**Herencia**

La herencia es uno de los cuatro pilares de la programación orientada a objetos. Permite definir una **clase derivada** que reutiliza, amplía o modifica el comportamiento de una **clase base**. Permite crear nuevas clases a partir de otras ya existentes.

Las clases derivadas son versiones especializadas de la clase base -> **LA CLASE DERIVADA ES UN TIPO DE LA CLASE BASE.**

C# y .NET sólo admiten **herencia simple o única**, es decir, sólo se puede heredar de una clase. Si bien sólo se puede heredar de una clase, **todas las clases heredan implícitamente de object**. Sin embargo, la herencia es **transitiva**.

**No se pueden heredar** constructores ni finalizadores. Se heredan todos los demás miembros pero los **miembros private** no son visibles en las clases derivadas. Los **miembros protected** son visibles para todas las clases derivadas directamente o indirectamente de la clase base.

Se puede **modificar o invalidar la implementación de los miembros heredados** de la clase base si están marcados como virtual y se debe hacerlo si están marcados como abstract (Polimorfismo).

Si se declara una clase como **sealed** no se podrá heredar de ella.

Normalmente cuando tenemos sobrecarga de constructores los constructores con menos parámetros proporcionan valores predeterminados al llamar al constructor con el mayor número de parámetros. El constructor con más parámetros usa la **palabra clave BASE** para pasar de forma explícita los parámetros al constructor de la clase base. Si no se realiza una llamada explícita a un constructor de clase base, el compilador de C# proporciona automáticamente una llamada al constructor sin parámetros o predeterminado de la clase base.

Una clase derivada no puede ser más accesible que su clase base.