Programación orientada a objetos.

Recordemos el vocabulario común de la POO:

* Clase
* Objeto
* Instancia de clase
* Modularización
* Encapsulamiento
* Herencia
* Polimorfismo

Para ver ejemplos seguiremos el objeto ***Consola de videojuegos***.

Bueno pues una ***Clase*** es una especie de modelo/plantilla donde redactaremos las características comunes de un grupo de objetos.

En este caso con el objeto dicho anteriormente la ***Clase*** podrían ser el modelo cada consola tendrá un diferente modelo asi como tambien una marca, un SO ya que todas las consolas trabajan con uno, almacenamiento, tarjeta madre, etc

Entonces lo primero que deberíamos hacer es una ***Clase*** que define las características comunes de las consolas que fabricaremos.

Bueno ahora para ello veremos ahora ***instancia de una clase*** básicamente seria un objeto o ejemplar de una clase, o sea una vez tengamos la Clase (plantilla) a base de esa misma podremos crear dos o más diferentes consolas.

Entonces digamos creamos en base a esa Clase (plantilla) dos consolas, aunque sea de diferentes fabricantes van a seguir esa misma estructura que nosotros definimos, van a tener cosas en común:

Los dos van a tener un almacenamiento minimo de 500gb, van a tener un sistema operativo, tendrán una tarjeta madre, etc. Y luego cada una de estas consolas tendrán sus propias características que las harán únicas.

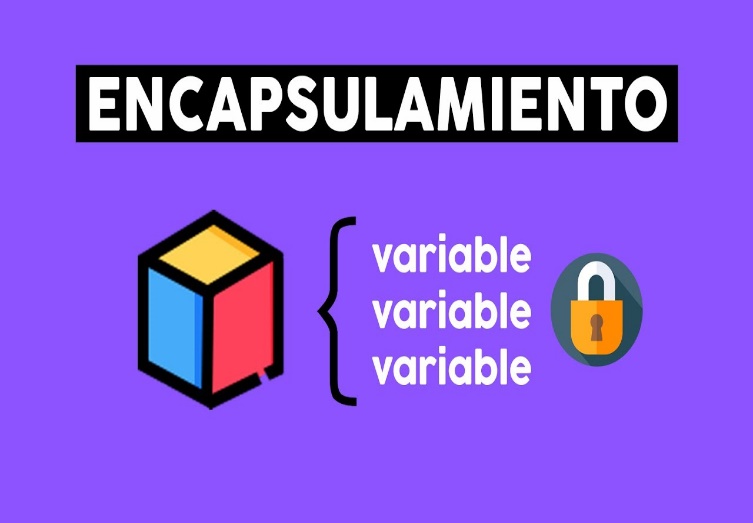
Entonces podemos decir que si creamos un objeto a base de la Clase (plantilla) que definimos antes este objeto es perteneciente a dicha clase o mejor dicho como que ese objeto es una ***instancia de esa clase***.

Ahora veamos que es la ***modularización.***

Hablando en términos de programación la hora de nosotros crear un programa complejo, lo normal es que lo dividamos en diferentes clases. Entonces eso indica la ***modularización*** un programa puede estar definido por una o más clases.

Esto va a indicar que cada módulo de nuestro código va a funcionar y manejarse de manera independiente, entonces podemos agarrar un modulo y llevarlo a trabajar a otro modulo mas dentro de nuestro programa.

Otra ventaja que tiene es que si falla un modulo no afectara a los demas módulos, pero no tendrás acceso a ese modulo que ha fallado.

Ligado a la ***modularización*** da paso a un siguiente concepto que es:

El ***encapsulamiento*** que quiere decir digamos, tenemos en nuestro programa 3 Clases (Modulos), y agarramos en modulo 3, todo el funcionamiento interno de ese módulo 3 corresponde únicamente a ese módulo 3, entonces los otros dos modulos no entienden ni saben el funcionamiento interno del módulo 3.

Aunque de alguna forma cada módulo está conectado entre sí, pero sin saber el funcionamiento del otro.

En programacion para conectar dichas Clases entre si se hace uso de los ***métodos de acceso,*** pero solo tendrán acceso a ciertas características de cada una de las clases.

Digamos desde la clase 1 estamos realizando cierta acción, pero despues le pedimos al programa que realice una accion que venga desde la clase 2 que complemente a la accion principal.

En programación para acceder a propiedades y comportamientos de un objeto usaremos la nomenclatura del punto.

Objeto ***Consola***:

* Accedemos a propiedades:
  + Consola.marca = “Sony”;
  + Consola.Modelo = “PS5”;
  + Consola.color = “Negro”;
  + Consola.peso = 5000;
* Accedemos a comportamientos:
  + Consola.enciende();
  + Consola.apaga();
  + Consola.lectura\_disco();
  + Consola.Acceso\_juego();