Московский государственный технический университет им. Н.Э. Баумана.

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

Курс «Базовые компоненты интернет-технологий»

Отчет по рубежному контролю №1 по курсу БКИТ

Выполнил:
студент группы ИУ5-31Б
Чичикин Тимофей Дмитриевич

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич

Подпись и дата:

Подпись и дата:

Москва, 2021 г.

Задание. (Для 22 варианта предметной области и варианта запросов А)
Рубежный контроль представляет собой разработку программы на языке Python, которая

выполняет следующие действия:

1) Необходимо создать 2 класса данных в соответствии с Вашим вариантом предметной области, которые связанны отношением один-ко-многим и многие-ко-многим.
2) Необходимо создать списки объектов классов, содержащих тестовые данные (3-5 записей), чтобы первичные и вторичные ключи соответствующих записей были связаны по идентификаторам.
3) Необходимо разработать запросы в соответствии с Вашим вариантом. Запросы сформулированы в терминах классов «Сотрудник» и «Отдел», которые используются в примере. Необходимо перенести эти требования в свой вариант предметной области.

Результатом рубежного контроля является документ в формате PDF, который содержит текст программы и результат выполнения.

Запросы

1) «Библиотека» и «Язык программирования» связаны соотношением один-ко-многим. Выведите список всех связанных библиотек и языков, отсортированный по языкам, сортировка библиотек произвольная.
2) «Библиотека» и «Язык программирования» связаны соотношением один-ко-многим. Выведите список языков с суммарным объемом всех библиотек каждого языка, отсортированный по объему.
3) «Библиотека» и «Язык программирования» связаны соотношением многие-ко-многим. Выведите список всех языков, в названии которых есть слово «Lang» и список всех их библиотек.

Текст программы:

```
from operator import itemgetter

class Lib:
    """Библиотека"""
    def __init__(self, id, Lib_name, volum, Lang_id):
        self.Lib_id = id
        self.Lib_name = Lib_name
        self.volum = volum      #Объем в килобайтах
        self.Lang_id = Lang_id

class Lang:
    """Язык программирования"""
    def __init__(self, id, name):
        self.Lang_id = id
        self.Lang_name = name

class Lib_Lang:
    """'Библиотеки языков' для реализации связи многие-ко-многим"""
    def __init__(self, Lang_id, Lib_id):
        self.Lang_id = Lang_id
        self.Lib_id = Lib_id
# Языки
Langs = [
    Lang(1, 'Python'),
    Lang(2, 'C++ Lang'),
    Lang(3, 'C# Lang'),

    Lang(11, 'Ruby'),
    Lang(22, 'Swift'),
    Lang(33, 'Pascal'),
]
# Библиотеки
Libs = [
    Lib(1, 'Colorama', 3, 1),
    Lib(2, 'SyST', 14, 2),
    Lib(3, 'EBalance', 4, 3),
    Lib(4, 'Rusty', 8, 3),
    Lib(5, 'XYZ', 10, 3),
```

```
]
Libs_Langs = [
    Lib_Lang(1, 1),
    Lib_Lang(2, 2),
    Lib_Lang(3, 3),
    Lib_Lang(3, 4),
    Lib_Lang(3, 5),

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

def main():
    """Основная функция"""
    # Соединение данных один-ко-многим
    one_to_many = [(e.Lib_name, e.volum, d.Lang_name)
                   for d in Langs
                   for e in Libs
                   if e.Lang_id == d.Lang_id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(d.Lang_name, ed.Lang_id, ed.Lib_id)
                         for d in Langs
                         for ed in Libs_Langs
                         if d.Lang_id == ed.Lang_id]

    many_to_many = [(e.Lib_name, e.volum, Lang_name)
                    for Lang_name, dep_id, Lib_id in many_to_many_temp
                    for e in Libs if e.Lib_id == Lib_id]

    print('Задание А1')
    res_11 = sorted(one_to_many, key=itemgetter(2))
    print(res_11)
    print("#Список связанных библиотек и языков, отсортированный по языкам")

    print('\nЗадание А2')
    res_12_unsorted = []
    for d in Langs:
        # Сортированный список языков по их имени
        d_Langs = list(filter(lambda i: i[2] == d.Lang_name, one_to_many))
        if len(d_Langs) > 0:
            d_Libs = [val for _, val, _ in d_Langs]
            # Суммарный объем библиотек языка
            d_Vol_sum = sum(d_Libs)
            res_12_unsorted.append((d.Lang_name, d_Vol_sum))
    # Сортировка по объему
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)
    print("#Список языков программирования и суммарного объема их библиотек,")
    print("отсортированный по объему")

    print('\nЗадание А3')
    res_13 = {}
    for d in Langs:
        if 'Lang' in d.Lang_name:
            # Список библиотек языка программирования
            d_Langs = list(filter(lambda i: i[2] == d.Lang_name, many_to_many))
            # Только название библиотек
            d_Libs_names = [x for x, _, _ in d_Langs]
            # Добавляем результат в словарь
            # ключ - язык, значение - список имен библиотек
            res_13[d.Lang_name] = d_Libs_names
    print(res_13)
```

```
    print("#Список всех языков, у которых в названии есть 'Lang' и список всех их
библиотек")

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

Результат:

```
Задание A1
[('EBalance', 4, 'C# Lang'), ('Rusty', 8, 'C# Lang'), ('XYZ', 10, 'C# Lang'), ('SyST', 14, 'C++ Lang'), ('Colorama', 3, 'Python')]
#Список связанных библиотек и языков, отсортированный по языкам

Задание A2
[('C# Lang', 22), ('C++ Lang', 14), ('Python', 3)]
#Список языков программирования и суммарного объема их библиотек,
отсортированный по объему

Задание A3
{'C++ Lang': ['SyST'], 'C# Lang': ['EBalance', 'Rusty', 'XYZ']}
#Список всех языков, у которых в названии есть 'Lang' и список всех их библиотек

Process finished with exit code 0
```