

JAVASCRIPT

1. \ sirve para incluir unas comillas dentro un texto
Ejemplo: "Hola \"Comillas\"" -> Hola "Comillas"
2. Se puede mezclar comillas simples con dobles
Ejemplo: 'Hola "Comillas"' -> Hola "Comillas"
Ejemplo: "Hola 'Comillas'" -> Hola 'Comillas'
3. Secuencias de escape

Código	Resultado
\'	Comilla Simple
\"	Comilla Doble
\\	Barra invertida
\n	Línea Nueva
\r	Retorno de Carro
\t	Tabulación
\b	Retroceso
\f	Salto de Página

4. length-> se usa para saber el numero de caracteres de una cadena
5. Ejemplo: var a = "Hola";
console.log(a[1]) -> o
console.log(a[3]) -> a
6. Crear Arreglo
Ejemplo: var miArreglo= ["hola",24]
7. Modificar un arreglo
Ejemplo: var a=[10,20,30] ;
a[0] = 40; a-> [40,20,30]

Ejemplo: var a=[10,20,30] ;
a[1] = [1,2,3]; a-> [10, [1,2,3],30]
8. Arreglos multidimensionales
Ejemplo: var a = [[1,2,3], [4,2,6]]
a[1] -> [4,2,6]
a[0][1] -> 2
9. push -> Sirve para adicionar un elemento al final de un objeto
Ejemplo: var a = ["a","b"];
a.push("c"); a-> ["a","b","c"]
10. pop -> Sirve para remover el último elemento de un objeto
Ejemplo: var a = ["a","b"];
a.pop(); a-> ["a"]

Ejemplo: var a = ["a","b"];
var b;
b = a.pop(); b-> ["b"]
11. shift() -> Sirve para remover el primer elemento de un objeto
Ejemplo: var a = ["a","b"];
a.shift(); a-> ["b"]

12. unshift() -> Sirve para adicionar un elemento al principio de un objeto

Ejemplo: var a = ["a","b"];
a.unshift("c"); a-> ["c","a","b"]

13. Trabajar con funciones

Ejemplo: function mostrarMensaje(){
 Console.log("Hola");
}

Ejemplo: function sumar(a,b){
 var suma = a + b;
 Console.log(suma);
}
sumar(2,3); -> 5

Ejemplo: function concatenar(cad1,cad2){
 Console.log(cad1+"-"+cad2);
}
concatenar("Hi","Carlos"); -> Hi-Carlos

Ejemplo: function sumar(a,b){
 return a + b;
}
console.log(sumar(2,3)); -> 5

14. JSON.stringify() -> sirve para mejorar la presentación de arreglos

15. Igualdad -> se hace con ==

Ejemplo: console.log(5==5); -> true
Ejemplo: console.log([1,2]== [1,2]); -> false
NOTA: No sirve para comparar arreglos.

16. Comparación estricta -> se hace con ===

Ejemplo: console.log(5=="5"); -> true
Ejemplo: console.log(5==="5"); -> false
NOTA: Tiene que ser el mismo valor y tipo de dato.

17. Comparador desigualdad -> se hace con !=

Ejemplo: console.log(9!=9); -> false

18. Comparador desigualdad estricto -> se hace con !==

Ejemplo: console.log(9!="9"); -> false
Ejemplo: console.log(9!==9); -> true

19. Comparador Mayor

```
// console.log(6 > 5);    // true
// console.log(3 > 10);   // false

// console.log("B" > "A");    // true
// console.log("ACB" > "ABC"); // true

// console.log("AB" > "A"); // true

// console.log("Mundo" > "Hola"); // true
// console.log("M" > "H");    // true
```

20. Comparadores && y ||

X	Y	X && Y
T	T	T
T	F	F
F	T	F
F	F	F

y

X	Y	X Y
T	T	T
T	F	T
F	T	T
F	F	F

21. Comparador Not

Para: !X

X	!X
T	F
F	T

*/

```
console.log(!true); // false
```

22. If condicional

```
Ejemplo: if(true){
    console.log("1");
} else{
    console.log("2");
}
```

La salida es "1"

23. Switch

Para un valor numerico

```
Ejemplo: switch (valor) {
    case 1:
        respuesta = "uno"
        break;
    case 2:
        respuesta = "dos"
        break;
    case 3:
        respuesta = "tres"
        break;
}
```

Para un valor string y usamos "default" que sirve si no se cumplen ninguno de los casos anteriores

```
Ejemplo: switch (valor) {
    case "1":
        respuesta = "uno"
        break;
    case "2":
        respuesta = "dos"
        break;
    default:
        respuesta = "tres"
        break;
}
```

Para un valor numerico y tomar dos números a la vez

Ejemplo: switch (valor) {

```
    case 1:
        respuesta = "uno"
        break;
    case 2:
    case 3:
        respuesta = "dosytres"
        break;
}
```

24. Declarar un objeto, las propiedades se pueden dejar sin comillas si toda la palabra es pegada.

Ejemplo: var miPerro={

```
    "nombre": "Marcuz",
    "edad": 7,
    "peso": "29kg",
    "Ciudad de Nac.": "Cucuta"
};
```

console.log(miPerro.nombre)-> Marcuz

console.log(miPerro["Ciudad de Nac."])-> Cucuta

miPerro.edad = 8 //Actualizamos el valor de 7 a 8

console.log(miPerro.edad)-> 8

Ejemplo: var mochila = {

```
    color: "rojo",
    tamaño: "grande",
    contenido: ["lapiz", "cuaderno"]
}
```

console.log(mochila.contenido); -> ['lapiz','cuaderno']

mochila.contenido.push("hoja");

console.log(mochila.contenido); -> ['lapiz','cuaderno','hoja']

mochila.contenido= []; →Vaciamos el arreglo

mochila.marca = "totto" →Agregamos una propiedad

delete mochila.marca; →Borramos la propiedad marca

25. Uso de un objeto para una búsqueda

```
function buscarElementoQuimico(simbolo) {
    var simbolosQuimicos = {
        "Al": "Aluminio",
        "S": "Azufre",
        "Cl": "Cloro",
        "He": "Helio",
        "B": "Boro",
        "Li": "Litio"
    };

    return simbolosQuimicos[simbolo];
}
```

```
console.log(buscarElementoQuimico("Al"));
```

→ Aluminio

26. hasOwnProperty -> Este método se usa para saber si un objeto contiene una propiedad.

```
var mochila = {  
  color: "rojo",  
  tamaño: "grande",  
  contenido: ["lapiz", "cuaderno"]  
}  
  
mochila.hasOwnProperty("color"); // Retorna true
```

27. Declaración de un arreglo de objetos

```
Ejemplo: var animales=[{color:"rojo",edad:29},{ color:"verde",edad:15}];  
Console.log(animales[0].color) -> rojo
```

28. Objeto anidado

```
Ejemplo: var empleado = {  
  nombre: "carlos",  
  edad: 28,  
  origen: {  
    ciudad: "bogota",  
    departamento: "cundinamarca"  
  }  
}  
  
console.log(empleado.origen.ciudad); -> bogota
```

29. Ciclo While

```
var i = 0;  
  
while (i <= 3) {  
  console.log("Hola, Mundo");  
  i++;  
}
```

30. Ciclo For

```
1 var miArreglo = [];  
2  
3 for (var i = 0; i < 10; i++) {  
4   miArreglo.push(i);  
5 }  
6  
7 console.log(miArreglo);
```

```
[ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 ]
```

31. toUpperCase -> Deja todo el texto en mayúsculas

32. toLowerCase -> Deja todo el texto en mayúsculas

33. Do While -> se ejecuta el código por lo menos una sola vez así la condición sea falsa.

```
1 var x = 16;  
2  
3 do {  
4   console.log(x);  
5   x++; // 17  
6 } while (x < 10);  
7
```

34. Para retornar los datos de una propiedad en una búsqueda usamos el siguiente ejemplo.

```
var contactos = [
  {
    "nombre": "Nora",
    "apellido": "Nav",
    "numero": "0543236543",
    "gustos": ["Pizza", "Programación"]
  },
  {
    "nombre": "Harry",
    "apellido": "Potter",
    "numero": "0994372684",
    "gustos": ["Hogwarts", "Magia"]
  },
]

function buscarPerfil(nombre, propiedad) {
  for (var i = 0; i < contactos.length; i++) {
    if (contactos[i].nombre === nombre) {
      return contactos[i][propiedad] || "La propiedad no existe.";
    }
  }
  return "El contacto no está en la lista.";
}

console.log(buscarPerfil("Nora", "direccion"));
```

-> La propiedad no existe.

35. Math.random() -> Retorna un número aleatorio entre 0 y 1, sin incluir el 1.

36. Math.floor() -> Retorna la parte entera por debajo de un numero con decimales.

37. parseInt() -> Convierte una cadena de caracteres a un entero. En caso de número con decimales trunca la parte decimal ósea la elimina.

38. If de una sola línea

Ejemplo: `x < y ? x : y;`

39. Condicionales anidados

```
function compararNumeros(a, b) {
  return a == b ? "a y b son iguales"
    : a > b ? "a es mayor que b"
    : "b es mayor que a";
}
```

40. var y let son dos formas de declarar un variable, la diferencia es que var se puede declarar varias veces, en cambio let solo se puede una única vez. Otra diferencia es que si declaramos una variable dentro de un ciclo o un condicional, si la declaramos como var se puede usar fuera de el ciclo o la condición, pero si la declaramos como let no la podemos usar fuera del ciclo o la condición.

41. Const -> Sirve para declarar una variable e inicializarla en una sola línea, pero después no se puede cambiar su valor.

42. ** -> eleva un número al cuadrado

43. Única forma de modificar un objeto declarado tipo const

```
const MI_ARREGLO = [1, 2, 3, 4];

MI_ARREGLO[0] = 5;
MI_ARREGLO[1] = 6;
MI_ARREGLO[2] = 7;
MI_ARREGLO[3] = 8;
```

44. Object.freeze() -> Sirve para evitar que se pueda modificar un objeto.

45. concat -> Sirve para concatenar texto, arreglos

46. Funciones flecha

Ejemplo: `const sumarTres = (x) => x + 3;`
`console.log(sumarTres(4));` -> 7

Ejemplo: `const concatArray = (arr1,arr2) => arr1.concat(arr2);`
`console.log(concatArray([1,2],[3,4,5]));` -> [1,2,3,4,5]

Ejemplo: `const sumar = (a,b) => {`
 `let num = 6;`
 `return a + b + num;`
 `};`
`console.log(sumar(1,1));` -> 8

47. Valor por defecto

```
const incrementar = (num, valor = 1) => num + valor;
```

```
console.log(incrementar(5));
```

 -> 6

48. Funcion rest -> se usa para pasar los parámetros que uno desee sin necesidad de declarar los en la función inicial.

```
function miFuncion(...args) {  
  console.log(args);  
}
```

```
miFuncion(1, 2, 3, 4);
```

 -> [1,2,3,4]

49. reduce() -> Método para sumar los elementos de un arreglo usando una función flecha.

```
const array1 = [1, 2, 3, 4];  
  
// 0 + 1 + 2 + 3 + 4  
const initialValue = 0;  
const sumWithInitial = array1.reduce(  
  (accumulator, currentValue) => accumulator + currentValue,  
  initialValue  
);  
  
console.log(sumWithInitial);  
// Expected output: 10
```

50. spread -> Método que sirve para descomponer un arreglo en valores individuales.

```
const numeros = [1, 2, 3];
```

```
function sumar(x, y, z) {  
  return x + y + z;  
}
```

```
console.log(sumar(...numeros));
```

 -> 6

51. Des Estructuración -> Sirve para asignar los valores de un objeto a unas constantes en una sola línea, en vez de hacerlo uno a uno

Ejemplo 1:

```
var coordenadas = {  
  x: 4,  
  y: 6,  
  z: 12  
};  
  
const {x, y, z} = coordenadas;
```

Ejemplo 2:

```
const usuario = {  
  johnDoe: {  
    edad: 27,  
    correo: "johnDoe@freecodecamp.com"  
  }  
};
```

```
const {johnDoe: {edad, correo}} = usuario;
```

Ejemplo 3: Se puede modificar los nombres de las variables

```
const usuario = {  
  johnDoe: {  
    edad: 27,  
    correo: "johnDoe@freecodecamp.com"  
  }  
};
```

```
const {johnDoe: {edad: edadDelUsuario, correo: correoDelUsuario}} = usuario;
```

Ejemplo 4: Las dos penúltimas líneas nos muestran las dos formas de des estructurar un objeto, no se pueden aplicar las dos al tiempo.

```
const maleta = {  
  "carlos":{  
    libro: "harry potter",  
    peso: 25 },  
  "erika":{  
    libro: "eclipse",  
    peso: 12 }  
}  
const nombreLibro = maleta.erika.libro;  
const {erika:{libro: nombreLibro}}= maleta;  
console.log(nombreLibro);
```


Ejemplo 5: las comas dentro del arreglo también se pueden poner al principio para eliminar los primeros números.

```
1 var a;  
2 var b;  
3 var c;  
4 [a,b,,c]=[1,2,3,4,5,6,7]  
5 console.log(a,b,c)
```

1 2 4

Ejemplo 6:

```
1 var a;  
2 var b;  
3 var arr;  
4 [a,b,...arr]=[1,2,3,4,5,6,7]  
5 console.log(a,b,arr)
```

1 2 [3, 4, 5, 6, 7]

Ejemplo 7:

```
const estadisticas = {  
  max: 56.78,  
  desviacionEstandar: 4.34,  
  mediana: 34.54,  
  moda: 23.87,  
  min: -0.75,  
  promedio: 35.85  
};
```

```
const mitad = ({max, min}) => (max + min) / 2.0;
```

```
console.log(mitad(estadisticas));
```

.015

52. backtick -> Se usa comillas invertidas para colocar variables dentro de un texto

Ejemplo: var variable = 5

Console.log(`El valor es \${variable}`); -> El valor es 5

Ejemplo: var persona = { nombre: "carlos e", edad: 39};

const saludo = `Hola mi nombre es \${persona.nombre}`;

53. Crear un objeto conciso

```
const crearPersona = (nombre, edad, idioma) => {  
  return {  
    nombre: nombre,  
    edad: edad,  
    idioma: idioma  
  };  
};
```

```
console.log(crearPersona("Gino Smith", 28, "Español"));
```

Ó También se puede declarar de esta manera

```
const crearPersona = (nombre, edad, idioma) => ({nombre, edad, idioma});
```

54. Creación de objetos con métodos incluidos.

```
const persona = {  
  nombre: "Isabel",  
  presentarse: function() {  
    return `¡Hola! Mi nombre es ${this.nombre}.`;  
  }  
};
```

```
console.log(persona.presentarse());
```

Son iguales pero con menos código

```
const persona = {  
  nombre: "Isabel",  
  presentarse() {  
    return `¡Hola! Mi nombre es ${this.nombre}.`;  
  }  
};
```

```
console.log(persona.presentarse());
```

55. Para creación de clases los nombres deben iniciar con letras mayúsculas.

```
1  class Mascota {  
2    constructor(nombre, edad) {  
3      this.nombre = nombre;  
4      this.edad = edad;  
5    }  
6  }  
7  
8  var miMascota = new Mascota("Nora", 5);  
9  
10 console.log(miMascota.nombre);  
11 console.log(miMascota.edad);  
12  
13
```

Nora

5

56. Trabajando con get y set

```
class Libro {  
  constructor(autor) {  
    this._autor = autor;  
  }  
  
  get autor() {  
    return this._autor;  
  }  
  
  set autor(nuevoAutor) {  
    this._autor = nuevoAutor;  
  }  
}  
  
const libro = new Libro("anónimo");  
console.log(libro.autor);  
  
libro.autor = "Gino Smith";  
console.log(libro.autor);
```