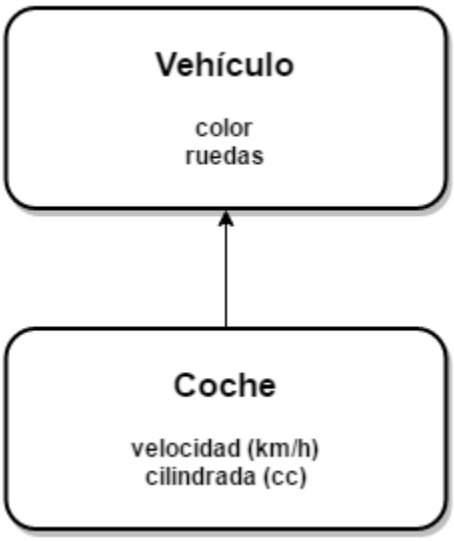
# Practica capítulo 9: Herencia

## Teoría previa

En este ejercicio vas a trabajar el concepto de herencia. Hasta ahora sabemos que una clase heredada puede fácilmente extender algunas funcionalidades, simplemente añadiendo nuevos atributos y métodos, o sobrescribiendo los ya existentes. Como en el siguiente ejemplo:




```
class Vehiculo():

    def __init__(self, color, ruedas):
        self.color = color
        self.ruedas = ruedas

    def __str__(self):
        return "Color {}, {} ruedas".format( self.color, self.ruedas )

class Coche(Vehiculo):

    def __init__(self, color, ruedas, velocidad, cilindrada):
        self.color = color
        self.ruedas = ruedas
        self.velocidad = velocidad
        self.cilindrada = cilindrada

    def __str__(self):
        return "color {}, {} km/h, {} ruedas, {} cc".format( self.color, self


coche = Coche("azul", 150, 4, 1200)
print(coche)
```

El inconveniente más evidente de ir sobrescribiendo es que tenemos que volver a escribir el código de la superclase y luego el específico de la subclase.

Para evitarnos escribir código innecesario, podemos utilizar un truco que consiste en llamar el método de la superclase y luego simplemente escribir el código de la clase:

```
class Vehiculo():

    def __init__(self, color, ruedas):
        self.color = color
        self.ruedas = ruedas

    def __str__(self):
        return "Color {}, {} ruedas".format( self.color, self.ruedas )


class Coche(Vehiculo):

    def __init__(self, color, ruedas, velocidad, cilindrada):
        Vehiculo.__init__(self, color, ruedas)
        self.velocidad = velocidad
        self.cilindrada = cilindrada

    def __str__(self):
        return Vehiculo.__str__(self) + ", {} km/h, {} cc".format(self.velocida


c = Coche("azul", 4, 150, 1200)
print(c)
```

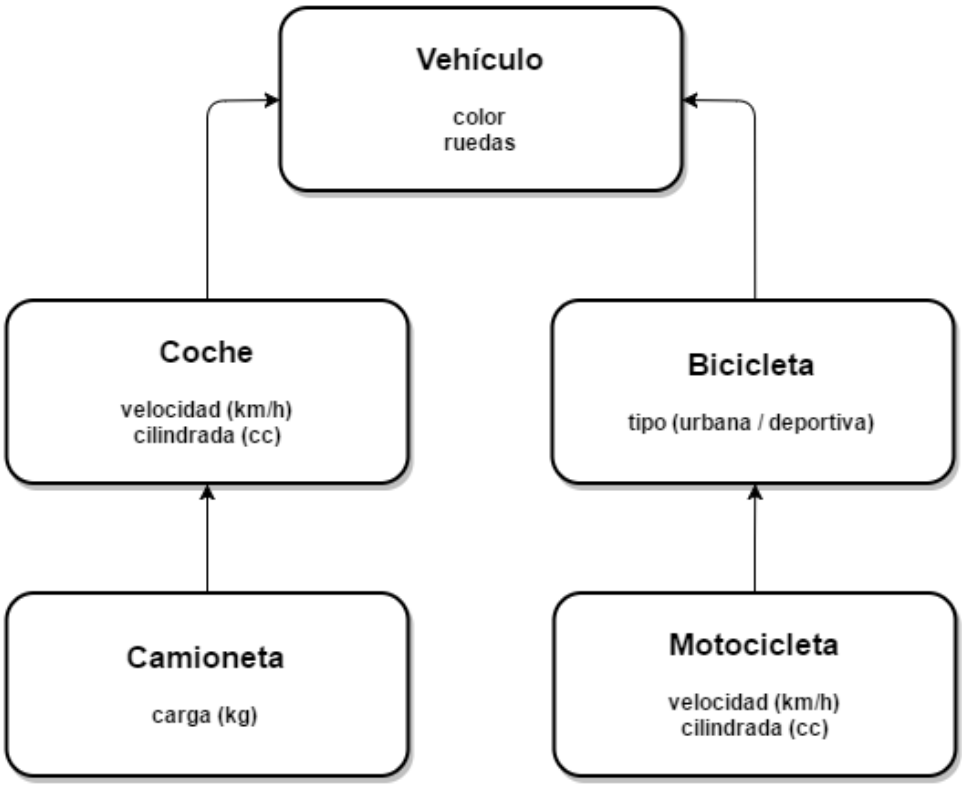
Como tener que determinar constantemente la superclase puede ser fastidioso, Python nos permite utilizar un acceso directo mucho más cómodo llamado *super()*.

Hacerlo de esta forma además nos permite llamar cómodamente los métodos o atributos de la superclase sin necesidad de especificar el self, pero ojo, **sólo se aconseja utilizarlo cuando tenemos una única superclase**:

```
class Vehiculo():

    def __init__(self, color, ruedas):
        self.color = color
        self.ruedas = ruedas

    def __str__(self):
        return "color {}, {} ruedas".format( self.color, self.ruedas )


class Coche(Vehiculo):

    def __init__(self, color, ruedas, velocidad, cilindrada):
        Vehiculo.__init__(self, color, ruedas)
        self.velocidad = velocidad
        self.cilindrada = cilindrada

    def __str__(self):
        return Vehiculo.__str__(self) + ", {} km/h, {} cc".format(self.velocid


c = Coche("azul", 4, 150, 1200)
print(c)
```

## Enunciado

Utilizando esta nueva técnica extiende la clase Vehículo y realiza la siguiente implementación:




- Crea al menos un objeto de cada subclase y añádelos a una lista llamada vehículos.
- Realiza una función llamada **catalogar()** que reciba la lista de vehículos y los recorra mostrando el nombre de su clase y sus atributos.
- Modifica la función **catalogar()** para que reciba un argumento optativo **ruedas**, haciendo que muestre únicamente los que su número de ruedas concuerde con el valor del argumento. También debe mostrar un mensaje **"Se han encontrado {} vehículos con {} ruedas:"** únicamente si se envía el argumento ruedas. Ponla a prueba con 0, 2 y 4 ruedas como valor.

```
PS C:\Users\USUARIO\Documents\Python V> & C:/Users/USUARIO/AppData/Local/Prog
9.py"
Coche color azul, 4 ruedas, 150 km/h, 1200 cc
Camioneta color blanco, 4 ruedas, 100 km/h, 1300 cc, 1500 kg de carga
Bicicleta color verde, 2 ruedas, urbana
Motocicleta color negro, 2 ruedas, deportiva, 180 km/h, 900 cc

Se han encontrado 0 vehículos con 0 ruedas:

Se han encontrado 2 vehículos con 2 ruedas:
Bicicleta color verde, 2 ruedas, urbana
Motocicleta color negro, 2 ruedas, deportiva, 180 km/h, 900 cc

Se han encontrado 2 vehículos con 4 ruedas:
Coche color azul, 4 ruedas, 150 km/h, 1200 cc
Camioneta color blanco, 4 ruedas, 100 km/h, 1300 cc, 1500 kg de carga
PS C:\Users\USUARIO\Documents\Python V>
```