# 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 модульного тесты с применением TDD – фреймворка (3 теста)

# Текст программы

## rk2.1.py

from operator import itemgetter

class Book:

def \_\_init\_\_(self, id, name, shop\_id, page):

self.id = id

self.name = name

self.shop\_id = shop\_id

self.page = page

class Shop:

def \_\_init\_\_(self, id, name):

self.id = id

self.name = name

class BookInShop:

def \_\_init\_\_(self, book\_id, shop\_id):

self.book\_id = book\_id

self.shop\_id = shop\_id

shops = [

Shop(1, 'Библио-Глобус'),

Shop(2, 'Читай - Москва'),

Shop(3, 'Молодая гвардия'),

Shop(4, 'Книжный лабиринт'),

Shop(5, 'Читай - город'),

]

books = [

Book(1, 'Преступление и наказание', 2, 200),

Book(2, 'Вишневый сад', 3, 150),

Book(3, 'Война и мир 1 том', 1, 450),

Book(4, 'Война и мир 2 том', 5, 300),

Book(5, 'Война и мир 3 том', 5, 200),

Book(6, 'Капитанская дочка', 4, 100),

Book(7, 'Евгений Онегин', 1, 250),

Book(8, 'Мертвые души', 3, 350),

]

booksinshops = [

BookInShop(1,2),

BookInShop(2,3),

BookInShop(3,1),

BookInShop(4,5),

BookInShop(5,5),

BookInShop(6,4),

BookInShop(7,1),

BookInShop(8,3),

]

def main():

one\_to\_many = [(b.name, b.page, s.name)

for s in shops

for b in books

if b.shop\_id==s.id]

many\_to\_many\_temp = [(s.name, bs.shop\_id, bs.book\_id)

for s in shops

for bs in booksinshops

if s.id == bs.shop\_id]

many\_to\_many = [(b.name, b.page, shop\_name)

for shop\_name, shop\_id, book\_id in many\_to\_many\_temp

for b in books if b.id==shop\_id]

print('Задание Е1\n')

res\_11 = list(filter(lambda i:i[2].find('Читай')!=-1, one\_to\_many))

print(res\_11)

print('\nЗадание Е2\n')

res\_12\_unsorted = []

for d in shops:

d\_Books = list(filter(lambda i: i[2]==d.name, one\_to\_many))

if len(d\_Books) > 0:

d\_Books\_page = [page for \_,page,\_ in d\_Books]

d\_page\_avg = round(sum(d\_Books\_page)/len(d\_Books\_page), 2)

res\_12\_unsorted.append((d.name, d\_page\_avg))

res\_12 = sorted(res\_12\_unsorted, key=itemgetter(1), reverse=True)

print(res\_12)

print('\nЗадание Е3\n')

print(list(filter(lambda i: i[0].find('В') != -1, many\_to\_many)))

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

main()

# rk2.2.py

﻿

import unittest

from rk2.1 import \*

class TestRK2(unittest.TestCase):

shops = [

Shop(1, 'Библио-Глобус'),

Shop(2, 'Читай - Москва'),

Shop(3, 'Молодая гвардия'),

Shop(4, 'Книжный лабиринт'),

Shop(5, 'Читай - город'),

]

books = [

Book(1, 'Преступление и наказание', 2, 200),

Book(2, 'Вишневый сад', 3, 150),

Book(3, 'Война и мир 1 том', 1, 450),

Book(4, 'Война и мир 2 том', 5, 300),

Book(5, 'Война и мир 3 том', 5, 200),

Book(6, 'Капитанская дочка', 4, 100),

Book(7, 'Евгений Онегин', 1, 250),

Book(8, 'Мертвые души', 3, 350),

]

def test1(self):

self.assertEqual(task\_1(books, shops), [('Преступление и наказание','Читай - Москва', 200 ),

('Вишневый сад','Молодая гвардия' , 150),

('Библио-Глобус','Библио-Глобус', 250)])

def test2(self):

self.assertEqual(task\_2(books, shops), [('Читай - город', 250),

('Молодая гвардия', 250),

('Библио-Глобус', 350)])

def test3(self):

self.assertEqual(task\_3(books, shops, booksinshops), [('Капитанская дочка', 100,'Молодая гвардия'),

('Война и мир 2 том', 300, 'Читай - Москва')])

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

unittest.main()