Equipo 2

Open /Closed Principle

Andrés Chambo, Andy Yara,
Julian Gutierrez, Juan Torrejano,
William Almario, Martin Mendez y
Danay Pereira

Introducción al OCP

- Hace parte de los 5 principios SOLID.
- Propuesto por Bertrand Meyer (1988).
- Popularizado por Robert C. Martin (Uncle Bob).
- Considerado uno de los principios más importantes en arquitectura de software.

Definición

“Las entidades de software deben estar abiertas a la extensión, pero cerradas a la modificación.”

- Clases
- Módulos
- Funciones
- Componentes

¿Qué significa realmente?

Abierto a la extensión

Podemos agregar nuevo comportamiento.

Cerrado a la modificación

No debemos cambiar el código que ya funciona.

Idea clave

El sistema debe poder crecer sin alterar constantemente lo que ya estaba funcionando.

Importancia

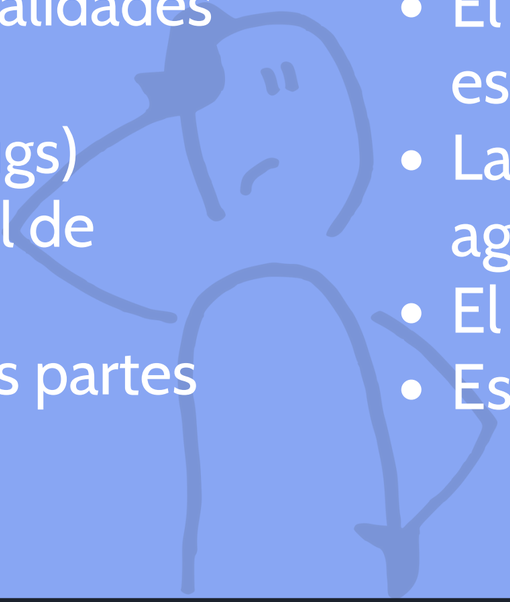
Cuando un sistema crece, aparecen nuevos requisitos:

- Tipo de usuario
- Método de pago
- Idioma
- Tipo de producto

¿Qué pasa si...?

No Aplicamos OCP

- Se pueden dañar funcionalidades existentes
- Aumentan los errores (bugs)
- El sistema se vuelve difícil de mantener
- Los cambios afectan otras partes del código



Aplicamos OCP

- El código existente permanece estable
- Las nuevas funcionalidades se agregan sin romper lo anterior
- El sistema es más flexible
- Es más fácil probar y mantener

¿Cómo se logra?

Se apoya en principios de la Programación Orientada a Objetos.

Herencia

Permite crear nuevas clases sin modificar la clase original.

Polimorfismo

Permite tratar distintos objetos como si fueran del mismo tipo.

Interfaces / Clases abstractas

Definen una estructura común que otras clases deben cumplir.

Ventajas

- Se reduzcan errores al agregar nuevas funciones
- El código sea más fácil de probar
- El sistema sea más mantenible
- El software sea más flexible
- El crecimiento del sistema sea más ordenado