Текст программы:

```
from typing import Union, NoReturn



class Component:
    """Класс Деталь"""

    def __init__(self, id: int, name: str, price: Union[int, float], fabric_id: int):
        self.__id = id
        self.__name = name
        self.__price = price
        self.__fabric_id = fabric_id

    @property
    def id(self) -> int:
        return self.__id

    @property
    def name(self) -> str:
        return self.__name

    @property
    def price(self) -> Union[int, float]:
        return self.__price

    @property
    def fabric_id(self) -> int:
        return self.__fabric_id
```

```
class Fabric:
    """Класс Производитель"""

    def __init__(self, id: int, name: str):
        self.__id = id
        self.__name = name

    @property
    def id(self) -> int:
        return self.__id

    @property
    def name(self) -> str:
        return self.__name


class FabricComponent:
    """Класс детали производителя"""

    def __init__(self, fabric_id: int, component_id: int):
        self.__fabric_id = fabric_id
        self.__component_id = component_id

    @property
    def fabric_id(self) -> int:
        return self.__fabric_id

    @property
```

```
    def component_id(self) -> int:
        return self.__component_id


def request1(components: list[Component], fabrics: list[Fabric]) -> NoReturn:
    print(
        "Запрос №1. Список всех связанных деталей и производителей, отсортированный по производителям, сортировка по деталям по цене."
    )
    response = [
        (c, f)
        for c in components
        for f in fabrics
        if c.fabric_id == f.id
    ]
    response.sort(key=lambda item: (item[1].name, item[0].price))

    for c, f in response:
        print(f"\tДеталь: {c.name} | Производитель: {f.name}")
    print()


def request2(components: list[Component], fabrics: list[Fabric]) -> NoReturn:
    print(
        "Запрос №2. Список производителей с суммарной стоимостью деталей, отсортированный по суммарной стоимости деталей"
    )
    response = {}
    for fabric in fabrics:
```

```
        sum_price = sum([c.price for c in components if c.fabric_id == fabric.id])
        response[fabric.name] = sum_price

    response_sorted = dict(sorted(response.items(), key=lambda item: item[1]))
    for fabric_name, sum_price in response_sorted.items():
        print(f"\tПроизводитель: {fabric_name} | Суммарная стоимость деталей: {sum_price}")
    print()


def request3(components: list[Component], fabrics: list[Fabric], fabric_components: list[FabricComponent]) -> NoReturn:
    print(
        "Запрос №3. Список всех производителей, у которых название оканчивается на 'АЗ' и список производимых ими деталей."
    )
    response = {
        f: [
            c
            for c in components
            for fc in fabric_components
            if fc.fabric_id == f.id and fc.component_id == c.id
        ]
        for f in fabrics
        if f.name.endswith("АЗ")
    }

    for fabric, components in response.items():
        print(f"\tПроизводитель: {fabric.name} | Детали: {[comp.name for comp in components]}")
```

```
    print()


def main() -> NoReturn:
    components = [
        Component(1, "Тормозные колодки", 12000, 1),
        Component(2, "Фары", 5000, 1),
        Component(3, "Заднее крыло", 10000, 2),
        Component(4, "Генератор", 15000, 3),
        Component(5, "Аккумулятор", 8000, 3),
        Component(6, "Тормозные диски", 9000, 4),
        Component(7, "Дворники", 3000, 4)
    ]

    fabrics = [
        Fabric(1, "АВТОВАЗ"),
        Fabric(2, "УАЗ"),
        Fabric(3, "КАМАЗ"),
        Fabric(4, "УВЗ"),
    ]

    fabric_component = [
        FabricComponent(1, 1),
        FabricComponent(1, 2),
        FabricComponent(2, 3),
        FabricComponent(3, 4),
        FabricComponent(3, 5),
        FabricComponent(4, 6),
        FabricComponent(4, 7)
```

```
    ]

    request1(components, fabrics)
    request2(components, fabrics)
    request3(components, fabrics, fabric_component)


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

Результат работы программы: