Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего **образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Рубежный контроль № 2

по дисциплине «Базовые компоненты интернет-технологий»

Выполнил:
студент группы ИУ5-33Б
Семенов В.А.

Проверил:
Гапанюк Ю.Е.

2021 г

Полученное задание:

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:

main.py

```
# Книга, Книжный магазин

from operator import itemgetter

class Kniga:
    """Книга"""

    def __init__(self, id, naz, stoim, mag_id):
        self.id = id
        self.naz = naz # название
        self.stoim = stoim # стоимость
        self.mag_id = mag_id

class Knizhnimagazin:
    """Книжный магазин"""

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

class Knigavmagazine:
    """Книга в магазине для реализации связи многие-ко-многим"""

    def __init__(self, kniga_id, magazin_id):
        self.kniga_id = kniga_id
        self.magazin_id = magazin_id

# Магазин
magazin = [
    Knizhnimagazin(1, 'Для малышей'),
    Knizhnimagazin(2, 'Современник'),
    Knizhnimagazin(3, 'Для школьников и их родителей'),
    Knizhnimagazin(4, 'Книга - друг человека'),
    Knizhnimagazin(5, 'Зарубежные книги'),
    Knizhnimagazin(6, 'Литература о саморазвитии'),
]

# Книга
knigi = [
    Kniga(1, 'Алфавит', 500, 1),
    Kniga(2, 'Приключения Тома Сойера', 2000, 5),
    Kniga(3, 'Магия утра', 2500, 6),
    Kniga(4, 'Красота мира', 450, 2),
    Kniga(5, 'Капитанская дочка', 3000, 4),
    Kniga(6, 'Алиса в стране чудес', 200, 2),
]

knigiandmagazin = [
    Knigavmagazine(1, 4),
    Knigavmagazine(2, 6),
    Knigavmagazine(3, 4),
    Knigavmagazine(3, 3),
    Knigavmagazine(1, 1),
    Knigavmagazine(2, 1),
```

```
    Knigavmagazine(5, 2),
    Knigavmagazine(6, 3),
    Knigavmagazine(4, 5),
    Knigavmagazine(4, 4),
]

def Task1(one_to_many):
    print('Задание B1')
    res1 = []
    for naz, stoim, Knizhnimagazin_name in one_to_many:
        if 'A' in naz[0]:
            res1.append((naz, Knizhnimagazin_name))
    return res1

def Task2(one_to_many):
    print('Задание B2')
    buff = []
    for a in magazin:
        magmin = list(filter(lambda i: i[2] == a.name, one_to_many))
        if len(magmin) > 0:
            a_stoim = [stoim for _, stoim, _ in magmin]
            min_stoim = min(a_stoim)
            buff.append((a.name, min_stoim))
    res2 = sorted(buff, key=itemgetter(1))
    return res2

def Task3(many_to_many):
    print('Задание B3')
    buff = []
    for naz, stoim, Knizhnimagazin_name in many_to_many:
        buff.append((naz, Knizhnimagazin_name))
    res3 = list(sorted(buff, key=itemgetter(0)))
    return res3

def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [(k.naz, k.stoim, m.name)
                   for m in magazin
                   for k in knigi
                   if k.mag_id == m.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(m.name, km.magazin_id, km.kniga_id)
                         for m in magazin
                         for km in knigiandmagazin
                         if m.id == km.magazin_id]

    many_to_many = [(k.naz, k.stoim, Knizhnimagazin_name)
                    for Knizhnimagazin_name, magazin_id, kniga_id in
many_to_many_temp
                    for k in knigi if k.id == kniga_id]

    print(Task1(one_to_many))
    print(Task2(one_to_many))
    print(Task3(many_to_many))

if __name__ == '__main__':
    main()
```

test.py

```
import unittest
import sys, os

sys.path.append(os.getcwd())
from main import *
```

```
class Test_task_1(unittest.TestCase):
    def test_task_1(self):
        one_to_many = [(k.naz, k.stoim, m.name)
                       for m in magazin
                       for k in knigi
                       if k.mag_id == m.id]
        self.assertEqual(Task1(one_to_many), [('Алфавит', 'Для малышей'), ('Алиса
в стране чудес', 'Современник')])

class Test_task_2(unittest.TestCase):
    def test_task_2(self):
        one_to_many = [(k.naz, k.stoim, m.name)
                       for m in magazin
                       for k in knigi
                       if k.mag_id == m.id]
        self.assertEqual(Task2(one_to_many), [('Современник', 200), ('Для
малышей', 500),
                                              ('Зарубежные книги', 2000),
('Литература о саморазвитии', 2500),
                                              ('Книга - друг человека', 3000)])


class Test_task_3(unittest.TestCase):
    def test_task_3(self):
        many_to_many_temp = [(m.name, km.magazin_id, km.kniga_id)
                             for m in magazin
                             for km in knigiandmagazin
                             if m.id == km.magazin_id]
        many_to_many = [(k.naz, k.stoim, Knizhnimagazin_name)
                        for Knizhnimagazin_name, magazin_id, kniga_id in
many_to_many_temp
                        for k in knigi if k.id == kniga_id]
        self.assertEqual(Task3(many_to_many), [('Алиса в стране чудес', 'Для
школьников и их родителей'),
                                               ('Алфавит', 'Для малышей'),
('Алфавит', 'Книга - друг человека'),
                                               ('Капитанская дочка',
'Современник'),
                                               ('Красота мира', 'Книга - друг
человека'),
                                               ('Красота мира', 'Зарубежные
книги'),
                                               ('Магия утра', 'Для школьников и их
родителей'),
                                               ('Магия утра', 'Книга - друг
человека'),
                                               ('Приключения Тома Сойера', 'Для
малышей'),
                                               ('Приключения Тома Сойера',
'Литература о саморазвитии')])
if __name__ == "__main__":
    unittest.main()
```

Результат выполнения программы:

```
Задание B1
Задание B2
Задание B3
...
----------------------
Ran 3 tests in 0.001s

OK
```