```java
public class Shop {
    private final List<AbstractTower> towers = new ArrayList<>();

    public Shop() {
        towers.add(new BCTower());
        towers.add(new MoneyTower());
        towers.add(new BioTower());
    }

    public List<AbstractTower> getTowers() { return Collections.unmodifiableList(towers); }
```

This principle shown is the Advanced SOLID principle of polymorphism. We define a List to contain AbstractTower objects, yet the ones being added are specifically instances of BCTower, MoneyTower, and BioTower. We do this so we can compile all of the different types of towers into one collection, but still allow them to have the variation necessary to be distinct Tower objects via unique methods and variables.