|  |  |
| --- | --- |
| **Single Responsibility Principle**  **(Принцип единственной ответственности)**  *У класса должен быть только один мотив для изменения.*  (лишнее поведение находится в собственном классе) | **Open/Closed Principle**  **(Принцип открытости/закрытости)**  *Расширяйте классы, но не изменяйте их первоначальный код.*  (новые способы доставки можно добавить, не трогая  класс заказов) |
| **Liskov Substitution Principle**  **(Принцип подстановки Лисков)**  *Подклассы должны дополнять, а не замещать поведение базового класса.*  (подкласс расширяет базовый класс новым поведением) | **Interface Segregation Principle**  **(Принцип разделения интерфейса)**  *Клиенты не должны зависеть от методов, которые они*  *не используют.*  (раздутый интерфейс разбит на части) |
| **Dependency Inversion Principle**  **(Принцип инверсии зависимостей)**  *Классы верхних уровней не должны зависеть от классов нижних уровней. Оба должны зависеть от абстракций. Абстракции не должны зависеть от деталей. Детали должны зависеть от абстракций.*  (низкоуровневые классы зависят от  высокоуровневой абстракции) |  |