**CICLO JAVASCRIPT – OBJETOS**

**EJEMPLOS**

**Ejercicio 1: Registro de Mascotas (Objetos y Métodos de Objetos)**

Crea un programa para gestionar un refugio de mascotas. Cada mascota estará representada como un objeto con las propiedades nombre, edad, raza y adoptado. Debes:

1. Crear un array llamado mascotas para almacenar las mascotas.
2. Agregar un método dentro de cada objeto de mascota llamado adoptar, que cambie la propiedad adoptada a true.
3. Escribir una función que permita agregar una nueva mascota al refugio.
4. Crear una función para listar todas las mascotas y sus estados (adoptadas o no).

const mascotas = [];

function agregarMascota(nombre, edad, raza, adoptado = false) {

const mascota = {

nombre,

edad,

raza,

adoptado,

adoptar() {

this.adoptado = true;

}

};

mascotas.push(mascota);

}

function listarMascotas() {

console.table(mascotas);

}

// Agrega y adopta mascotas

agregarMascota("Fido", 3, "Labrador");

agregarMascota("Michi", 2, "Siames");

mascotas[0].adoptar();

listarMascotas();

**Ejercicio 2: Administración de Productos (Clonación y Métodos de Objetos)**

Un ecommerce necesita manejar su catálogo de productos. Cada producto tendrá las propiedades id, nombre, precio, y stock.

1. Crea una función que reciba un producto (objeto) y devuelva un **nuevo producto clonado** con un descuento aplicado al precio.
2. Usa Object.assign() o el operador spread ... para clonar el objeto.

const producto = { id: 1, nombre: "Laptop", precio: 1000, stock: 10 };

function aplicarDescuento(producto, descuento) {

const productoConDescuento = { ...producto, precio: producto.precio \* (1 - descuento) };

return productoConDescuento;

}

const nuevoProducto = aplicarDescuento(producto, 0.2);

console.log("Producto Original:", producto);

console.log("Producto con Descuento:", nuevoProducto);

**Ejercicio 3: Información de Estudiantes (Desestructuración)**

Tienes un objeto que representa un estudiante:

const estudiante = {

nombre: "Juan",

edad: 20,

curso: "JavaScript",

notas: { practica: 90, teoria: 85, final: 88 }

};

1. Usa desestructuración para extraer las propiedades nombre y curso.
2. Extrae las notas y calcula el promedio.
3. Crea una función que reciba un objeto estudiante y devuelva una frase como:  
   "El estudiante Juan del curso JavaScript tiene un promedio de 87.67"

function promedioEstudiante({ nombre, curso, notas }) {

const { practica, teoria, final } = notas;

const promedio = (practica + teoria + final) / 3;

return `El estudiante ${nombre} del curso ${curso} tiene un promedio de ${promedio.toFixed(2)}`;

}

console.log(promedioEstudiante(estudiante));

**Ejercicio 4: Gestión de Tareas (Métodos y Arrays de Objetos)**

Crea un programa para gestionar una lista de tareas. Cada tarea tendrá las propiedades id, titulo, descripción, y completada. Debes:

1. Crear un array llamado tareas para almacenar las tareas.
2. Crear funciones para:

* Agregar una nueva tarea.
* Marcar una tarea como completada (por su id).
* Mostrar todas las tareas pendientes.

1. Usa métodos de objetos y funciones para organizar el programa.

const tareas = [];

// Función para agregar tareas al array

function agregarTarea(titulo, descripcion) {

const id = tareas.length + 1;

const tarea = { id, titulo, descripcion, completada: false };

tareas.push(tarea);

}

// Función para marcar una tarea como completada (sin usar find())

function completarTarea(id) {

for (let i = 0; i < tareas.length; i++) {

if (tareas[i].id === id) {

tareas[i].completada = true;

break; // Salimos del bucle una vez encontramos la tarea

}

}

}

// Función para mostrar todas las tareas pendientes (sin usar filter())

function mostrarPendientes() {

const pendientes = [];

for (let i = 0; i < tareas.length; i++) {

if (!tareas[i].completada) {

pendientes.push(tareas[i]);

}

}

console.table(pendientes);

}

// Ejemplo de uso

agregarTarea("Estudiar JavaScript", "Practicar objetos y métodos");

agregarTarea("Leer sobre APIs", "Entender cómo funcionan las REST APIs");

completarTarea(1);

mostrarPendientes();

**Ejercicio 5: Clonación y Modificación de Objetos (Spread y Object.assign)**

Crea un programa que reciba una lista de libros, donde cada libro es un objeto con las propiedades titulo, autor, y anio. Debes:

1. Clonar un libro existente y cambiar su título y año.
2. Crear una función que use Object.assign o el operador spread para clonar y modificar un libro.
3. Mostrar ambos libros, el original y el modificado.

const libro = { titulo: "Cien Años de Soledad", autor: "Gabriel García Márquez", anio: 1967 };

function clonarYModificarLibro(libro, nuevoTitulo, nuevoAnio) {

return { ...libro, titulo: nuevoTitulo, anio: nuevoAnio };

}

const nuevoLibro = clonarYModificarLibro(libro, "El Otoño del Patriarca", 1975);

console.log("Libro Original:", libro);

console.log("Libro Modificado:", nuevoLibro);

**BORRAR PROPIEDADES DE UN OBJETO:**

En JavaScript, puedes borrar una propiedad de un objeto usando el operador delete. Este operador elimina la propiedad especificada del objeto.

***Sintaxis***

delete objeto.propiedad;

o también se pude escribir así:

delete objeto["propiedad"];

***ejemplo:***

// Objeto inicial

const persona = {

nombre: "Juan",

edad: 25,

profesion: "Ingeniero"

};

console.log("Antes de borrar:", persona);

// Borrar la propiedad 'profesion'

delete persona.profesion;

console.log("Después de borrar:", persona);

***Resultado***

Antes de borrar: { nombre: 'Juan', edad: 25, profesion: 'Ingeniero' }

Después de borrar: { nombre: 'Juan', edad: 25 }