□ Tema 3: Arrays en JavaScript

□ Resumen del contenido

1. Introducción

- Un **array** es un tipo especial de objeto que almacena varios valores bajo un mismo nombre y se accede a ellos por **índices numéricos**.
- Se puede declarar con `[]` o con `new Array()`:

```
let frutas = ["manzana", "pera", "uva"];
let numeros = new Array(1, 2, 3);
```

- Los arrays pueden contener distintos tipos de datos, incluso otros arrays u objetos.
- Si se intenta acceder a una posición inexistente, devuelve `undefined`.

1.1 Desestructuración

Permite asignar los valores de un array a variables independientes:

```
const colores = ["rojo", "verde", "azul"];
const [c1, c2, c3] = colores;
```

1.2 Operador Spread (...)

Permite combinar arrays o agrupar parámetros variables:

```
const base = [1, 2, 3];
const completo = [...base, 4, 5];
function sumar(...valores) { return valores.reduce((a,b)=>a+b); }
```

2. Añadir y eliminar elementos

- `push()` añade al final.
- `unshift()` añade al principio.
- `pop()` elimina el último.

- `shift()` elimina el primero.
- `length` devuelve o cambia la longitud.

2.1 slice()

Devuelve una copia parcial sin modificar el original:

```
let subArray = array.slice(1, 3);
```

2.2 splice()

Permite eliminar, insertar o reemplazar elementos:

```
array.splice(pos, numEliminados, elem1, elem2);
```

3. Métodos comunes

- `sort()` ordena elementos.
- `join()` / `split()` convierte entre cadena y array.
- `concat()` une arrays.
- `reverse()` invierte el orden.
- `indexOf()` / `lastIndexOf()` busca posiciones.

4. Programación funcional con arrays

Método	Descripción
filter()	Filtra elementos que cumplan una condición.
find() / findIndex()	Devuelve el primer elemento o su índice.
every() / some()	Comprueban si todos o alguno cumplen la condición.
map()	Crea un nuevo array transformando los elementos.
reduce()	Reduce el array a un solo valor.
forEach()	Recorre el array ejecutando una acción.
includes()	Comprueba si un valor existe.
Array.from()	Crea un array desde otro iterable.

5. Copias de arrays

Asignar un array crea una **referencia**, no una copia. Para copiar sin vincular:

```
let copia = array.slice();
let copia2 = [...array];
```

□ Ejercicios integradores

□ Ejercicio 1 (Básico)

Crea un array con tus comidas favoritas:

```
let comidas = ["pasta", "pizza", "tacos"];
comidas.push("sushi");
comidas.shift();
console.log(comidas);
console.log(comidas.length);
```

□ Ejercicio 2 (Intermedio)

Manipula un array con `splice` y `slice`:

```
let materiales = ["tela", "bies", "hilo", "tijeras", "máquina de coser", "botón"];
materiales.splice(1, 1);
materiales.splice(2, 0, "hilo torzal");
materiales.splice(3, 1, "cúter rotatorio", "mesa de corte");
let parte1 = materiales.slice(0, 3);
let parte2 = materiales.slice(3);
console.log(materiales, parte1, parte2);
```

□ Ejercicio 3 (Avanzado)

Analiza un array de notas:

```
let notas = Array.from({length: 10}, () => Math.floor(Math.random() * 10) + 1);
let aprobadas = notas.filter(n => n >= 5);
let media = notas.reduce((a, b) => a + b, 0) / notas.length;
let todasAprobadas = notas.every(n => n >= 5);
let porcentajes = notas.map(n => n * 10);

console.log("Notas:", notas);
console.log("Aprobadas:", aprobadas);
console.log("Media:", media.toFixed(2));
console.log("¿Todas aprobadas?", todasAprobadas);
console.log("Porcentajes:", porcentajes);
```