Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологий»
Отчёт по РК 1

Выполнил:
студент группы ИУ5-31Б
Шимолина Полина
Кирилловна

Подпись:_____

Дата:_____

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий
Евгеньевич

Подпись:_____

Дата:_____

Москва, 2021 г.

Задание:

Рубежный контроль представляет собой разработку программы на языке Python, которая выполняет следующие действия:

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

Пример классов данных для предметной области Сотрудник-Отдел:

1. Класс «Глава», содержащий поля:
   - ID записи о главе;
   - Название главы;
   - Количество страниц в главе (количественный признак);
   - ID записи о книге. (для реализации связи один-ко-многим)
2. Класс «Книга», содержащий поля:
   - ID записи о книге;
   - Название книги.
3. (Для реализации связи многие-ко-многим) Класс «Главы книги», содержащий поля:
   - ID записи о главе;
   - ID записи о книге.

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

3) Необходимо разработать запросы в соответствии с вариантом.

1. «Книга» и «Глава» связаны соотношением один-ко-многим. Выведите список всех связанных глав и книг, отсортированный по книгам, сортировка по главам произвольная.
2. «Книга» и «Глава» связаны соотношением один-ко-многим. Выведите список книг с суммарным количеством страниц в каждой книге, отсортированный по суммарному количеству страниц.
3. «Книга» и «Глава» связаны соотношением многие-ко-многим. Выведите список всех книг, у которых в названии присутствует слово “оригинала”, и список содержащихся в них глав.

ыТекст программы:

```
from operator import itemgetter

class Chapter:
    """Глава"""
    def __init__(self, id, title, pages, book_id):
        self.id = id
        self.title = title
        self.pages = pages
        self.book_id = book_id

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

class ChapterBook:
    """
    'Глава книги' для реализации
    связи многие-ко-многим
    """
    def __init__(self, book_id, chapter_id):
        self.book_id = book_id
        self.chapter_id = chapter_id

# Книги
books = [
    Book(1, 'Понедельник начинается в субботу'),
    Book(2, 'Горе от ума'),
    Book(3, 'Убить пересмешника (на языке оригинала)'),

    Book(11, 'Повелитель мух (на языке оригинала'),
    Book(22, '1984 (на языке оригинала)'),
    Book(33, 'Пикник на обочине'),
]

# Главы
chapters = [
    Chapter(1, 'Глава 1', 11, 1),
    Chapter(2, 'Глава 2', 35, 2),
    Chapter(3, 'Глава 3', 40, 3),
    Chapter(4, 'Глава 4', 30, 3),
    Chapter(5, 'Глава 5', 15, 3),
]

chapters_books = [
    ChapterBook(1,1),
    ChapterBook(2,2),
    ChapterBook(3,3),
    ChapterBook(3,4),
    ChapterBook(3,5),
```

```
    ChapterBook(11,1),
    ChapterBook(22,2),
    ChapterBook(33,3),
    ChapterBook(33,4),
    ChapterBook(33,5),
]

def main():

    # Соединение данных один-ко-многим
    one_to_many = [(c.title, c.pages, b.title)
        for b in books
        for c in chapters
        if c.book_id == b.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(b.title, cb.book_id, cb.chapter_id)
        for b in books
        for cb in chapters_books
        if b.id == cb.book_id]

    many_to_many = [(c.title, c.pages, book_title)
        for book_title, book_id, chapter_id in many_to_many_temp
        for c in chapters if c.id == chapter_id]

    print('Задание А1')
    res_1 = sorted(one_to_many, key=itemgetter(2))
    print(res_1)

    print('\nЗадание А2')
    res_2_unsorted = []
    # Перебираем все книги
    for b in books:
        # Список глав книги
        b_chapters = list(filter(lambda i: i[2]==b.title, one_to_many))
        # Если книга не пустая
        if len(b_chapters) > 0:
            # Количество страниц глав книги
            b_pages = [pages for _,pages,_ in b_chapters]
            # Суммарное количество страниц глав книги
            b_pages_sum = sum(b_pages)
            res_2_unsorted.append((b.title, b_pages_sum))

    # Сортировка по суммарному количеству страниц
    res_2 = sorted(res_2_unsorted, key=itemgetter(1), reverse=True)
    print(res_2)

    print('\nЗадание А3')
    res_3 = {}
    # Перебираем все книги
    for b in books:
        if 'оригинала' in b.title:
            # Список глав книги
```

```
        b_chapters = list(filter(lambda i: i[2] == b.title, many_to_many))
        # Только названия глав
        b_chapters_titles = [x for x,_,_ in b_chapters]
        # Добавляем результат в словарь
        # ключ - книга, значение - список названий
        res_3[b.title] = b_chapters_titles

    print(res_3)

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

Результаты:

```
Задание A1
[('Глава 2', 35, 'Горе от ума'), ('Глава 1', 11, 'Понедельник начинается в субботу'), ('Глава 3', 40, 'Убить пересмешника (на языке оригинала)'), ('Глава 4', 30, 'Убить пересмешника (на языке оригинала)'), ('Глава 5', 15, 'Убить пересмешника (на языке оригинала)')]

Задание A2
[('Убить пересмешника (на языке оригинала)', 85), ('Горе от ума', 35), ('Понедельник начинается в субботу', 11)]

Задание A3
{'Убить пересмешника (на языке оригинала)': ['Глава 3', 'Глава 4', 'Глава 5'], 'Повелитель мух (на языке оригинала': ['Глава 1'], '1984 (на языке оригинала)': ['Глава 2']}
```