```python
from dataclasses import dataclass

# ---------- классы данных ----------

@dataclass
class Supplier: # Поставщик (аналог Отдела)
id: int
name: str

@dataclass
class Detail: # Деталь (аналог Сотрудника)
id: int
name: str
price: int
supplier_id: int # связь один-ко-многим: основной поставщик

@dataclass
class DetailSupplier: # связь многие-ко-многим
detail_id: int
supplier_id: int

# ---------- тестовые данные ----------

suppliers = [
Supplier(1, "Иванов ИП"),
Supplier(2, "МеталлПром"),
Supplier(3, "Детали+"),
Supplier(4, "Технолов"),
Supplier(5, "СтройСнаб"),
]

details = [
Detail(1, "Гайка", 7, 1),
Detail(2, "Шайба", 5, 1),
Detail(3, "Болт", 10, 2),
Detail(4, "Втулка", 12, 2),
Detail(5, "Крышка", 15, 3),
Detail(6, "Ручка", 20, 4),
Detail(7, "Прокладка", 8, 5),
]

# множество для связи многие-ко-многим
```

```
detail_supplier = [
DetailSupplier(1, 1),
DetailSupplier(2, 1),
DetailSupplier(3, 2),
DetailSupplier(4, 2),
DetailSupplier(5, 3),
DetailSupplier(6, 4),
DetailSupplier(7, 5),
DetailSupplier(2, 3), # Гайка у поставщика «Детали+»
DetailSupplier(4, 5), # Втулка у «СтройСнаб»
DetailSupplier(5, 2), # Крышка у «МеталлПром»
]

# ---------- построение связей ----------

# один-ко-многим (Detail -> Supplier)
one_to_many = [
(d.name, d.price, s.name)
for s in suppliers
for d in details
if d.supplier_id == s.id
]

# многие-ко-многим
many_to_many_temp = [
(s.name, ds.detail_id)
for s in suppliers
for ds in detail_supplier
if s.id == ds.supplier_id
]

many_to_many = [
(d.name, d.price, s_name)
for s_name, detail_id in many_to_many_temp
for d in details
if d.id == detail_id
]

def main():
# 1. Список всех связанных деталей и поставщиков (1-ко-многим),
# отсортированный по деталям
res_1 = sorted(one_to_many, key=lambda x: x[0])
print("1) Деталь – Поставщик (1-ко-многим), сортировка по детали:")
```

```
for name, price, supplier in res_1:
print(f"{name:10} | цена {price:2} | {supplier}")
print()

# 2. Список поставщиков с количеством деталей, отсортированный по количеству
res_2 = [
(s.name, len([d for d in details if d.supplier_id == s.id]))
for s in suppliers
]
# убираем поставщиков без деталей
res_2 = list(filter(lambda x: x[1] > 0, res_2))
res_2 = sorted(res_2, key=lambda x: x[1], reverse=True)

print("2) Поставщики и количество их деталей (по убыванию количества):")
for supplier, cnt in res_2:
print(f"{supplier:10} | деталей: {cnt}")
print()

# 3. Многие-ко-многим: все детали, название которых оканчивается на «ка»,
# и названия их поставщиков
res_3 = [
(detail_name, supplier_name)
for detail_name, _, supplier_name in many_to_many
if detail_name.endswith("ка")
]

print("3) Детали, оканчивающиеся на «ка», и их поставщики (многие-ко-многим):")
for detail_name, supplier_name in res_3:
print(f"{detail_name:10} | {supplier_name}")

name = "main"
if name == "main":
main()
```

```
oem@UbuntuArm:~/labs/Andreev-Sem03/labs/RK1$ python3 RK1.py
1) Деталь – Поставщик (1-ко-многим), сортировка по детали:
Болт       | цена 10 | МеталлПром
Втулка     | цена 12 | МеталлПром
Гайка      | цена  7 | Иванов ИП
Крышка     | цена 15 | Детали+
Прокладка  | цена  8 | СтройСнаб
```

Ручка | цена 20 | Технолов
Шайба | цена 5 | Иванов ИП

2) Поставщики и количество их деталей (по убыванию количества):
Иванов ИП | деталей: 2
МеталлПром | деталей: 2
Детали+ | деталей: 1
Технолов | деталей: 1
СтройСнаб | деталей: 1

3) Детали, оканчивающиеся на «ка», и их поставщики (многие-ко-многим):
Гайка | Иванов ИП
Втулка | МеталлПром
Крышка | МеталлПром
Крышка | Детали+
Ручка | Технолов
Прокладка | СтройСнаб
Втулка | СтройСнаб