**Unidad 2: Objetos y Tipos de datos.**

**Ejercicios (2)**

#### EJERCICIO 14: Transforma borde-left-width en borderLeftWidth

Escribe la función camelize(str) que convierta palabras separadas por guión como “mi-cadena-corta” en palabras con mayúscula “miCadenaCorta”.

Es decir, eliminar todos los guiones y que cada palabra después de un guión comience con mayúscula.

Ejemplo:

camelize("background-color") == 'backgroundColor';

camelize("list-style-image") == 'listStyleImage';

camelize("-webkit-transition") == 'WebkitTransition';

#### EJERCICIO 15: Obtener un subarray

Escribe una función filterRange(arr, a, b) que obtenga un array arr, busque los elementos con valor mayor o igual a a y menor o igual a b y devuelva un array con los resultados.

La función no debe modificar el array. Debe devolver un nuevo array.

Por ejemplo:

let arr = [5, 3, 8, 1];

let filtrado = filterRange(arr, 1, 4);

console.log( filtrado ); // 3,1 (valores dentro del rango)

console.log( arr ); // 5,3,8,1 (array original no modificado)

#### EJERCICIO 16: Ordenar decreciente

Escribe una función ordenaDecreciente(arr) que recibe un array como parámetro y lo ordena en orden decreciente matemático.

Suponemos que todos los elementos del array son números.

La función modifica el array.

Ejemplo

let arr = [5, 2, 1, -10, 8];

ordenaDecreciente(arr);

console.log(arr);

#### EJERCICIO 17: Copia y ordena un array

Escribe una función copiaOrdenada(arr) que devuelva una copia del array ordenada con orden alfabético.

El array original no se tiene que modificar.

Ejemplo

let arr= ["HTML", "CSS", "JavaScript"];

let arrOrdenado = copiaOrdenada(arr);

console.log(arr); // HTML,CSS,JavaScript

console.log(arrOrdenado); // CSS,HTML,JavaScript

#### EJERCICIO 18: Mapear a nombres

Tenemos un array de objetos users, y cada uno de los objetos tiene una propiedad user, que es una cadena, el nombre del usuario.

Escribe una función a la que pasamos el array users y devuelve un array con los nombres de los usuarios.

Ejemplo:

let john = { name: "John", age: 25 };

let pete = { name: "Pete", age: 30 };

let mary = { name: "Mary", age: 28 };

let users = [ john, pete, mary ];

let names = mapaANombres(users);

console.log( names ); // John, Pete, Mary

#### EJERCICIO 19: ordenarPorEdad

Escribe la función ordenarPorEdad(users) que recibe como parámetro un array de objetos con la propiedad edad y los ordena según edad.

Por ejemplo:

let john = { name: "John", edad: 25 };

let pete = { name: "Pete", edad: 30 };

let mary = { name: "Mary", edad: 28 };

let arr = [ pete, john, mary ];

ordenarPorEdad(arr);

// ahora: [john, mary, pete]

console.log(arr[0].name); // John

console.log(arr[1].name); // Mary

console.log(arr[2].name); // Pete

#### EJERCICIO 20: mapear a objetos

Tienes un array de objetos usuarios, cada uno tiene nombre, apellido e id.

Escribe el código para crear otro array a partir de este, de objetos con id y nombrCompleto, donde nombreCompleto es generado a partir de nombre y apellido.

Por ejemplo:

let john = { nombre: "John", apellido: "Smith", id: 1 };

let pete = { nombre: "Pete", apellido: "Hunt", id: 2 };

let mary = { nombre: "Mary", apellido: "Key", id: 3 };

let usuarios = [ john, pete, mary ];

let usuariosMapeados = /\* ... tu código ... \*/

/\*

usuariosMapeados= [

{ nombreCompleto: "John Smith", id: 1 },

{ nombreCompleto: "Pete Hunt", id: 2 },

{ nombreCompleto: "Mary Key", id: 3 }

]

\*/

alert( usuariosMapeados[0].id ) // 1

alert(usuariosMapeados [0].nombreCompleto ) // John Smith

Para hacerlo, se puede mapear un array de objetos a otro.

Si se utilizan funciones flecha (=>), hay un pequeño truco.