Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Московский государственный технический университет имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ	**«Информатики и систем управления»**
КАФЕДРА	ИУ5

Дисциплина «Разработка интернет-приложений»

Отчет по рубежному контролю №1
Вариант Б-19

Студент	группы ИУ5-52Б	Нищук Р.С.
Преподаватель		Гапанюк Ю.Е.

Классы для предметной области 1.

Класс «Детали», содержащий поля:

- ID записи о детали;
- Название детали;
- ID записи о производителе. (для реализации связи одинко-многим)

2. Класс «Производители», содержащий поля:
   - ID записи о производителе;
   - Наименование фирмы производителя.
3. (Для реализации связи многие-ко-многим) Класс «Детали фирмы производителя», содержащий поля:
   - ID записи о детали; • ID записи о производителе.

Задание

1. «Производитель» и «Деталь» связаны соотношением один-ко-многим. Выведите список всех связанных деталей и производителей, отсортированный по деталям, сортировка по производителю произвольная.
2. «Производитель» и «Деталь» связаны соотношением один-ко-многим. Выведите список производителей с количеством деталей у каждого производителя, отсортированный по количеству деталей
3. «Производитель» и «Деталь» связаны соотношением многие-ко-многим. Выведите список всех деталей, у которых название заканчивается на «l», и названия производителей.

Код :

```
#Вариант Б 19; Нищук Роман Сергеевич ИУ5-52Б

# используется для сортировки
from operator import itemgetter

class Detail:
    """Деталь"""
    def __init__(self, id, name, Fabricator_id):
        self.id = id
        self.name = name
        self.Fabricator_id = Fabricator_id
```

```python
class Fabricator:
    """Производитель"""
    def __init__(self, id, name):
        self.id = id
        self.name = name

class DetailFabricator:
    """
    'Производители деталей' для реализации
    связи многие-ко-многим
    """
    def __init__(self, Fabricator_id, Detail_id):
        self.Fabricator_id = Fabricator_id
        self.Detail_id = Detail_id

# Производители
Fabricators = [
    Fabricator(1, 'Ford'),
    Fabricator(2, 'Ferrari'),
    Fabricator(3, 'Nissan'),
    Fabricator(11, 'Renault'),
    Fabricator(22, 'Peugeot'),
    Fabricator(33, 'Maserati'),
]

# Детали
Details = [
    Detail(1, 'Wheel', 1),
    Detail(2, 'Carcase', 1),
    Detail(3, 'Engine', 3),
    Detail(4, 'Painting', 11),
    Detail(5, 'Glasses', 11),
    Detail(6, 'Headlights', 33),
    Detail(7, 'Battery', 11),
]

Details_Fabricators = [
    DetailFabricator(1,1),
    DetailFabricator(2,2),
    DetailFabricator(3,3),
    DetailFabricator(22,4),
    DetailFabricator(3,5),

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


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

    # Соединение данных один-ко-многим
    one_to_many = [(l.name, p.name)
        for p in Fabricators
        for l in Details
        if l.Fabricator_id==p.id]

    # Соединение данных многие-ко-многим
```

```
    many_to_many_temp = [(l.name, pl.Fabricator_id, pl.Detail_id)
        for l in Fabricators
        for pl in Details_Fabricators
        if l.id==pl.Fabricator_id]

    many_to_many = [(p.name, Fabricator_name)
        for Fabricator_name, Fabricator_id, Detail_id in many_to_many_temp
        for p in Details if p.id==Detail_id]

    print('Задание Б1')
    res_11 = sorted(one_to_many, key=itemgetter(0))
    print(res_11)

    print('\nЗадание Б2')
    res_12_unsorted = []
    # Перебираем всех производителей
    for p in Fabricators:
        # Список деталей
        l_Details = list(filter(lambda i: i[1]==p.name, one_to_many))
        # Если производитель не пустой
        if len(l_Details) > 0:
            res_12_unsorted.append((p.name, len(l_Details)))

    # Сортировка по количеству библиотек
    res_12 = sorted(res_12_unsorted, key=itemgetter(1), reverse=True)
    print(res_12)

    print('\nЗадание Б3')
    res_13 = {}
    # Перебираем все Детали
    for l in Details:
        if l.name[-1]=="l":
            # Список Деталей
            l_Details = list(filter(lambda i: i[0]==l.name, many_to_many))
            # Только наименования производителей
            d_Details_names = [x for _,x in l_Details]
            # Добавляем результат в словарь
            # ключ - Деталь, значение - список названий производителей
            res_13[l.name] = d_Details_names

    print(res_13)

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

Результат выполнения программы:

Задание Б1
[('Glasses', 'Renault'), ('Painting', 'Renault'), ('Wheel', 'Ford'), ('Battery', 'Renault'), ('Engine', 'Nissan') , ('Headlights', 'Maserati'), ('Carcase', 'Ford')]
Задание Б2
[('Renault', 3), ('Ford', 2), ('Nissan', 1), ('Maserati', 1)]
Задание Б3
{'Wheel': ['Ford', 'Renault']}