Composición

La **composición** es una **agregación** fuerte entre una **clase compuesta** y una **clase componente** en la que instancias de la clase componente no suelen existir independiente de instancias de la clase compuesta

Delegación

La **delegación** es un mecanismo en programación por el cual cuando un objeto recibe un mensaje para realizar una operación, **no la realiza él mismo**, sino que **la encarga** a otro objeto

Este otro objeto suele ser un objeto componente

Composición y reutilización

La **composición y delegación** es una forma de **reutilización** de código pues permite crear nuevas clases a partir de clases existentes

Composición vs. herencia

La composición y delegación es una **alternativa** a la herencia

En el contexto de la reutilización toda implementación que use herencia se puede cambiar por una equivalente que use composición y delegación

Composición vs. herencia

Caja negra	Caja blanca
Dinámica	Estática
Ejecución	Compilación
Por código	Declarativa
Más código	Menos código
Reuso selectivo	Reuso todo o nada
1+ clases	1 clase (simple)
Tipos s/relación	Impone subtipo

Cohesión

La **cohesión** es la forma y el grado en el que las responsabilidades de una clase o de las clases contenidas en un paquete **están relacionas** unas con otras

Cuando la **cohesión** es alta es mejor

Acoplamiento

El **acoplamiento** es la forma y el grado de **interdependencia** entre clases y entre paquetes

Cuando el **acoplamiento** es bajo es mejor

Modularidad

La **modularidad** es una propiedad de las clases y paquetes cuando son altamente cohesivos y están poco acoplados