Шибанова Варвара ИУ5-31Б РК№2 Вариант№24

1) рефакторинг текста программы рубежного контроля №1

```python
from operator import itemgetter

class Chapter:
    def __init__(self, id, tit, pag, b_id):
        self.id = id
        self.tit = tit
        self.pag = pag
        self.b_id = b_id

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

class ChapBook:
    def __init__(self, b_id, chap_id):
        self.b_id = b_id
        self.chap_id = chap_id

def join_one_to_many(books, chapters):
    """Соединяет данные один-ко-многим"""
    return [(c.tit, c.pag, b.name)
            for b in books
            for c in chapters
            if c.b_id == b.id]

def join_many_to_many(books, chap_b, chapters):
    """Соединяет данные многие-ко-многим"""
    many_to_many_temp = [(b.name, c.b_id, c.chap_id)
                         for b in books
                         for c in chap_b
                         if b.id == c.b_id]
    return [(c.tit, c.pag, book_name)
            for book_name, b_id, chap_id in many_to_many_temp
            for c in chapters
            if c.id == chap_id]

def sort_one_to_many_by_book(one_to_many):
    """Сортирует данные один-ко-многим по названию книги"""
    return sorted(one_to_many, key=itemgetter(2))

def sort_books_by_total_pages(books, one_to_many):
    """Сортирует книги по суммарному количеству страниц"""
    res_12_unsorted = []
    for b in books:
        b_chaps = list(filter(lambda i: i[2] == b.name, one_to_many))
        if b_chaps:
```

```
            b_pag = [pag for _, pag, _ in b_chaps]
            b_pag_sum = sum(b_pag)
            res_12_unsorted.append((b.name, b_pag_sum))
    return sorted(res_12_unsorted, key=itemgetter(1), reverse=True)

def find_books_with_i(books, many_to_many):
    """Находит книги с буквой "и" в названии"""
    res_13 = {}
    for b in books:
        if 'и' in b.name:
            b_chaps = list(filter(lambda i: i[2] == b.name, many_to_many))
            b_chaps_names = [x for x, _, _ in b_chaps]
            res_13[b.name] = b_chaps_names
    return res_13

# Данные (остаются без изменений)
books = [
    Book(1, 'Маленький принц'),
    Book(2, 'Американский психопат'),
    Book(3, 'Палач'),
    Book(11, 'Гарри Поттер и узник Азкабана'),
    Book(22, 'Путешествие к центру Земли'),
    Book(33, 'Обломов'),
]

chapters = [
    Chapter(1, 'Введение', 10, 3),
    Chapter(2, 'Заключение', 12, 1),
    Chapter(3, 'Глава 6', 20, 2),
    Chapter(4, 'Роза', 15, 1),
    Chapter(5, 'Расследование', 27, 3),
]

chap_b = [
    ChapBook(1, 4),
    ChapBook(2, 3),
    ChapBook(3, 5),
    ChapBook(3, 1),
    ChapBook(3, 2),
    ChapBook(11, 1),
    ChapBook(22, 2),
    ChapBook(33, 1),
    ChapBook(33, 2),
    ChapBook(22, 1),
]

def main():
    one_to_many_data = join_one_to_many(books, chapters)
    many_to_many_data = join_many_to_many(books, chap_b, chapters)

    print('Задание А1')
```

```
    print(sort_one_to_many_by_book(one_to_many_data))

    print('\nЗадание А2')
    print(sort_books_by_total_pages(books, one_to_many_data))

    print('\nЗадание А3')
    print(find_books_with_i(books, many_to_many_data))

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

Обработка результатов:

```
PS C:\Users\VARVARA\Desktop\Парадигмы в конструкции языков програмирования\Labs.py> python RK2.py
Задание А1
[('Глава 6', 20, 'Американский психопат'), ('Заключение', 12, 'Маленький принц'), ('Роза', 15, 'Маленький принц'), ('Введение', 10, 'Палач'), ('Расследование', 27, '
алач')]

Задание А2
[('Палач', 37), ('Маленький принц', 27), ('Американский психопат', 20)]

Задание А3
{'Маленький принц': ['Роза'], 'Американский психопат': ['Глава 6'], 'Гарри Поттер и узник Азкабана': ['Введение'], 'Путешествие к центру Земли': ['Заключение', 'Введ
ние']}
PS C:\Users\VARVARA\Desktop\Парадигмы в конструкции языков програмирования\Labs.py>
```

2) модульные тесты с применением TDD - фреймворка (3 теста).

```
import unittest
from RK2 import (join_one_to_many, join_many_to_many,
                    sort_one_to_many_by_book, sort_books_by_total_pages,
                    find_books_with_i, Book, Chapter, ChapBook)

class TestProgram(unittest.TestCase):

    def setUp(self):
        self.books = [
            Book(1, 'Маленький принц'),
            Book(2, 'Американский психопат'),
            Book(3, 'Палач')
        ]
        self.chapters = [
            Chapter(1, 'Введение', 10, 1),
            Chapter(2, 'Глава 2', 20, 2),
        ]
        self.chap_b = [
            ChapBook(1, 1),
            ChapBook(2,2)
        ]

    def test_join_one_to_many(self):
        result = join_one_to_many(self.books, self.chapters)
        self.assertEqual(len(result),2)
        self.assertEqual(result[0], ('Введение', 10, 'Маленький принц'))

    def test_join_many_to_many(self):
      result = join_many_to_many(self.books,self.chap_b,self.chapters)
```

```
      self.assertEqual(len(result),2)
      self.assertEqual(result[0], ('Введение', 10, 'Маленький принц'))

    def test_sort_books_by_total_pages(self):
      one_to_many_data = join_one_to_many(self.books, self.chapters)
      result = sort_books_by_total_pages(self.books, one_to_many_data)
      self.assertEqual(len(result),2)
      self.assertEqual(result[0][0], 'Американский психопат') #проверка
сортировки по суммарному количеству страниц

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

Обработка результатов:

```
PS C:\Users\VARVARA\Desktop\Парадигмы в конструкции языков програмирования\Labs.py> python testRK2.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
PS C:\Users\VARVARA\Desktop\Парадигмы в конструкции языков програмирования\Labs.py>
```