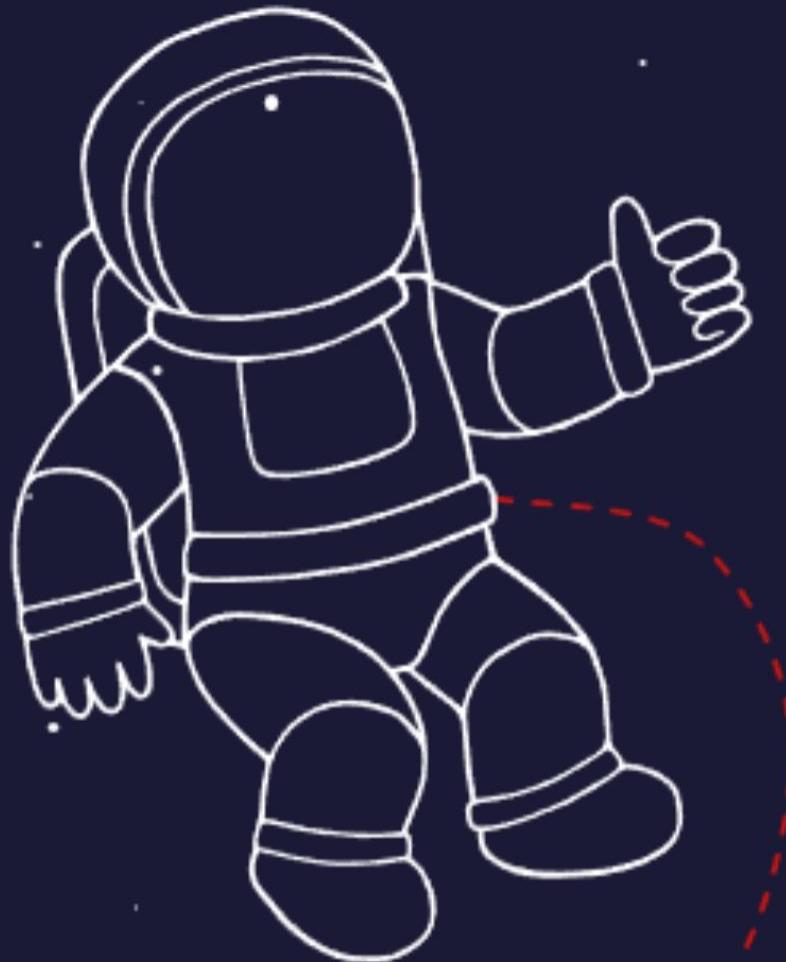




Programa académico **CAMPUS**



JAVASCRIPT

ARRAYS

Creación de Arrays

Los **Arrays** (*vectores y matrices*) de JavaScript es una sola variable que se utiliza para almacenar diferentes elementos. Suele utilizarse cuando queremos almacenar una lista de elementos y acceder a ellos mediante una sola variable.

Sintaxis de creación de Arrays

Sintaxis Método 1

```
let arrayName = [valor1, valor2, ...];
```

Sintaxis de creación de Arrays

Sintaxis Método 2

```
let arrayName = new Array(); // Método 2
```

Ejemplo de creación de Arrays

Ejemplo Método 1

```
// Inicializando mientras se declara  
let casa = ["1BHK", "2BHK", "3BHK", "4BHK"]
```

Ejemplo de creación de Arrays

Ejemplo Método 2

```
//Crea un array de 5 elementos 10, 20, 30, 40, 50
let house = new Array(10, 20, 30, 40, 50);
console.log(hous);
//Salida: [ 10, 20, 30, 40, 50 ]
```

Ejemplo de creación de Arrays

Ejemplo Método 2

```
// Crea un array de 5 elemento undefined
let house1 = new Array(5);
console.log(house1);
//Salida: [ <5 empty items> ]
```

Ejemplo de creación de Arrays

Ejemplo Método 2

```
// Crea un array con el string 1BHK
let home = new Array("1BHK");
console.log(home)
//Salida: [ '1BHK' ]
```

Ejemplo de creación de Arrays

Ejemplo Método 2

```
// Crea un Array de 2 elementos undefined
let house1 = new Array(2);
// Ahora asigna cada posición
house1[0] = "1BHK"
house1[1] = "2BHK"
console.log(house1) //Salida: ["1BHK", "2BHK", "3BHK", "4BHK"]
```

Creación de Arrays

Un array en JavaScript puede contener diferentes elementos que pueden almacenar números, cadenas y valores booleanos en una sola matriz.

Creación de Arrays

```
// Almacenando numeros, booleanos y cadenas en un array
let house = ["1BHK", 25000, "2BHK", 50000, "Rent", true];
console.log(house)
//Salida: [ '1BHK', 25000, '2BHK', 50000, 'Rent', true ]
```

Acceso a los elementos del Array

Se indexan desde 0, por lo que podemos acceder a los elementos de la matriz de la siguiente manera.

Acceso a los elementos del Array

```
let house = ["1BHK", 25000, "2BHK", 50000, "Rent", true];
console.log(house[0]+" cost= "+house[1]); // 1BHK cost= 25000

let cost_1BHK = house[1];
let is_for_rent = house[5];
console.log("Cost of 1BHK = "+ cost_1BHK); // Cost of 1BHK = 25000

console.log("Is house for rent = "+ is_for_rent); // Is house for rent = true
```

Recorrido de Array

```
let house = [ "1BHK", 2500, "2BHK", 50000, "Rent", true];  
  
// length: Longitud del array  
for (let i = 0; i < house.length; i++)  
    console.log(house[i]);
```

Métodos más usados Array

Método	Definición
push()	Agrega uno o más elementos al final del array y devuelve la nueva longitud del array.
pop()	Elimina el último elemento del array y lo devuelve.
shift()	Elimina el primer elemento del array y lo devuelve.
unshift()	Agrega uno o más elementos al principio del array y devuelve la nueva longitud del array.
slice()	Devuelve una parte del array seleccionada desde el índice inicial hasta el índice final (no incluido).

Métodos más usados Array

Método	Definición
splice()	Agrega o elimina elementos del array y devuelve los elementos eliminados.
concat()	Une dos o más arrays y devuelve un nuevo array.
join()	Une todos los elementos del array en una cadena de texto separada por un separador especificado.
indexOf()	Devuelve el índice del primer elemento encontrado en el array, o -1 si no lo encuentra.

Método PUSH

Agrega uno o más elementos al final del array y devuelve la nueva longitud del array.

```
let arr = [1, 2, 3];
arr.push(4);
console.log(arr); // [1, 2, 3, 4]
```

Método POP

Elimina el último elemento del array y lo devuelve.

```
let arr = [1, 2, 3];
let lastElement = arr.pop();
console.log(lastElement); // 3
console.log(arr); // [1, 2]
```

Método SHIFT

Elimina el primer elemento del array y lo devuelve.

```
let arr = [1, 2, 3];
let firstElement = arr.shift();
console.log(firstElement); // 1
console.log(arr); // [2, 3]
```

Método UNSHIFT

Agrega uno o más elementos al principio del array y devuelve la nueva longitud del array.

```
let arr = [1, 2, 3];
arr.unshift(0, -1);
console.log(arr); // [0, -1, 1, 2, 3]
```

Método SLICE

Devuelve una parte del array seleccionada
desde el [índice inicial hasta el índice final]

```
let arr = [1, 2, 3, 4, 5];
let slicedArr = arr.slice(1, 4);
console.log(slicedArr); // [2, 3, 4]
console.log(arr); // [1, 2, 3, 4, 5]
```

Método SPLICE

Agrega o elimina elementos del array y devuelve los elementos eliminados.

```
let arr = [1, 2, 3, 4, 5];
let removedElements = arr.splice(1, 2);
console.log(removedElements); // [2, 3]
console.log(arr); // [1, 4, 5]
```

Método CONCAT

Une dos o más arrays y devuelve un nuevo array.

```
let arr1 = [1, 2, 3];
let arr2 = [4, 5, 6];
let newArr = arr1.concat(arr2);
console.log(newArr); // [1, 2, 3, 4, 5, 6]
```

Método JOIN

Une todos los elementos del array en una cadena de texto separada por un separador especificado.

```
let arr = ['apple', 'banana', 'orange'];
let str = arr.join(',');
console.log(str); // 'apple, banana, orange'
```

Método INDEXOF

Devuelve el índice del primer elemento encontrado en el array, o -1 si no lo encuentra.

```
let arr = ['apple', 'banana', 'orange'];
let index = arr.indexOf('banana');
console.log(index); // 1
```

Métodos más usados Array

Método	Definición
forEach()	Ejecuta una función para cada elemento del array.
map()	Crea un nuevo array con los resultados de llamar a una función para cada elemento del array.
filter()	Crea un nuevo array con todos los elementos que cumplen una condición especificada en una función.
reduce()	Reduce el array a un único valor mediante la ejecución de una función para cada elemento del array.

Método FOREACH

Ejecuta una función para cada elemento del array.

```
let arr = [1, 2, 3];
arr.forEach(function(element) {
  console.log(element); }))
```

Método MAP

Crea un nuevo array con los resultados de llamar a una función para cada elemento del array.

```
let arr = [1, 2, 3];
let mappedArr = arr.map(function(element) {
  return element * 2; });
console.log(mappedArr); // [2, 4, 6]
```

Método FILTER

Crea un nuevo array con todos los elementos que cumplan una condición especificada en una función.

```
let arr = [1, 2, 3, 4, 5];
let filteredArr = arr.filter(function(element) {
  return element % 2 === 0; });
console.log(filteredArr); // [2, 4]
```

Método REDUCE

Reduce el array a un único valor mediante la ejecución de una función para cada elemento del array.

```
let arr = [1, 2, 3, 4, 5];
let sum = arr.reduce(function(accumulator, currentValue) {
  return accumulator + currentValue; });
console.log(sum); // 15
```

Método REDUCE

Recibe dos argumentos: el acumulador (que al principio es igual al primer elemento del array) y el valor actual (que va tomando cada elemento del array en cada iteración).

Método REDUCE

```
let arr = [5, 10, 2, 8, 3];
let maxNumber = arr.reduce(function(accumulator, currentValue) {
  return Math.max(accumulator, currentValue);
});
console.log(maxNumber); // 10
```

Método REDUCE

En cada iteración, la función reduce() va sumando el acumulador (que al principio es igual al primer elemento del array) con el valor actual (que va tomando cada elemento del array en cada iteración). Al final, el método reduce() devuelve la suma total de los elementos del array.

Método REDUCE

```
let arr = [5, 10, 2, 8, 3];
let sum = arr.reduce(function(accumulator, currentValue) {
  return accumulator + currentValue;
});
let average = sum / arr.length;
console.log(average); // 5.6
```

Ejercicios