МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ

КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования   «Московский технический университет связи и информатики»



Кафедра"Системное программирование"

Лабораторная работа № 2 Вариант № 14

По дисциплине

“Информационные технологии и программирование”

                                      Выполнил: студент гр. БВТ2201

Шамсутдинов Р.Ф.

Москва, 2023 г

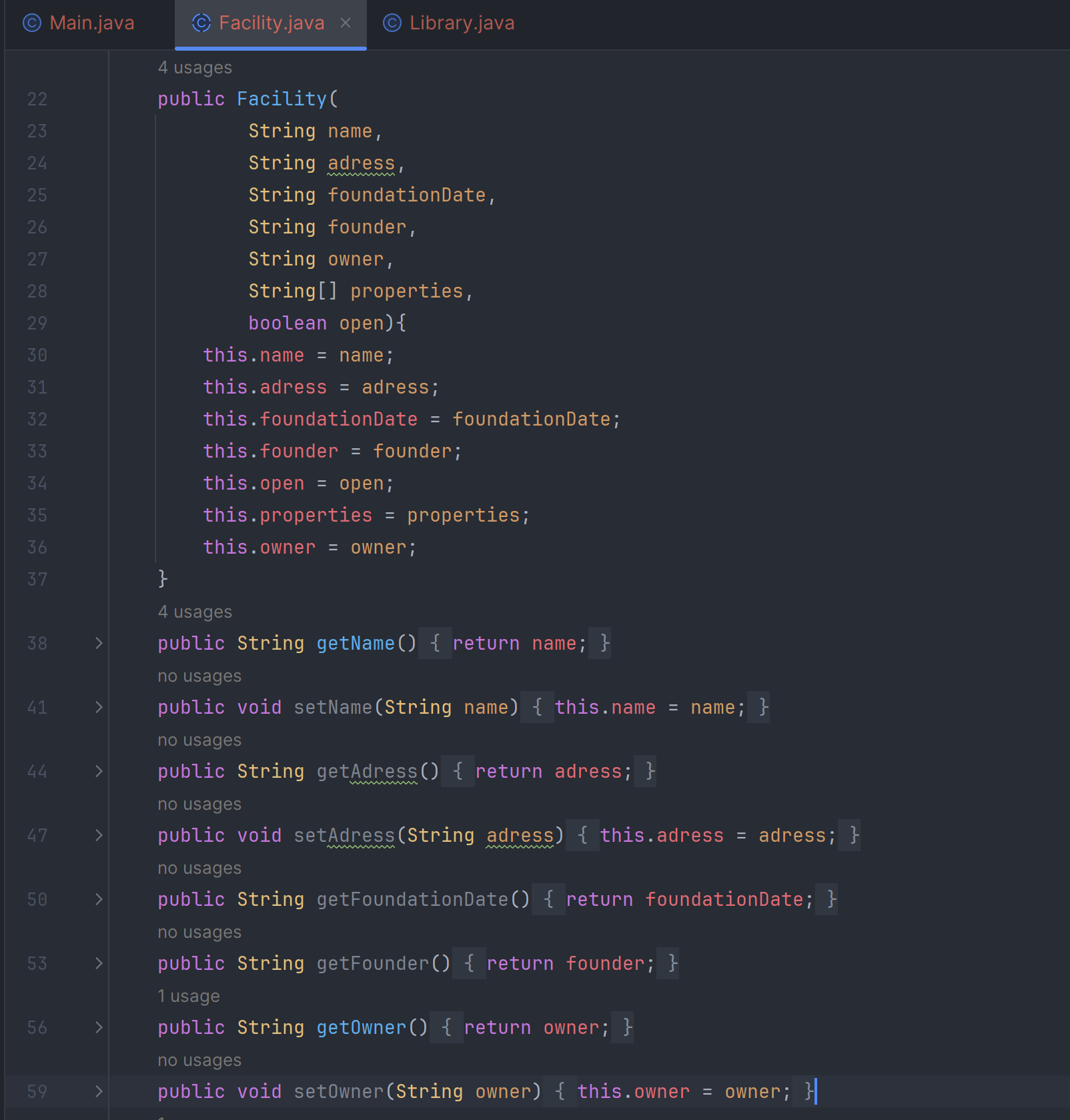
Цели и задачи:

Изучить основы ООП в языке java. Выполнить задание.

Ход работы:

1)Создаем базовый класс Facility. Описываем его поля, методы и конструкторы.





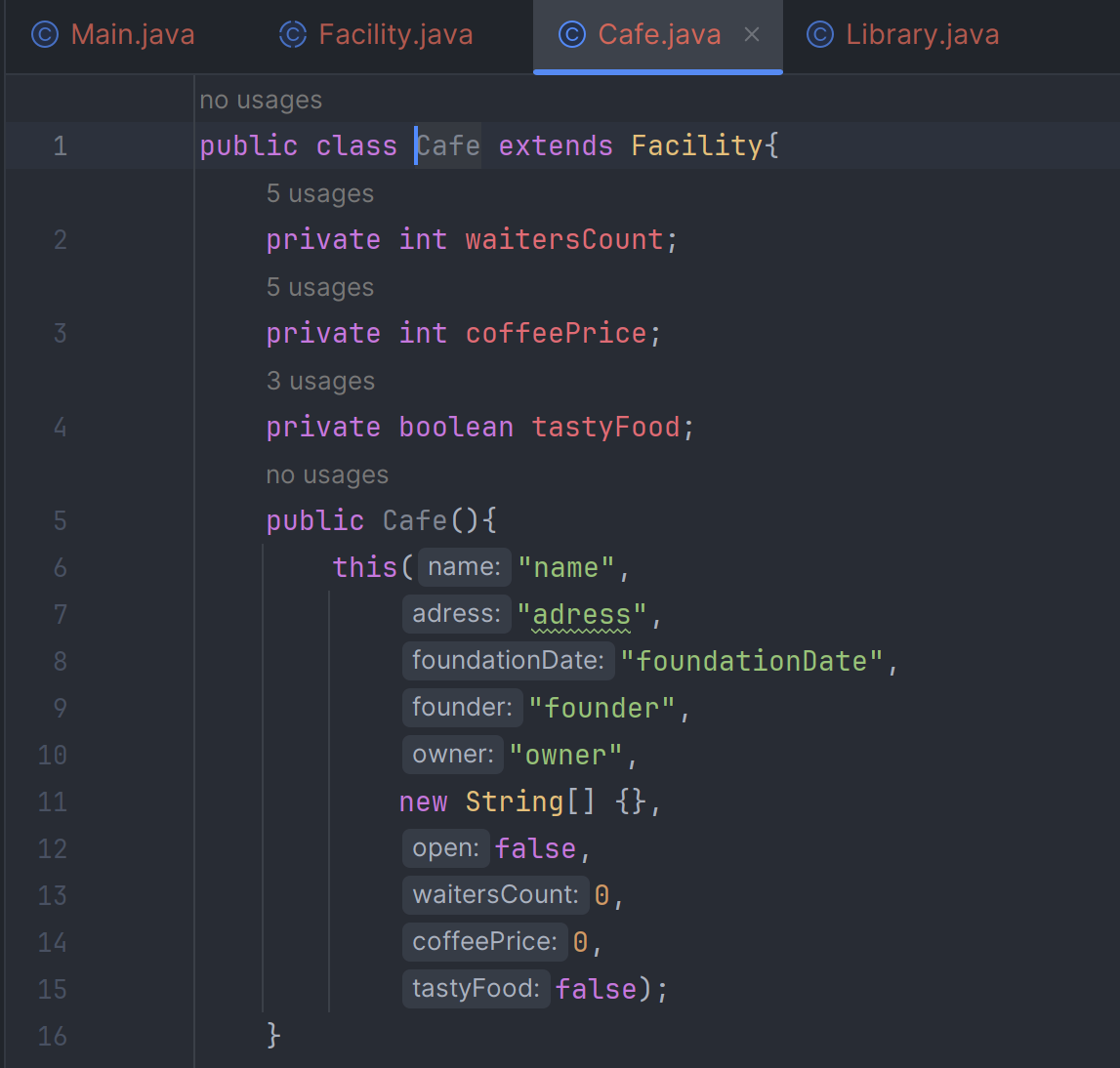


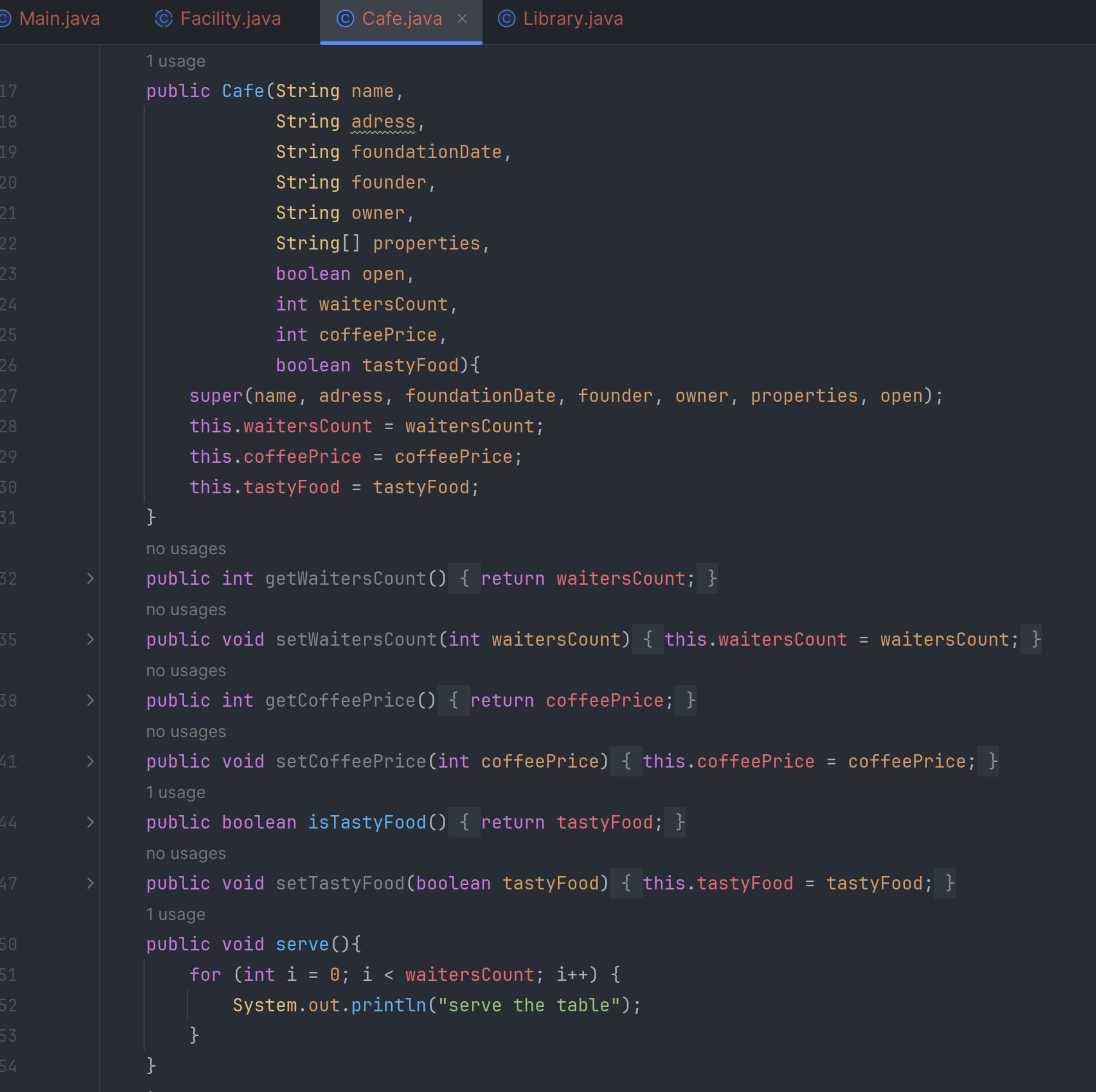
Комментарии:

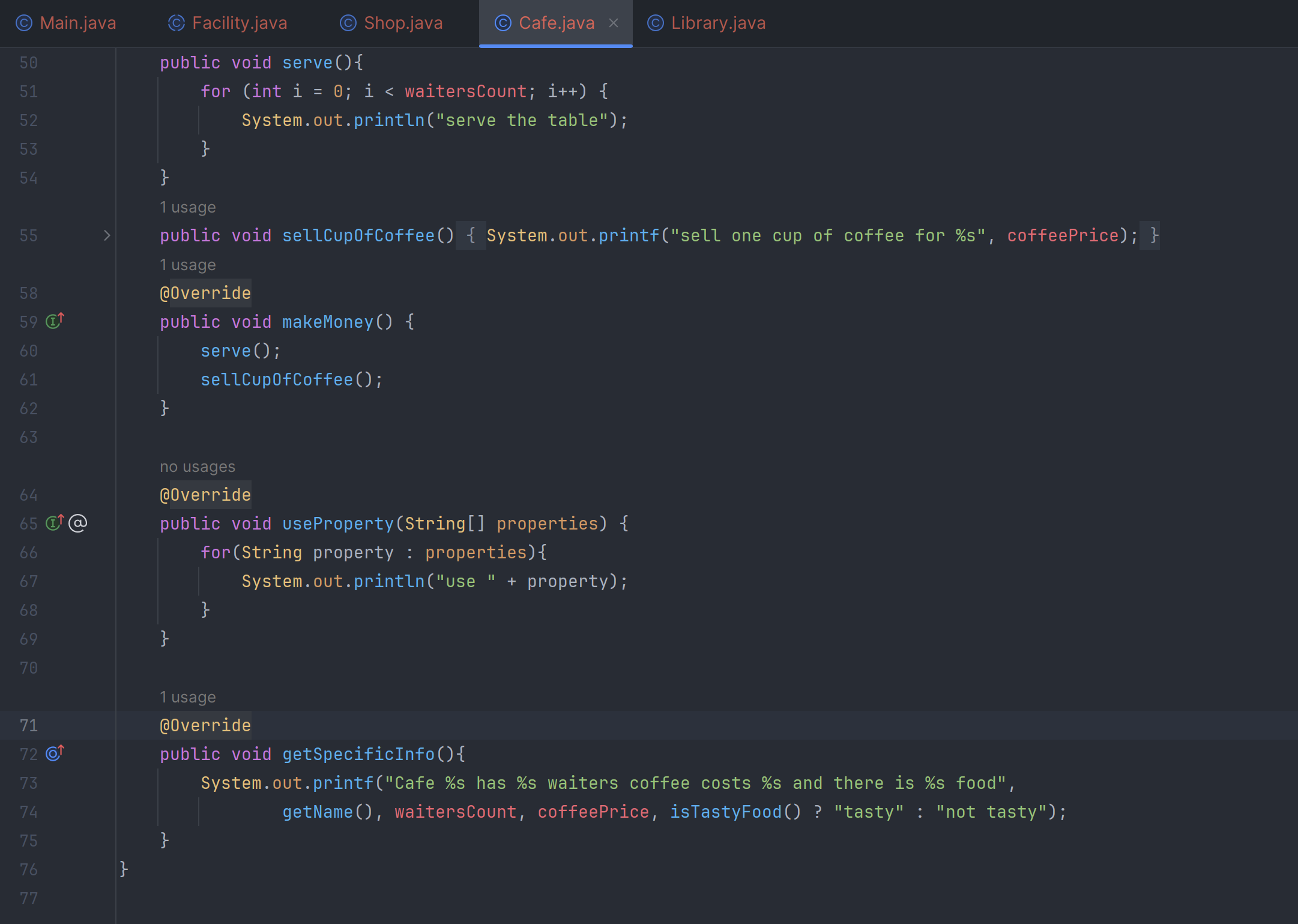
1. Нельзя создать объект абстрактного класса
2. Поля делаем private, чтобы ограничить доступ к ним вне тела данного класса
3. Создаем конструкторы, они бывают с параметрами и без, создаем оба варианта. Конструкторы нужны при создании объекта класса, с помощью них мы можем установить значения полей
4. Создаем геттеры и сеттеры. Так как поля имеют тип доступа private, то чтобы получить их значения или поменять извне тела класса, создаются геттеры и сеттеры. Геттеры возвращают значение поля, сеттеры изменяют его.
5. У класса есть методы. Можно сделать абстрактный метод, то есть объявить его, но не указывать что он делает, тогда в дочерних классах нужно будет определить их. Обычные методы объявляются и определяются сразу.

2) Создадим дочерние классы : Café, Library, Shop.

2.a) Café class





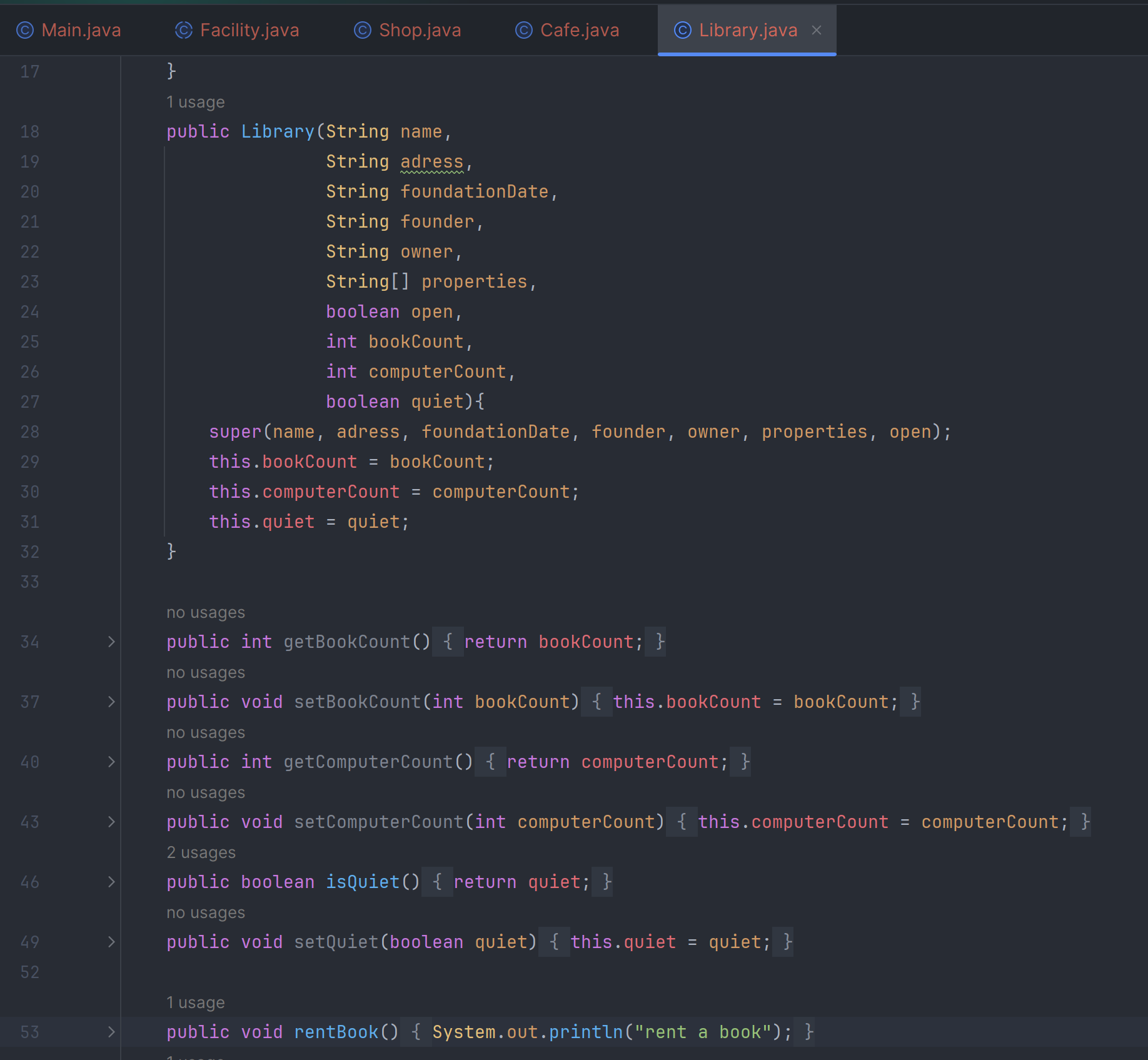


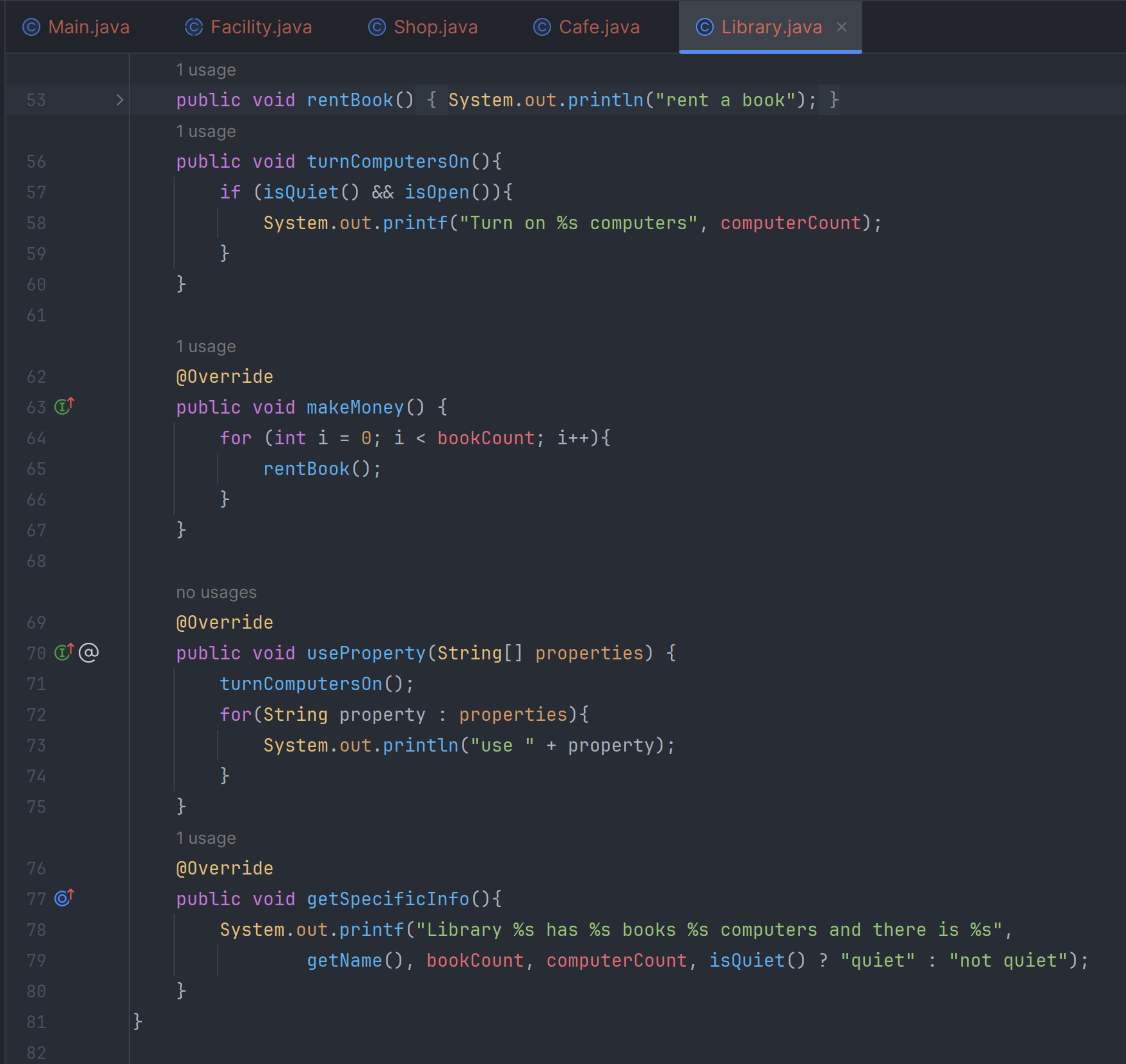
Комментарии:

1. Класс Café наследует класс Facility, то есть является его дочерним классом. (public class Cafe extends Facility). Это значит, что теперь классу Café доступны все методы класса Facility, а с помощью super можно обратиться к полям\методам базового класса Facility
2. У дочернего класса могут быть поля и методы, которых нет у базового класса
3. В конструкторах дочернего класса нужно указывать поля как и дочернего так и родительского класса (super(name, adress, foundationDate, founder, owner, properties, open)
4. Так как в базовом классе Facility были абстрактные методы, то нам нужно их иплементировать (написать тело метода).
5. При переопределении метода из базового класса рекомендуется использовать @Override

2.б) Library class

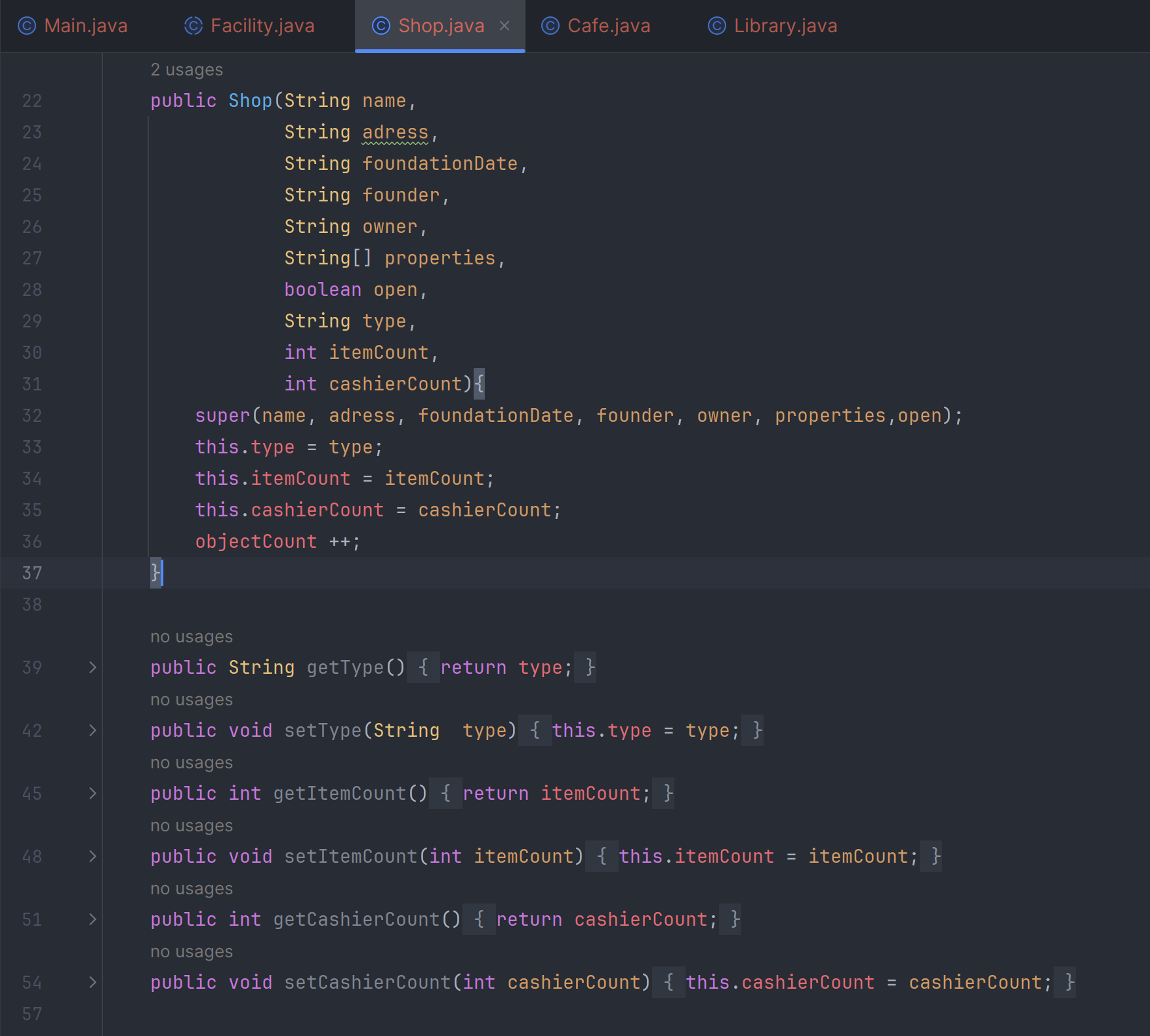


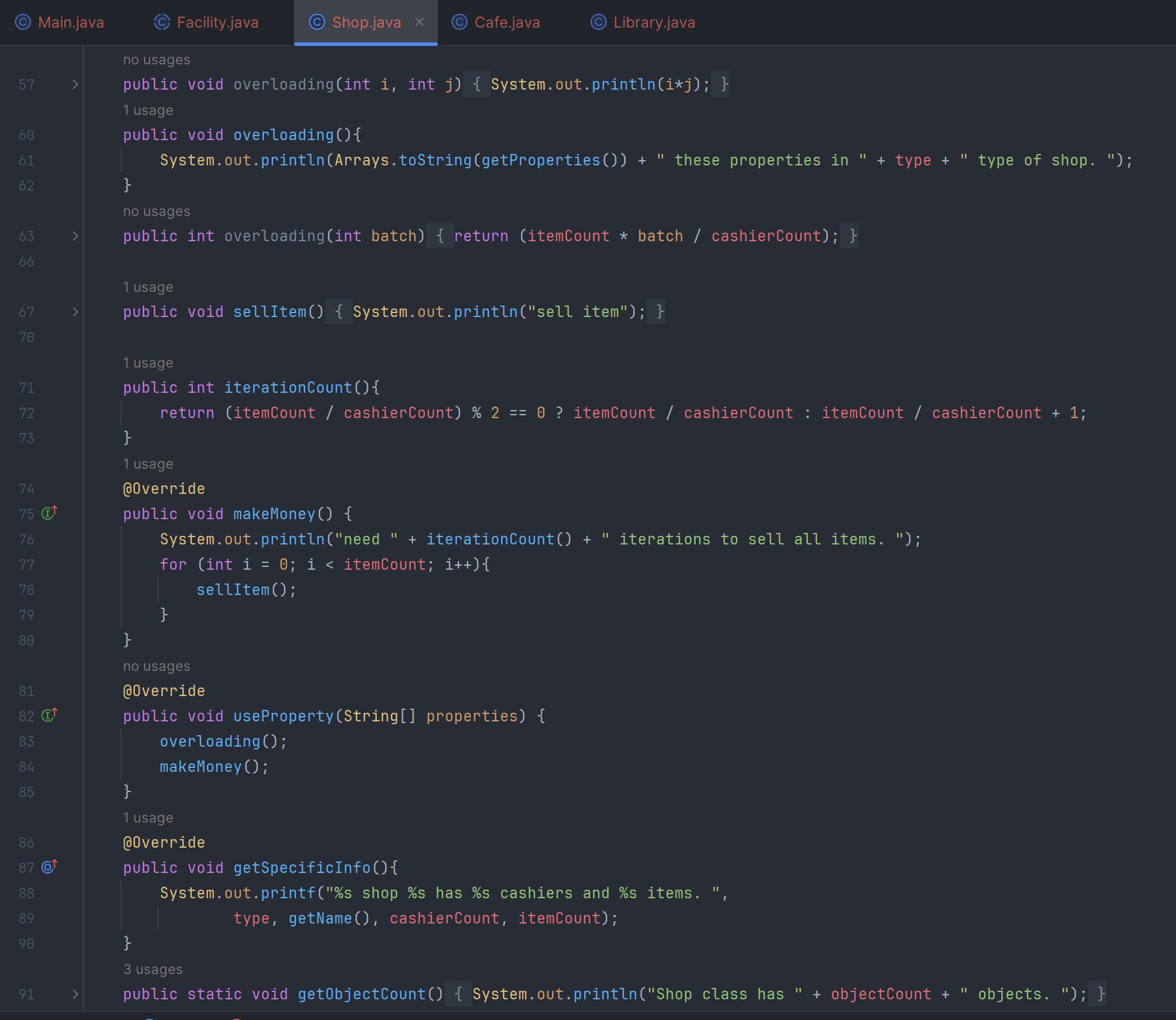




2.в) Shop class



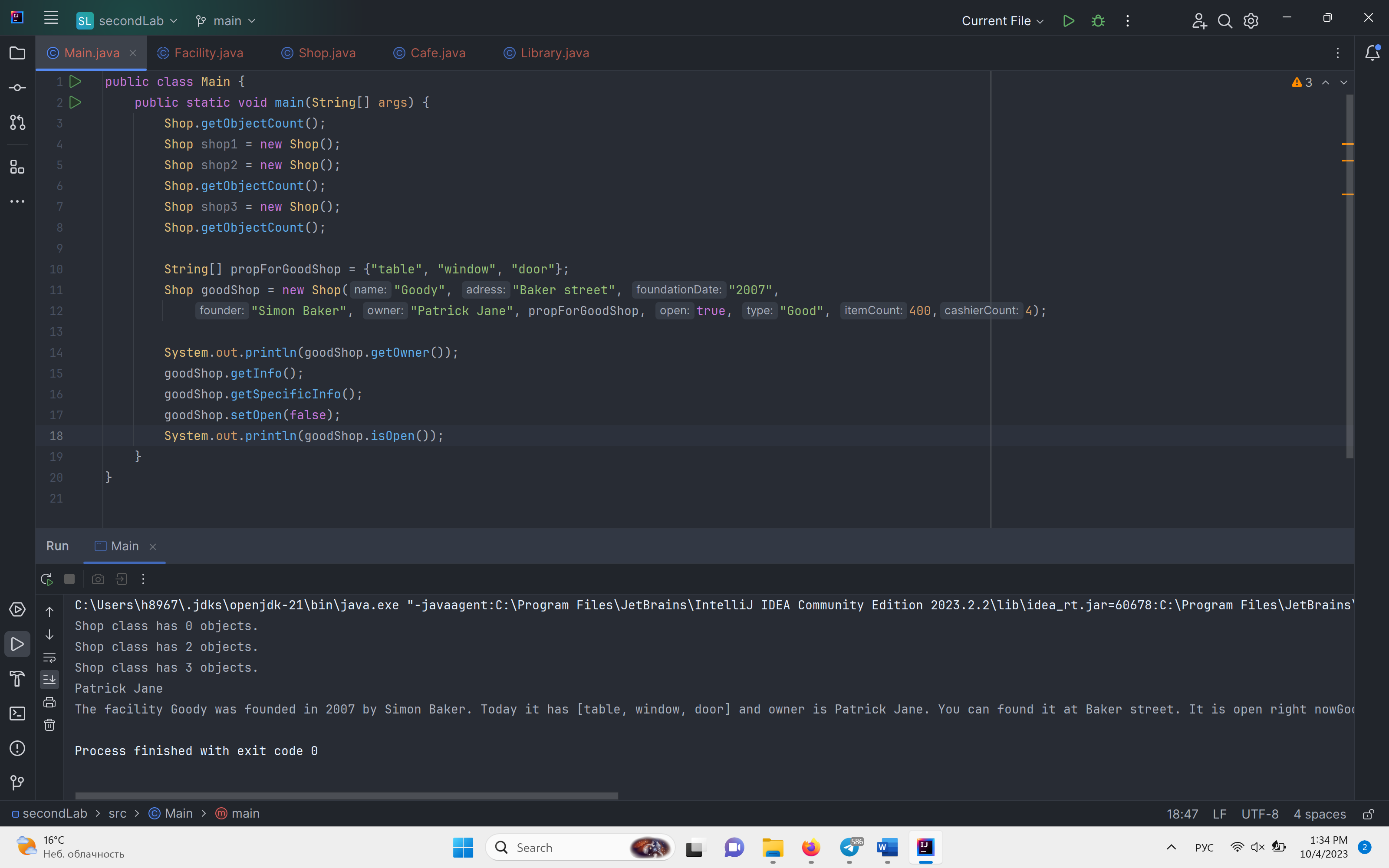




Комментарии:

1. Static поля\методы относятся к самому классу, а не к объекту
2. При создании объекта класса переменная objectCount будет увеличиваться на 1
3. Перегрузка – создание методов с одинаковым названием, но с разными типами переменных или их количеством.

3) Демонстрация работы



Вывод: Проделав данную работу, я познакомился с основами ООП на Java.

Ссылка на репозиторий: https://github.com/Sypoo1/mtuci-java-labs/tree/main/secondLab