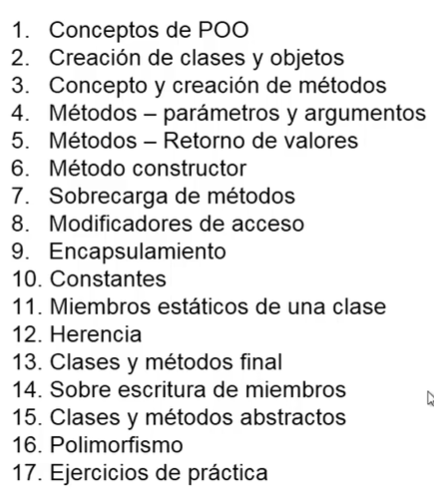
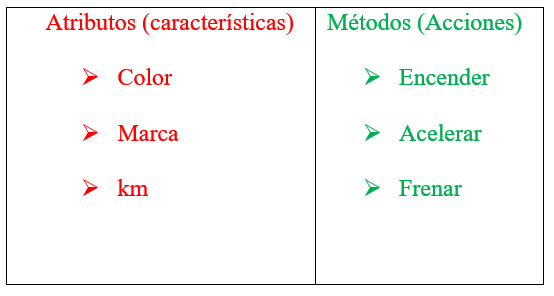
Java Programación orientada a objetos

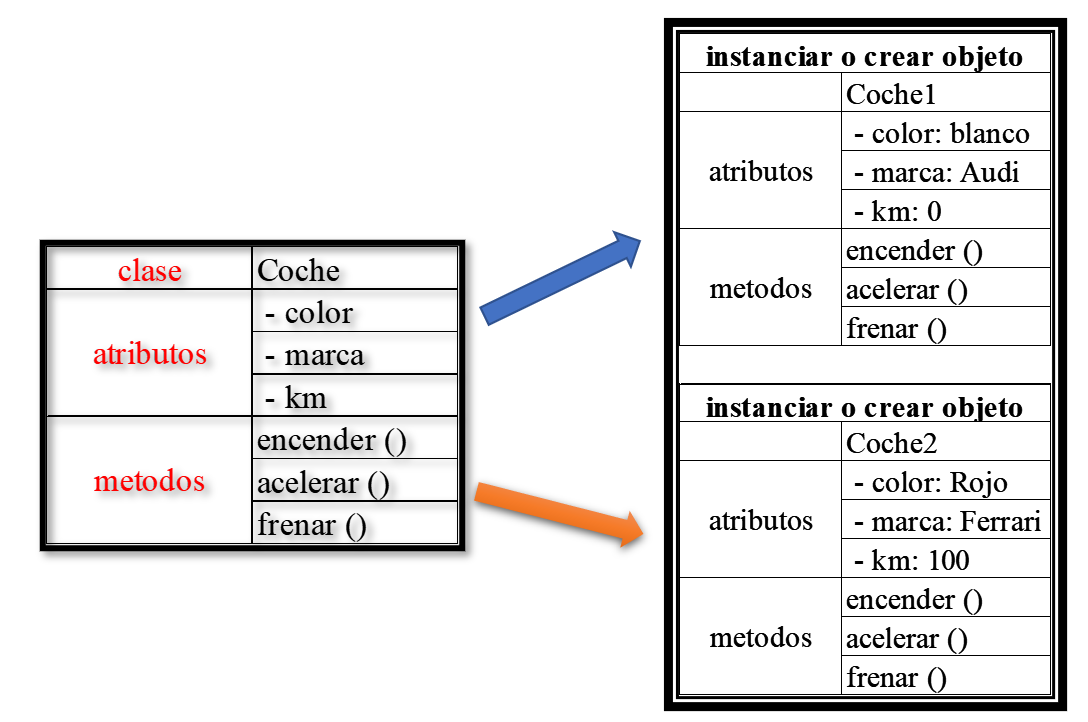
**La POO:** Es una forma especial de programar, más cercana a como expresaríamos las cosas en la vida real que otros tipos de programación.

Como pensar en objetos:

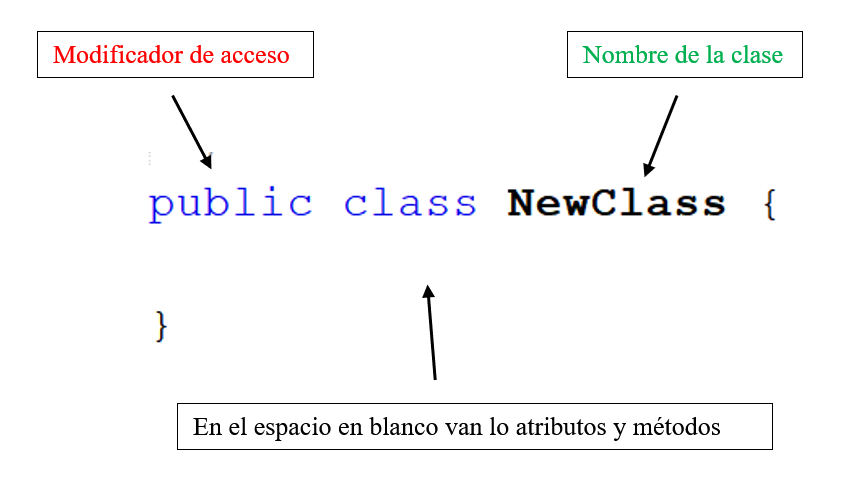


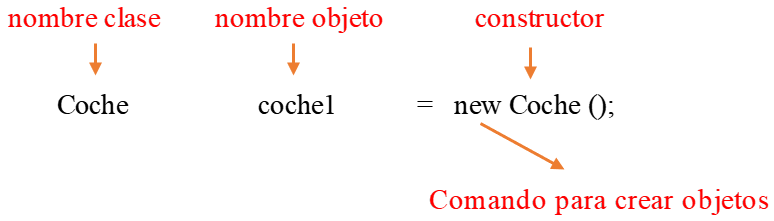
**Clase en POO:** es un conjunto de objetos con características similares.

Para crear una clase siempre se inicializa la primera letra en mayúscula y en singular ejemplo: Coche, Auto etc.



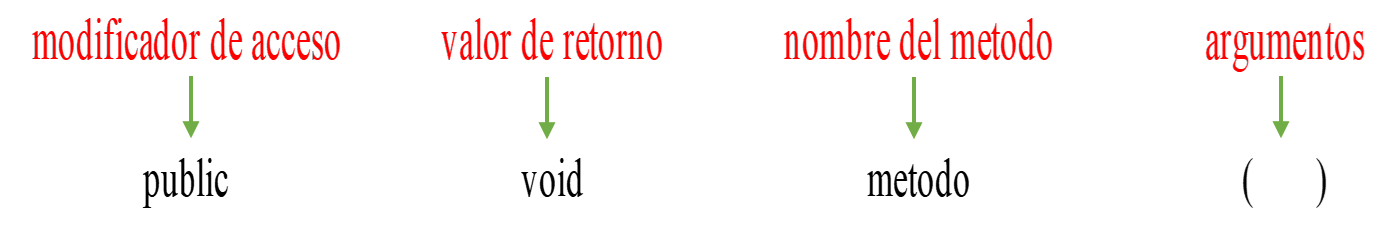
**Creación de clases y objetos.**



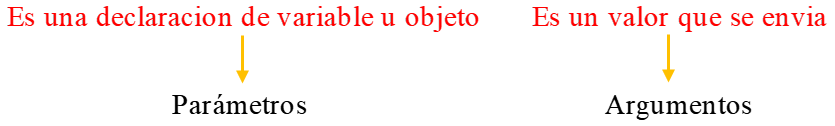


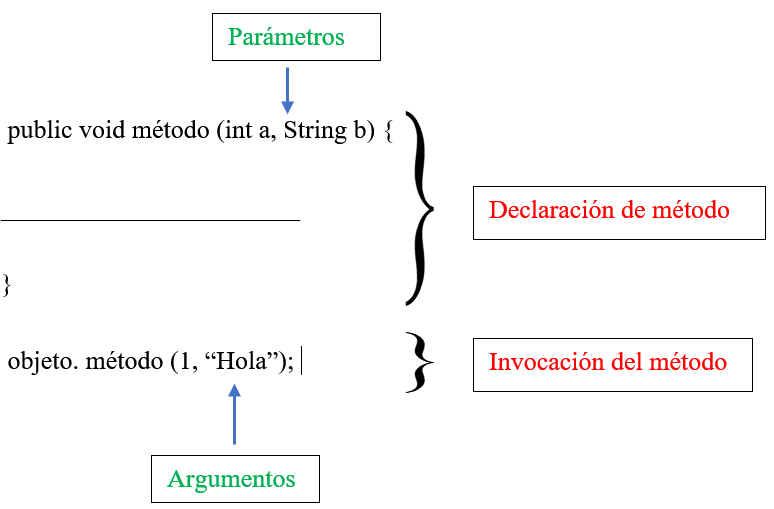
**Concepto y creación de métodos:**

Un método es una acción o comportamiento de los objetos.

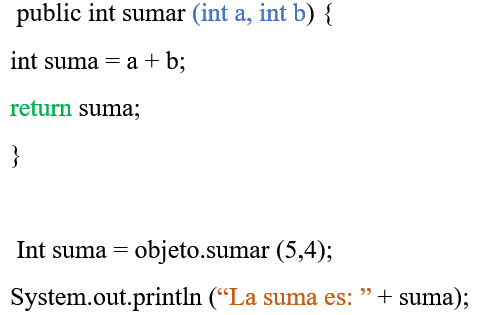


**Métodos: Parámetros y argumentos.**





**Métodos: Retorno de valores.**



**Método constructor:** es un método especial de una clase que se invoca siempre que se crea un objeto de clase.

Cuando se crea un objeto ocurren 3 cosas:

1. Se asigna memoria para el objeto.
2. Se inicializan los atributos.
3. Se invoca al constructor de la clase que puede ser uno entre varios.

Ejemplo: Persona p1 = new Persona ();

Características de los constructores.

1. Tiene el mismo nombre de la clase.
2. No devuelve ningún valor.
3. Debe declararse público.

Encapsulamiento: Se encarga de ocultar los atributos o métodos para que solamente puedan ser accedidos desde la misma clase y previene errores en programas extensos. Ejemplo

private int edad; la palabra private indica que el atributo edad esta encapsulado.

Setters: Permite establecer un dato para un atributo encapsulado.

Getters: Permite mostrar un dato para un atributo encapsulado.

Constantes: es un espacio de memoria donde se va a almacenar un valor y no va cambiar a lo largo del programa. private final String nombre; final se usa para indicar que es una constante y no va a cambiar.

Miembros estáticos de una clase: son los atributos y los métodos. También se conocen como miembros de clase por que le pertenecen ya no al objeto si no ahora a la clase.

Herencia: es una forma de reutilización de software en la que se crea una nueva clase al absorber los miembros de una ya existente.

Polimorfismo: es una relación de tipo herencia un objeto de la superclase puede almacenar un objeto de cualquiera de sus subclases. Esto significa que la clase padre o superclase es compatible con los tipos que derivan de ella. Pero no al revés.

Herencia Múltiple: Hace referencia a la característica de los lenguajes de programación orientada a objetos en la que una clase puede heredar atributos y métodos de más de una super clase.

Interfaces en java

* Permite simular la herencia múltiple.
* La interfaz solo es public o default.
* Todos sus métodos son abstractos.
* Todos sus atributos son final.