Programación Orientada a Objetos (OOP)

La programación orientada a objetos expresa un programa como un conjunto de objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar.

Objeto: Un objeto representa a una entidad, (como por ejemplo un coche),

- que tiene una identidad concreta (el coche con matrícula 8888CFG)
- que tiene un estado determinado (color=rojo, cilindrada=1905cc, velocidad_actual=50km/h, marcha 2ª, ...
- un comportamiento (cómo cambia de estado, por ejemplo: arranca, cambia de velocidad… y cómo se relaciona con otros objetos (como por ejemplo con el objeto llave)

En realidad los objetos no se definen directamente, sino que lo que se define es la clase.

Clase: Una clase es una PLANTILLA o MOLDE a partir del cual posteriormente crearemos los objetos

Cuando definamos una clase, tenemos que definir dos cosas:

- Los ATRIBUTOS: Los datos que dicha clase puede manipular o contener,

  Por ejemplo: la clase coche tendrá los atributos matricula, color, cilindrada, velocidad actual, marcha, encendido/apagado, ….)

- Los MÉTODOS: La forma de acceder a esos datos y manipularlos

  Por ejemplo: la clase coche tendrá los métodos asignar_matricula, acelerar, encender, apagar, cambiar_de_marcha, leer_velocidad, …, que en realidad lo que hacen es modificar los valores de los atributos.

Una clase en sí es algo abstracto, que sólo dice las características que tendrán los objetos de esa clase (*en nuestro ejemplo la **clase coche** dice que los objetos que pertenezcan a esa clase tendrán una matrícula, un color, etc, pero no dice qué matrícula será, ni que color, ni que velocidad lleva en este momento, etc*).

Un objeto es la concreción (una instancia) de esa clase. Por ejemplo:

*- el **objeto coche_1** es un objeto de la clase coche, cuya matrícula será 2222ABC, color verde, motor en marcha, velocidad actual 25km/h;*
*el **objeto coche_2** es otro objeto de la clase coche, cuya matrícula es 3333GFD, color rosa, motor apagado, velocidad actual 0 km/h, …*

Por tanto, a partir de una clase, podemos crear un montón de objetos de esa clase

Características de un Lenguaje orientado a objetos:

1.- La **herencia** es la cualidad de crear clases que estén basadas en otras clases.
- La nueva clase heredará todos los atributos y métodos de la clase de la que está derivada,
- Además en la nueva clase heredada se podrá modificar el comportamiento de los procedimientos que ha heredado,
-También en la nueva clase heredada se podrán añadir atributos y métodos nuevos.

Resumiendo: Gracias a la herencia podemos ampliar cualquier clase existente, además de aprovecharnos de todo lo que esa clase haga...

***El ejemplo del gato**: Supongamos que tenemos una clase Gato que está derivada de la clase Animal.*
La clase Gato hereda de la clase Animal todas las características comunes a los animales, además de añadirle algunas características particulares a su condición felina.
Podemos decir que un Gato es un Animal, lo mismo que un Perro es un Animal, ambos están derivados (han heredado) de la clase Animal, pero cada uno de ellos es diferente, aunque en el fondo los dos son animales.
Esto es herencia: usar una clase base (Animal) y poder ampliarla sin perder nada de lo heredado, pudiendo ampliar la clase de la que se ha derivado (o heredado).

2.- La **encapsulación** nos oculta el código y las interioridades de la clase, para que sólo veamos lo que tenemos que ver.
Si tomamos el ejemplo de la clase Gato, sabemos que araña, come, se mueve, etc., pero el cómo lo hace no es algo que deba preocuparnos,

3.- El **polimorfismo** se refiere a la posibilidad de definir clases diferentes, pero con métodos o propiedades denominados de forma idéntica, que pueden utilizarse de manera intercambiable
Siguiendo con el ejemplo de los animales, si el Gato y el Perro pueden morder podríamos tener un "animal" que muerda sin importarnos que sea el Gato o el Perro, simplemente podríamos usar el método Morder ya que ambos animales tienen esa característica "animal mordedor".