**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 «Системы обработки информации и управления»**

**Курс «Базовые компоненты интернет-технологий»**

Отчёт по Рубежному Контролю №1

по курсу «Базовые компоненты и интернет-технологии»

Вариант 16

Выполнил:

студент группы ИУ5-33Б

Рыбин Владислав

Проверил:

к.т.н., доц., Ю. Е. Гапанюк

2022 г.

Полученное задание:

Необходимо создать два класса данных в соответствии с Вашим вариантом предметной области, которые связаны отношениями один-ко-многим и многие-ко-многим.

Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), таким образом, чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.

Необходимо разработать запросы в соответствии с Вашим вариантом.

Предметная область: класс\_1 – Книга, класс\_2 – Книжный магазин, вариант запросов: В.

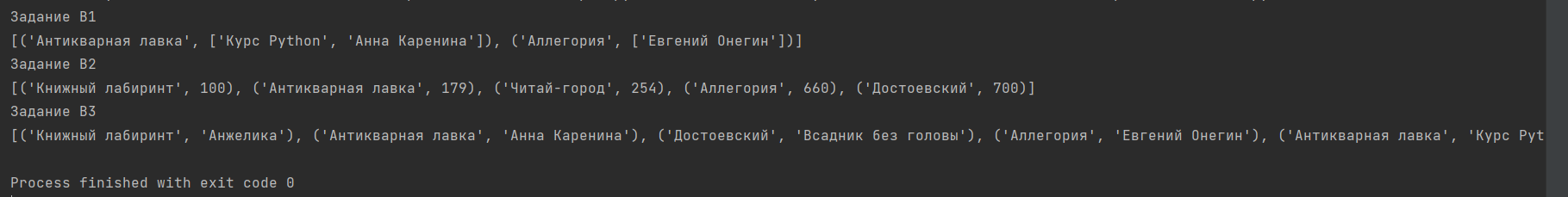
Запросы:

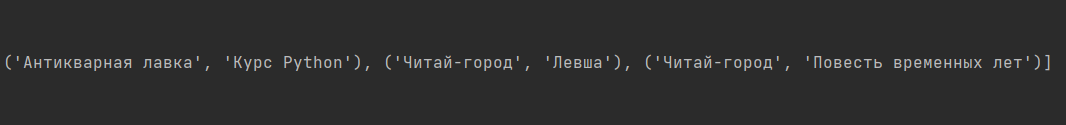
1. «Книга» и «Книжный магазин» связаны соотношением один-ко-многим. Выведите список всех книжных магазинов, у которых название начинается с буквы «А», и их книги.
2. «Книга» и «Книжный магазин» связаны соотношением один-ко-многим. Выведите список магазинов с минимальной ценой книги в каждом, отсортированных по минимальной цене.
3. «Книга» и «Книжный магазин» связаны соотношением многие-ко-многим. Выведите список всех связанных книг и магазинов, отсортированных по книгам, сортировка по магазинам произвольная.

Текст программы:

from operator import itemgetter  
  
  
class Book:  
 *"""Книга"""* def \_\_init\_\_(self, id, price, name, shop\_id):  
 self.id = id  
 self.price = price  
 self.name = name  
 self.shop\_id = shop\_id  
  
class Shop:  
 *"""Книжный магазин"""* def \_\_init\_\_(self, id, name):  
 self.id = id  
 self.name = name  
  
  
class BookShop:  
 *""" 'Книги магазина' для реализации связи многие-ко-многим """* def \_\_init\_\_(self, shop\_id, book\_id):  
 self.shop\_id = shop\_id  
 self.book\_id = book\_id  
  
  
Shops = [  
 Shop(1, 'Книжный лабиринт'),  
 Shop(2, 'Читай-город'),  
 Shop(3, 'Антикварная лавка'),  
 Shop(4, 'Достоевский'),  
 Shop(5, 'Аллегория')  
]  
  
Books = [  
 Book(1, 100, 'Анжелика', 1),  
 Book(2, 339, 'Левша', 2),  
 Book(3, 254, 'Повесть временных лет', 2),  
 Book(4, 468, 'Курс Python', 3),  
 Book(5, 179, 'Анна Каренина', 3),  
 Book(6, 700, 'Всадник без головы', 4),  
 Book(7, 660, 'Евгений Онегин', 5)  
]  
  
Books\_Shops = [  
 BookShop(1, 1),  
 BookShop(2, 2),  
 BookShop(2, 3),  
 BookShop(3, 4),  
 BookShop(3, 5),  
 BookShop(4, 6),  
 BookShop(5, 7)  
]  
  
  
def main():  
 one\_to\_many = [(b.name, b.price, s.name)  
 for s in Shops  
 for b in Books  
 if b.shop\_id == s.id]  
  
 many\_to\_many\_temp = [(s.name, bs.shop\_id, bs.book\_id)  
 for s in Shops  
 for bs in Books\_Shops  
 if s.id == bs.shop\_id]  
  
 many\_to\_many = [(name, b.name)  
 for name, shop\_id, book\_id in many\_to\_many\_temp  
 for b in Books if b.id == book\_id]  
  
 print('Задание В1')  
 res\_1 = []  
 for i in Shops:  
 book = list(filter(lambda a: a[2] == i.name, one\_to\_many))  
 if i.name[0] == 'А':  
 s\_name = [x for x, \_, \_ in book]  
 res\_1.append((i.name, s\_name))  
 print(res\_1)  
  
 print('Задание В2')  
 res\_unsorted = []  
 for s in Shops:  
 s\_book = list(filter(lambda i: i[2] == s.name, one\_to\_many))  
 if len(s\_book) > 0:  
 prices = [price for \_, price, \_ in s\_book]  
 prices\_min = min(prices)  
 res\_unsorted.append((s.name, prices\_min))  
 res\_2 = sorted(res\_unsorted, key=itemgetter(1))  
 print(res\_2)  
  
 print('Задание В3')  
 res\_13 = {}  
 # Перебираем все отделы  
 many\_to\_many.sort(key=lambda i: i[1])  
 print(many\_to\_many)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Результаты выполнения:





(продолжение результата на третий запрос)