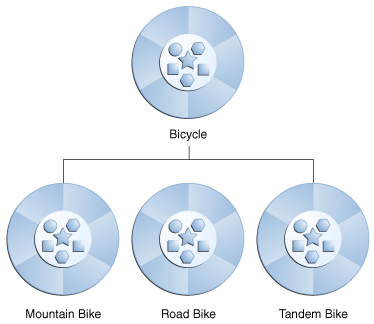
HERENCIA:

**Herencia es cuando un objeto o clase se basa en otro objeto o clase, usando la misma implementación o comportamiento**. Esto es un mecanismo para la reutilización de código para permitirnos extensiones independientes del software original mediante clases públicas e interfaces.



Ventajas:

* La herencia es atractiva porque permite reutilizar código de una manera rápida, sencilla y evidente.
* La herencia es poderosa porque me permite jugar con el polimorfismo con facilidad.
* La herencia me permite sobrescribir métodos, así que si algo no es exactamente como espero siempre puedo cambiarlo si fuera necesario, por lo que es versátil.
* La herencia usando algo parecido al patrón de diseño [Template Method](https://es.wikipedia.org/wiki/Template_Method_(patr%C3%B3n_de_dise%C3%B1o)) me permite crear clases abstractas que hacen que al heredar de ellas me quede un desarrollo guiado por la plantilla creada, ahorrando así tiempo y quedando un desarrollo guiado muy sencillo.

COMPOSICION:

**Composición quiere decir que tenemos una instancia de una clase que contiene instancias de otras clases que implementan las funciones deseadas**.

Es decir, estamos delegando las tareas que nos mandan a hacer a aquella pieza de código que sabe hacerlas. El código que ejecuta esa tarea concreta está sólo en esa pieza y todos delegan el ella para ejecutar dicha tarea. Por lo tanto estamos reutilizando código de nuevo. [En la composición, delegamos responsabilidades en colaboradores designados para ello](https://twitter.com/intent/tweet?url=https://devexperto.com/herencia-vs-composicion/&text=En%20la%20composici%C3%B3n%2C%20delegamos%20responsabilidades%20en%20colaboradores%20designados%20para%20ello&via=maros87&related=maros87" \t "_blank)

