Принцип единственной ответственности (Single Responsibility Principle, SRP):

* Каждый класс должен иметь только одну причину для изменения.
* Этот принцип утверждает, что класс должен быть ответственен только за одну функцию или задачу. Это уменьшает связность в классе и делает его более гибким и легко поддерживаемым.

Принцип открытости/закрытости (Open/Closed Principle, OCP):

* Система должна быть открыта для расширения, но закрыта для изменения.
* Этот принцип подразумевает, что программное обеспечение должно быть легко расширяемым без изменения существующего кода. Это достигается путем создания абстракций и интерфейсов, которые могут быть расширены новыми классами.

Принцип подстановки Барбары Лисков (Liskov Substitution Principle, LSP):

* Объекты в программе должны быть заменяемыми экземплярами их базовых типов без изменения правильности программы.
* Этот принцип гарантирует, что подклассы могут быть использованы вместо их базовых классов без изменения ожидаемого поведения программы.

Принцип разделения интерфейсов (Interface Segregation Principle, ISP):

* Клиенты не должны зависеть от интерфейсов, которые они не используют.
* Этот принцип утверждает, что интерфейсы должны быть небольшими и специфичными для потребностей клиентов. Это помогает избежать ненужных зависимостей и уменьшает связность.

Принцип инверсии зависимостей (Dependency Inversion Principle, DIP):

* Высокоуровневые модули не должны зависеть от низкоуровневых модулей. Оба типа модулей должны зависеть от абстракций.
* Этот принцип утверждает, что классы должны зависеть от абстракций, а не от конкретных реализаций. Это позволяет создавать гибкие и легко поддерживаемые системы, которые легко адаптировать к изменениям.