**Taller de conceptos y principios de programación orientada a objetos**

Heidy Dayana Otagrí Rodríguez

Tecnología en Análisis y Desarrollo de Software

Servicio Nacional de Aprendizaje (SENA)

Ficha 2627060

Ing. Yerman Augusto Hernández Sáenz

Bogotá D.C.

**Programación Orientada Objetos**

Programación Orientada a Objetos, es un paradigma de programación que tiene como base la estructura del código en objetos. Permite la reutilización de código, la modularidad, la abstracción y la encapsulación, haciendo los programas más flexibles y escalables.

**Clase**

Las clases son un modelo o plantilla que define las propiedades o atributos y los métodos también llamados comportamientos de los objetos que se pueden instanciar de ella.

**Herencia**

La Herencia tiene como baje la jerarquía que tiene como esquema la estructura de un árbol. Consiste en la creación de clases a partir de una más grande, está clase también llamada superclase que tiene subclases que están obligadas a implementar los métodos de la clase padre, así como hereda los atributos. En ocasiones esta superclase se vuelve una clase abstracta ya que no se necesita una instancia de esta clase si no que se crean instancias de las subclases.

**Objeto**

Es una entidad que contiene datos y los comportamientos relacionados con ellos. A su vez, es instancia de una clase, es decir, se crea a partir de una plantilla llamada Clase

**Método**

Hace referencia a las acciones o comportamientos que puede realizar un objeto.

**Atributo**

Son las características, propiedades o datos relacionados con un objeto.

**Abstracción**

Es la representación del mundo real en forma de Clases que luego serán instanciadas por objetos.

**Encapsulamiento**

Es la capacidad que tiene una clase de ocultar ciertos detalles sobres sus atributos y comportamientos. Algunos ejemplos en JAVA son: **public, private, default** y **static**

**Polimorfismo**

Se refiere a la capacidad que tiene un objeto de que un método tenga diferentes comportamientos, lo que permite que dependiendo de lo que necesite una instancia de una clase actúe de forma diferente.

Ejemplo

Un **Mascota** (Por convención una clase se nombre en mayúscula) es la superclase y la plantilla, tiene como **atributos** (Por lo general los atributos de las clases son privadas y tienen a modificarse a través de getter y setter):

* Id
* Nombre
* Edad
* Genero
* Tamaño
* NombreDueño

Sus **métodos** son:

* Dormir
* Comer

Mascota sería una clase abstracta o super clase, no tiene necesidad de ser instanciada, ya que, para este ejemplo tendremos tres clases Perro, gato y canario, son subclases de mascota, estas heredan sus atributos y métodos. Aquí entra el concepto de sobreescritura, que hace referencia a la capacidad que tiene una subclase de implementar el método heredado a su forma.