**PROGRAMACIÓN ORIENTADA A OBJETOS**

**¿Qué es un objeto?**

**Es un elemento del programa que contiene un conjunto de atributos y métodos que ha sido definido o creado mediante la instanciación de una clase.**

**En el lenguaje de programación orientado a objeto, los objetos interactúan mediante el uso de métodos y las propiedades de un objeto se denominan atributos.**

**Ejemplo de un objeto:**

**En la figura N° 1 se tiene un objeto denominado Toyota y un conjunto de métodos.**

**Los métodos hacen referencia a funciones que posee un objeto en particular.**

**METODOS**

**OBJETO**

**avanzar**

**Toyota**

**detenerse**

**retroceder**

**Figura N° 1**

**En el ejemplo el objeto Toyota posee los métodos de avanzar, detenerse, retroceder.**

**En la figura N° 2 se tiene un objeto denominado Toyota y un conjunto de atributos.**

**Los atributos se refieren a las características propias de un objeto.**

**En el ejemplo el objeto Toyota posee los atributos kilometraje, color y el número de llantas**

**ATRIBUTOS**

**OBJETO**

**color**

**Toyota**

**N° llantas**

**kilometraje**

**++**

**Figura N°2**

**DEFINICIÓN DE UNA CLASE:**

**Una clase es una plantilla o molde a partir del cual se crean objetos siguiendo un modelo predefinido.**

**En el ejemplo anterior la clase seria vehículos**

**CLASE**

**VEHICULOS**

NISSAN

WOLSBAGEN

CHEVROLET

TOYOTA

**Figura N°3**

**INSTANCIAS:**

**El proceso de crear un objeto se denomina INSTANCIAR además los objetos instanciados de una clase compartirán los METODOS (funciones) y ATRIBUTOS (propiedades).**

**Los métodos y atributos se tienen que definir en la creación de una clase.**

**Los objetos instanciados de una clase poseerán dichos métodos y atributos.**

**En la Figura N°3 se tiene la clase VEHICULOS y la instanciación de 4 objetos denominados TOYOTA, CHEVROLET,WOLSBAGEN Y NISSAN.**

**PROPOSITOS DE LA POO:**

**ENCAPSULAMIENTO:**

**Dividir el código fuente en una interfaz publica y una interfaz privada en la implementación de tal interfaz.**

**POLIMORFISMO:**

**Capacidad de sobrecargar operaciones de tal manera que posean comportamiento apropiados dependiendo del contexto.**

**HERENCIA:**

**La capacidad de crear sub-clases que posee más especificaciones que de su ancestro**

**EJEMPLO 1 EN PYTHON**

**DEFINIR LA CLASE VEHICULO Y 3 METODOS**

**class** vehiculo():

**def** avanzar(self):  
 print(**"avanzar"**)

**def** retroceder(self):  
 print(**"retroceder"**)

**def** detenerse(self):  
 print(**"detener carro"**)

*#Instanciar un objeto*toyota=vehiculo()

*#acceder al metodo avanzar*toyota.avanzar()

*#acceder al metodo retroceder*toyota.retroceder()

*#acceder al metodo detenerse*toyota.detenerse()

**CONSOLA DE PYTHON**

avanzar

retroceder

detener carro

**CONSTRUCTOR \_\_init\_\_(self)**

Realiza algún trabajo de inicialización

**EJEMPLO 2 EN PYTHON**

**class** vehiculo():  
 **def** \_\_init\_\_(self,name):

#self.atributo=variable  
 self.nombre=name  
 print(**"MI NOMBRE ES "**+self.nombre)  
 **def** avanzar(self):  
 print(**"avanzar"**)  
 **def** retroceder(self):  
 print(**"retroceder"**)  
 **def** detenerse(self):  
 print(**"detener carro"**)  
*#Instanciar un objeto y pasar argumento con un nombre*toyota=vehiculo(**"VEHICULO TOYOTA"**)

Toyota.avanzar()

**CONSOLA DE PYTHON**

MI NOMBRE ES VEHICULO TOYOTA