**Programación orientada a objetos**

Esto consiste en modelar objetos del mundo real colocando características (atributos)a los mismos, como por ejemplo:

En un establo se tiene un burro, al burro se le asigna un nombre que en este será el objeto como tal, y por consecuente se le asignan las características y unas acciones.

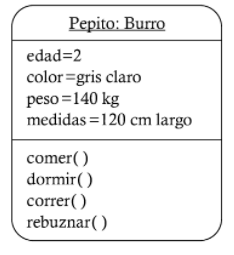
**Clase:** es aquella que se toma como una plantilla para crear objetos, de la cual pueden existir muchas como por ejemplo clases de personas, cuentas, burros, entre otros.

**Variables:** es donde se almacenan los datos que se quieren encapsular o agrupar.

**Métodos:** Son aquellos que definen las acciones de la clase.

**Objetos:** los objetos son instancias de las clases y es allí cuando se le asignan los datos a las variables de las clases.

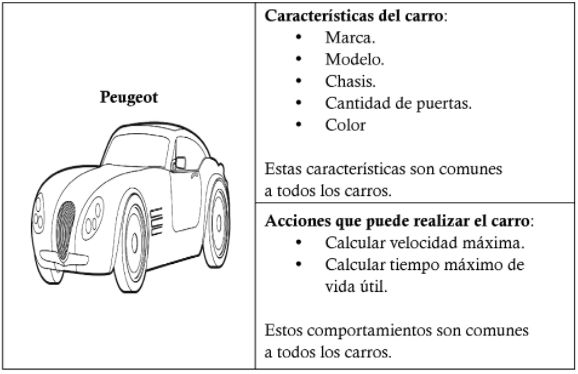
**Ejemplo:**



Inicialmente el nombre en este caso “pepito” es el objeto instanciado de la clase “burro” y lo demás por consecuente serian los atributos que se le asignan a la clase, asi como los métodos que en sí son las acciones de la clase.

**Abstracción: Esta es encargada de identificar los atributos y acciones del objeto.**

**Ejemplo: abstracción carro compuesta de varias características y acciones**



**Encapsulamiento y ocultamiento:**

El encapsulamiento consiste en que se agrupan las características y acciones (variables y métodos) relacionadas con la abstracción, pero no los detalles internos del mismo, por ejemplo: se conoce lo que hace la clase, pero no como lo hace.

Lo que hace el ocultamiento de la información es protegerla para que una entidad externa no la modifique sino que se programe que variables y métodos serán visibles para otras clases.

**Especificadores o modificadores de acceso**

* **Publico (+):** El elemento puede ser accedido desde otras clases sea datos miembros o métodos.
* **Privado (-):** Solo puede ser accedido desde que el método o dato pertenezca a la misma clase.
* **Protegido (#):** Solo se le proporciona acceso cuando la clase se le realiza herencia, de resto es privado para otras clases.
* **Sin modificar ():** Se accede desde cualquier clase mientras todo este dentro de la misma carpeta.

**Herencia y reutilización**

En la herencia se recoge todas las características la clase padre y se le deja la clase hija realizando una reutilización de contenido, para ahorrar líneas de código.

En la reutilización se realiza por medio de funciones esto permite llamar líneas de código cuantas veces se necesite.

**Polimorfismo**

Es cuando existen objetos que comparten el mismo nombre de método, y cuando el mismo es invocado depende al método que se llame tendrá un comportamiento distinto ya que no van a pertenecer a la misma entidad.

**Ejemplo:** Dos carros tienen el método de avanzar, pero los carros por su marca o modelo pueden variar en atributos los cuales van hacer que su comportamiento cambie asi como su velocidad.

**Bibliografía:** Oviedo Regino, E. Lógica de programación orientada a objetos. Bogotá: Ecoe Ediciones, 2015. p. https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/70431?page=271