Arrays

Características

- Objetos que permiten manejar conjuntos de datos.
- Se les llaman también listas, vectores, arreglos.
- Son dinámicos, su tamaño se puede modificar después de la declaración.
- **Son heterogéneos. Cada elemento del array puede ser de un tipo diferente.**
- La primera posición siempre es 0

Declarar Arrays

- Declarar array vacio:

```
let a = [];
let a = new Array();
```

- Declarar con valores:

```
let a = new Array(3,4,5,6,7);
let b =[1,true,[1,2,3],”cad”];
let c = [3,4,5,6,7];
let d= Array(100).fill(false);

console.log(a[1]); // muestra 4
console.log(a); // muestra [3,4,5,6,7]
```

- Para asignar valores:

```
a[0] = ”Rojo”;
a[1] = “Azul”;
console.log(a[1]); // muestra azul
```

Valores indefinidos

✔ Ejemplo:

```
let a =[“Saul”, “Rocio”];
a[3]=”Maria”;
console.log(a[2]);     // escribe “undefined”
```

✔ Ejemplo:

```
let a = [“Saul”, “Rocio”, ,“Maria”];
console.log(a[2]);     // escribe “undefined”
```

Borrar elementos: delete

let dias=[“Lunes”,”Martes”,”Miercoles”];

dias[3]=”Jueves”;

delete dias[2]; → Ojo, no afecta a la longitud de la cadena

console.log(dias); // Como queda el array? ¿Qué longitud tiene? ¿que valor tiene dias[2]?

Arrays heterogéneos

✔ Ejemplo:

```
let a =[3, 4, ”Hola”, true, Math.random()];
```

✔ Ejemplo:

```
let b=[3, 4, ”Hola”, [ 99, 55, 33]];
console.log(b[3][1]); // Que devuelve?
console.log(b[2][1]); // Que devuelve?
```

Recorridos de Arrays

- **Con for**

```
let notas =[5,7,3,8,5,3,8];
for(let i =0; i< notas.length; i++)
{
    console.log(“La nota ” + i + “ es ” +  notas[i]);
}
```

Resultado: notas[i] sacará: 5,7,3,8,5,3,8

Y si hay elementos indefinidos en el array?

```
notas =[5,7,,,5,3,8];
for(let i =0; i< notas.length; i++)
{
    console.log(“La nota ” + i + “ es ” +  notas[i]);
}
```

Resultado: notas[i] sacará: 5,7,undefined,undefined ,5,3,8

Hay que controlar los elementos undefined si no queremos que aparezcan

```
notas =[5,7,,,5,3,8];
for(let i =0; i< notas.length; i++)
{
    if(notas[i]!=undefined)
        console.log(“La nota ” + i + “ es ” +  notas[i]);
}
```

Resultado: notas[i] sacará: 5,7,5,3,8

- **Con for .. in**
  - Se saltan los indefinidos

```
notas =[5,7,,,5,3,8];
for(let i in notas)
{
    console.log("La nota " + i + " es " +  notas[i]);
}
```

Resultado: notas[i] sacará: 5,7,5,3,8

- **Con for .. of**
  - Surge a partir del ES2015
  - NO Se saltan los indefinidos

```
notas =[5,7,,,5,3,8];
for(let i of notas)
{
    console.log(i);   //¿Que muestra?
}

notas =[5,7,,,5,3,8];
for(let i of notas)
{
    if(i!=undefined)
        console.log(i);
}
```

Resultado: i sacará: 5,7,5,3,8

Métodos de Arrays

- **Length.** Devuelve la longitud del array

```
dias =["Lunes","Martes","Miércoles","Jueves","Viernes","Sábado","Domingo"];

console.log(dias.length);
```

- **Instanceof.** Permite saber si el objeto es un array. Devuelve un valor booleano.

```
let a =[1,2,3,4,5,6,7];
let b="Hola";
console.log(a instanceOf Array); // Que devuelve?
console.log(b instanceOf Array); // Que devuelve?
```

- **Push**: Añade elementos al final de un array

```
let colores=["verde"];
colores.push("blanco"); // el array tendrá dos elementos
```

- **Pop**: retira del array el último elemento

```
colores.pop(); // el array solo tendrá un elemento, "verde"
```

- **Shift**: Quita el primer elemento de un array
- **Unshift**: Añade un elemento al inicio del array

colores.unshift(“naranja”); // el array tendrá: “naranja, verde”

colores.shift(); // el array tendrá verde

- **concat**: Para unir dos arrays en uno nuevo. No se modifica ninguno de los arrays originales:

  b.concat(a);

- **slice**: Obtiene un subarray, indicando el indice del primer elemento que deseamos obtener y el índice del final (el último no se incluye y es optativo)

  let nombres=[“Juana”,”Pedro”,”Miguel”,”Ana”,”Pepa”];

  let masculinos = nombres.slice(1,3); // masculinos contiene [“Pedro”,”Miguel”]

- **splice**: permite añadir y eliminar elementos al array: splice(start[, deleteCount[, item1[, item2[, ...]]]])

  Se indica el índice desde donde comienza la eliminación y cuantos elementos eliminamos.

  Eje: colores.splice(1,1); //Borra el elemento de la posición 1.

  let colores =[‘blanco’,’negro’,’azul’,’lila’];

  let removed = colores.splice(2,0,’verde’); // añade el color verde en la posición 2

  colores=[‘blanco’,’negro’,’verde’,’azul’,’lila’];

  removed = colores.splice(3,1); // elimino el color azul

- **Join**: permite convertir un array en un string con todos los elementos del array separados por coma. Se puede indicar otro separador.

  colores.join();

  colores.join(“-”);

- **IndexOf**: Busca el índice de un elemento de un array. Si no lo encuentra, devuelve -1. Permite indicar el inicio de la búsqueda.
- **LastIndexOf**. Empieza a buscar desde el último elemento. Permite indicar el inicio de la búsqueda
- **Includes**. Busca un elemento y devuelve true si lo encuentra o false en caso contrario
- **Reverse**: Invierte un array.
- **Sort**: Permite ordenar los elementos de un array.
- **ForEach**: Sirve para recorrer arrays y puede llamar a una función por cada elemento que puede recibir 3 parametros: (elemento actual, [indice, array objetivo]) // [] opcionales

  ✔ Ejemplo de recorrido con foreach;

```
let ranks = ['A', 'B', 'C'];
ranks.forEach(function (e) {
    console.log(e);
});
```

```
let ranks = ['A', 'B', 'C'];
ranks.forEach((e) => {
    console.log(e);
});
```

  ✔ Ejemplo de recorrido con foreach; → ¿**Cómo obtenemos el índice de cada elemento**?

```
ranks.forEach(function (e,index) {
            console.log(e + “ indice: ” +index);
});
```

- **Filter**: devuelve un array nuevo con los elementos que cumplan una condición
  - ✔ Ejemplo: Dado un array de números, seleccionar los pares.

```
const numbers = [1, 2, 3];
const evens = numbers.filter(number => number % 2 === 0); // es par
console.log(evens); // [2]
```

- **Map:** Devuelve un nuevo array con los valores que se calculen en cada elemento
  - ✔ Ejemplo: Dado un array de números, devolver otro con sus valores dobles:

```
const arr =[2,3,4]
const newArr = arr.map(el => el * el);
return console.log(`Array original ${arr} \nArray con el doble ${newArr}`);
```

Objetos literales

- Objetos en los que se pueden definir directamente sus propiedades y sus métodos:

```
let perro = {
nombre:"Scott",
color:"Cafe",
edad: 5,
macho: true
};

console.info(perro.nombre); // Scott
console.info(perro.edad); // 5
console.info(perro['nombre']); // Scott

console.info(perro['edad']); // 5
```

- Un objeto tiene **propiedades** y **métodos**. Accedemos a ellos a través **de un punto**.
  - **Objeto.propiedad;**
    - Ejemplo:

      coche.color=”verde”; // modificamos la propiedad color del coche
  - **Objeto.metodo();**
    - Ejemplo:
    - coche.acelerar(25);

- Ejemplo:

```
let perro = {
    nombre:"Scott",
    color:"Cafe",
    edad: 5,
    macho: true,
    ladrar: function(){
        return(`${this.nombre} puede ladrar`)
    }
};
console.log(perro.ladrar()); // Scott puede ladrar
```

- **Inserción** de una nueva propiedad

```
perro.tamaño = "Grande";
console.log(perro);
```

- **Modificar** una propiedad:

```
perro.edad = 8;
console.log(perro);
```

- **Eliminar** una propiedad: delete nombre_del_objeto.clave;

```
delete perro.color;
console.log(perro);
```

- **Recorrer** propiedades de uno objeto: nombreObjeto[propiedad]

  Dado un objeto literal 'punto':

```
for (let prop in punto)
{
    console.log (`${prop} tiene el valor ${punto[prop]}`);
}
```

  - Si no queremos mostrar las funciones:

```
for (let prop in punto)
{
    if(typeof punto[prop] != "function")
        console.log (`${prop} tiene el valor ${punto[prop]}`);
}
```

- **Objeto this**
  - Hace referencia al objeto actual. Nos permite llegar al objeto propietario de la propiedad o del método.

Ejercicios

1. Dado un array de números: const arr = [2, 3, 4, 5, 0]:
   1. Escribir por consola la suma del array. Hacerlo con el método forEach y arrow functions.
   2. Escribir por consola la media.
   3. Obtener otro array con el triple de cada elemento y mostrarlo por consola.
   4. Obtener el mismo array con el triple de cada elemento y mostrarlo por consola.
2. Crear un array con 3 palabras que se introducirán por mensajes al usuario palabra a palabra (3 veces)
   Si cancela, se insertará una cadena vacía en el array.
   Escribir por consola y por pantalla, el array inicial y el array filtrando sólo las que comienzan por la letra C o c. Mostrarlas las palabras separadas por coma.
   Si no hay ninguna, escribir: “No hay ninguna palabra que comience por C.”
3. Crear un array de 5 elementos que representan personas, donde cada elemento tiene las propiedades: nombre, edad y ciudad.
   1. Mostrar en la página las personas mayores de edad en forma de lista html.
   2. Mostrar en la página las personas que son de Sevilla en forma de lista html.
4. Crear un array de objetos que representan productos en una tienda, donde cada objeto tiene las propiedades **nombre**, **precio** y **categoria:**
   Obtener un nuevo array que contenga sólo los nombres de los productos, transformando cada nombre en mayúsculas.
   Mostrar en consola el array de nombres transformados y en la página HTML separados por guión.
5. Realizar programa donde el usuario introduce una palabra y devuelva el número total de vocales contenidas. Escribir por pantalla el resultado: “La palabra X tiene Y vocales”.
   Utilizar la función forEach (PISTA: pasar de cadena a array.)
6. Dado un array con los 7 días de la semana, mostrar por pantalla la longitud de cada palabra, así como el día de la semana más largo. Escribir un fichero js externo para javascript. Usar forEach.
7. Hacer un programa que pida por pantalla al usuario una palabra. Seguir pidiéndola si se introduce vacía. Mostrar la palabra original y la invertida.

   Ejemplo: La palabra hello invertida es olleh
8. Crear una función flecha, que reciba un array y que se escriba por pantalla dos arrays: uno con los números pares y otro con los impares. Realizar las siguientes comprobaciones: que se reciba un array, que no esté vacío y que sólo contenga números.

   Llamar a la función con [1,3,2,5,7,4], [], [1,”3”,2] , “pepe”
9. Crear un objeto literal llamado factura con las propiedades:
   1. numero, cliente, divisa, subtotal e IVA, dándole valores a cada uno de ellos.
   2. Tendrá también un método que calcula el total (subtotal + iva)
   3. Imprimir por consola: La factura X(numero) tiene un importe de Y(subtotal) Z(divisa)
10. Crea un objeto literal llamado ‘user’ vacío.
    1. Agrega la propiedad *name* con el valor John.
    2. Agrega la propiedad *surname* con el valor Smith.
    3. Cambia el valor de name a Peter.
    4. Elimina la propiedad name del objeto.

11. Tienes un array de objetos que representan productos en un inventario. Cada objeto tiene las propiedades nombre, cantidad y precio. Crear un nuevo array de objetos que contenga el nombre del producto y el valor total en stock (cantidad * precio , de cada producto.). Luego, imprimir el nombre del producto y el valor total en la consola con dos decimales.

12. Tenemos un objeto que almacena los salarios de nuestro equipo:
```
let salarios = {
    John: 100,
    Ann: 160,
    Peter: 130
};
```
Escribe el código para sumar todos los salarios y mostrar por pantalla el resultado de la suma. En el ejemplo de arriba nos debería dar 390. Si salarios está vacío entonces el resultado será 0.

13. Dado un array de usuarios donde cada usuario tiene un nombre y una edad, obtener los usuarios que tienen 18 años o más. Luego, crear un array de nombres de estos usuarios en formato "Nombre (Edad años)". Muestra el resultado en la consola.

14. Dado un array de objetos de libros, donde cada libro tiene un título y un número de páginas, obtener los libros que tienen más de 300 páginas. Luego, extraer solo los títulos de estos libros. Finalmente, imprimir los títulos en la consola.

15. Tienes un array de objetos que representan estudiantes. Cada objeto de estudiante tiene un nombre y un array de calificaciones. Utiliza las funciones de arrays para realizar las siguientes tareas:

    1. Calcular el promedio de calificaciones para cada estudiante. Para cada estudiante, calcular su media. Debe devolver un array con objetos con el nombre y la media. Imprimir.
    2. Del array obtenido en el punto 1, filtrar los estudiantes que tienen un promedio superior a 7.
    3. Imprimir el nombre de los estudiantes que cumplen con el criterio de promedio anterior.

16. Tienes un array de objetos que representan productos en una tienda. Cada objeto tiene un nombre, un precio, y una propiedad categoria, que es otro objeto con un nombre y una descripción. Realiza las siguientes tareas:

    1. Crear un array de productos donde cada producto tiene una categoría con nombre y descripción.
    2. Obtener un array de nombres de productos pertenecientes a una categoría específica (por ejemplo, "Electrónica"). Imprimirlo
    3. Imprimir los detalles de los productos filtrados, incluyendo el nombre de la categoría y su descripción.

17. Crear una página HTML con un botón para llamar a la función flecha crearObjeto. Esta función debe definir un objeto Taxi con 5 propiedades: tipoMotor, numeroPasajeros, carga, velocidad y ruedas. Darles valor. Tendrá un método saludar, que mostrará un mensaje alert de la forma: ‘Hola soy un taxi de X ruedas y Y pasajeros’. Después de crear el objeto taxi, invocar la función saludar.