Московский государственный технический университет им. Н.Э. Баумана

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

Курс «Парадигмы и конструкции языков программирования»

Отчет по РК №1
Вариант запросов: В
Вариант предметной области: 19

Выполнил:
студент группы ИУ5-33Б
Нагапетян Валерий

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю. Е.

Москва, 2023 г.

Вариант запросов В.

1. «Деталь» и «Производитель» связаны соотношением один-ко-многим. Выведите список всех деталей, которые начинаются с «Тормоз» и названия их производителей.

2. «Деталь» и «Производитель» связаны соотношением один-ко-многим. Выведите список производителей с минимальной стоимостью деталей, отсортированный их по минимальной стоимости.

3. «Деталь» и «Производитель» связаны соотношением многие-ко-многим. Выведите список всех связанных деталей и производителей, отсортированный по производителям, сортировка деталей по цене.

Код 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
```

```python
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 деталей, которые начинаются с "Тормоз", и их производителей.'
    )
    response = [
        (c, f)
        for c in components
        for f in fabrics
        if c.fabric_id == f.id and c.name.startswith("Тормоз")
    ]
    for c, f in response:
        print(f"    Деталь: {c.name} | Производитель: {f.name}")
    print()
```

```python
def request2(components: list[Component], fabrics: list[Fabric]) -> NoReturn:
    print(
        "Запрос №2. Список производителей с минимальной стоимостью деталей."
    )
    response = {}
    for fabric in fabrics:
        minimal_price = min([c.price for c in components if c.fabric_id == fabric.id])
        response[fabric.name] = minimal_price

    response_sorted = dict(sorted(response.items(), key=lambda item: item[1]))
    for fabric_name, minimal_price in response_sorted.items():
        print(f"    Производитель: {fabric_name} | Минимальная стоимость детали:
{minimal_price}")
    print()


def request3(components: list[Component], fabrics: list[Fabric], fabric_components:
list[FabricComponent]) -> NoReturn:
    print(
        "Запрос №3. Список всех связанных деталей и производителей, отсортированный по
производителям. Сортировка деталей по цене."
    )
    response = [
        (f, c)
        for fc in fabric_components
        for f in fabrics
        for c in components
        if fc.fabric_id == f.id and fc.component_id == c.id
    ]
    response.sort(key=lambda item: (item[0].name, item[1].price))

    for fabric, component in response:
        print(f"    Производитель: {fabric.name} | Деталь: {component.name}")
    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

```
Запрос №1. Список деталей, которые начинаются с "Тормоз", и их производителей.
    Деталь: Тормозные колодки | Производитель: АВТОВАЗ
    Деталь: Тормозные диски | Производитель: УВЗ

Запрос №2. Список производителей с минимальной стоимостью деталей.
    Производитель: УВЗ | Минимальная стоимость детали: 3000
    Производитель: АВТОВАЗ | Минимальная стоимость детали: 5000
    Производитель: КАМАЗ | Минимальная стоимость детали: 8000
    Производитель: УАЗ | Минимальная стоимость детали: 10000

Запрос №3. Список всех связанных деталей и производителей, отсортированный по производителям. Сортировка деталей по цене.
    Производитель: АВТОВАЗ | Деталь: Фары
    Производитель: АВТОВАЗ | Деталь: Тормозные колодки
    Производитель: КАМАЗ | Деталь: Аккумулятор
    Производитель: КАМАЗ | Деталь: Генератор
    Производитель: УАЗ | Деталь: Заднее крыло
    Производитель: УВЗ | Деталь: Дворники
    Производитель: УВЗ | Деталь: Тормозные диски
```