 150, 1),
    Book(2, 'Преступление и наказание', 70, 2),
    Book(3, 'Отцы и дети', 100, 3),
    Book(4, 'Евгений Онегин', 50, 1),
    Book(5, 'Ревизор', 40, 6),
]
```

```
books_libraries = [
    BookLibrary(1,1),
    BookLibrary(1,2),
    BookLibrary(1,3),
    BookLibrary(1,4),
    BookLibrary(1,5),
    BookLibrary(2,2),
    BookLibrary(3,2),
    BookLibrary(4,3),
    BookLibrary(5,4),
    BookLibrary(6,1),
    BookLibrary(6,2),
    BookLibrary(6,4),
    BookLibrary(6,5),
]

    #Соединение данных один-ко-многим
def one_to_many(libraries, books):
    return[(b.title, b.number, l.name)
        for l in libraries
        for b in books
        if b.library_id == l.id]

    #Соединение данных многие-ко-многим
def many_to_many(libraries, books):
    many_to_many_temp = [(l.name, bl.library_id, bl.book_id)
        for l in libraries
        for bl in books_libraries
        if l.id == bl.library_id]
    return [(b.title, b.number, library_name)
        for library_name, library_id , book_id in many_to_many_temp
        for b in books
        if b.id == book_id]

def example_A1(libraries, books):
    res_11 = sorted(one_to_many(libraries, books), key = itemgetter(2))
    return res_11

def example_A2(libraries, books):
    res_12_unsorted = []
    for l in libraries:
        l_books = list(filter(lambda i: i[2] == l.name,
one_to_many(libraries, books)))
        if len(l_books) > 0:
            l_numbers = [number for _,number,_ in l_books]
            l_numbers_sum = sum(l_numbers)
            res_12_unsorted.append((l.name, l_numbers_sum))

    res_12 = sorted(res_12_unsorted, key = itemgetter(1), reverse=True)
    return res_12

def example_A3(libraries, books):
    res_13 = {}
    for l in libraries:
        if "имени" in l.name:
```

```
            l_books = list(filter(lambda i: i[2] == l.name,
many_to_many(libraries, books)))
            l_books_titles = [x for x,_,_ in l_books]
            res_13[l.name] = l_books_titles
    return res_13

if __name__ == '__main__':
    print('Задание А1')
    print(example_A1(libraries, books))
    print('Задание А2')
    print(example_A2(libraries, books))
    print('Задание А3')
    print(example_A3(libraries, books))
```

test_TDD.py

```
import unittest
from rk1 import *

class rk1_test(unittest.TestCase):

    def test_example_A1(self):
        expected_result = [
            ('Преступление и наказание', 70, 'Библиотека имени
Достоевского'),
            ('Отцы и дети', 100, 'Библиотека имени Некрасова'),
            ('Горе от ума', 150, 'Российская государственная библиотека'),
            ('Евгений Онегин', 50, 'Российская государственная библиотека'),
            ('Ревизор', 40, 'Российская национальная библиотека')
        ]
        result = example_A1(libraries, books)
        self.assertEqual(result, expected_result)

    def test_example_A2(self):
        expected_result = [
            ('Российская государственная библиотека', 200),
            ('Библиотека имени Некрасова', 100),
            ('Библиотека имени Достоевского', 70),
            ('Российская национальная библиотека', 40)
        ]
        result = example_A2(libraries, books)
        self.assertEqual(result, expected_result)

    def test_example_A3(self):
        expected_result = {
            'Библиотека имени Достоевского': ['Преступление и наказание'],
            'Библиотека имени Некрасова': ['Преступление и наказание'],
            'Библиотека-читальня имени И.С. Тургенева': ['Отцы и дети'],
            'Центральная библиотека имени Добролюбова': ['Евгений Онегин']
        }
        result = example_A3(libraries, books)
        self.assertEqual(result, expected_result)
```

```python
if __name__ == '__main__':
    unittest.main()
```

Результаты тестирования: