**CONSULTA**

**SARA BARRERA MARTINEZ**

**CENTRO DE COMERCIO Y TURISMO**

**ANALISIS Y DESARROLLO DE SOFTWARE**

**FICHA: 3144886**

**ARLE MORALES**

**ARMENIA-QUINDIO**

**21/02/2025**

**1. CLASES**

**¿Qué es?**Una clase es como un molde de plastilina. Si quieres hacer varios muñecos parecidos (por ejemplo, varios perritos), haces un molde. La clase es ese molde que dice cómo es un objeto.

**Ejemplo 1:**

class Perro {

}

**Ejemplo 2:**

class Carro {

}

**2. PROPIEDADES**

¿Qué es?  
Son como los colores o detalles del muñeco de plastilina. Por ejemplo, que el perro sea blanco o negro. Es la información que tiene un objeto.

**Ejemplo 1:**

class Perro {

constructor() {

this.color = "marrón"; // propiedad

}

}

**Ejemplo 2:**

class Carro {

constructor() {

this.marca = "Toyota"; // propiedad

}

}

**3. MÉTODOS**

¿Qué es?  
Son las cosas que puede hacer tu figura de plastilina, como ladrar, correr, prenderse, etc.

**Ejemplo 1:**

class Perro {

ladrar() {

console.log("Guau guau");

}

}

**Ejemplo 2:**

class Carro {

encender() {

console.log("El carro está encendido");

}

}

**4. MODIFICADORES DE ACCESO**

¿Qué es?  
Son candados o puertas que dicen si puedes ver o tocar esa parte del objeto desde fuera. En JavaScript moderno, usamos # para indicar privado (acceso solo desde dentro de la clase).

**Ejemplo 1:**

class Perro {

#edad = 3; // privado

mostrarEdad() {

console.log(this.#edad);

}

}

**Ejemplo 2:**

class Carro {

#codigoMotor = "AB123";

mostrarCodigo() {

console.log(this.#codigoMotor);

}

}

**5. POLIMORFISMO**

¿Qué es?  
Es como tener varios muñecos que hacen lo mismo pero de forma diferente. Por ejemplo, todos pueden hacer “sonido”, pero uno ladra, otro maúlla.

**Ejemplo 1:**

class Animal {

hacerSonido() {

console.log("Algún sonido");

}

}

class Perro extends Animal {

hacerSonido() {

console.log("Guau");

}

}

class Gato extends Animal {

hacerSonido() {

console.log("Miau");

}

}

**Ejemplo 2:**

function reproducirSonido(animal) {

animal.hacerSonido();

}

let perro = new Perro();

let gato = new Gato();

reproducirSonido(perro); // Guau

reproducirSonido(gato); // Miau

**6. ABSTRACCIÓN**

¿Qué es?  
Es como esconder los detalles de cómo hiciste el muñeco. Solo le dices a alguien: "este muñeco puede bailar", pero no le explicas cómo mueves cada parte.

**Ejemplo 1:**

class Cafetera {

prepararCafe() {

this.#calentarAgua();

console.log("Café listo");

}

#calentarAgua() {

console.log("Agua caliente");

}

}

**Ejemplo 2:**

let cafetera = new Cafetera();

cafetera.prepararCafe(); // Solo puede usar lo que está público

**7. HERENCIA**

¿Qué es?  
Es como hacer un nuevo molde basado en otro que ya hiciste. Por ejemplo, haces una clase “Animal” y después una clase “Perro” que hereda lo que tenía el animal.

**Ejemplo 1:**

class Animal {

comer() {

console.log("Estoy comiendo");

}

}

class Perro extends Animal {}

**Ejemplo 2:**

let miPerro = new Perro();

miPerro.comer(); // Puede usar el método de Animal

**8. MÉTODO CONSTRUCTOR**

¿Qué es?  
Es como el paso inicial para crear tu figura. El constructor dice cómo nacerá tu objeto.

**Ejemplo 1:**

class Persona {

constructor(nombre) {

this.nombre = nombre;

}

}

Ejemplo 2:

let p1 = new Persona("Ana");

console.log(p1.nombre); // Ana

**9. INTERFACES**

¿Qué es?  
Es como un contrato que dice “si haces un muñeco de este tipo, tienes que ponerle estas partes”. En JavaScript no hay interfaces puras, pero podemos simularlas con clases o comentarios.

**Ejemplo 1 (simulado):**

class Volador {

volar() {

throw new Error("Debes implementar volar()");

}

}

**Ejemplo 2:**

class Pajaro extends Volador {

volar() {

console.log("Estoy volando");

}

}

**10. INSTANCIA**

¿Qué es?  
Es como el muñeco ya hecho con tu molde (clase). Una instancia es un objeto creado a partir de una clase.

**Ejemplo 1:**

class Gato {

constructor(nombre) {

this.nombre = nombre;

}

}

let miGato = new Gato("Michi"); // instancia

**Ejemplo 2:**

console.log(miGato.nombre); // Michi