Группа: ИУ5-31Б

Студент: Нагдасёв Дмитрий

RK2.py

```python
from operator import itemgetter

class Book:
    """Книга"""
    def __init__(self, id, title, price, shop_id):
        self.id = id
        self.title = title
        self.price = price
        self.shop_id = shop_id

class BookShop:
    """Книжный магазин"""
    def __init__(self, id, shop_name):
        self.id = id
        self.shop_name = shop_name

class Sales:
    """Книги в магазинах для реализации связи многие-ко-многим"""
    def __init__(self, book_id, shop_id):
        self.book_id = book_id
        self.shop_id = shop_id

def create_one_to_many(books, bookshops):
    """Создает связь один-ко-многим"""
    return [
        (b.title, b.price, bs.shop_name)
        for bs in bookshops
        for b in books
        if b.shop_id == bs.id
    ]

def create_many_to_many(books, bookshops, sales):
    """Создает связь многие-ко-многим"""
    many_to_many_temp = [
        (bs.shop_name, bb.shop_id, bb.book_id)
        for bs in bookshops
        for bb in sales
        if bs.id == bb.shop_id
    ]

    return [
        (b.title, b.price, bs_name)
        for bs_name, shop_id, book_id in many_to_many_temp
        for b in books
        if b.id == book_id
```

```
    ]

def task_a1(one_to_many):
    """Задача A1: Сортировка один-ко-многим по имени магазина"""
    return sorted(one_to_many, key=itemgetter(2))

def task_a2(one_to_many, bookshops):
    """Задача A2: Сумма цен книг по магазинам"""
    res = []
    for bs in bookshops:
        bs_books = list(filter(lambda i: i[2] == bs.shop_name, one_to_many))
        if bs_books:
            bs_prices = [price for _, price, _ in bs_books]
            bs_prices_sum = sum(bs_prices)
            res.append((bs.shop_name, bs_prices_sum))
    return sorted(res, key=itemgetter(1), reverse=True)

def task_a3(many_to_many, bookshops):
    """Задача A3: Книги в магазинах с 'Book' в названии"""
    res = {}
    for bs in bookshops:
        if "Book" in bs.shop_name:
            bs_books = list(filter(lambda i: i[2] == bs.shop_name, many_to_many))
            bs_books_titles = [x for x, _, _ in bs_books]
            res[bs.shop_name] = bs_books_titles
    return res

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

    books = [
        Book(1, "Python 101", 25.99, 1),
        Book(2, "Advanced Python", 35.50, 2),
        Book(3, "Learning AI", 40.00, 1),
        Book(21, "Advanced C++", 50.0, 4),
        Book(22, "Rust For Beginners", 35.99, 5),
        Book(23, "ML: Profile Level", 49.99, 3)
    ]

    bookshops = [
        BookShop(1, "Tech Books"),
        BookShop(2, "Programming Hub"),
        BookShop(3, "AI Books"),
        BookShop(4, "IT Heaven"),
        BookShop(5, "Book++")
    ]

    sales = [
        Sales(1, 1),
        Sales(2, 2),
        Sales(3, 1),
```

```
        Sales(3, 2),
        Sales(3, 5),
        Sales(21, 4),
        Sales(22, 2),
        Sales(22, 1),
        Sales(23, 3),
        Sales(23, 3)
    ]

    one_to_many = create_one_to_many(books, bookshops)
    many_to_many = create_many_to_many(books, bookshops, sales)

    print("A1")
    print(task_a1(one_to_many))

    print("\nA2")
    print(task_a2(one_to_many, bookshops))

    print("\nA3")
    print(task_a3(many_to_many, bookshops))


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

tests.py

```
import unittest

from RK2 import *

class TestBookshopAnalysis(unittest.TestCase):

    def setUp(self):
        """Инициализация данных для тестов"""
        self.books = [
            Book(1, "Python 101", 25.99, 1),
            Book(2, "Advanced Python", 35.50, 2),
            Book(3, "Learning AI", 40.00, 1),
        ]

        self.bookshops = [
            BookShop(1, "Tech Books"),
            BookShop(2, "Programming Hub")
        ]

        self.sales = [
            Sales(1, 1),
            Sales(2, 2),
            Sales(3, 1),
```

```
        ]

    def test_task_a1(self):
        """Тест для задачи A1"""
        one_to_many = create_one_to_many(self.books, self.bookshops)
        result = task_a1(one_to_many)
        expected = [
            ("Advanced Python", 35.50, "Programming Hub"),
            ("Python 101", 25.99, "Tech Books"),
            ("Learning AI", 40.00, "Tech Books")
        ]
        self.assertEqual(result, expected)


    def test_task_a2(self):
        """Тест для задачи A2"""
        one_to_many = create_one_to_many(self.books, self.bookshops)
        result = task_a2(one_to_many, self.bookshops)
        expected = [("Tech Books", 65.99), ("Programming Hub", 35.50)]
        self.assertEqual(result, expected)

    def test_task_a3(self):
        """Тест для задачи A3"""
        many_to_many = create_many_to_many(self.books, self.bookshops,
self.sales)
        result = task_a3(many_to_many, self.bookshops)
        expected = {"Tech Books": ["Python 101", "Learning AI"]}
        self.assertEqual(result, expected)

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

Результат:

```
...
----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
PS C:\Users\dmnag\OneDrive\Desktop\labs_paradigms\RK2>
```