Выполнил: Щепетов Дмитрий

Группа: ИУ5-32Б

**Рубежный контроль №2 по дисциплине «Базовые компоненты интернет-технологий»**

Вариант запросов: Б.

Вариант предметной области: 24.

**Условия задания**

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**

**Файл main.py (измененный под условия РК2)**

from operator import itemgetter

'''Вариант Б.

1. «Книга» и «Глава» связаны соотношением один-ко-многим.

Выведите список всех связанных глав и книг, отсортированный по главам, сортировка по книгам произвольная.

2. «Книга» и «Глава» связаны соотношением один-ко-многим.

Выведите список книг с количеством глав в каждой книге, отсортированный по количеству глав.

3. «Книга» и «Глава» связаны соотношением многие-ко-многим.

Выведите список всех глав, у которых названия закачиваются на "ми", и названия их книг.

'''

class book() :

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

        self.id = ID # Номер книги

        self.name = name # Название книги

class chapter() :

    def \_\_init\_\_ (self, ID, chapter\_name, chapter\_page, book\_ID):

        self.id = ID # Номер главы

        self.name = chapter\_name # Название главы

        self.page = chapter\_page # Номер страницы, где находится глава

        self.book\_id = book\_ID # Номер книги

class bookchapter() :

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

        self.book\_id = book\_ID # Номер книги

        self.chapter\_id = chapter\_ID # Номер главы

books = [

    book(1, 'Google Android. Программирование для мобильных устройств'),

    book(2, 'Java 2. Самоучитель'),

    book(3, '1С: предприятие 8.1. Конфигурирование и администрирование'),

    book(4, 'Думай и богатей'),

    book(5, 'Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds'),

    book(6, 'Горизонт событий'),

        ]

chapters = [

    chapter(1, 'Компоненты Android-приложения', 43,1),

    chapter(2, 'Управление деятельностями', 249,1),

    chapter(3, 'Ресурсы, активы и локализация приложений', 349,1),

    chapter(4, 'Все, что надо знать о программах', 14,2),

    chapter(5, 'Наследование', 203, 2),

    chapter(6, 'Объекты конфигурации', 80 ,3),

    chapter(7, 'Механизм Web-сервисов', 612, 3),

    chapter(8, 'Изумительная сила секрета доказывается фактами', 3,4),

    chapter(9, 'Шесть принципов возрождения бренда', 46,5),

    chapter(10, 'Принцип 2. Восстановление соответствия бренда запросам потребителей', 68,5),

    chapter(11, 'Альфа Цефея', 2048,6),

        ]

chapters\_of\_books = [

    bookchapter(1, 1),

    bookchapter(1, 2),

    bookchapter(1, 3),

    bookchapter(2, 4),

    bookchapter(2, 5),

    bookchapter(3, 6),

    bookchapter(3, 7),

    bookchapter(4, 8),

    bookchapter(5, 9),

    bookchapter(5, 10),

    bookchapter(6, 11)]

def task1(oneToMany):

    try:

        print ('\_\_'\*35+'\n')

        print ('Задание Б1         |')

        print ('\_\_'\*35+'\n')

        B1 = sorted(oneToMany, key = itemgetter(0))

        return B1

    except(TypeError):

        raise TypeError("Выражение должно быть типа list[tuple[str, int, str]]")

def task2(oneToMany):

    print ('\_\_'\*35+'\n')

    print ('Задание Б2         |')

    print ('\_\_'\*35+'\n')

    B2 = []

    for i in range(6):

        count\_chap = list(filter(lambda j: j[2] == books[i].name, oneToMany))

        B2.append((books[i].name, len(count\_chap)))

    B2 = sorted(B2, key=itemgetter(1), reverse=True)

    return B2

def task3(manyToMany):

    print ('\_\_'\*35+'\n')

    print ('Задание Б3         |')

    print ('\_\_'\*35+'\n')

    B3 = []

    for i in manyToMany:

        if (i[0][-2:]=='ми'):

            B3.append(i)

    return B3

def main():

     # Соединение данных один-ко-многим

    oneToMany = [(chap.name, chap.page, book.name)

                    for book in books

                    for chap in chapters

                    if chap.book\_id == book.id

                ]

    # Соединение данных многие-ко-многим

    manyToMany\_temp = [ (book.name, chobs.book\_id, chobs.chapter\_id)

                        for book in books

                        for chobs in chapters\_of\_books

                        if book.id == chobs.book\_id

                        ]

    manyToMany =  [(chap.name, chap.page, book\_name)

                    for book\_name, book\_id, chapter\_id in manyToMany\_temp

                    for chap in chapters if chap.id == chapter\_id]

    print(task1(oneToMany))

    print(task2(oneToMany))

    print(task3(manyToMany))

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Файл modul\_test.py (файл с модульными тестами с TDD)**

import unittest

from unittest import main

from main import task1,task2,task3

class testiki(unittest.TestCase):

    def testik1(self):

        self.assertEqual(task1(

            [('Альфа Цефея', 2048, 'Горизонт событий'),

            ('Изумительная сила секрета доказывается фактами', 3, 'Думай и богатей'),

            ('Объекты конфигурации', 80, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Управление деятельностями', 249, 'Google Android. Программирование для мобильных устройств'),

            ('Все, что надо знать о программах', 14, 'Java 2. Самоучитель'),

            ('Наследование', 203, 'Java 2. Самоучитель'),

            ('Механизм Web-сервисов', 612, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Ресурсы, активы и локализация приложений', 349, 'Google Android. Программирование для мобильных устройств'),

            ('Компоненты Android-приложения', 43, 'Google Android. Программирование для мобильных устройств')]),

            [('Альфа Цефея', 2048, 'Горизонт событий'),

            ('Все, что надо знать о программах', 14, 'Java 2. Самоучитель'),

            ('Изумительная сила секрета доказывается фактами', 3, 'Думай и богатей'),

            ('Компоненты Android-приложения', 43, 'Google Android. Программирование для мобильных устройств'),

            ('Механизм Web-сервисов', 612, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Наследование', 203, 'Java 2. Самоучитель'),

            ('Объекты конфигурации', 80, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Ресурсы, активы и локализация приложений', 349, 'Google Android. Программирование для мобильных устройств'),

            ('Управление деятельностями', 249, 'Google Android. Программирование для мобильных устройств')])

    def testik2(self):

        with self.assertRaises(TypeError) as err:

            task1([1,'str',1.0])

        self.assertEqual("Выражение должно быть типа list[tuple[str, int, str]]", err.exception.args[0])

    def testik3(self):

        self.assertEqual(task2(

            [('Альфа Цефея', 2048, 'Горизонт событий'),

            ('Все, что надо знать о программах', 14, 'Java 2. Самоучитель'),

            ('Изумительная сила секрета доказывается фактами', 3, 'Думай и богатей'),

            ('Компоненты Android-приложения', 43, 'Google Android. Программирование для мобильных устройств'),

            ('Механизм Web-сервисов', 612, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Наследование', 203, 'Java 2. Самоучитель'),

            ('Объекты конфигурации', 80, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Принцип 2. Восстановление соответствия бренда запросам потребителей', 68, 'Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds'),

            ('Ресурсы, активы и локализация приложений', 349, 'Google Android. Программирование для мобильных устройств'),

            ('Управление деятельностями', 249, 'Google Android. Программирование для мобильных устройств'),

            ('Шесть принципов возрождения бренда', 46, 'Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds')]),

            [('Google Android. Программирование для мобильных устройств', 3),

            ('Java 2. Самоучитель', 2),

            ('1С: предприятие 8.1. Конфигурирование и администрирование', 2),

            ('Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds', 2),

            ('Думай и богатей', 1),

            ('Горизонт событий', 1)])

    def testik4(self):

        self.assertEqual(task3(

            [('Альфа Цефея', 'Горизонт событий'),

            ('Управление деятельностями', 'Google Android. Программирование для мобильных устройств'),

            ('Изумительная сила секрета доказывается фактами', 'Думай и богатей'),

            ('Все, что надо знать о программах', 14, 'Java 2. Самоучитель'),

            ('Компоненты Android-приложения', 43, 'Google Android. Программирование для мобильных устройств'),

            ('Механизм Web-сервисов', 612, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Наследование', 203, 'Java 2. Самоучитель'),

            ('Объекты конфигурации', 80, '1С: предприятие 8.1. Конфигурирование и администрирование'),

            ('Принцип 2. Восстановление соответствия бренда запросам потребителей', 68, 'Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds'),

            ('Ресурсы, активы и локализация приложений', 349, 'Google Android. Программирование для мобильных устройств'),

            ('Шесть принципов возрождения бренда', 46, 'Возрождение бренда. Шесть принципов. Вдохните в свой бренд новую жизнь вместе с McDonalds')]),

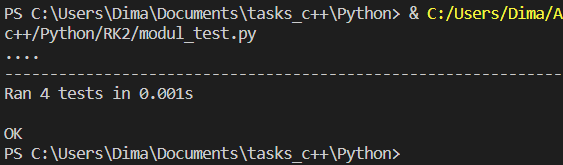
            [('Управление деятельностями', 'Google Android. Программирование для мобильных устройств'),

            ('Изумительная сила секрета доказывается фактами', 'Думай и богатей')])

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Результаты выполнения**

****