**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 A1')

print(example\_A1(libraries, books))

print('Задание A2')

print(example\_A2(libraries, books))

print('Задание A3')

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)

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

unittest.main()

**Результаты тестирования:**

