**PROGRAMACIÓN ORIENTADA A OBJETOS**

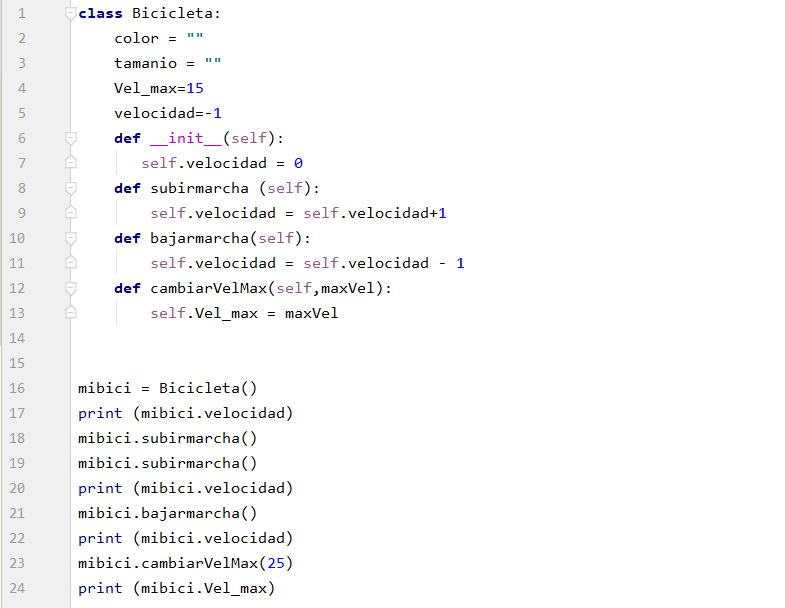
* **En Python se define una clase con la palabra reservada class seguido del nombre la clase.**

**class Nombre**

**Propiedades y métodos de la clase**

* **El constructor en Python se define con el método \_\_init\_\_**
* **Las propiedades o atributos de una clase en Python serán las variables.**
* **Las acciones o comportamiento conocidos en POO como métodos, en Python serán las funciones.**
* **Todos los métodos tienen de primer parámetro uno que se llama self. Será una referencia a la instancia del objeto.**

Vamos a crear de ejemplo una clase Bicicleta con cuatro propiedades (variables), constructor y tres métodos.



RESULTADO



**Clase Bicicleta:**

* **Incluimos cuatro propiedades para el color, tamaño, velocidad máxima y velocidad actual.**
* **Constructor donde inicializamos la velocidad a cero.**
* **Tres métodos para poder subir de velocidad, bajar de velocidad y cambiar la velocidad máxima.**

**class** Bicicleta:  
 color = **""** tamanio = **""** Vel\_max=15  
 velocidad=-1  
 **def** \_\_init\_\_(self):  
 self.velocidad = 0  
 **def** subirmarcha (self):  
 self.velocidad = self.velocidad+1  
 **def** bajarmarcha(self):  
 self.velocidad = self.velocidad - 1  
 **def** cambiarVelMax(self,maxVel):  
 self.Vel\_max = maxVel

**Creamos una instancia de la clase Bicicleta y accedemos a las propiedades (variables) y métodos (funciones)**

mibici = Bicicleta()  
print (mibici.velocidad)  
mibici.subirmarcha()  
mibici.subirmarcha()  
print (mibici.velocidad)  
mibici.bajarmarcha()  
print (mibici.velocidad)  
mibici.cambiarVelMax(25)  
print (mibici.Vel\_max)

* **MÉTODO \_\_str\_\_(self)**

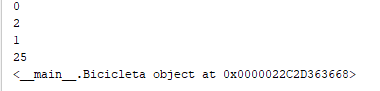
Este método también debería devolver una representación en forma de cadena del objeto. Se le llama cuando imprimimos una instancia del objeto con print.

En el ejemplo anterior vamos a imprimir el objeto mibici y veremos el resultado que obtenemos:

**class** Bicicleta:  
 color = **""** tamanio = **""** Vel\_max=15

velocidad=-1  
 **def** \_\_init\_\_(self):  
 self.velocidad = 0  
 **def** subirmarcha (self):  
 self.velocidad = self.velocidad+1  
 **def** bajarmarcha(self):  
 self.velocidad = self.velocidad - 1  
 **def** cambiarVelMax(self,maxVel):  
 self.Vel\_max = maxVel  
mibici = Bicicleta()  
print (mibici.velocidad)  
mibici.subirmarcha()  
mibici.subirmarcha()  
print (mibici.velocidad)  
mibici.bajarmarcha()  
print (mibici.velocidad)  
mibici.cambiarVelMax(25)  
print (mibici.Vel\_max)  
**print (mibici)**

**RESULTADO**



Ahora vamos a incluir el método \_\_str\_\_ para que devuelva la velocicad y velocidad máxima al imprimir el objeto mibici

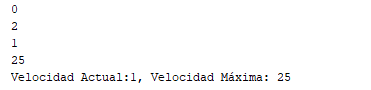
**class** Bicicleta:  
 color = **""**  
tamanio = **""**  
Vel\_max=15

velocidad=-1  
 **def** \_\_init\_\_(self):

self.velocidad = 0  
 **def** subirmarcha (self):  
 self.velocidad = self.velocidad+1  
 **def** bajarmarcha(self):  
 self.velocidad = self.velocidad - 1  
 **def** cambiarVelMax(self,maxVel):  
 self.Vel\_max = maxVel  
 **def** \_\_str\_\_(self):  
 **return "Velocidad Actual:"** + str(self.velocidad) + **", Velocidad Máxima: "** + str(self.Vel\_max)

mibici = Bicicleta()  
print (mibici.velocidad)  
mibici.subirmarcha()  
mibici.subirmarcha()  
print (mibici.velocidad)  
mibici.bajarmarcha()  
print (mibici.velocidad)  
mibici.cambiarVelMax(25)  
print (mibici.Vel\_max)  
print (mibici)

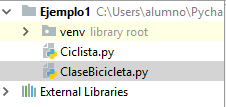
**RESULTADO**



* **SEPARANDO LÓGICA DE NEGOCIO Y CLIENTE EN DIFERENTES FICHEROS**

Podemos separar la lógica de funcionamiento de la bicicleta y el cliente que hace uso de ella en dos ficheros diferentes.

1. Creamos dos ficheros \*.py



1. Incluimos el código de funcionamiento de la bicicleta en el fichero ClaseBicicleta.py

**class** Bicicleta:  
 color = **""** tamanio = **""** Vel\_max=15

velocidad=-1  
 **def** \_\_init\_\_(self):  
 self.velocidad = 0  
 **def** subirmarcha (self):  
 self.velocidad = self.velocidad+1  
 **def** bajarmarcha(self):  
 self.velocidad = self.velocidad - 1  
 **def** cambiarVelMax(self,maxVel):  
 self.Vel\_max = maxVel  
 **def** \_\_str\_\_(self):  
 **return "Velocidad Actual:"** + str(self.velocidad) + **", Velocidad Máxima: "** + str(self.Vel\_max)

1. La clase Ciclista.py tendrá como primera línea de código la importación de la clase Bicicleta que está dentro del fichero ClaseBicicleta.

**from** ClaseBicicleta **import** Bicicleta

mibici = Bicicleta()  
print (mibici.velocidad)  
mibici.subirmarcha()  
mibici.subirmarcha()  
print (mibici.velocidad)  
mibici.bajarmarcha()  
print (mibici.velocidad)  
mibici.cambiarVelMax(25)  
print (mibici.Vel\_max)  
print (mibici)