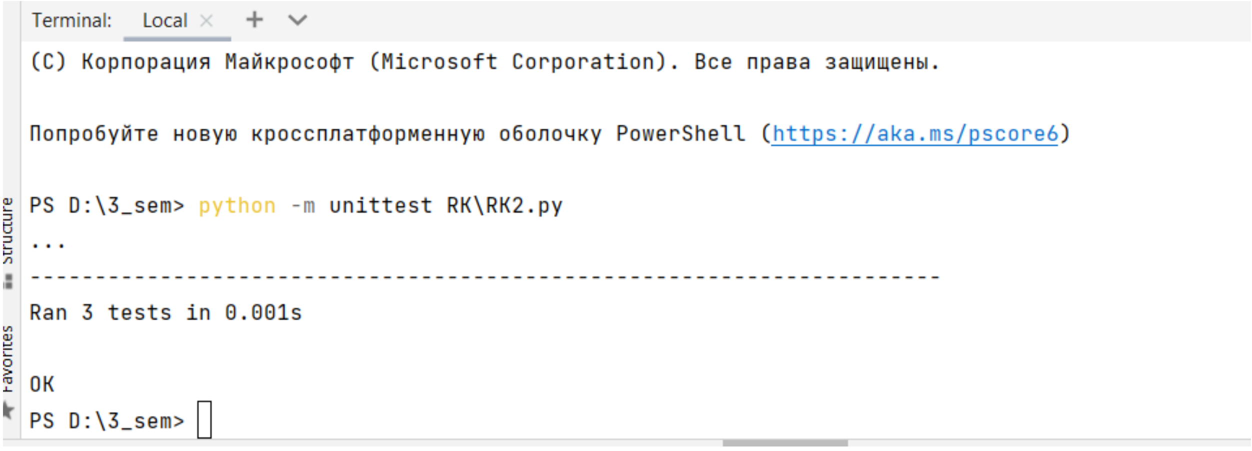
```python
from operator import itemgetter
import unittest


class Book:
    def __init__(self, id, title, pages, lib_id):
        self.id = id
        self.title = title
        self.pages = pages
        self.lib_id = lib_id


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


class BookLib:
    def __init__(self, lib_id, book_id):
        self.lib_id = lib_id
        self.book_id = book_id


class DataProcessor:
    def __init__(self, libraries, books, books_libs):
        self.libraries = libraries
        self.books = books
        self.books_libs = books_libs

    def one_to_many_relationship(self):
        one_to_many = [(b.title, b.pages, lib.name)
                for b in self.books
                for lib in self.libraries
                if b.lib_id == lib.id]
        return sorted(one_to_many, key=itemgetter(2))

    def many_to_many_relationship(self):
        many_to_many_temp = [(lib.name, bl.lib_id, bl.book_id)
                             for lib in self.libraries
                             for bl in self.books_libs
                             if lib.id == bl.lib_id]

        return [(b.title, b.pages, lib_name)
                for lib_name, lib_id, book_id in many_to_many_temp
                for b in self.books if b.id == book_id]

    def total_pages_per_library(self, one_to_many):
        res = []
        for lib in self.libraries:
            l_books = list(filter(lambda i: i[2] == lib.name, one_to_many))
            if l_books:
                l_pages_sum = sum(pages for _, pages, _ in l_books)
                res.append((lib.name, l_pages_sum))
        return sorted(res, key=itemgetter(1), reverse=True)

    def books_in_libraries(self, many_to_many):
        res = {}
        for lib in self.libraries:
            if 'библиотека' in lib.name.lower():
                l_books = list(filter(lambda i: i[2] == lib.name,
many_to_many))
                l_books_names = [x for x, _, _ in l_books]
                res[lib.name] = l_books_names
        return res


class TestDataProcessor(unittest.TestCase):
    def setUp(self):
        self.libraries = [
            Library(1, 'Российская государственная библиотека'),
            Library(2, 'Библиотека им. Н.А. Некрасова'),
            Library(3, 'Библиотека № 19 им. Ф. М. Достоевского'),
            Library(4, 'Дом Н.В. Гоголя'),
            Library(5, 'Библиотека им. Н.А. Добролюбова')
        ]
        self.books = [
            Book(1, 'Война и мир', 1400, 1),
            Book(2, 'Мастер и Маргарита', 400, 2),
            Book(3, 'Тихий Дон', 1500, 2),
            Book(4, 'Преступление и наказание', 450, 3),
            Book(5, 'Мертвые души', 400, 4),
            Book(6, 'Герой нашего времени', 350, 4),
            Book(7, 'Евгений Онегин', 200, 5)
        ]
        self.books_libs = [
            BookLib(1, 1),
            BookLib(1, 2),
            BookLib(2, 2),
            BookLib(2, 3),
            BookLib(3, 1),
            BookLib(3, 4),
            BookLib(4, 5),
            BookLib(4, 6),
            BookLib(5, 7)
        ]
        self.processor = DataProcessor(self.libraries, self.books,
self.books_libs)

    def test_one_to_many_relationship(self):
        expected = [
            ('Евгений Онегин', 200, 'Библиотека им. Н.А. Добролюбова'),
            ('Мастер и Маргарита', 400, 'Библиотека им. Н.А. Некрасова'),
            ('Тихий Дон', 1500, 'Библиотека им. Н.А. Некрасова'),
            ('Преступление и наказание', 450, 'Библиотека № 19 им. Ф. М.
Достоевского'),
            ('Мертвые души', 400, 'Дом Н.В. Гоголя'),
            ('Герой нашего времени', 350, 'Дом Н.В. Гоголя'),
            ('Война и мир', 1400, 'Российская государственная библиотека')
        ]
        result = self.processor.one_to_many_relationship()
        self.assertEqual(result, expected)

    def test_total_pages_per_library(self):
        one_to_many = self.processor.one_to_many_relationship()
        expected = [
            ('Библиотека им. Н.А. Некрасова', 1900),
            ('Российская государственная библиотека', 1400),
            ('Дом Н.В. Гоголя', 750),
            ('Библиотека № 19 им. Ф. М. Достоевского', 450),
            ('Библиотека им. Н.А. Добролюбова', 200)
        ]
        result = self.processor.total_pages_per_library(one_to_many)
        self.assertEqual(result, expected)

    def test_books_in_libraries(self):
        many_to_many = self.processor.many_to_many_relationship()
        expected = {
            'Российская государственная библиотека': ['Война и мир', 'Мастер
и Маргарита'],
            'Библиотека им. Н.А. Некрасова': ['Мастер и Маргарита', 'Тихий
Дон'],
            'Библиотека № 19 им. Ф. М. Достоевского': ['Война и мир',
'Преступление и наказание'],
            'Библиотека им. Н.А. Добролюбова': ['Евгений Онегин']
        }
        result = self.processor.books_in_libraries(many_to_many)
        self.assertEqual(result, expected)


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

Результат тестирования



```
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

PS D:\3_sem> python -m unittest RK\RK2.py
...
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
PS D:\3_sem>
```