PRINCIPIOS SOLID.

1. **Principio de responsabilidad única**.
   1. Una clase debe tener una única responsabilidad, debe siempre existir el desacoplamiento
2. **Principio de abierto y cerrado**.
   1. Las clases deben estar abiertas para su extensión y cerradas para su modificación.
3. **Principio de sustitución de Liskov**.
   1. Declara que una subclase debe ser sustituible por su superclase, y si al hacer esto, el programa falla, estaremos violando este principio.
4. **Principio de segregación de la interfaz.**
   1. Si una clase implementa una interfaz y se ve obligada a sobreecribir método que no va a usar entonces se rompe este principio. En estos casos deberían existir interfaces más concretas para evitar estos escenarios.

**COHESION Y ACOPLAMIENTO**

**Cohesión:** Grado en que los elementos de un módulo están relacionados entre sí, siempre nos interesa una cohesión muy alta.

**Acoplamiento**: Grado en el que dos módulos de software están relacionados entre si, un buen diseño de software tiene pocos modulo acoplados, si se modifica un módulo, debe afectar lo menos posible a los demás.

**CREATIONAL PATTERNS.**

* **ABSTRACT FACTORY PATTERN.**
  + Este patrón está orientado, para cuando se necesita crear objetos de una misma familia sin especificar su clase concreta.

