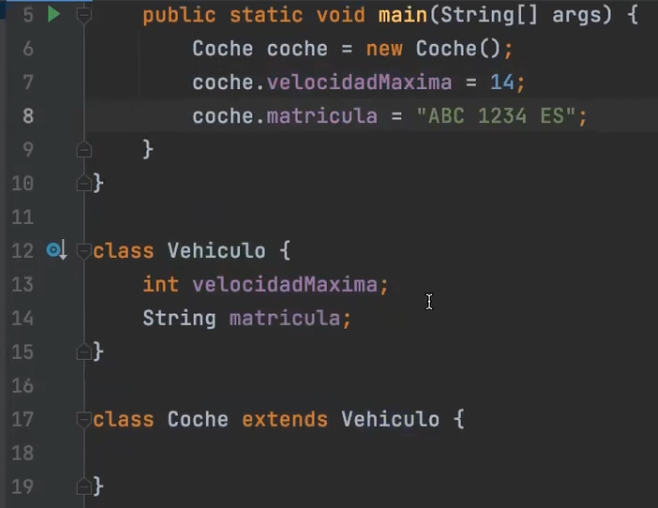
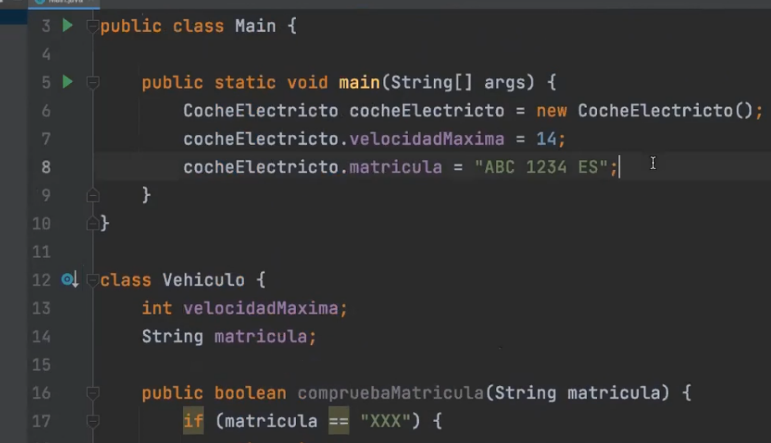
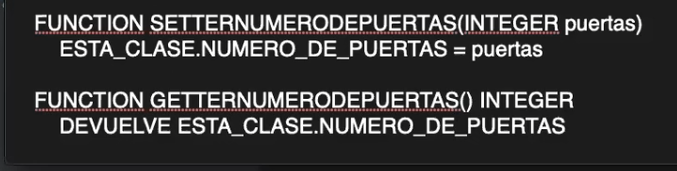
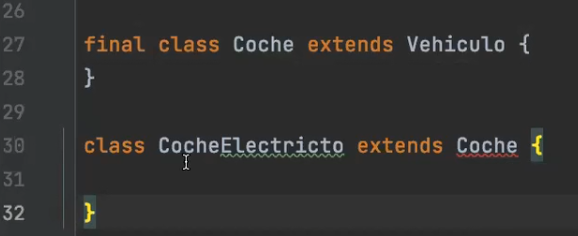
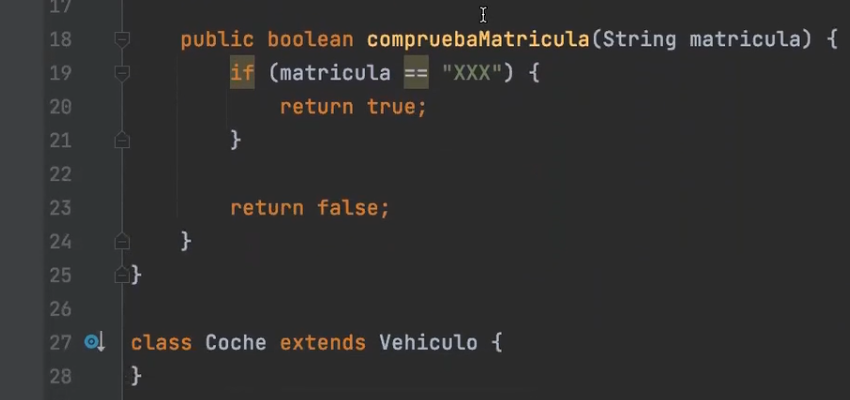
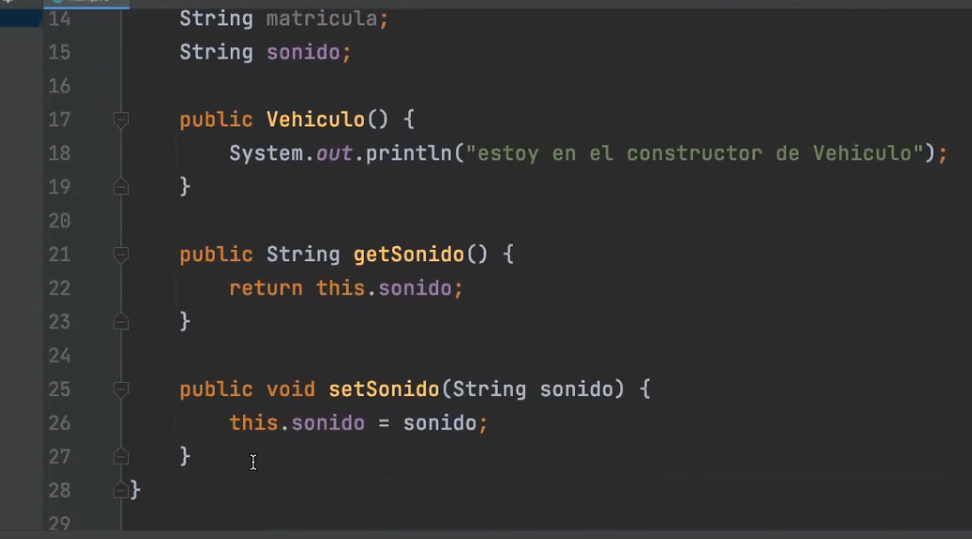
**Herencia** **simple y multinivel** cada clase derivada hereda funciones, métodos de su padre y añade las suyas propias.

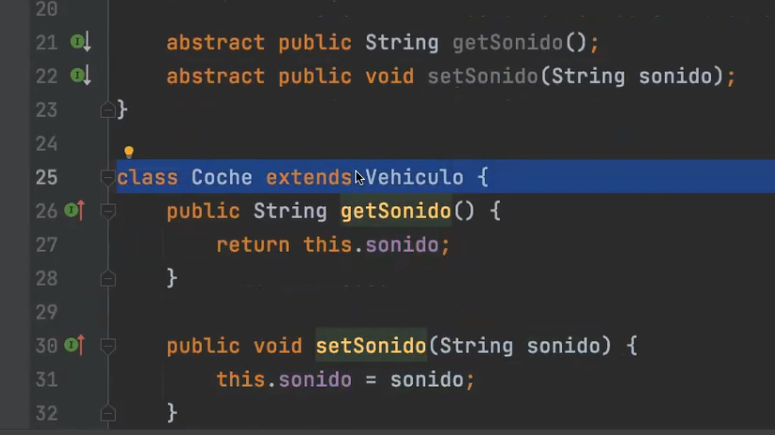


No heredar mas de 3 veces

Se añade “final” para que esa clase no herede hacia abajo.

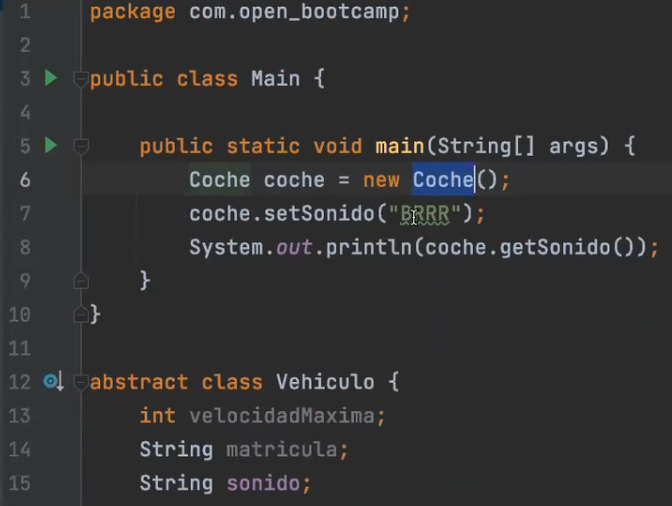
“coche” da error ahora dice que no puede heredar esa clase al ser final.

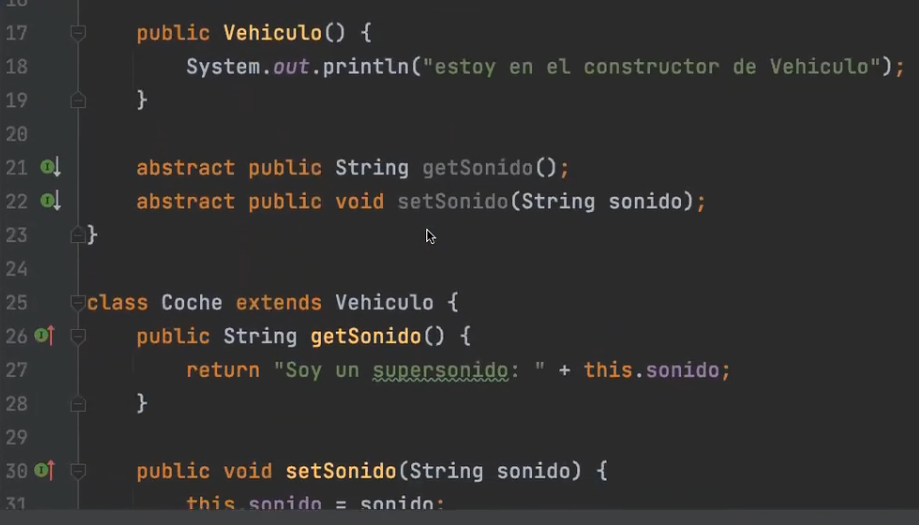


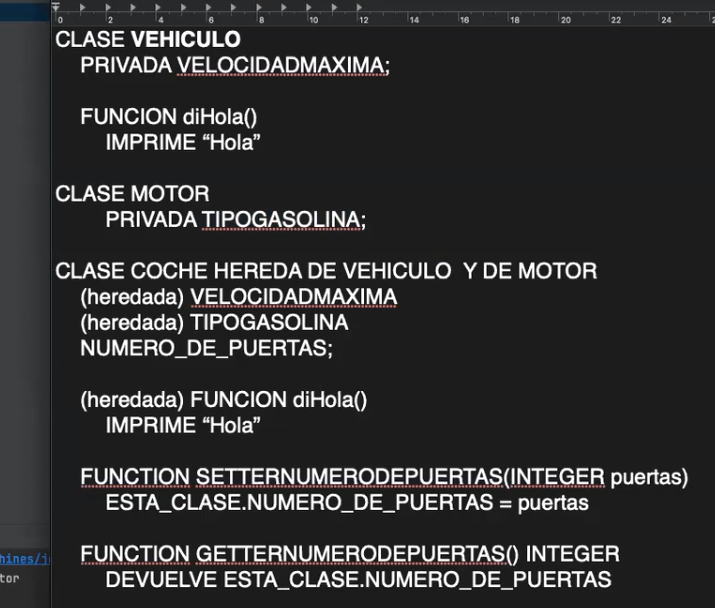
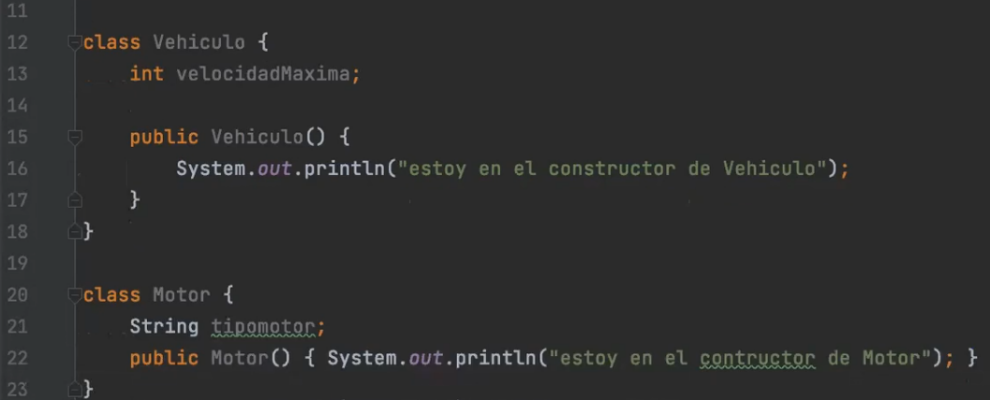
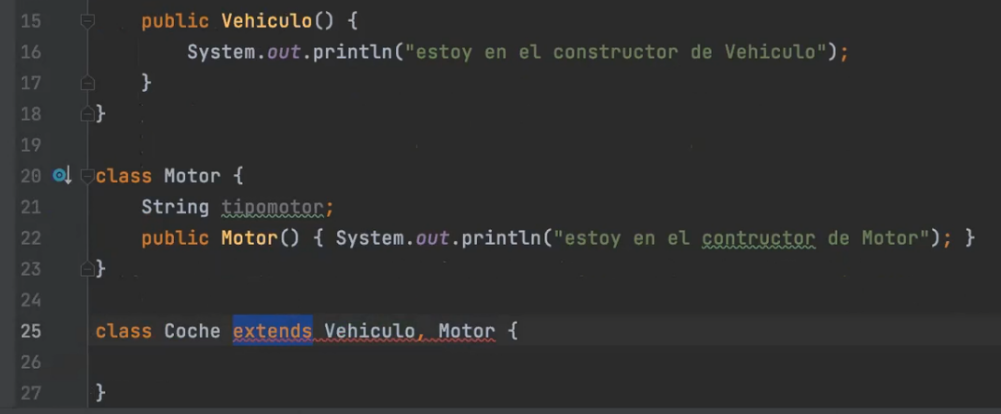


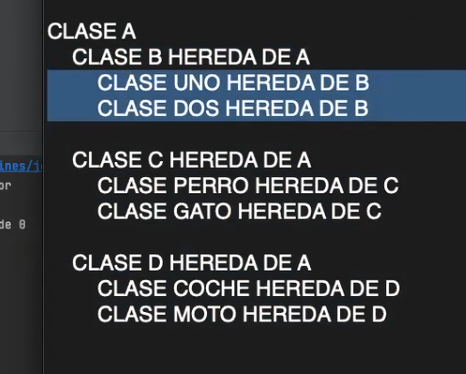
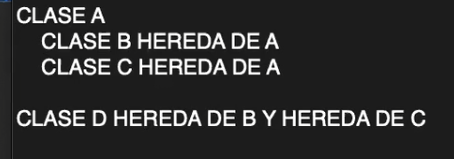
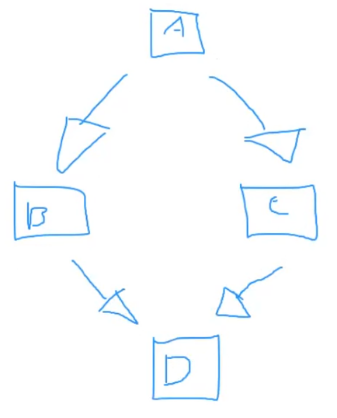
Una clase abstracta Nos dice que tenemos que hacer **algo** pero no nos dice **cómo** debemos hacerlo mientras cumplamos sus normas.

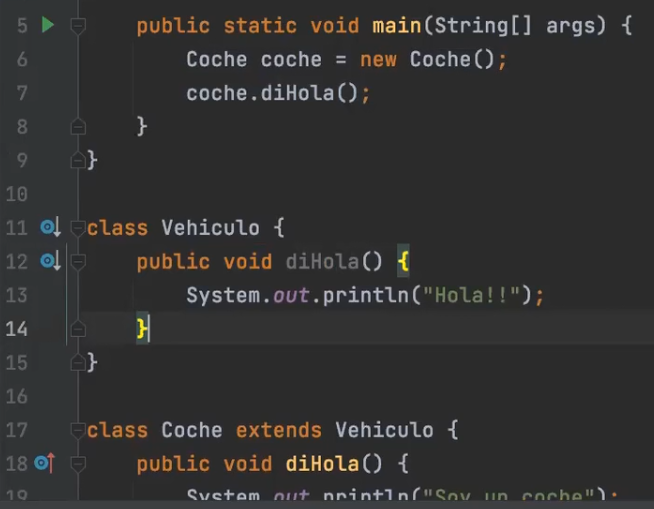
La clase derivada implementa la función de **cómo hacerlo**

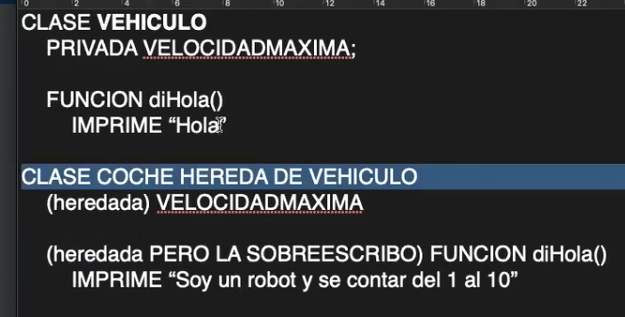


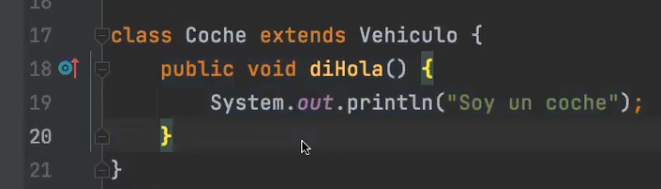


**Herencia múltiple:** una clase hereda de dos o más clases padres. (Puede que ya no esté funcionando porque ha habido un cambio de lenguaje y lo han prohibido)

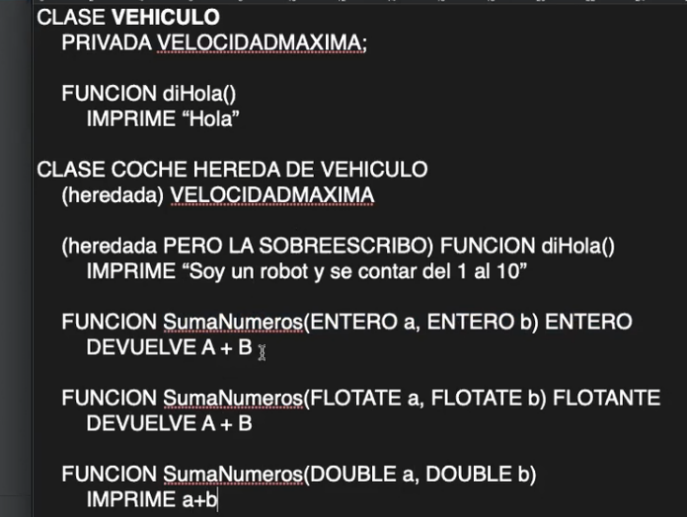
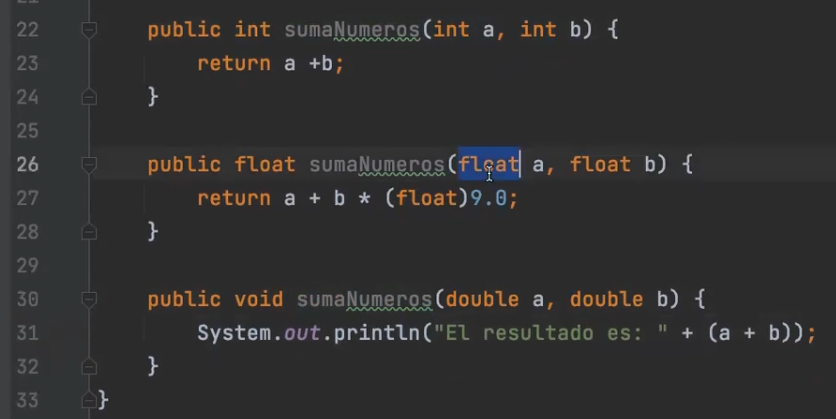
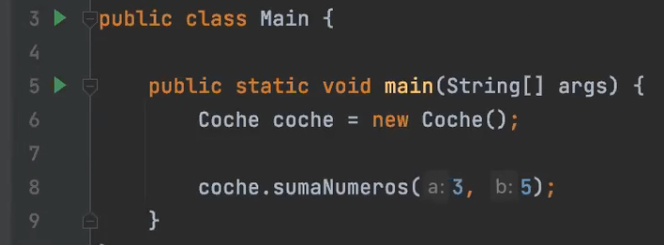
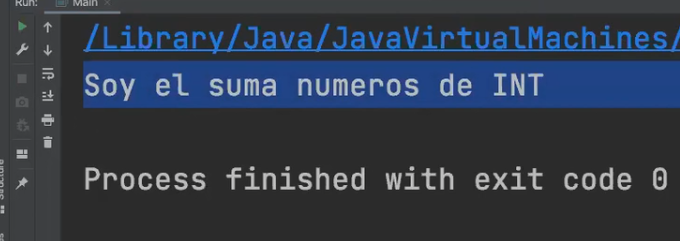
Herencia jerárquica  
 Herencia híbrida

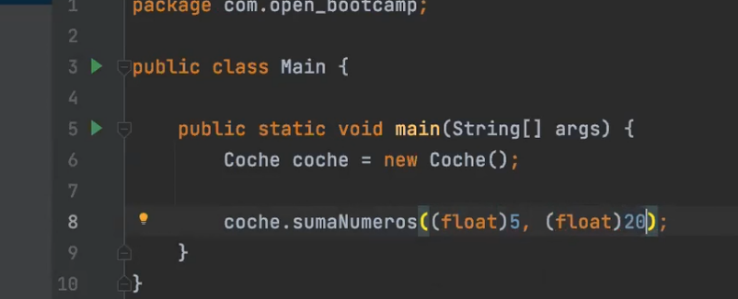
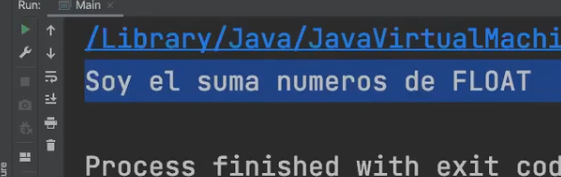
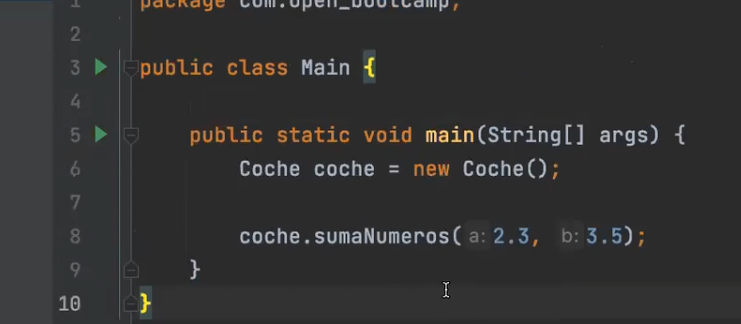
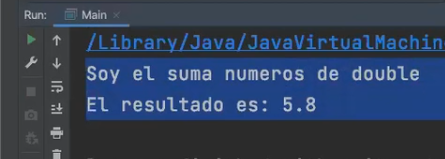
Poliformismo

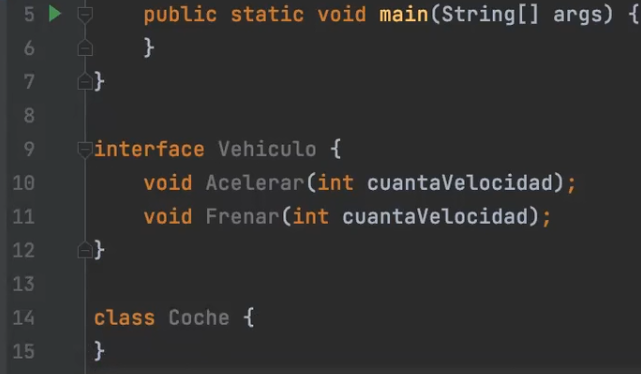


Creo una función en el padre y en el hijo, en este caso escribe un texto, pero el texto de ambos es distinto.

Si quito la función del hijo, el código sigue funcionando porque no he quitado la función del padre.   
Se ejecuta el método del hijo porque hay método, si no hubiera método se ejecuta el método de su padre.





INTERFACES



Sirven para unificar métodos

