Unidad 4

```
/*
Unidad 4: Funciones en JavaScript
1. ¿Qué es una función?
Grupo de instrucciones con nombre identificativo.
Puede recibir datos de entrada (parámetros).
Devuelve un resultado con return.
Facilitan dividir programas grandes y reutilizar código.
*/
/*
2. Definición y llamada de funciones
*/
function mostrarMensaje() {
    console.log("Función llamada");
}
mostrarMensaje(); // Ejecuta la función
```

```
/*
3. Parámetros y argumentos
Parámetros son variables definidas en la función.
Argumentos son los valores que se pasen cuando se llama.
*/
function saludar(nombre) {
    console.log("Hola " + nombre);
}
saludar("Ana");
/*
4. Valor de retorno
Se usa return para devolver un resultado.
Solo una instrucción return por función para claridad.
*/
function suma(a, b) {
    return a + b;
}
let resultado = suma(3, 5); // 8
/*
```

```
5. Ámbito de variables

Variables locales solo visibles dentro de la función.

Variables globales visibles en todo el programa.

*/

let globalVar = 10;

function prueba() {

    let localVar = 20;

    console.log(globalVar); // 10

    console.log(localVar); // 20

}

prueba();

/*

6. Parámetros avanzados

Valores por defecto:

*/

function multiplicar(a, b = 2) {

    return a * b;

}

multiplicar(3); // 6
```

```
/*
Parámetros variables con "arguments":
*/
function mostrarArgumentos() {
    for (let i = 0; i < arguments.length; i++) {
        console.log(arguments[i]);
    }
}
mostrarArgumentos("uno", "dos", "tres");
/*
Parámetro rest (spread) ... para recoger varios argumentos en un array:
*/
function sumar(...numeros) {
    return numeros.reduce((total, num) => total + num, 0);
}
sumar(1, 2, 3); // 6
/*
7. Formas de definir funciones
Declaración tradicional:
```

```
function nombre() {}

Expresión asignada a variable:

const nombre = function() {};

Función flecha (más corta y moderna):

const nombre = () ⇒ {};

*/

/*

8. Funciones flecha ejemplos

/

const sumaFlecha = (a, b) ⇒ a + b;

const saludarFlecha = nombre ⇒ "Hola " + nombre;

/*

9. Funciones anónimas y callbacks

Función sin nombre, usada como argumento para otra función.

*/

setTimeout(function () {

    console.log("Ejecutado luego de 2 segundos");

}, 2000);

/*
```

```
10. Funciones autoejecutables (IIFE)

Se definen y ejecutan inmediatamente.

*/

(function () {

    console.log("Función autoejecutable");

})();

/*

11. Recursividad

Función que se llama a sí misma.

Debe tener un caso base que detenga la llamada.

*/

/* Ejemplo: factorial */

function factorial(n) {

    if (n === 0) return 1;

    return n * factorial(n - 1);

}

/*

12. Métodos avanzados para arrays

sort(): ordenar con función comparadora personalizada
```

```
forEach(): recorrer elementos con función

map(): transformar elementos y devolver nuevo array

filter(): filtrar elementos que cumplen una condición

*/

/* Ejemplos de arrays */

let frutas = ["pera", "manzana", "kiwi"];

/* Ordenar por longitud */

frutas.sort((a, b) => a.length - b.length);

/* Recorrer con forEach */

frutas.forEach(fruta => console.log(fruta));

/* Transformar con map */

let longitudes = frutas.map(fruta => fruta.length);

/* Filtrar con filter */

let cortas = frutas.filter(fruta => fruta.length < 5);
```