

Curso Complementario: JavaScript

Instructor: Jorge E. Andrade C.

Correo: [jandradec@sena.edu.co](mailto:jandradec@sena.edu.co)

SENA Centro Agropecuario La Granja

Test de conocimientos previos

El siguiente es un operador de asignación:

- a. \*
- b. +
- c. =
- d. >

La siguiente es una variable tipo entera:

- a. "25"
- b. 25
- c. Veinticinco
- d. 25.0

La siguiente es una variable tipo cadena de texto:

- a. "25"
- b. 25
- c. Veinticinco
- d. 25.0

El siguiente es un operador de comparación:

- a. =
- b. >
- c. \*
- d. %

La siguiente instrucción asigna el valor 2 a una variable:

- a. variable equal 2
- b. variable is 2
- c. variable ++
- d. variable = 2

2 elevado al cuadrado es:

- a. 2

- b. 4
- c. 6
- d. 8

3 elevado al cubo es:

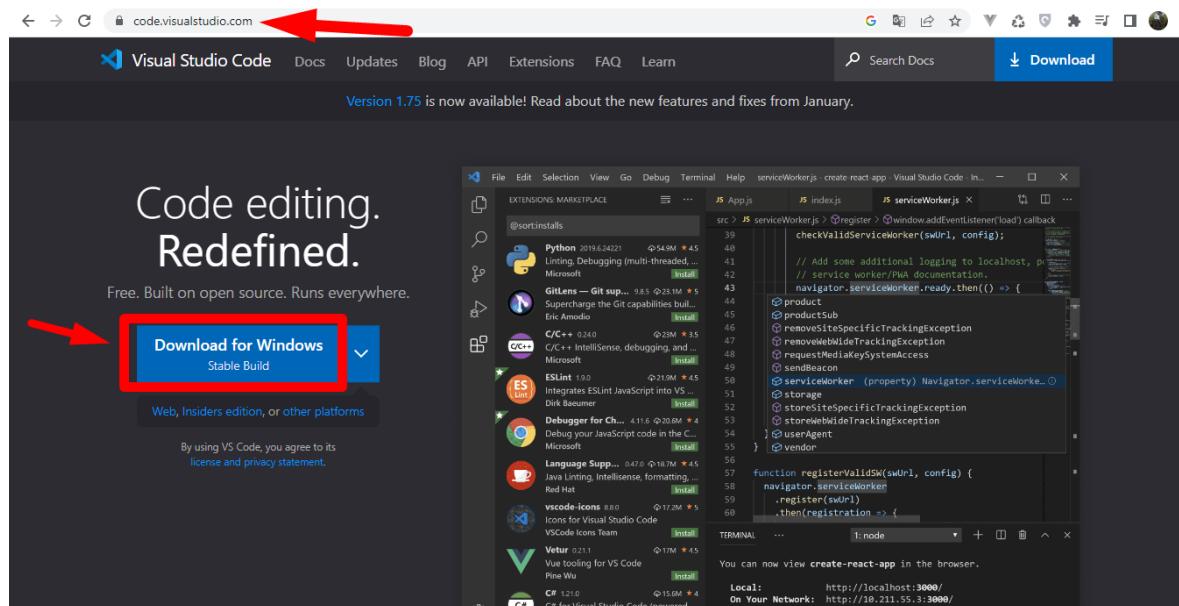
- a. 6
- b. 9
- c. 18
- d. 27

¿Qué es JavaScript?

Teniendo en cuenta la definición por AWS JavaScript es un lenguaje de programación que los desarrolladores utilizan para hacer páginas web interactivas. Desde actualizar fuentes de redes sociales a mostrar animaciones y mapas interactivos, las funciones de JavaScript pueden mejorar la experiencia del usuario de un sitio web. Como lenguaje de scripting del lado del servidor, se trata de una de las principales tecnologías de la World Wide Web. Por ejemplo, al navegar por Internet, en cualquier momento en el que vea un carrusel de imágenes, un menú desplegable “click-to-show” (clic para mostrar), o cambien de manera dinámica los elementos de color en una página web, estará viendo los efectos de JavaScript.

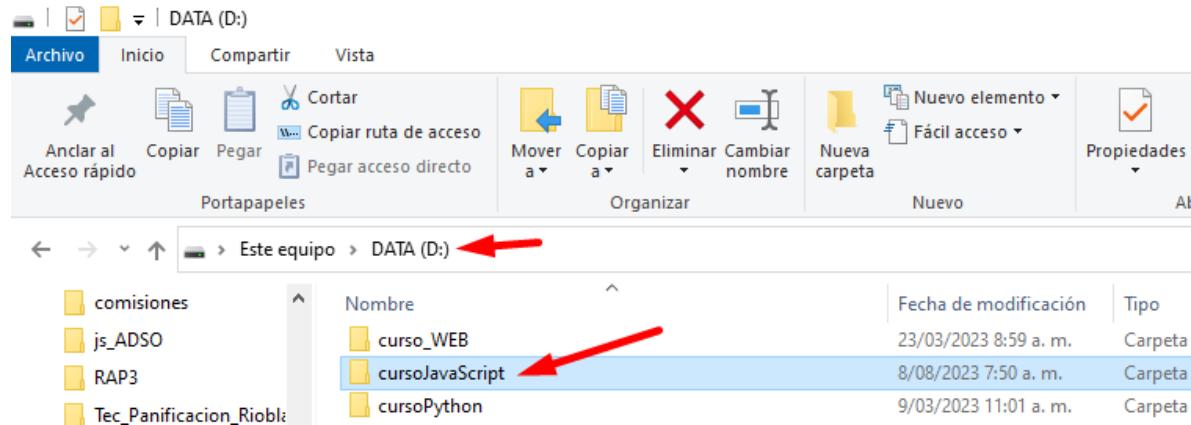
## Editor de código para programar en JavaScript

Para programar sitios web se puede utilizar desde el bloc de notas en adelante, sin embargo, para este tutorial se hará uso del IDE (Entorno de Desarrollo Integrado) Visual Studio Code (VSC), el cual se puede descargar del sitio <https://code.visualstudio.com/>

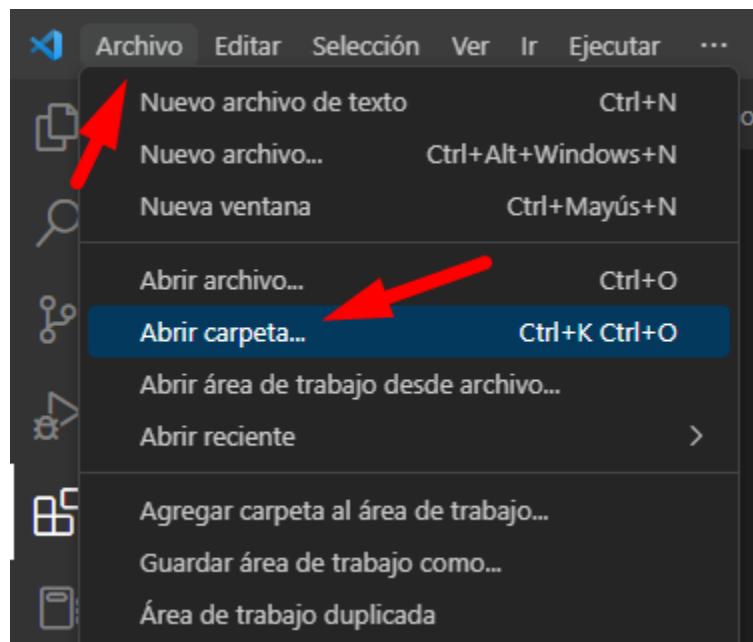


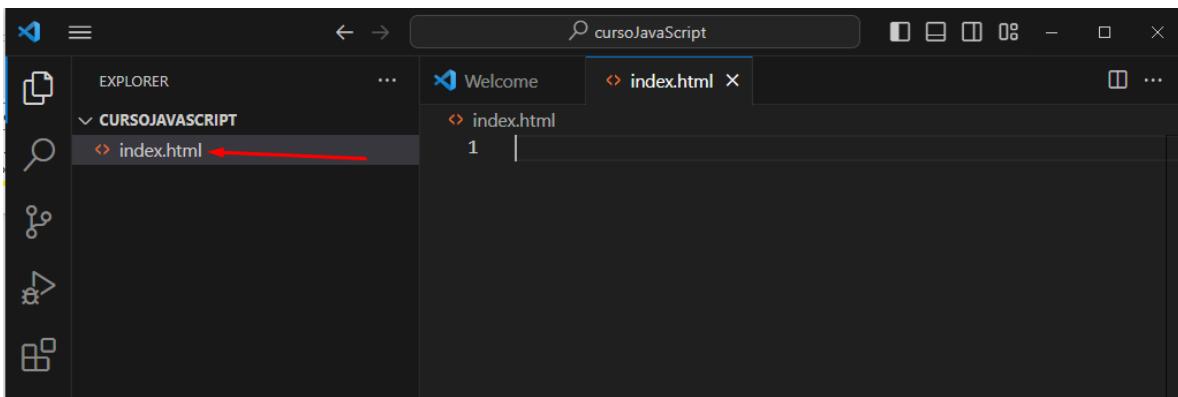
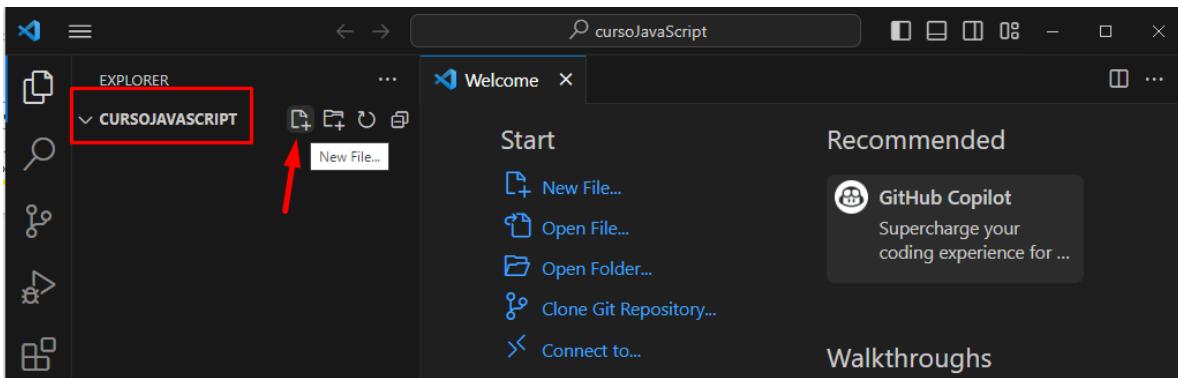
Instructor Jorge E. Andrade C.  
 jandradec@sena.edu.co  
 Centro Agropecuario La Granja  
 Regional Tolima

Crear una carpeta de trabajo, puede ser en cualquier lugar del disco duro, en este caso se crea en el disco D con el nombre de cursoJavaScript:



Abrir carpeta con el editor Visual Studio Code y crear un archivo llamado index.html





En el archivo crear la estructura básica de HTML (html, head y body) e incluir al final, antes de terminar el body, la etiqueta <script text="text/javascript"></script> la cual permite incluir JavaScript en la página web:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
<script text="text/javascript">
</script>
</body>
</html>
```

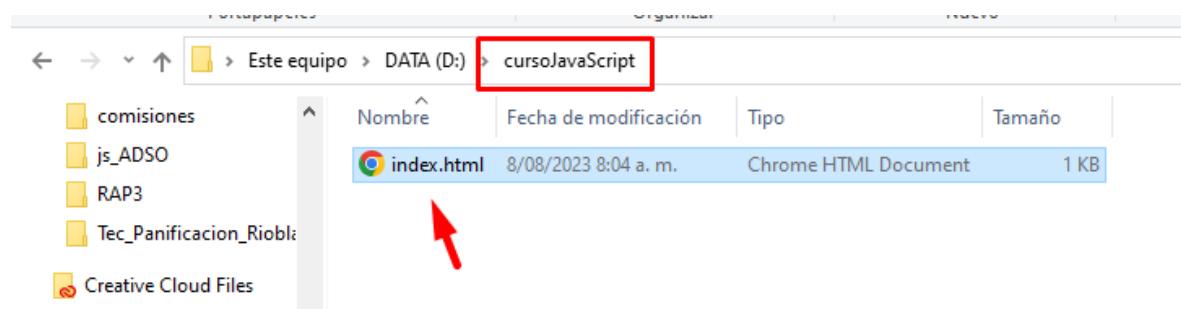
Primer línea de código en JS: imprimir en consola un mensaje con la función console.log(mensaje)

```

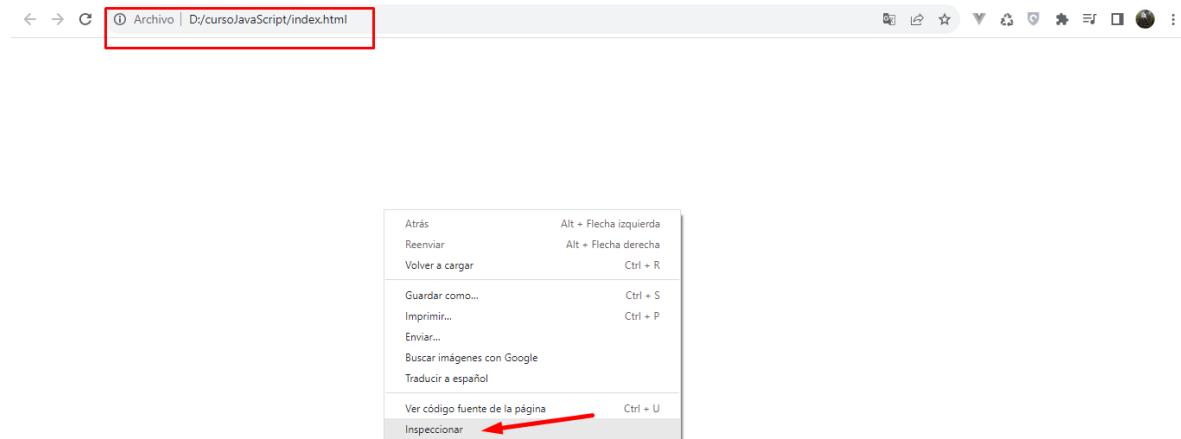
8 <body>
9
10
11   <script text="text/javascript">
12     |   console.log("Hola mundo")
13   </script>
14 </body>
15 </html>

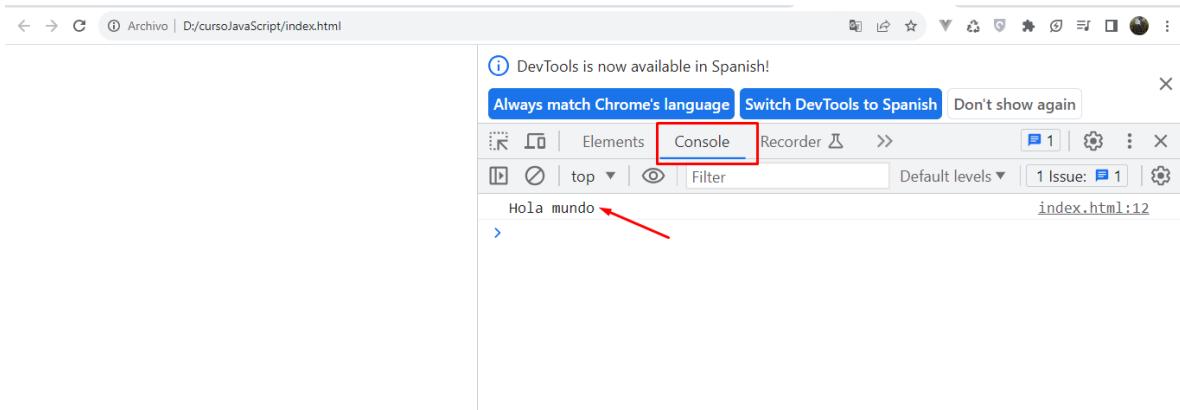
```

Para ver el resultado, es necesario abrir la página en el navegador, para ello, en el explorador de Windows abra la carpeta y de doble clic en el archivo index.html



La página se abrirá totalmente en blanco, sin embargo, al dar clic derecho y seleccionar la opción **inspeccionar**, el navegador mostrará información detallada sobre la página, allí seleccione la opción de **Console** y podrá ver la ejecución del código JS:





### Comentarios:

Los comentarios son mensajes que el programador escribe en lenguaje natural para mencionar algo en relación con el código que está digitando, estos comentarios no hacen parte del programa y el navegador no los ejecuta como código JavaScript:

El comentario para una línea se realiza con //

El comentario para un bloque (varias líneas) se abre con /\* y se cierra con \*/

```
8 <body>
9
10 <script text="text/javascript">
11   console.log("Hola mundo")
12
13 //Este es un comentario de línea
14
15 /*
16   Este
17   es
18   un
19   comentario
20   de
21   bloque
22 */
23 </script>
24 </body>
```

The code block shows a script tag within a body tag. Line 13 contains a single-line comment starting with //. Lines 15 through 22 are enclosed in a multi-line comment block starting with /\* and ending with \*/. Both the single-line and multi-line comments are highlighted with red boxes.

## Variables y tipos de variables

- string – cadena de texto
- number – numero
- boolean – booleano (verdadero o falso)
- object – Objeto
- function -funciones
- null – nulo
- undefined – valor sin definir

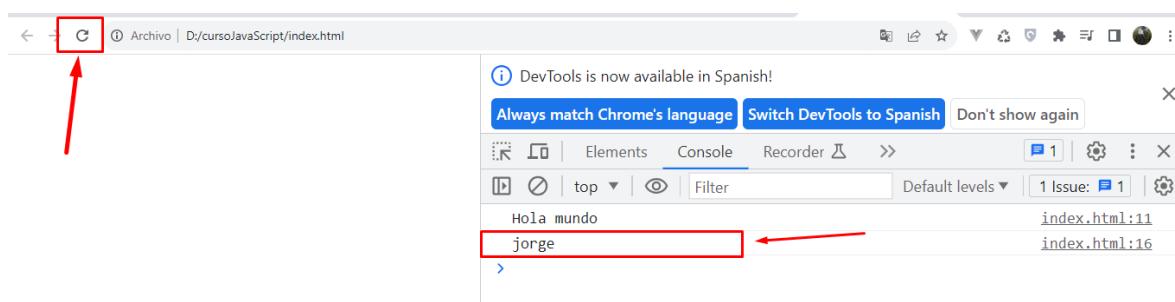
Digitar el siguiente código en el mismo archivo **index.html**, tenga en cuenta los comentarios

```

10   <script text="text/javascript">
11     console.log("Hola mundo")
12
13 //Ejercicio 1
14 var nombre //Definición de la variable
15 nombre = 'jorge' //Asignación de valor tipo String
16 console.log(nombre) //Imprimir variable

```

Para ver el resultado basta con actualizar la página en el navegador:



Cuando se imprimen en la consola, **se muestra el contenido más NO el nombre de la variable**.

Digitar el siguiente código para ver los tipos de variable:

```

18   //Variable tipo number
19   var numero
20   numero = 9
21   console.log(numero)
22
23   //variable tipo boolean
24   var mayorDeEdad
25   mayorDeEdad = true
26   console.log(mayorDeEdad)

```

## Resultado en consola:

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The output area displays the following text:  
jorge  
9  
true

### Tipo arreglo:

```
28 //variable tipo objeto arreglo  
29 var objetoArreglo  
30 objetoArreglo = [2, 4, 6]  
31 console.log(objetoArreglo)
```

## Resultado:

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The output area displays the following log entries:

- jorge
- 9
- true
- ▶ (3) [2, 4, 6] ←

A red arrow points to the last entry, indicating the current selection or focus.

Al abrir el arreglo dando clic en la pestaña (triángulo) se observan los detalles:

```
▼ (3) [2, 4, 6] i
  0: 2
  1: 4
  2: 6
  length: 3
▶ [[Prototype]]: Array(0)
```

## Tipo Objeto (JSON)

```

33 //variable tipo objeto (JSON)
34 var objetoJSON = {
35     'nombre':'fulanito',    //estructura clave:valor
36     'edad':30,
37     'estatura': 1.78
38 }
39 console.log(objetoJSON)

```

DevTools is now available in Spanish!

[Always match Chrome's language](#) [Switch DevTools to Spanish](#) [Don't show again](#)

Elements Console Recorder > Default levels 1 Issue: 1 | [Filter](#)

```

jorge index.html:16
9 index.html:21
true index.html:26
▶ (3) [2, 4, 6] index.html:31
▶ {nombre: 'fulanito', edad: 30, estatura: 1.78} index.html:39

```



Al abrir el objeto se muestran los detalles:

DevTools is now available in Spanish!

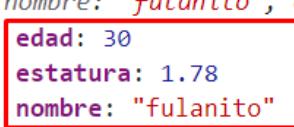
[Always match Chrome's language](#) [Switch DevTools to Spanish](#) [Don't show again](#)

Elements Console Recorder > Default levels 1 Issue: 1 | [Filter](#)

```

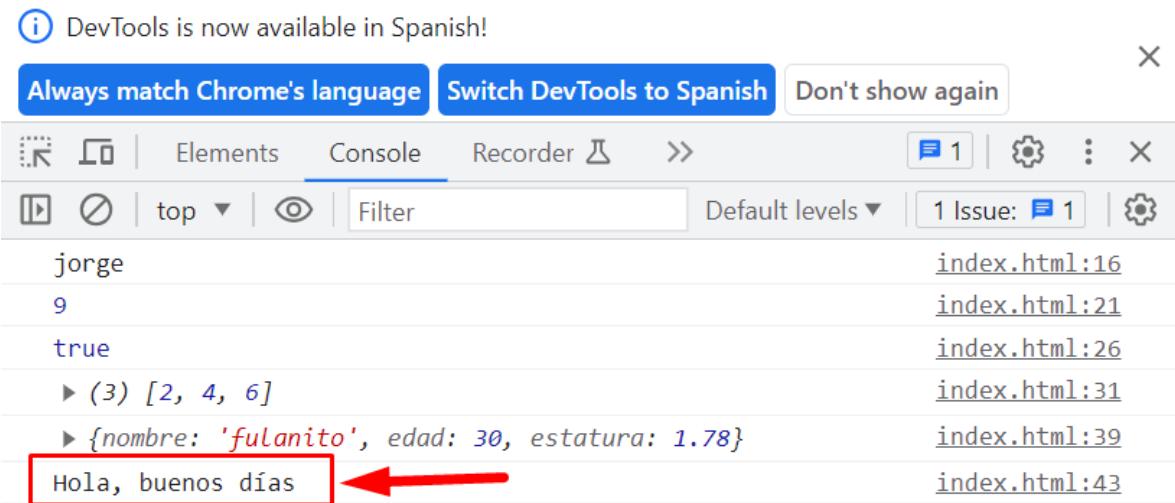
jorge index.html:16
9 index.html:21
true index.html:26
▶ (3) [2, 4, 6] index.html:31
▼ {nombre: 'fulanito', edad: 30, estatura: 1.78} ⓘ index.html:39
  ▶ edad: 30
  ▶ estatura: 1.78
  ▶ nombre: "fulanito"
▶ [[Prototype]]: Object

```



Tipo función:

```
41     //variable tipo función
42     var saludo = function(){
43         console.log("Hola, buenos días")
44     }
45     saludo()
```



## Tipo null e indefinida

```
47 //variable tipo null  
48 var dato  
49 dato = null  
50 console.log(dato)  
  
52 //variable indefinida  
53 var tomate  
54 console.log(tomate)
```

ⓘ DevTools is now available in Spanish!

Always match Chrome's language **Switch DevTools to Spanish** Don't show again

Elements Console Recorder > Default levels ▾ 1 Issue: 1 | Filter

jorge	<a href="#">index.html:16</a>
9	<a href="#">index.html:21</a>
true	<a href="#">index.html:26</a>
▶ (3) [2, 4, 6]	<a href="#">index.html:31</a>
▶ {nombre: 'fulanito', edad: 30, estatura: 1.78}	<a href="#">index.html:39</a>
Hola, buenos días	<a href="#">index.html:43</a>
null	<a href="#">index.html:50</a>
undefined	<a href="#">index.html:54</a>

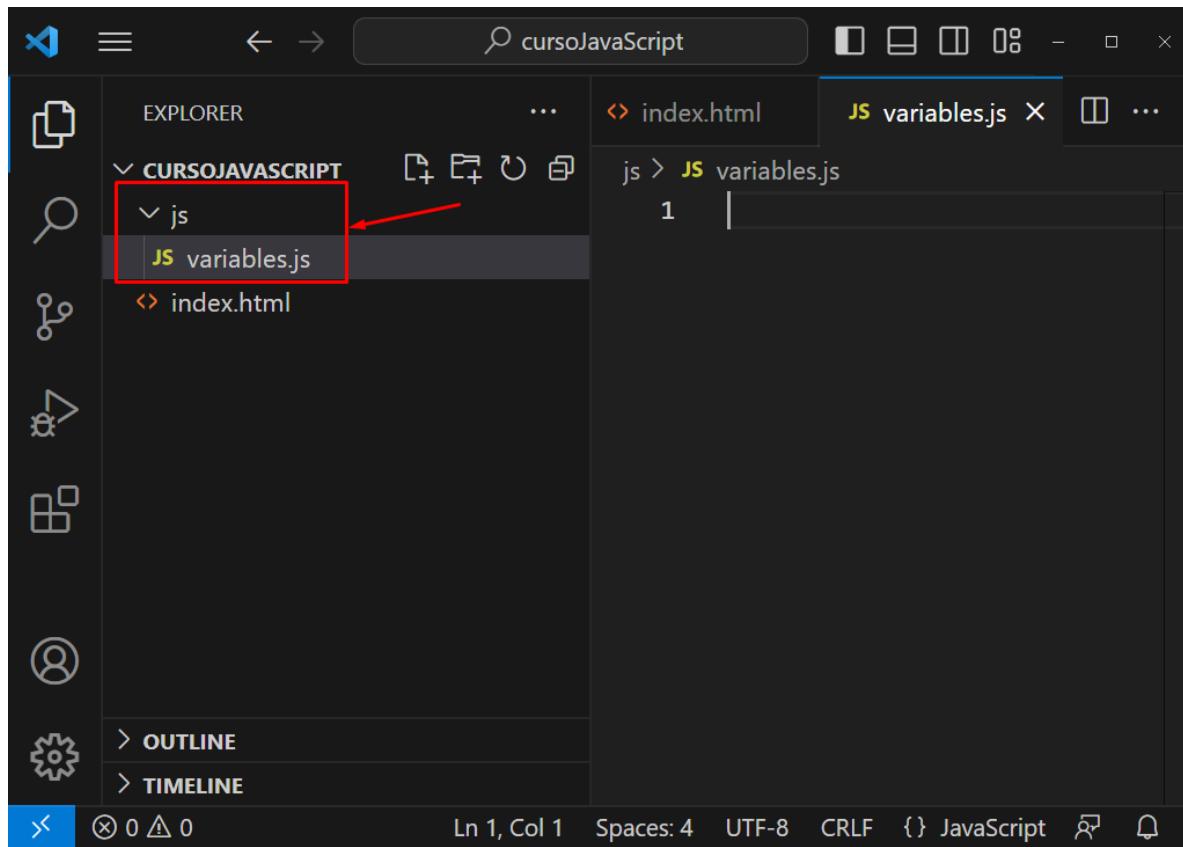
▶



## Archivo JS externo

La manera más organizada de trabajar con JS es dejando un archivo a parte solo con los scripts e invocarlo desde el archivo .html

Dentro de la carpeta cursoJavaScript crear una carpeta llamada js y allí crear el archivo variables.js



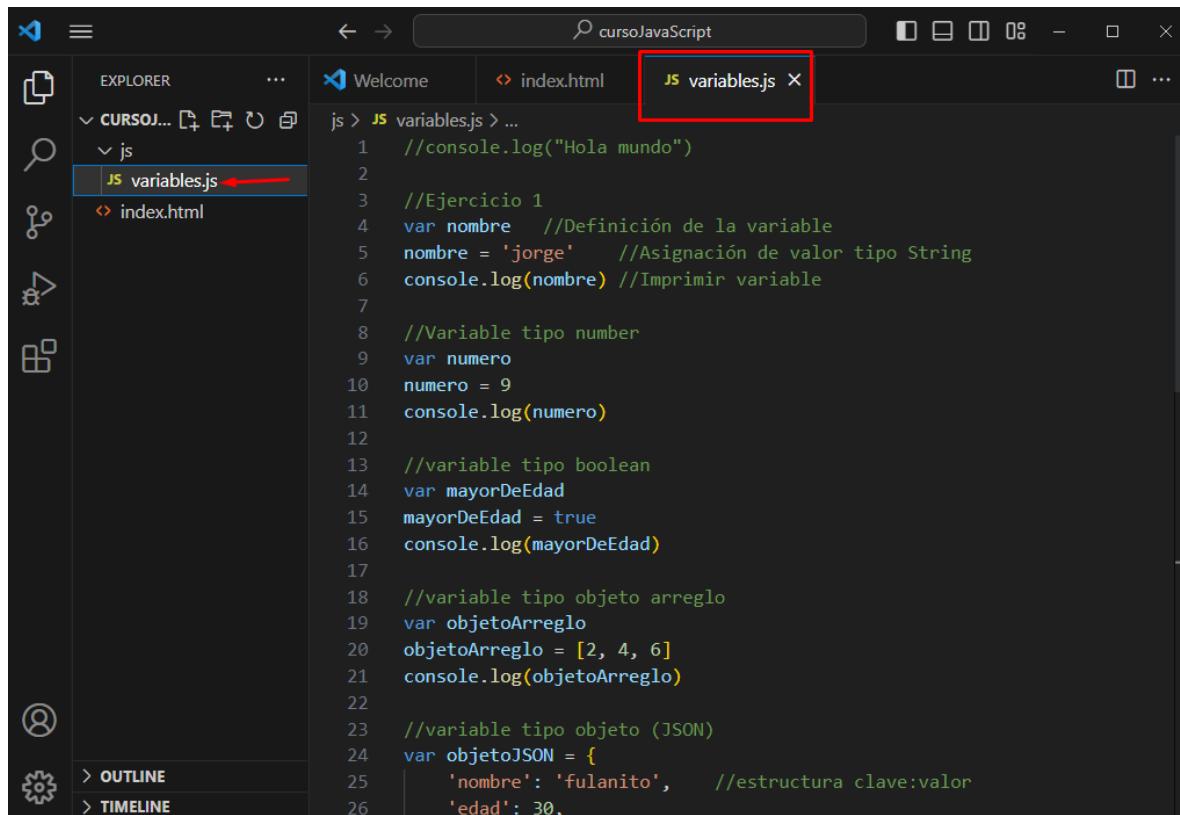
Corte y pegue todo el código que hay dentro de las etiquetas `<script></script>`

A screenshot of the Visual Studio Code (VS Code) interface. The left sidebar contains icons for file operations like Open, Save, Find, Replace, and others. The main area shows two tabs: 'index.html' and 'JS variables.js'. The 'variables.js' tab is active, displaying a script with several code snippets. A context menu is open over the line 'var nombre = 'jorge'' at line 15. The menu items are:

- Go to Definition F12
- Go to References Shift+F12
- Peek >
- Find All References Shift+Alt+F12
- Rename Symbol F2
- Change All Occurrences Ctrl+F2
- Format Document Shift+Alt+F
- Format Selection Ctrl+K Ctrl+F
- Refactor... Ctrl+Shift+R
- Cut Ctrl+X (highlighted)
- Copy Ctrl+C
- Paste Ctrl+V
- Command Palette... Ctrl+Shift+P

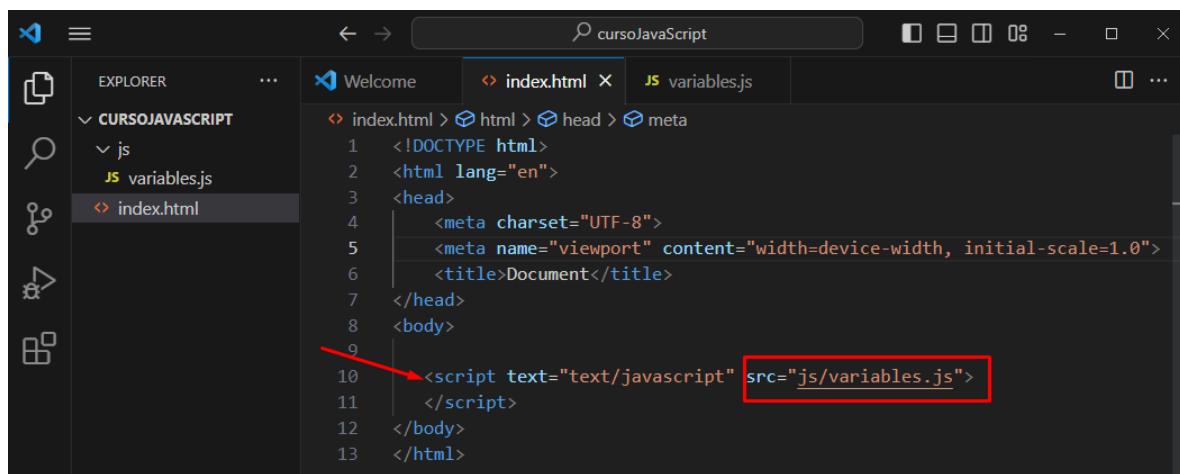
A red arrow points to the 'Cut' option in the menu.

```
<script text="text/javascript">
    //console.log("Hola mundo")
    //Ejercicio 1
    var nombre //Definición de la variable
    nombre = 'jorge' //Asignación de valor tipo String
    console.log(nombre) //Imprimir el valor de la variable
    //Variable tipo number
    var numero
    numero = 9
    console.log(numero)
    //variable tipo boolean
    var mayorDeEdad
    mayorDeEdad = true
    console.log(mayorDeEdad)
    //variable tipo objeto arreglo
    var objetoArreglo
    objetoArreglo = [2, 4, 6]
    console.log(objetoArreglo)
    //variable tipo objeto JSON
    var objetoJSON = {
        'nombre': 'fulanito',
        'edad': 30,
        'estatura': 1.78
    }
    console.log(objetoJSON)
    //variable tipo función
    var saludo = function(){
        console.log("Hola, buenos días")
    }
    saludo()
    //variable tipo null
    var dato
    dato = null
    console.log(dato)
    //variable indefinida
    var tomate
    console.log(tomate)
</script>
```



```
1 //console.log("Hola mundo")
2
3 //Ejercicio 1
4 var nombre //Definición de la variable
5 nombre = 'jorge' //Asignación de valor tipo String
6 console.log(nombre) //Imprimir variable
7
8 //Variable tipo number
9 var numero
10 numero = 9
11 console.log(numero)
12
13 //variable tipo boolean
14 var mayorDeEdad
15 mayorDeEdad = true
16 console.log(mayorDeEdad)
17
18 //variable tipo objeto arreglo
19 var objetoArreglo
20 objetoArreglo = [2, 4, 6]
21 console.log(objetoArreglo)
22
23 //variable tipo objeto (JSON)
24 var objetoJSON = {
25   'nombre': 'fulanito', //estructura clave:valor
26   'edad': 30,
```

En las etiquetas que quedaron en el archivo .html agregue la propiedad src y digite la ruta del archivo .js



```
<script text="text/javascript" src="js/variables.js">
```

Al actualizar la página no se verán cambios, si aparece un error en la consola del navegador es porque no direccionó correctamente el archivo .js

Otra manera de vincular el archivo .js en la página web es dejando la etiqueta <script> en medio de las etiquetas <head></head> e incluir la propiedad defer para que el navegador espere a que carguen los elementos de la página web (DOM, tema que se verá más adelante).

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7
8      <script defer text="text/javascript" src="js/variables.js"></script>
9
10 </head>
11 <body>
12
13
14 </body>
15 </html>

```

## Operadores matemáticos

Nombre	Operador	Descripción
Suma	a + b	Suma el valor de a al valor de b.
Resta	a - b	Resta el valor de b al valor de a.
Multiplicación	a * b	Multiplica el valor de a por el valor de b.
División	a / b	Divide el valor de a entre el valor de b.
Módulo	a % b	Devuelve el resto de la división de a entre b.
Exponenciación	a ** b	Eleva a a la potencia de b, es decir, $a^b$ . Equivalente a Math.pow(a, b).

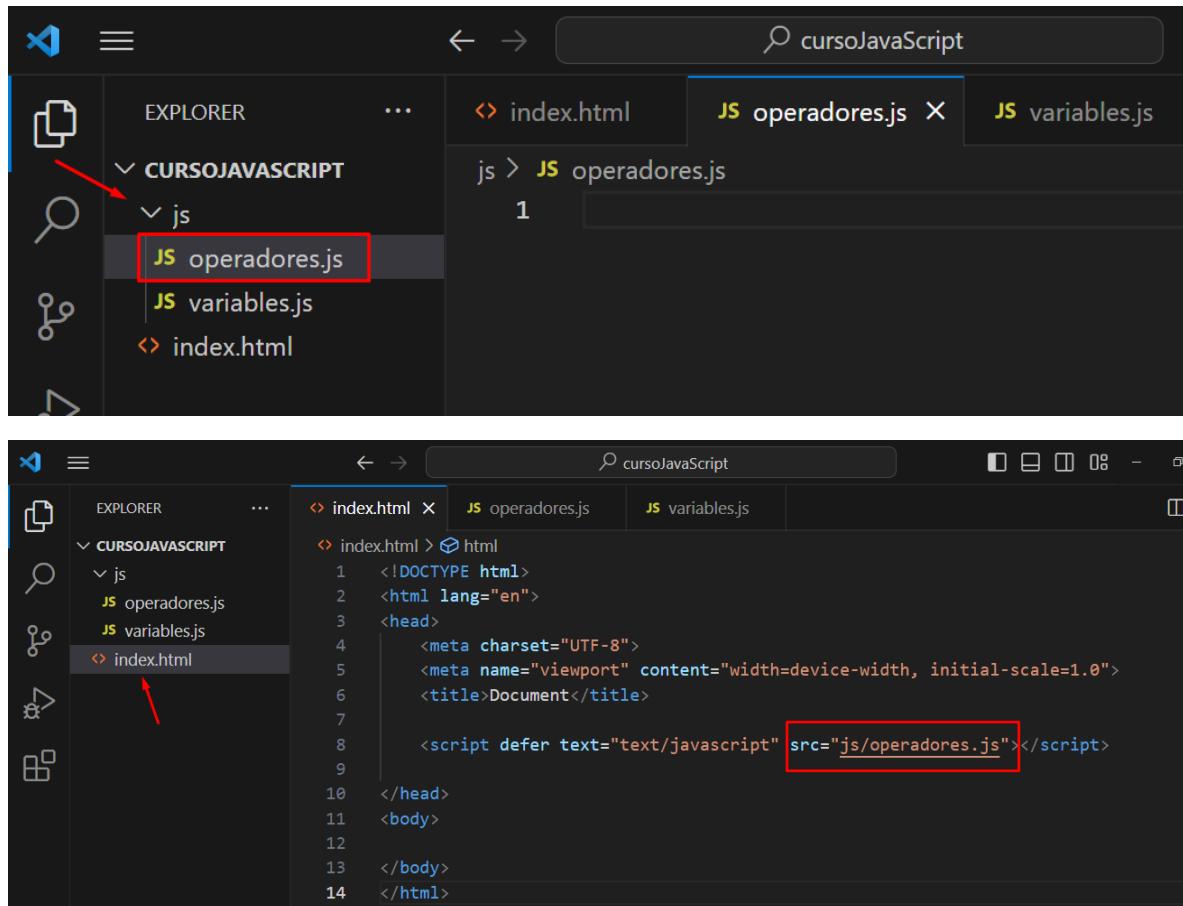
## Operadores unarios

Los operadores unarios son aquellos que, en lugar de tener dos operandos, como los anteriores, sólo tienen uno. Es decir, se realizan sobre un sólo valor almacenado en una variable.

Nombre	Operador	Descripción
Incremento	a++	Usa el valor de a y luego lo incrementa. También llamado <b>postincremento</b> .
Decremento	a--	Usa el valor de a y luego lo decrementa. También llamado <b>postdecremento</b> .
Incremento previo	++a	Incrementa el valor de a y luego lo usa. También llamado <b>preincremento</b> .
Decremento previo	--a	Decrementa el valor de a y luego lo usa. También llamado <b>predecremento</b> .
Resta unaria	-a	Cambia de signo (niega) a a.

Fuente: <https://lenguajejs.com/javascript/introduccion/operadores-basicos/>

Para poner en práctica el tema de los operadores, se creará dentro de la carpeta js un archivo llamado operadores.js y en el archivo index.html debe cambiar el nombre del archivo a enrutar:



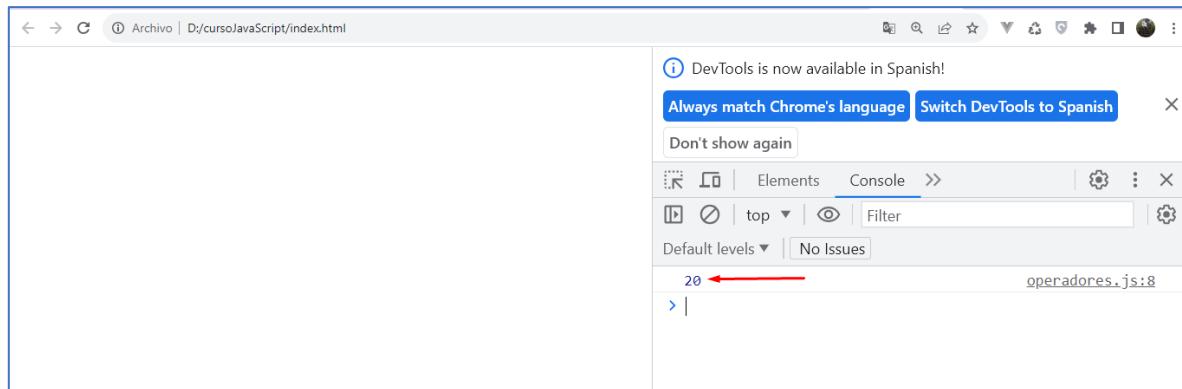
The screenshot shows two views of the VS Code interface. The top view is the Explorer sidebar, which lists files in the 'CURSOJAVASCRIPT' folder. A red arrow points to the 'operadores.js' file, which is highlighted with a red border. The bottom view is the code editor showing 'index.html'. A red arrow points to the line of code where the script tag is defined. The 'src' attribute of the script tag is also highlighted with a red border.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
<script defer text="text/javascript" src="js/operadores.js"></script>
```

En el archivo operadores.js empezar a digitar el siguiente código:

```
index.html          JS operadores.js X
js > JS operadores.js > ...
1 //Definición de variables
2 let num1, num2, resultado
3
4 //Asignación de valores a variables
5 num1 = 5
6 num2 = 15
7
8 //suma
9 resultado = num1 + num2
10 //Imprimir resultado en la consola del navegador
11 console.log(resultado)
12
```

En la consola el resultado será el siguiente:



## Concatenar

Concatenar: Es unir variables con texto para enviar mensajes de salida.

Hay dos (2) maneras de concatenar en JavaScript:

1. Uniendo las cadenas con el operador más (+)
2. Usando el template string

Opción 1:

```
↳ index.html          JS operadores.js X
js > JS operadores.js > ...
1  //Definición de variables
2  let num1, num2, resultado
3
4  //Asignación de valores a variables
5  num1 = 5
6  num2 = 15
7
8  //suma
9  resultado = num1 + num2
10 //Imprimir resultado en la consola del navegador
11 console.log("El resultado de la suma es: " + resultado)
12
13
```

ⓘ DevTools is now available in Spanish!

[Always match Chrome's language](#) [Switch DevTools to Spanish](#) [X](#)

[Don't show again](#)

Elements Console > [⚙️](#) [⋮](#) [X](#)

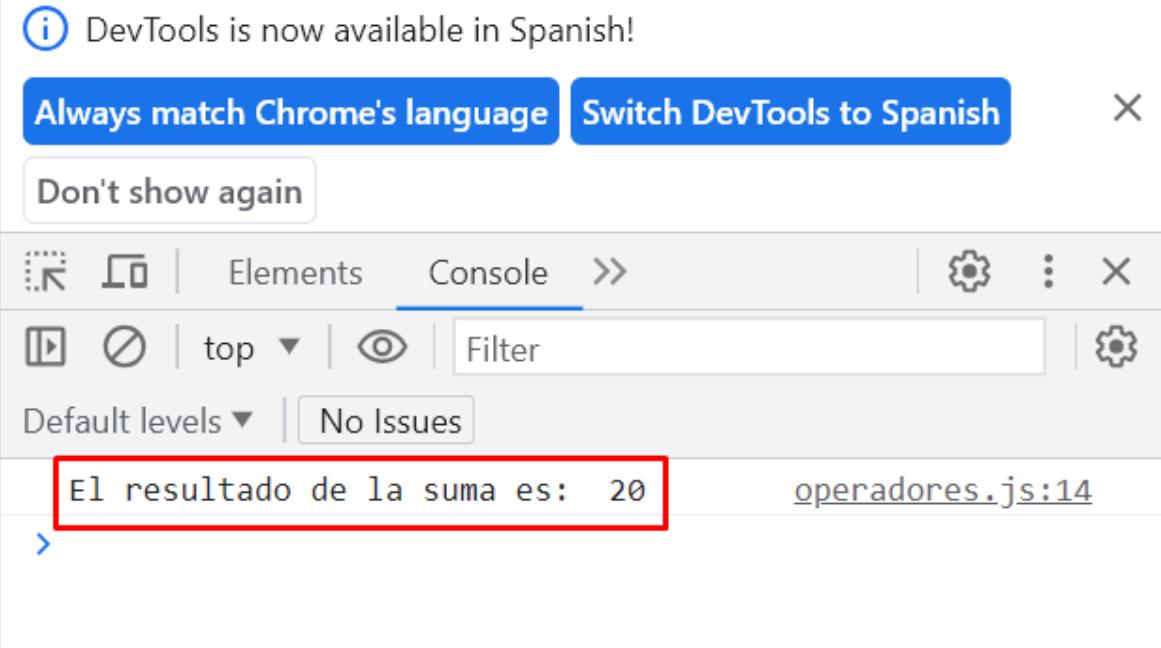
top [Filter](#) [⚙️](#)

Default levels [▼](#) [No Issues](#)

**El resultado de la suma es: 20** [operadores.js:11](#)

Opción 2: Template String **sin usar** el operador +

```
11
12 //Para imprimir con Template String es necesario usar la tilde invertida y las
variables van dentro de la estructura ${...}
13
14 console.log(` El resultado de la suma es:  ${resultado}  `)
15
16
```



```
↳ index.html JS operadores.js X
js > JS operadores.js > ...
  ↳ //Para imprimir con template string es necesario usar la tilde invertida y las variables
    estructura ${...}
13
14   console.log(`El resultado de la suma es: ${resultado}`)
15
16   //resta
17   resultado = num1 - num2
18   console.log(`El resultado de la resta es: ${resultado}`)
19
20   //multiplicación
21   resultado = num1 * num2
22   console.log(`El resultado de la multiplicación es: ${resultado}`)
23
24   //división
25   resultado = num1 / num2
26   console.log(`El resultado de la división es: ${resultado}`)
27
28   //módulo
29   resultado = num1 % num2
30   console.log(`El resultado del módulo es: ${resultado}`)
31
32   //exponenciación
33   resultado = num1 ** num2
34   console.log(`El resultado de la exponenciación es: ${resultado}`)
35
36 |
```

 DevTools is now available in Spanish!

Always match Chrome's language Switch DevTools to Spanish Don't show again X

Elements Console Recorder > ⋮ X

top Filter Default levels ▾ No Issues ⚙️

El resultado de la suma es: 20	operadores.js:14
El resultado de la resta es: -10	operadores.js:18
El resultado de la multiplicación es: 75	operadores.js:22
El resultado de la división es: 0.3333333333333333	operadores.js:26
El resultado del módulo es: 5	operadores.js:30
El resultado de la exponenciación es: 30517578125	operadores.js:34

Cambiar el orden de las variables para observar mejor el resultado de la división y el módulo:

```
index.html JS operadores.js X  
js > JS operadores.js > ...  
13  
14     console.log(`El resultado de la suma es: ${resultado}`)  
15  
16 //resta  
17 resultado = num1 - num2  
18 console.log(`El resultado de la resta es: ${resultado}`)  
19  
20 //multiplicación  
21 resultado = num1 * num2  
22 console.log(`El resultado de la multiplicación es: ${resultado}`)  
23  
24 //división  
25 resultado = num2 / num1  
26 console.log(`El resultado de la división es: ${resultado}`)  
27  
28 //módulo  
29 resultado = num2 % num1  
30 console.log(`El resultado del módulo es: ${resultado}`)  
31  
32 //exponenciación  
33 resultado = num1 ** num2  
34 console.log(`El resultado de la exponenciación es: ${resultado}`)  
35  
36
```

ⓘ DevTools is now available in Spanish!

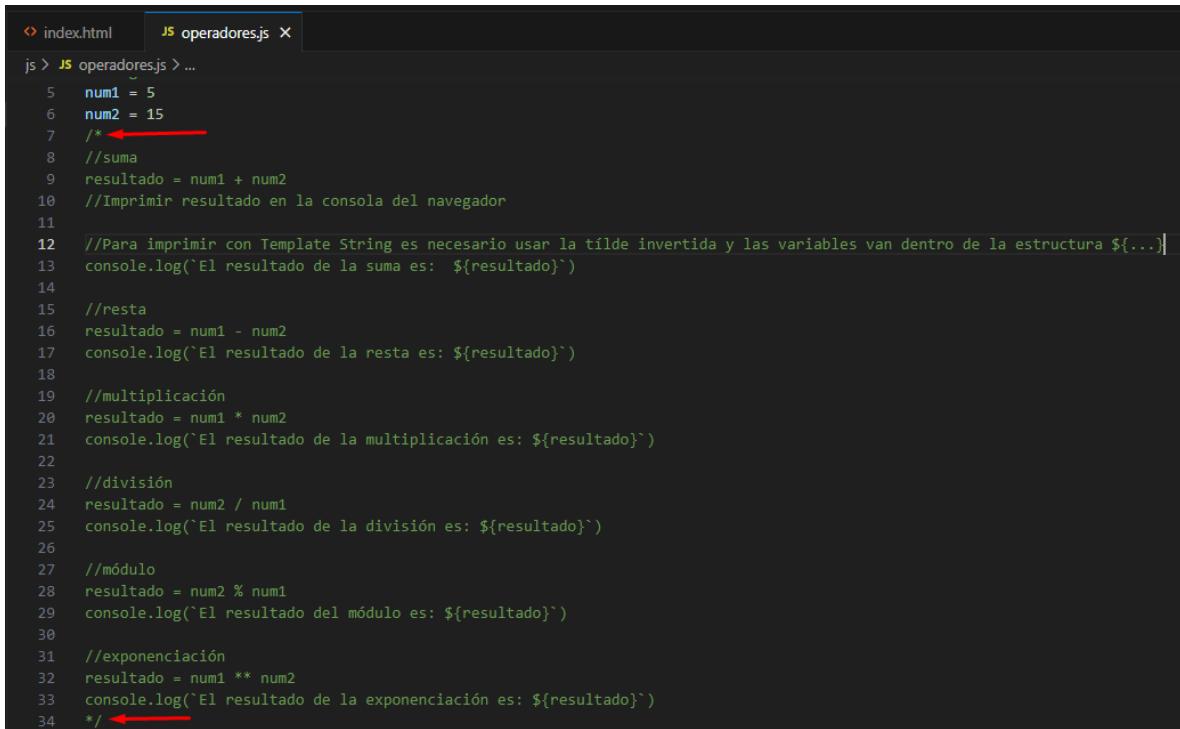
Always match Chrome's language

Switch DevTools to Spanish

Don't show again

```
Elements Console Recorder > | ⚙️ ⚙️ X  
top ▾ | Filter Default levels ▾ | No Issues | ⚙️  
El resultado de la suma es: 20 operadores.js:14  
El resultado de la resta es: -10 operadores.js:18  
El resultado de la multiplicación es: 75 operadores.js:22  
El resultado de la división es: 3 operadores.js:26  
El resultado del módulo es: 0 operadores.js:30  
El resultado de la exponenciación es: 30517578125 operadores.js:34  
>
```

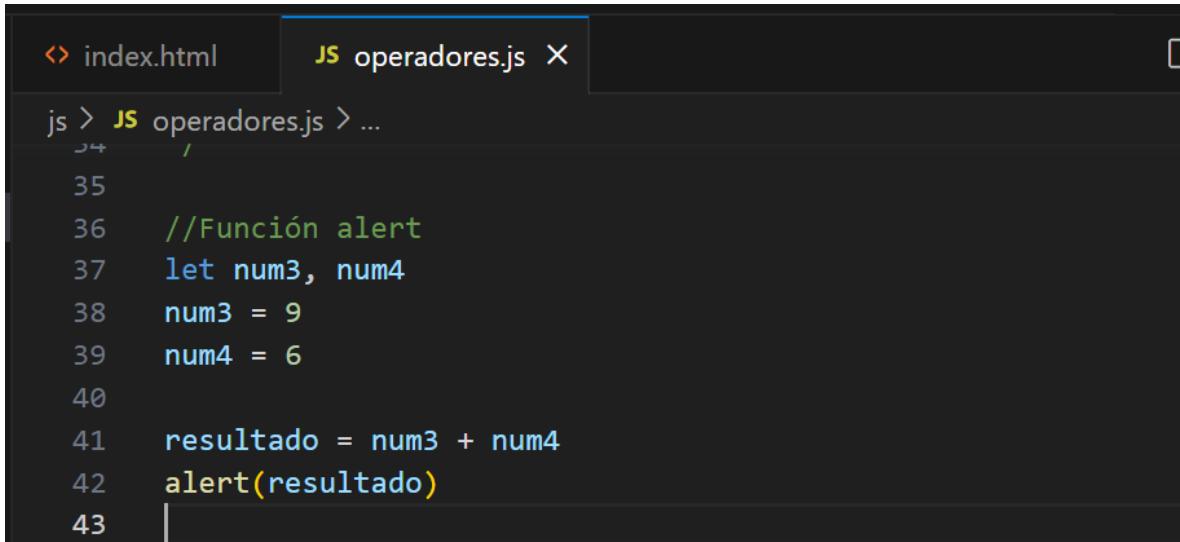
## Comentario de bloque



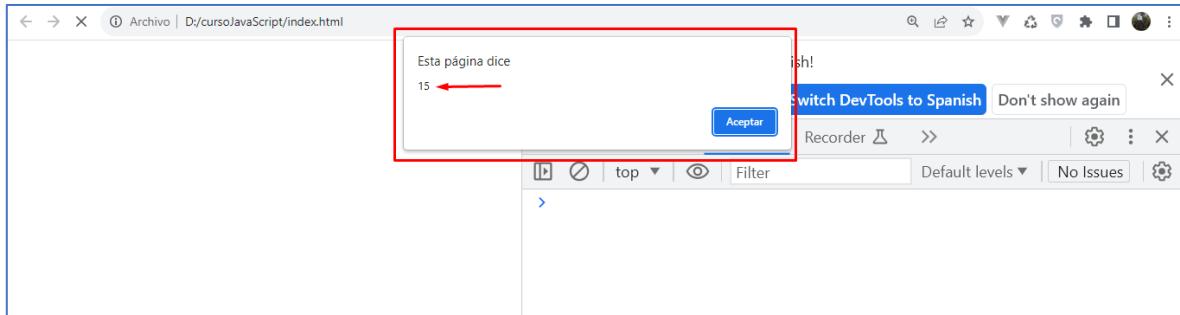
```
index.html JS operadores.js
js > JS operadores.js > ...
5 num1 = 5
6 num2 = 15
7 /* ←
8 //suma
9 resultado = num1 + num2
10 //Imprimir resultado en la consola del navegador
11
12 //Para imprimir con Template String es necesario usar la tilde invertida y las variables van dentro de la estructura ${...}
13 console.log(`El resultado de la suma es: ${resultado}`)
14
15 //resta
16 resultado = num1 - num2
17 console.log(`El resultado de la resta es: ${resultado}`)
18
19 //multiplicación
20 resultado = num1 * num2
21 console.log(`El resultado de la multiplicación es: ${resultado}`)
22
23 //división
24 resultado = num2 / num1
25 console.log(`El resultado de la división es: ${resultado}`)
26
27 //módulo
28 resultado = num2 % num1
29 console.log(`El resultado del módulo es: ${resultado}`)
30
31 //exponenciación
32 resultado = num1 ** num2
33 console.log(`El resultado de la exponenciación es: ${resultado}`)
34 */ ←
```

## Entrada y salida de datos (alert y prompt)

### Función alert

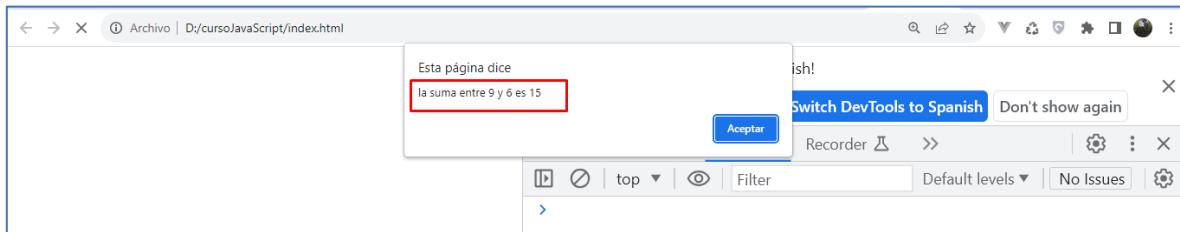


```
index.html JS operadores.js
js > JS operadores.js > ...
34 /
35
36 //Función alert
37 let num3, num4
38 num3 = 9
39 num4 = 6
40
41 resultado = num3 + num4
42 alert(resultado)
43 |
```



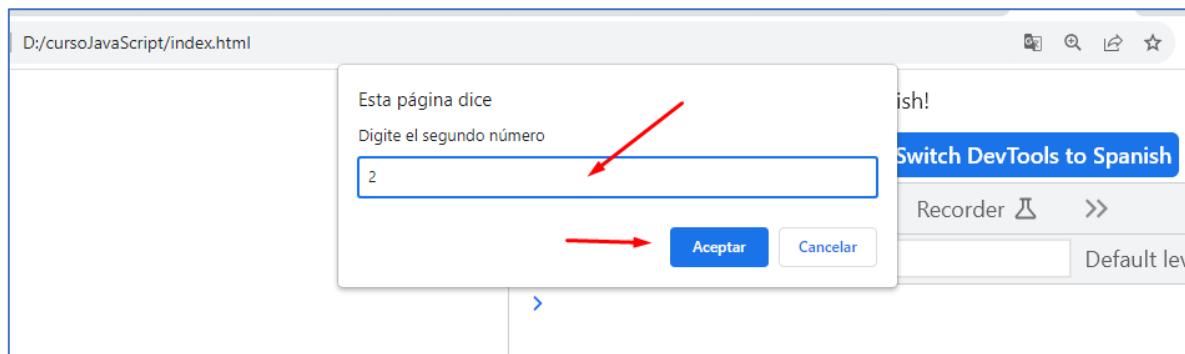
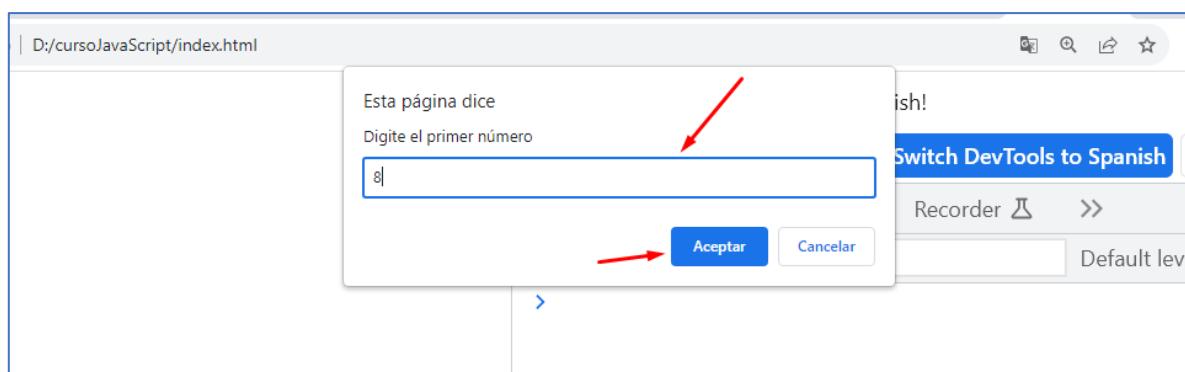
También se puede usar Template String:

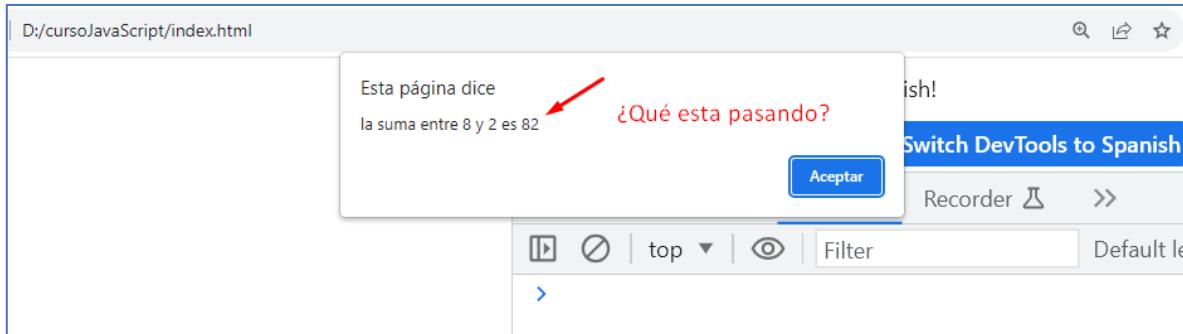
```
index.html JS operadores.js ...
js > JS operadores.js > ...
34
35
36 //Función alert
37 let num3, num4
38 num3 = 9
39 num4 = 6
40
41 resultado = num3 + num4
42 alert(`la suma entre ${num3} y ${num4} es ${resultado}`)
43
44
```



Comentar el bloque a continuación y digitar el código para recibir datos:

```
index.html JS operadores.js ...
js > JS operadores.js > ...
37 let num3, num4
38 /* ←
39 num3 = 9
40 num4 = 6
41
42 resultado = num3 + num4
43 alert(`la suma entre ${num3} y ${num4} es ${resultado}`)
44 */ ←
45 //Función prompt para ingreso de datos
46
47 num3 = prompt('Digite el primer número')
48 num4 = prompt('Digite el segundo número')
49 resultado = num3 + num4
50
51 alert(`la suma entre ${num3} y ${num4} es ${resultado}`)
52
```

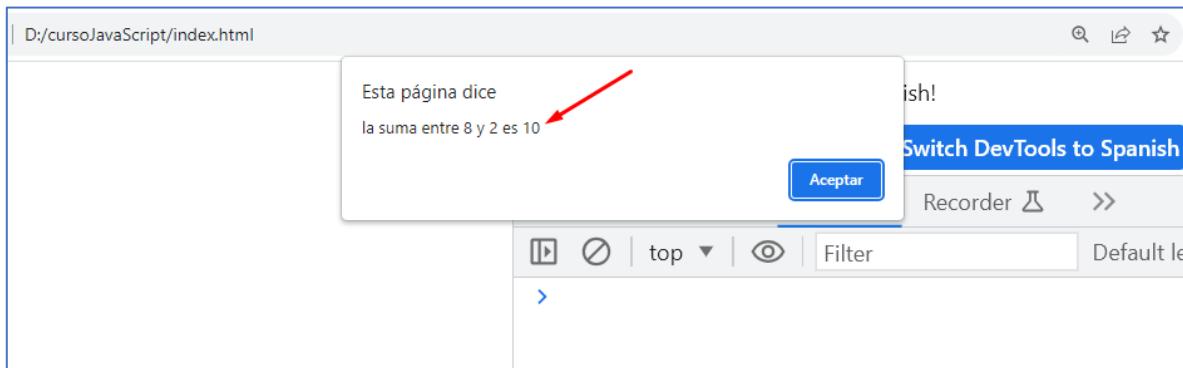




Lo que pasa es que los datos están ingresando como si fueran de tipo String, y lo que hizo JS fue concatenar los dos números, por ello es necesario convertirlos a números enteros, para eso se utiliza la función parseInt:

```
47  num3 = prompt('Digite el primer número')
48  num4 = prompt('Digite el segundo número')
49
50  resultado = parseInt(num3) + parseInt(num4)
51
52  alert(`la suma entre ${num3} y ${num4} es ${resultado}`)
53
54
```

De esta manera, el cálculo se realiza según lo esperado:



## Ejercicios de práctica:

1. Crear un programa que pida un número y lo imprima en un alert y en consola.

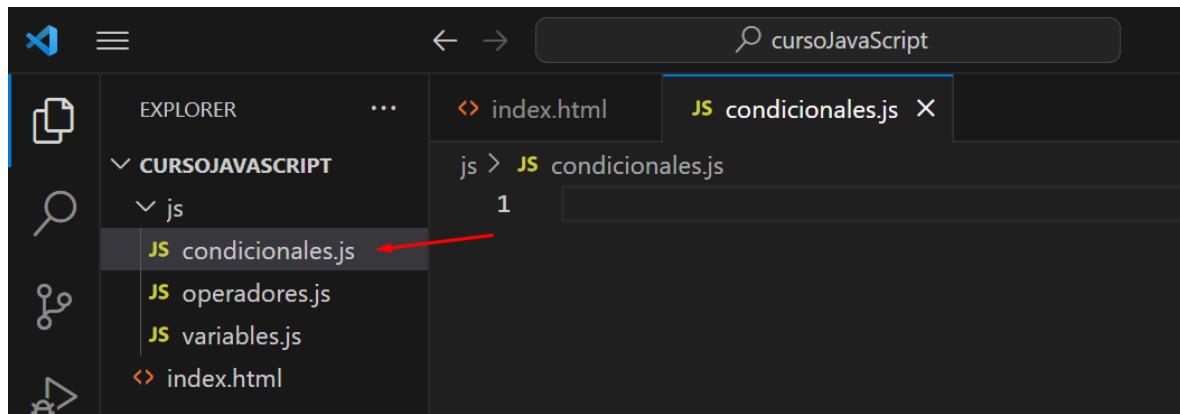
2. Crear un programa que pida un número y que imprima ese número multiplicado por 2 (alert y consola).
3. Crear un programa que pida un número e imprima el cuadrado del mismo número (alert y consola).
4. Crear un programa que pida un número e imprima el cubo del mismo número (alert y consola).
5. Crear un programa que pida dos números y muestre la suma entre ellos (alert y consola).

## Condicionales

Un condicional, como su nombre lo indica, es una condición para discernir entre una opción u otra, y en el proceso mental normalmente se manifiesta con un “Si”; por ejemplo: Si (va a llover), coge el paraguas. (Fuente: <https://codenotch.com/blog/condicionales-y-ciclos/>)

### Condicional IF

Para ver este tema debe crear un nuevo archivo llamado condicionales.js



Recuerde cambiar la ruta en el archivo index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
<script defer text="text/javascript" src="js/condicionales.js"></script>
```

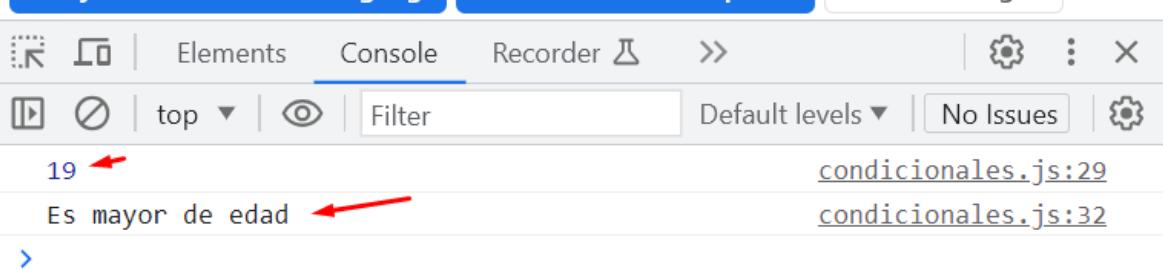
```
index.html condicionales.js
js > condicionales.js
1  /*
2   En el idioma español funciona de la siguiente manera:
3
4   SI se cumple esta condición {
5       se ejecutan unas instrucciones
6   } SI NO {
7       se ejecutan otras instrucciones
8   }
9
10 Ejemplo:
11
12 SI esta lloviendo{
13     llevo paraguas
14 }SI NO{
15     llevo gafas
16 }
17
18 En el lenguaje JavaScript
19 if(condición){
20     ejecutar unas instrucciones
21 }else{
22     ejecutar otras instrucciones
23 }
24
25 */
```

## Sentencia IF

```
↳ index.html JS condicionales.js X  
js > JS condicionales.js > ...  
26  
27 let edad = 19  
28  
29 console.log(edad)  
30  
31 if(edad > 18){  
32   console.log('Es mayor de edad')  
33 }else{  
34   console.log('Es menor de edad')  
35 }  
36  
37
```

 DevTools is now available in Spanish!

Always match Chrome's language Switch DevTools to Spanish Don't show again X



Console

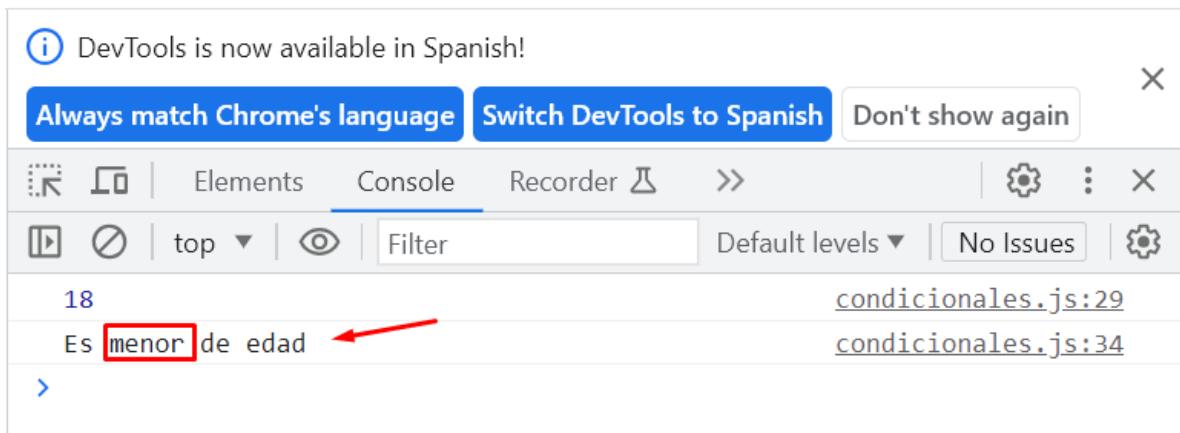
19 ←

Es mayor de edad ←

condicionales.js:29  
condicionales.js:32

Cambiando el valor de la variable por 18

```
↳ index.html JS condicionales.js X  
js > JS condicionales.js > [e] edad  
26  
27 let edad = 18 ←  
28  
29 console.log(edad)  
30  
31 if(edad > 18){  
32   console.log('Es mayor de edad')  
33 }else{  
34   console.log('Es menor de edad')  
35 }  
36  
37
```



En este caso el resultado de la toma de decisión fue **FALSO** y se ejecutaron las instrucciones del **ELSE**. En Colombia, una persona que cumpla sus 18 años ya es mayor de edad, sin embargo, en este programa no se utilizó el operador de comparación indicado, el cual debe ser **mayor o igual que >=**

### Operadores de comparación:

Los operadores de comparación son aquellos que utilizamos en nuestro código (generalmente, en el interior de un if, aunque no es el único sitio donde podemos utilizarlos) para realizar comprobaciones. **Estas expresiones de comparación devuelven un booleano con un valor de true o false.**

Nombre	Operador	Descripción
Operador de igualdad <b>=</b>	<b>a = b</b>	Comprueba si el <b>valor</b> de <b>a</b> es igual al de <b>b</b> . <b>No comprueba tipo de dato</b> .
Operador de desigualdad <b>!=</b>	<b>a != b</b>	Comprueba si el <b>valor</b> de <b>a</b> no es igual al de <b>b</b> . <b>No comprueba tipo de dato</b> .
Operador mayor que <b>&gt;</b>	<b>a &gt; b</b>	Comprueba si el <b>valor</b> de <b>a</b> es mayor que el de <b>b</b> .
Operador mayor/igual que <b>&gt;=</b>	<b>a &gt;= b</b>	Comprueba si el <b>valor</b> de <b>a</b> es mayor o igual que el de <b>b</b> .
Operador menor que <b>&lt;</b>	<b>a &lt; b</b>	Comprueba si el <b>valor</b> de <b>a</b> es menor que el de <b>b</b> .
Operador menor/igual que <b>&lt;=</b>	<b>a &lt;= b</b>	Comprueba si el <b>valor</b> de <b>a</b> es menor o igual que el de <b>b</b> .
Operador de identidad <b>==</b>	<b>a == b</b>	Comprueba si el <b>valor y el tipo de dato</b> de <b>a</b> es igual al de <b>b</b> .
Operador no idéntico <b>!=</b>	<b>a != b</b>	Comprueba si el <b>valor y el tipo de dato</b> de <b>a</b> no es igual al de <b>b</b> .

Fuente: <https://lenguajejs.com/javascript/introduccion/operadores-basicos/>

Realizando la corrección al código el programa queda de la siguiente manera:

index.html      JS condicionales.js X

```

js > JS condicionales.js > ...
26
27 let edad = 18
28
29 console.log(edad)
30
31 if(edad >= 18){
32     console.log('Es mayor de edad')
33 }else{
34     console.log('Es menor de edad')
35 }
36
37

```

(i) DevTools is now available in Spanish!

[Always match Chrome's language](#) [Switch DevTools to Spanish](#) [Don't show again](#)

Elements      Console      Recorder

top Filter      Default levels No Issues

18      [condicionales.js:29](#)  
 Es **mayor** de edad      [condicionales.js:32](#)

>

## Operadores lógicos (Y, O, NO)

Estos operadores se observan en la toma de decisiones con la sentencia IF

### Tabla de verdad conjunción (Y / AND)

P	Q	$P \wedge Q$
V	V	V
V	F	F
F	V	F
F	F	F

O puede ser

P	Q	$P \wedge Q$
1	1	1
1	0	0
0	1	0
0	0	0

### Tabla de verdad disyunción (O / OR)

P	Q	PvQ
V	V	V
V	F	V
F	V	V
F	F	F

O puede ser

P	Q	PvQ
1	1	1
1	0	1
0	1	1
0	0	0

### Tabla de verdad negación (no / NOT)

P	-P
V	F
F	V

O puede ser

P	-P
1	0
0	1

Operador Lógico	Operador en JS
O	
Y	&&
NO	!

#### Caso:

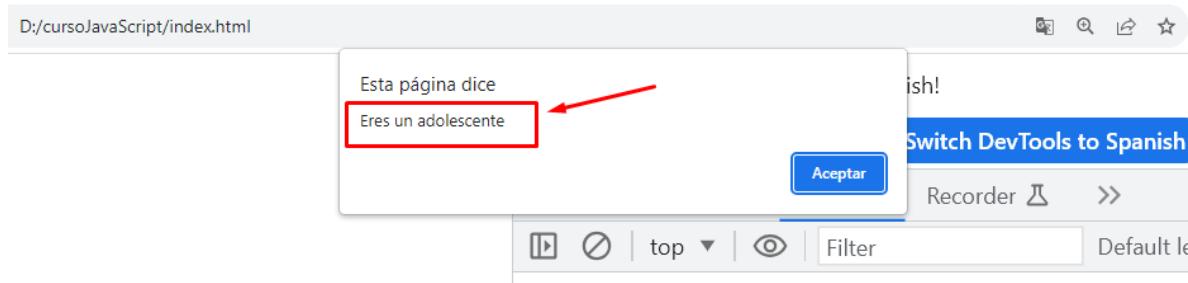
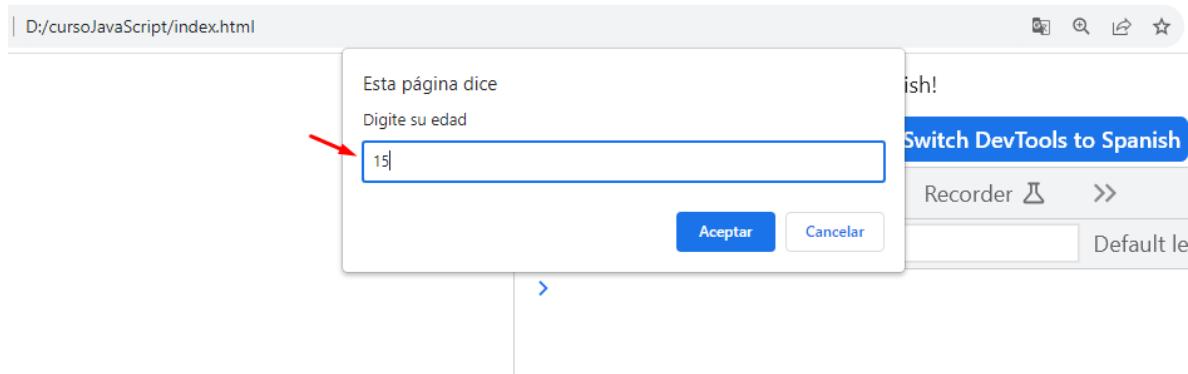
Pedir al usuario su edad y determinar si es un adolescente, un joven adulto, o un adulto.

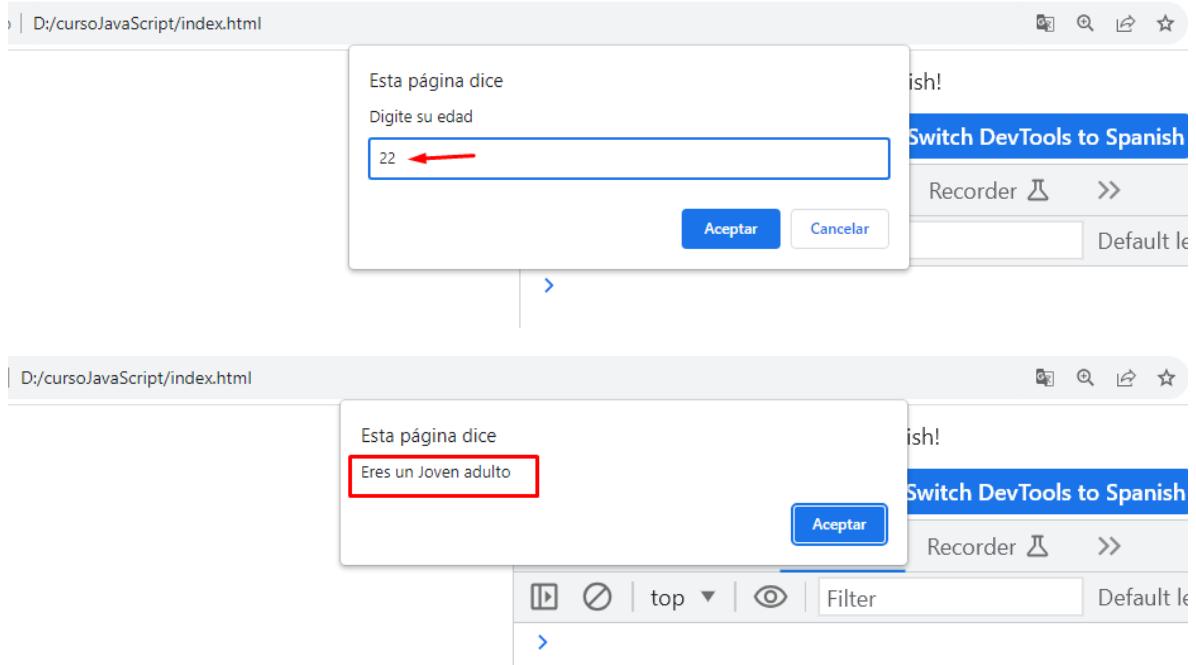
Un adolescente tiene entre los 14 y 17 años

Un Joven adulto tiene entre los 18 y 25 años

Un adulto tiene más de 26 años

```
index.html      JS condicionales.js ×  
js > JS condicionales.js > ...  
39 let edad = parseInt(prompt('Digite su edad'))  
40  
41 if(edad >= 14 && edad <= 17){  
42  
43     alert('Eres un adolescente')  
44  
45 }else if(edad >= 18 && edad <= 25){  
46  
47     alert('Eres un Joven adulto')  
48  
49 }else if(edad >= 26){  
50  
51     alert('Eres un adulto')  
52  
53 }
```





Caso:

Crear un programa que identifique el idioma que se habla en el país donde nació el usuario. Comente el código anterior y digite el siguiente:

```

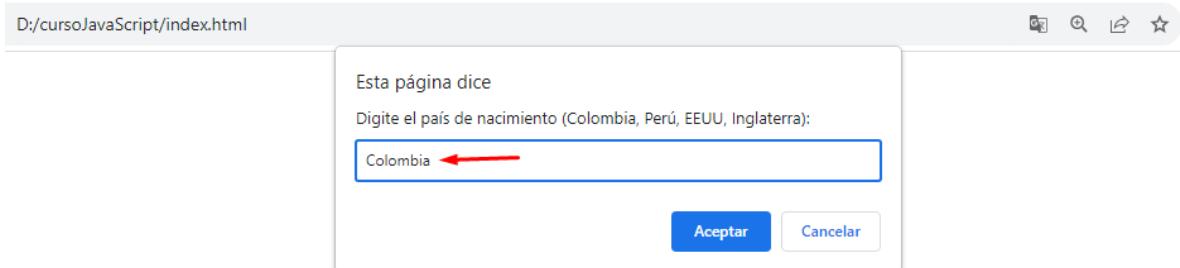
index.html   JS condicionales.js X

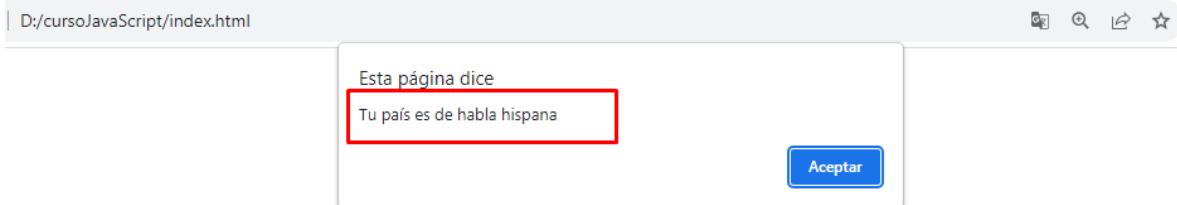
js > JS condicionales.js
      /*
38     let edad = parseInt(prompt('Digite su edad'))
39
40     if(edad >= 14 && edad <= 17){
41
42         alert('Eres un adolescente')
43
44     }else if(edad >= 18 && edad <= 25){
45
46         alert('Eres un Joven adulto')
47
48     }else if(edad >= 26){
49
50         alert('Eres un adulto')
51
52     }
53 }
54 */
55
56 let pais = prompt('Digite el país de nacimiento (Colombia, Perú, EEUU, Inglaterra): ')
57
58 if(pais == 'Colombia' || pais == 'Perú'){
59     alert('Tu país es de habla hispana')
60 }else if(pais == 'EEUU' || pais == 'Inglaterra'){
61     alert('Tu país es de habla inglesa')
62 }
63
64

```

OR

OR

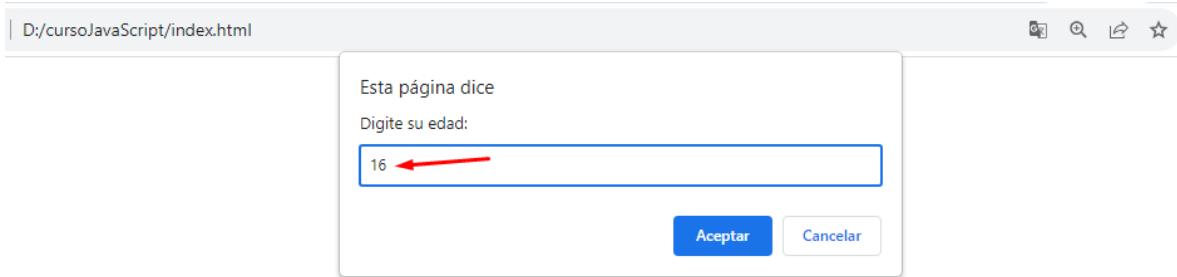


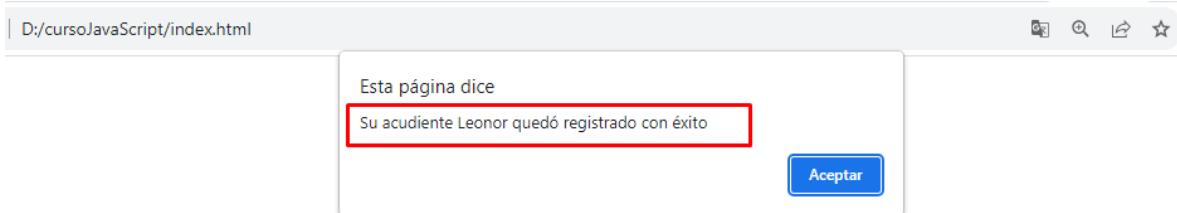
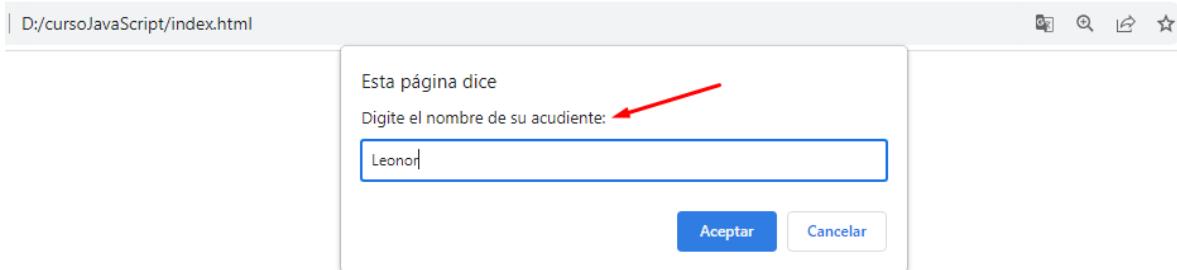


Caso:

Crear un programa que pida al usuario la edad y SI NO es mayor, debe digitar el nombre del acudiente.

```
index.html JS condicionales.js ...
js > JS condicionales.js > ...
64
65 let edad = parseInt(prompt('Digite su edad:'))
66 NOT
67 if(!(edad >= 18)){
68     nombreAcudiente = prompt('Digite el nombre de su acudiente:')
69     alert(`Su acudiente ${nombreAcudiente} quedó registrado con éxito`)
70 }else{
71     alert('Usted no necesita acudiente')
72 }
73
74
```

A screenshot of a code editor showing a file named "condicionales.js". The code contains a conditional statement that checks if the user's age is greater than or equal to 18. If the condition is false (NOT), it prompts the user for the name of their guardian and displays a success message. An annotation with a red arrow points to the logical NOT operator "!" in the if condition. The code editor interface shows tabs for "index.html" and "condicionales.js".



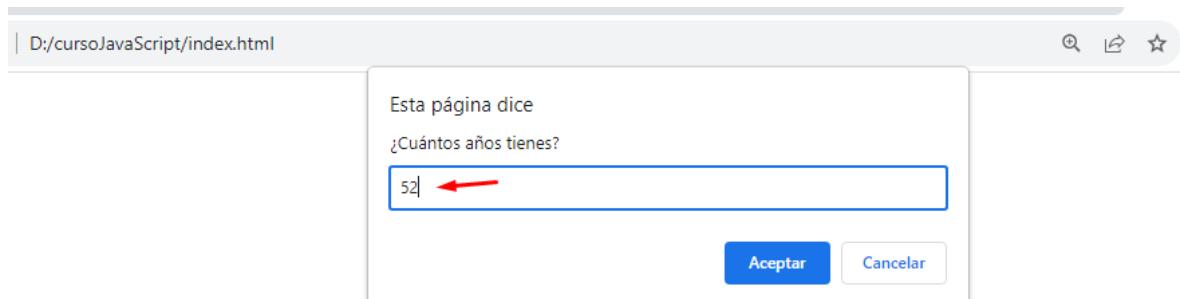
### Condiciones anidadas

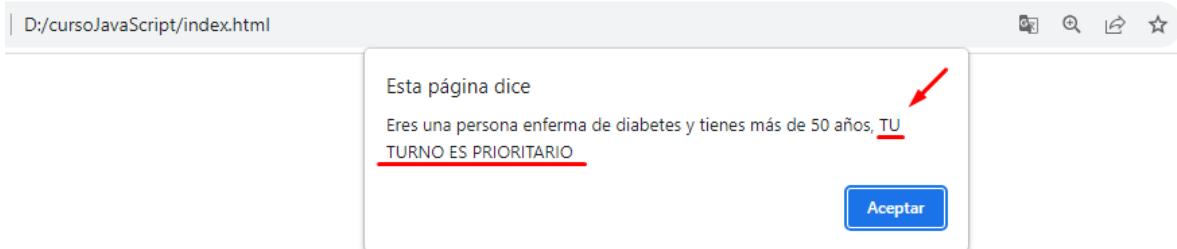
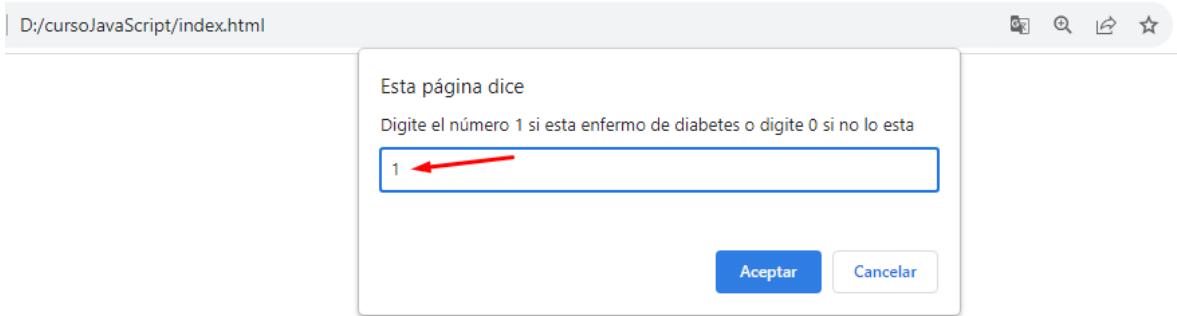
Caso:

Una clínica necesita un sistema de información que priorice los turnos dependiendo si la personas son mayores o iguales de 50 años y si sufren de diabetes:

Comentar el ejercicio anterior y avanzar con el nuevo código:

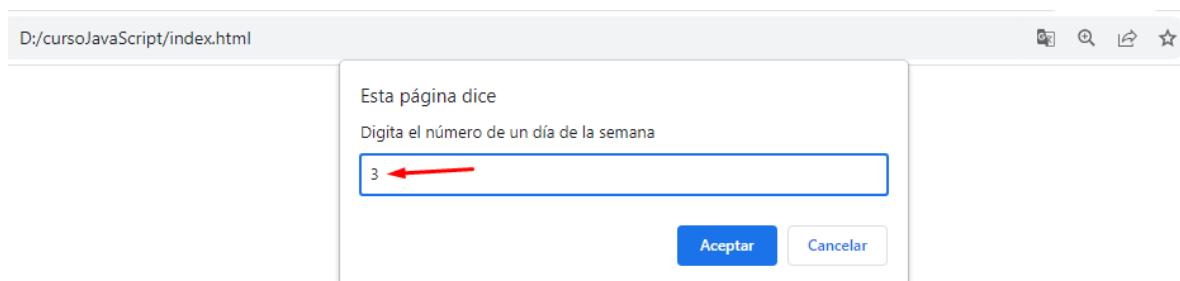
```
index.html JS condicionales.js ...
js > JS condicionales.js > ...
55
56     let edad = parseInt(prompt('¿Cuántos años tienes?'))
57     let diabetes = parseInt(prompt('Digite el número 1 si esta enfermo de diabetes o
digite 0 si no lo esta'))
58
59     if(edad >= 50){
60         if(diabetes == 1){
61             alert('Eres una persona enferma de diabetes y tienes más de 50 años, TU
TURNO ES PRIORITARIO')
62         }else{
63             alert('Eres una persona que tienes más de 50 años SIN diabetes, TU TURNO
ES NORMAL')
64         }
65
66     }else{
67         alert('Eres una persona menor de 50 años, TU TURNO ES NORMAL')
68     }
69
```

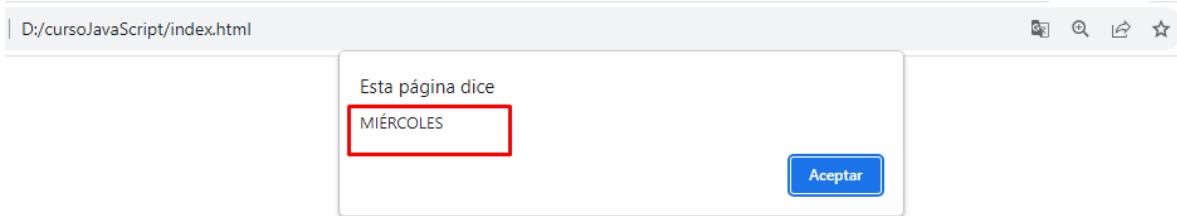




## Switch

```
index.html JS condicionales.js
js > JS condicionales.js > ...
  60  jeisei
  61  |     alert('Eres una persona menor de 50 años, TU TURNO ES NORMAL')
  62  | */
  63
  64  let dia = parseInt(prompt('Digita el número de un día de la semana'))
  65
  66  switch (dia) {
  67      case 1:
  68          alert('LUNES')
  69          break;
  70      case 2:
  71          alert('MARTES')
  72          break;
  73      case 3:
  74          alert('MIÉRCOLES')
  75          break;
  76      case 4:
  77          alert('JUEVES')
  78          break;
  79      case 5:
  80          alert('VIERNES')
  81          break;
  82      case 6:
  83          alert('SÁBADO')
  84          break;
  85      case 7:
  86          alert('DOMINGO')
  87          break;
  88      default:
  89          alert('No existe')
  90          break;
  91  }
  92
  93
  94
  95
  96
  97 }
```





## Ejercicios de práctica:

1. Inventarse un ejercicio donde aplique un if con un else
2. Inventarse un ejercicio donde aplique un if anidado
3. Inventarse un ejercicio donde aplique la estructura switch
4. Inventarse un ejercicio donde aplique el operador AND
5. Inventarse un ejercicio donde aplique el operador OR
6. Inventarse un ejercicio donde aplique el operador NOT

## Navigator, window y document

Internamente desde JavaScript podemos acceder a todas las opciones del navegador a través de sus objetos nativos, como lo son:

- Navegador (JavaScript lo llama **navigator**)
- Ventanas (JavaScript lo llama **window**)
- Contenido de página web (todo el HTML, JavaScript lo llama **document**)

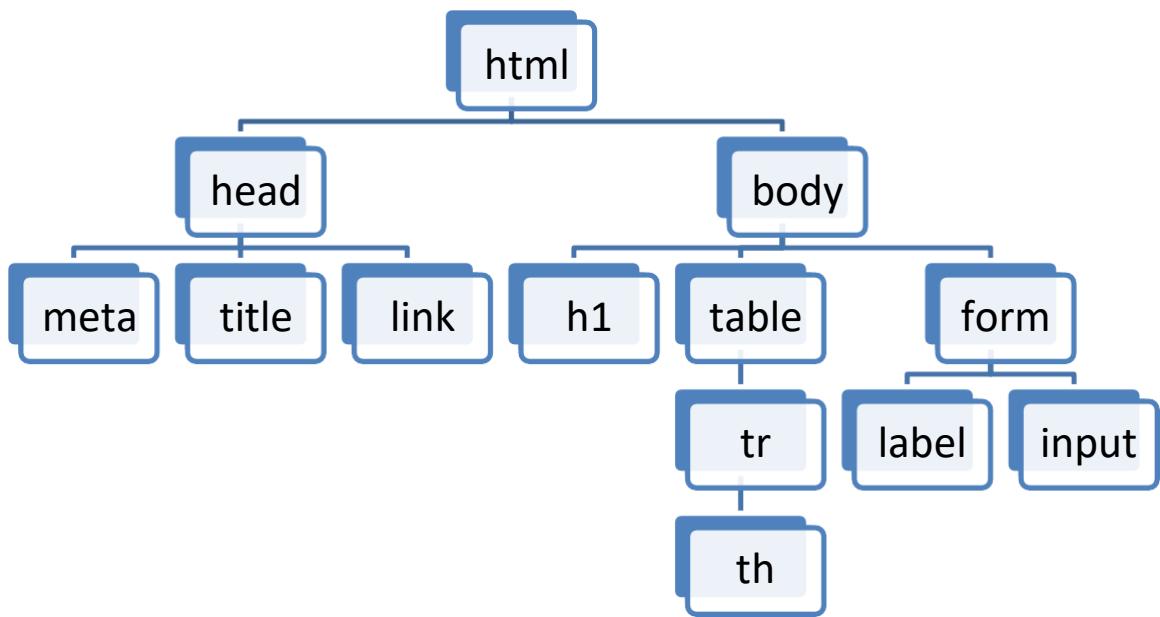


## Document Object Model

### (Modelo de objetos del documento o DOM)

El **Modelo de Objetos del Documento (DOM)** es un API para documentos HTML. Proporciona una representación estructural del documento, permitiendo la modificación de su contenido o su presentación visual. Esencialmente, comunica las páginas web con los scripts o los lenguajes de programación. **DOM** es un estándar del W3C.

El DOM es el árbol o la estructura del documento:



Para ver este tema debe crear un nuevo archivo llamado **dom.js** y cambiar la ruta del archivo en **index.html**

```
index.html <--> JS dom.js
index.html > html > head > script
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7
8      <script defer text="text/javascript" src="js/dom.js"></script>
9
10 </head>
11 <body>
12
13
14 </body>
15 </html>
```

Obtener elementos de DOM y manipularlos (`document.getElementById`):

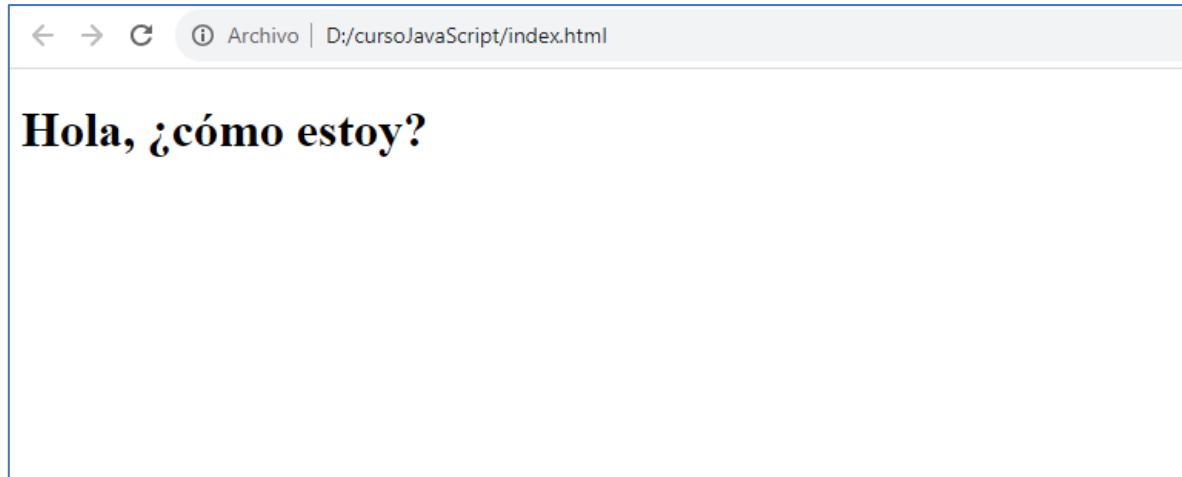
Por ejemplo, a un `<h1>` en la página HTML se le puede agregar la propiedad `id` y asignar un identificador, en este caso ‘`titulo1`’ y en el archivo `dom.js` obtener ese elemento por el id y darle valores dinámicamente:

```
index.html <--> JS dom.js
index.html > html > body > h1#titulo1
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7
8      <script defer text="text/javascript" src="js/dom.js"></script>
9
10 </head>
11 <body>
12
13 <h1 id="titulo1"></h1>
14
15
16 </body>
17 </html>
```

### Uso de innerText

```
↳ index.html   JS dom.js  X  □  
js > JS dom.js  
1  //Mostrar texto en la página actual en un h1  
2  document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'  
3
```

Resultado:

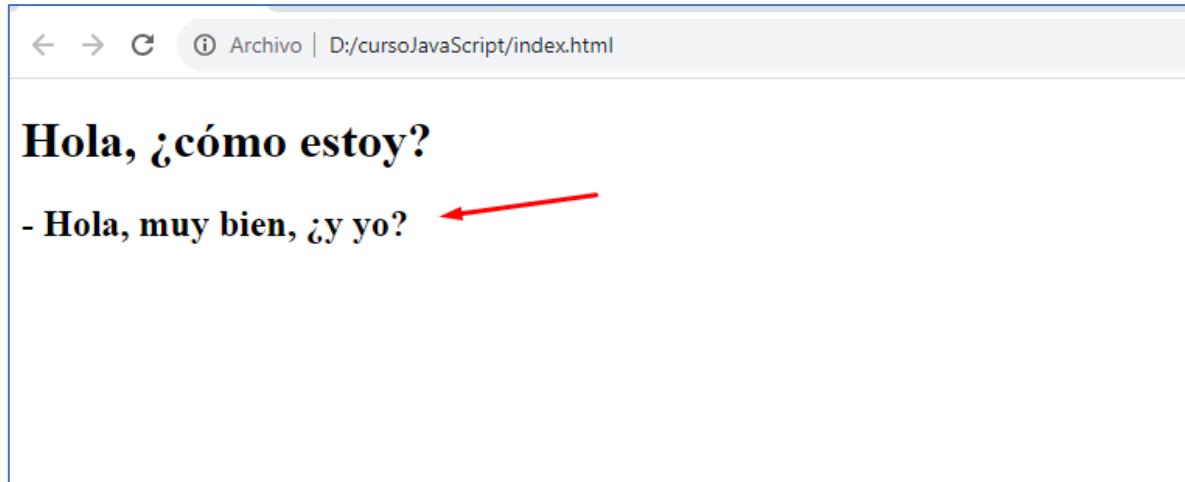


### Uso de innerHTML

```
↳ index.html  X  JS dom.js  
↳ index.html > ⏷ html > ⏷ body > ⏷ div#recuadro1  
1  <!DOCTYPE html>  
2  <html lang="en">  
3  <head>  
4  |   <meta charset="UTF-8">  
5  |   <meta name="viewport" content="width=device-width, initial-scale=1.0">  
6  |   <title>Document</title>  
7  
8  |   <script defer text="text/javascript" src="js/dom.js"></script>  
9  
10 |</head>  
11 <body>  
12  
13 <h1 id="titulo1"></h1>  
14  
15  
16 <div id="recuadro1">  
17  
18 </div>  
19 </body>  
20 </html>
```

```
index.html | JS dom.js | X
js > JS dom.js
1 //Mostrar texto en la página actual en un h1
2 document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'
3
4 //Insertar etiquetas HTML desde JS
5 document.getElementById('recuadro1').innerHTML = '<h2>- Hola, muy bien, ¿y yo?</h2>'
```

Resultado:



Insertar una tabla con innerHTML

```
< index.html > JS dom.js
< index.html > html > body > div#recuadro2
9
10   </head>
11   <body>
12
13     <h1 id="titulo1"></h1>
14
15
16     <div id="recuadro1">
17
18       </div>
19
20     <div id="recuadro2">
21
22       </div>
23
24
25   </body>
26   </html>
```

En este caso, el código de la tabla debe quedar en una sola línea, en Visual Studio Code puede presionar Alt + Z para ajustar el texto:

```
js > JS dom.js
1 //Mostrar texto en la página actual en un h1
2 document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'
3
4 //Insertar etiquetas HTML desde JS
5 document.getElementById('recuadro1').innerHTML = '<h2>- Hola, muy bien, ¿y yo?</h2>'
6
7 //Incluir una tabla con innerHTML
8
9 document.getElementById('recuadro2').innerHTML = '<table border="1"><thead><tr><th>Nombre</th><th>Nº Calzado</th></tr></thead><tbody><tr><td>Martha</td><td>38</td></tr><tr><td>Jorge</td><td>40</td></tr></tbody></table>'
```

Resultado:

← → ⌂ Archivo | D:/cursoJavaScript/index.html

# Hola, ¿cómo estoy?

- Hola, muy bien, ¿y yo?

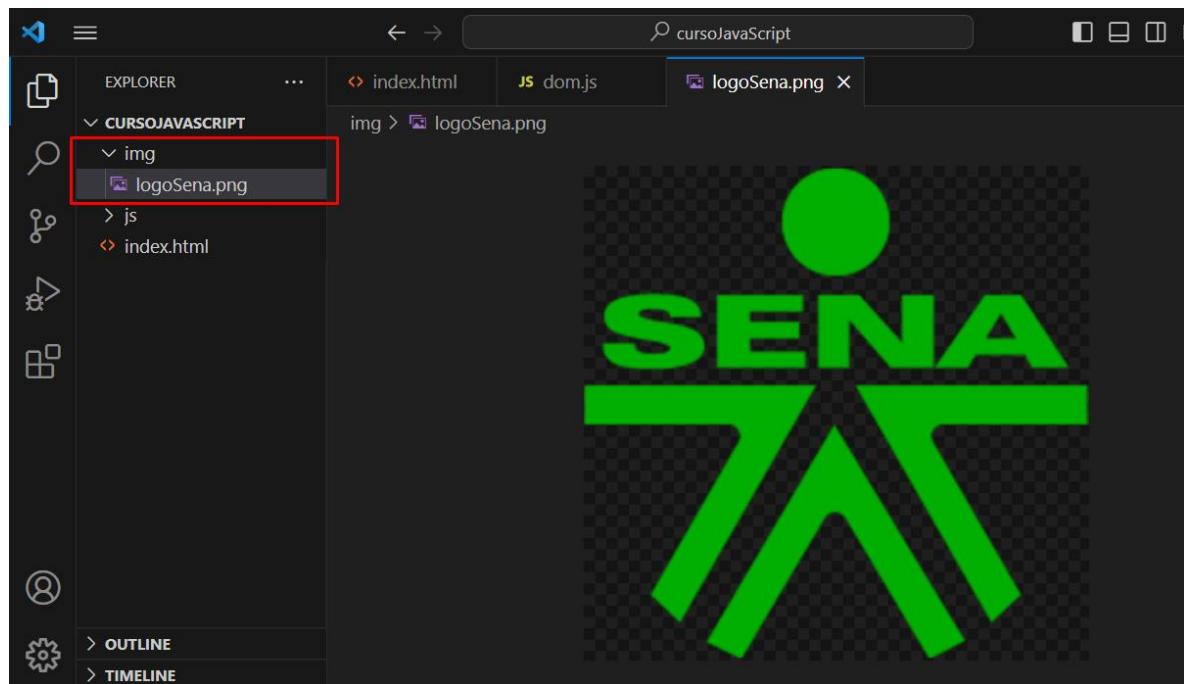
Nombre	Nº Calzado
Martha	38
Jorge	40

A red arrow points to the second row of the table.

### Acceder a propiedades de las etiquetas HTML

Incluir una imagen de manera dinámica con JavaScript usando la propiedad `src` de la etiqueta `img`:

1. Crear una carpeta llamada `img` en el sitio de trabajo.
2. Poner en la carpeta una imagen con un nombre corto.



Código HTML:

```
↳ index.html X JS dom.js logoSena.png  
↳ index.html > ⚒ html > ⚒ body > ⚒ img#logo  
17  
18     </div>  
19  
20     <div id="recuadro2">  
21  
22     </div>  
23  
24     <img src="" id="logo">  
25  
26  
27 </body>  
28 </html>
```

```
↳ index.html JS dom.js logoSena.png ..  
js > JS dom.js  
1 //Mostrar texto en la página actual en un h1  
2 document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'  
3  
4 //Insertar etiquetas HTML desde JS  
5 document.getElementById('recuadro1').innerHTML = '<h2>- Hola, muy bien, ¿y yo?</h2>'  
6  
7 //Incluir una tabla con innerHTML  
8  
9 document.getElementById('recuadro2').innerHTML = '<table border="1"><thead><tr><th>Nombre</th><th>Nº Calzado</th></tr></thead><tbody><tr><td>Martha</td><td>38</td></tr><tr><td>Jorge</td><td>40</td></tr></tbody></table>'  
10  
11 document.getElementById('logo').src = 'img/logoSena.png'  
↑
```

Resultado:

**Hola, ¿cómo estoy?**

- Hola, muy bien, ¿y yo?

Nombre	Nº Calzado
Martha	38
Jorge	40



Cambiar estilos de un elemento desde JS:

```
index.html      JS dom.js      logoSena.png      ...  
js > JS dom.js  
1 //Mostrar texto en la página actual en un h1  
2 document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'  
3  
4 //Insertar etiquetas HTML desde JS  
5 document.getElementById('recuadro1').innerHTML = '<h2>- Hola, muy bien, ¿y yo?</h2>'  
6  
7 //Incluir una tabla con innerHTML  
8  
9 document.getElementById('recuadro2').innerHTML = '<table border="1"><thead><tr><th>Nombre</th><th>Nº Calzado</th></tr></thead><tbody><tr><td>Martha</td><td>38</td></tr><tr><td>Jorge</td><td>40</td></tr></tbody></table>'  
10  
11 document.getElementById('logo').src = 'img/logoSena.png'  
12  
13 document.getElementById('logo').style.width = '100px'  
14 document.getElementById('logo').style.border = 'solid 2px red'
```

El resultado es una imagen más pequeña y con un borde rojo de 2 pixeles:

The screenshot shows a web browser window with the address bar displaying "Archivo | D:/cursoJavaScript/index.html". The page content is as follows:

**Hola, ¿cómo estoy?**

**- Hola, muy bien, ¿y yo?**

Nombre	Nº Calzado
Martha	38
Jorge	40

The SENA logo is displayed, consisting of a green stylized figure and the text "SENA".

## Ejercicios de práctica:

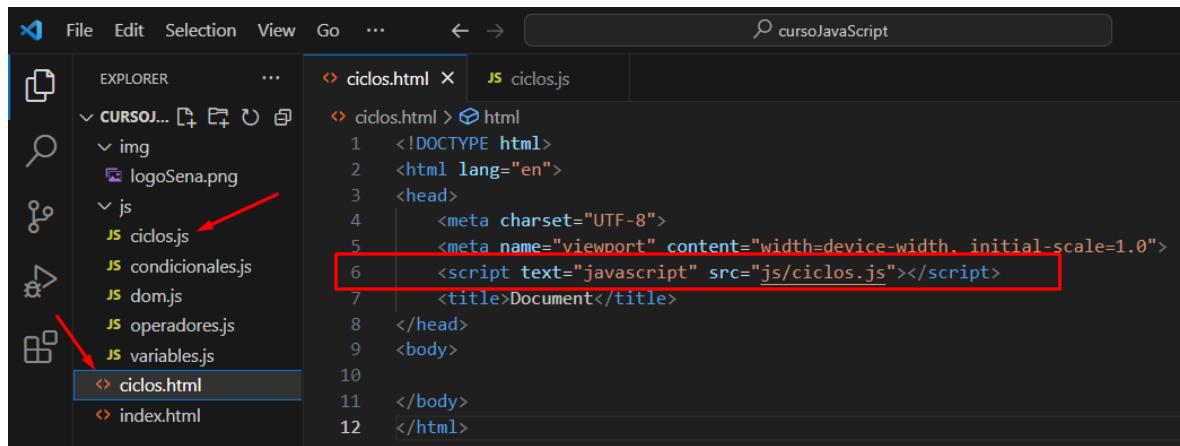
1. Título y párrafos
  - a. Insertar etiquetas para títulos `<h1>` – `<h6>` o párrafos `<p>`
  - b. Usar el `innerText` para mostrar allí sus datos básicos (nombres, apellidos, teléfono, correo, edad, hobby, entre otros).
  - c. Por cada dato usar una etiqueta html diferente.
2. Imágenes
  - a. Manera fácil
    - i. Crear 6 etiquetas `<img>` vacías con id diferentes.
    - ii. Buscar en internet 6 imágenes de dados, cada imagen debe contener una de las caras de un dado.
    - iii. Con código JS insertar en cada `<img>` las caras de los dados.
    - iv.** Cada imagen debe tener características diferentes (tamaño y borde).
  - b. Manera difícil
    - i. Crear 6 etiquetas `<div>` vacías con id diferentes.
    - ii. Buscar en internet 6 imágenes de dados, cada imagen debe contener una de las caras de un dado.
    - iii. Con código JS insertar en cada `<div>` las caras de los dados.
    - iv.** Cada imagen debe tener características diferentes (tamaño y borde).
3. Tabla
  - a. Definir con código JS una tabla que tenga 4 columnas y 4 filas.
  - b. Los títulos de las columnas son DOCUMENTO, NOMBRE, EDAD Y HOBBY.
  - c. Crear un `<div>` con id = 'tabla' y dibujar allí la tabla.

## Estructuras de control

Bucle FOR (Ciclo PARA)

Crear dos nuevos archivos, uno llamado ciclos.html y otro llamado ciclos.js

Ubicarlos en el lugar respectivo:



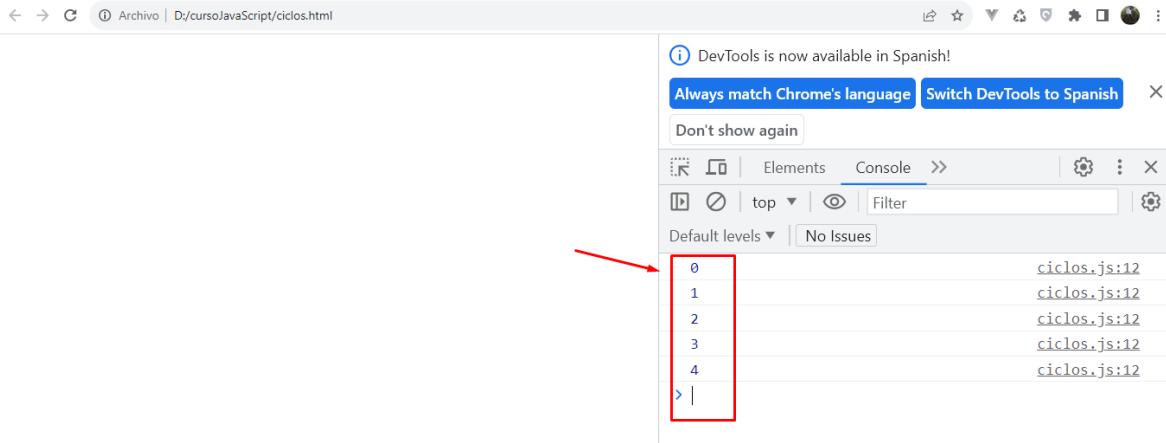
Ejemplo 1: en el archivo ciclos.js digitar el siguiente código y ejecutar en .html en el navegador:

The screenshot shows the 'ciclos.js' code again:

```
/*
    for(desde; hasta; y contador){
        aquí se ejecutan las instrucciones
    }
*/
var arreglo = [2, 4, 6, 8, 10]
for(var indice = 0; indice <= 4; indice++){
    //Esta línea imprime el contador (variable indice)
    console.log(indice)
}
```

Two red arrows point to specific lines of code: one to the explanatory comment in the first loop iteration and another to the 'console.log(indice)' line within the loop.

Resultado:

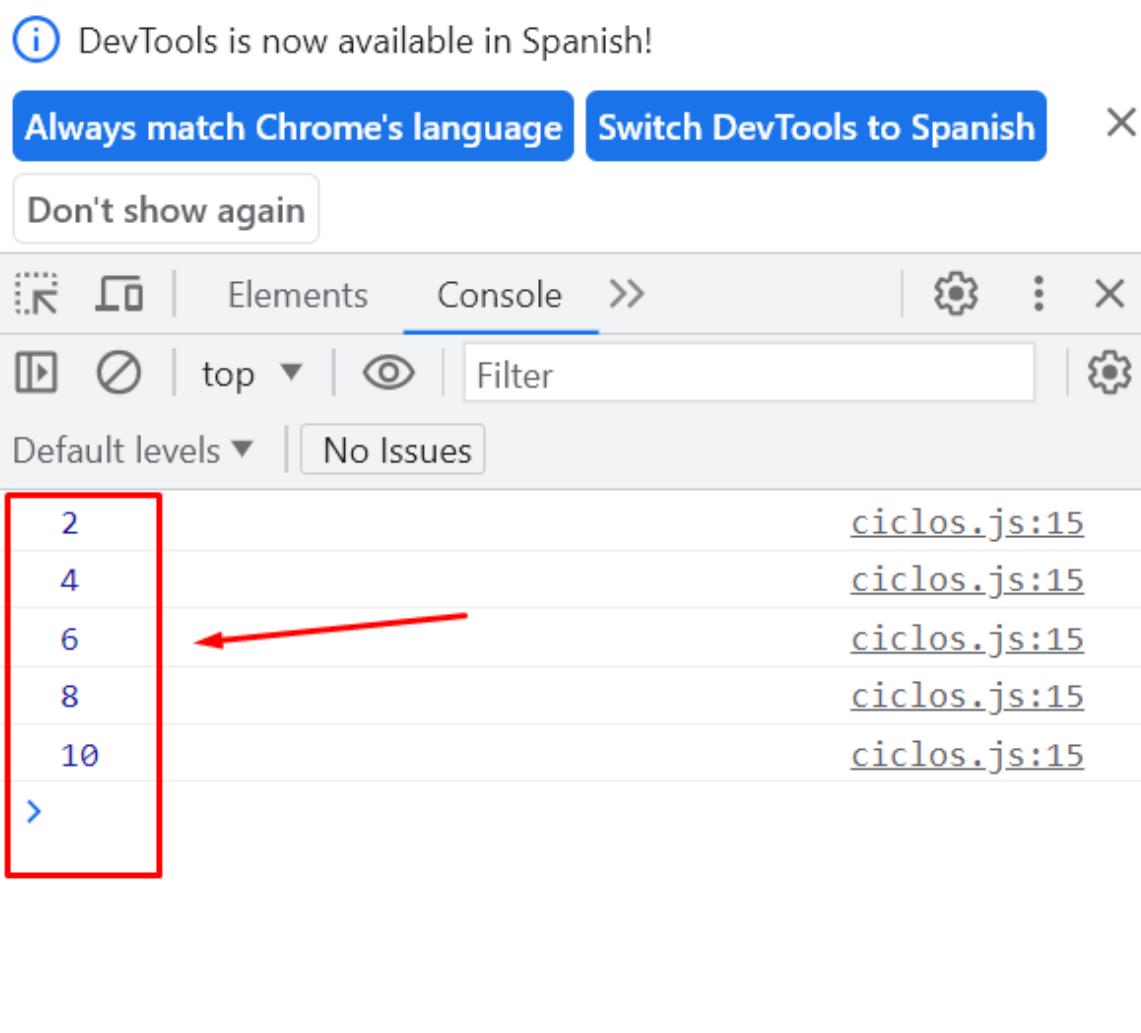


En este caso se ha mostrado el valor del índice, el cual, va incrementando de uno en uno, sin embargo, para imprimir el contenido del arreglo hay que nombrar el arreglo y dentro de los corchetes ubicar la variable índice:

arreglo[indice]

```
8  var arreglo = [2, 4, 6, 8, 10]
9
10 for(var indice = 0; indice <= 4; indice++){
11     //Esta línea imprime el contador (variable indice)
12     //console.log(indice)
13
14     //Esta línea imprime cada elemento del arreglo gracias a la variable indice
15     console.log(arreglo[indice])
16 }
17 }
```

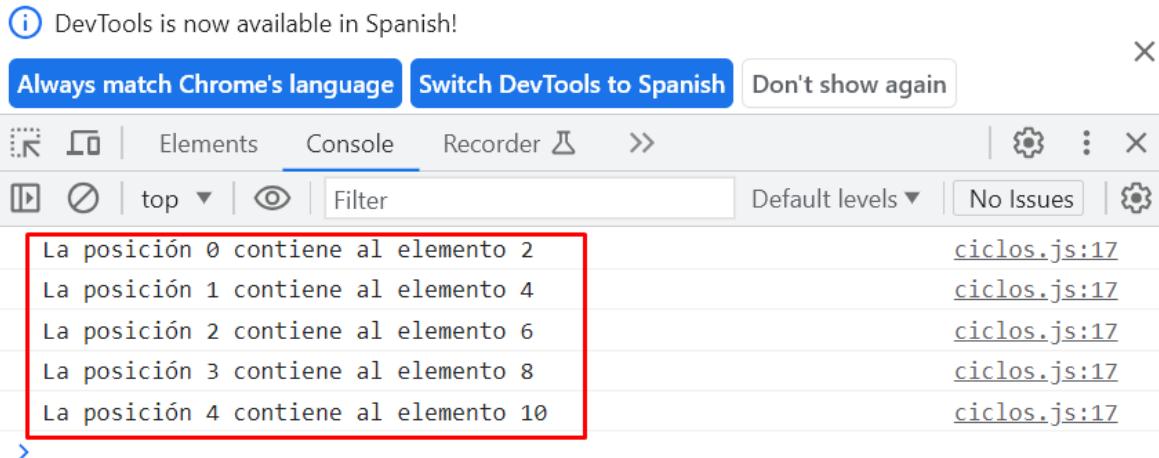
Resultado:



Para comprenderlo mejor se puede imprimir lo siguiente usando template String:

```
10  for(var indice = 0; indice <= 4; indice++){  
11      //Esta línea imprime el contador (variable indice)  
12      //console.log(indice)  
13  
14      //Esta línea imprime cada elemento del arreglo gracias a la variable indice  
15      //console.log(arreglo[indice])  
16  
17      console.log(`La posición ${indice} contiene al elemento ${arreglo[indice]}`)  
18  }  
19 }
```

Resultado:



Se puede reemplazar el número 4 en la segunda sección del ciclo FOR por la propiedad `length` del arreglo, de esta manera si el arreglo cambia su tamaño no habrá necesidad de modificar este número manualmente, sin embargo, antes de realizar el cambio revise cual es la longitud del arreglo así:

```
8  var arreglo = [2, 4, 6, 8, 10]
9
10 console.log(`longitud del arreglo = ${arreglo.length}`) ←
11
12 for(var indice = 0; indice <= 4; indice++){
13     //Esta línea imprime el contador (variable indice)
14     //console.log(indice)
15
16     //Esta línea imprime cada elemento del arreglo gracias a la variable indice
17     //console.log(arreglo[indice])
18
19     console.log(`La posición ${indice} contiene al elemento ${arreglo[indice]}`)
20
21 }
```

Arroja que es 5:

ⓘ DevTools is now available in Spanish!

Always match Chrome's language [Switch DevTools to Spanish](#) [Don't show again](#)

Elements Console Recorder >> [Filter](#) Default levels ▾ No Issues

```
longitud del arreglo = 5 ← ciclos.js:10
La posición 0 contiene al elemento 2 ciclos.js:19
La posición 1 contiene al elemento 4 ciclos.js:19
La posición 2 contiene al elemento 6 ciclos.js:19
La posición 3 contiene al elemento 8 ciclos.js:19
La posición 4 contiene al elemento 10 ciclos.js:19
```

Si se cambia sin tener en cuenta el signo de comparación `<=` (menor o igual) la variable índice sobrepasará las posiciones del arreglo y al final tratará de acceder a una posición que no existe:

```
8 var arreglo = [2, 4, 6, 8, 10]
9
10 for(var indice = 0; indice <= arreglo.length; indice++){
11     //Esta línea imprime el contador (variable indice)
12     //console.log(indice)
13
14     //Esta línea imprime cada elemento del arreglo gracias a la variable indice
15     //console.log(arreglo[indice])
16
17     console.log(`La posición ${indice} contiene al elemento ${arreglo[indice]}`)
18
19 }
```

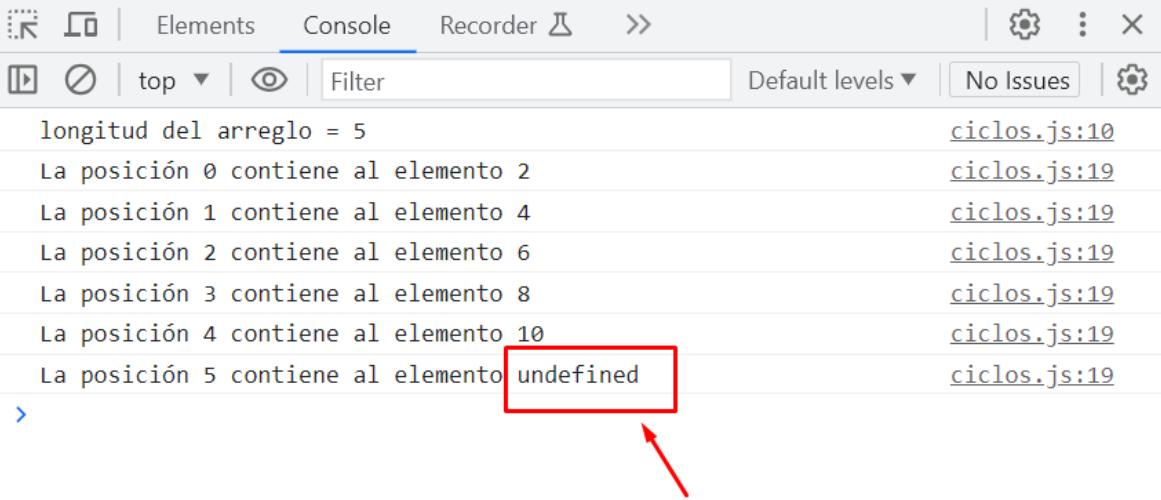
En consola nos mostrará que el último elemento está indefinido:

ⓘ DevTools is now available in Spanish!

Always match Chrome's language

Switch DevTools to Spanish

Don't show again



```
longitud del arreglo = 5
La posición 0 contiene al elemento 2
La posición 1 contiene al elemento 4
La posición 2 contiene al elemento 6
La posición 3 contiene al elemento 8
La posición 4 contiene al elemento 10
La posición 5 contiene al elemento undefined
```

Por ese motivo, es necesario corregir también el operador de comparación, el ciclo debe ir hasta que se MENOR a 5:

```
8 var arreglo = [2, 4, 6, 8, 10]
9
10 console.log(`longitud del arreglo = ${arreglo.length}`)
11
12 for(var indice = 0; indice < arreglo.length; indice++){
13     //Esta linea imprime el contador (variable indice)
14     //console.log(indice)
15
16     //Esta linea imprime cada elemento del arreglo gracias a la variable indice
17     //console.log(arreglo[indice])
18
19     console.log(`La posición ${indice} contiene al elemento ${arreglo[indice]}`)
20
21 }
```

Y con eso se solucionará el problema:

ⓘ DevTools is now available in Spanish!

Always match Chrome's language **Switch DevTools to Spanish** Don't show again

Elements Console Recorder »

top Filter Default levels No Issues

```
longitud del arreglo = 5
La posición 0 contiene al elemento 2
La posición 1 contiene al elemento 4
La posición 2 contiene al elemento 6
La posición 3 contiene al elemento 8
La posición 4 contiene al elemento 10
```

ciclos.js:10
ciclos.js:19
ciclos.js:19
ciclos.js:19
ciclos.js:19
ciclos.js:19

### Ciclo FOR con arreglo de DOS dimensiones

En el mismo archivo .js comentar el código digitado hasta el momento y definir el siguiente arreglo:

```
js > ciclos.js > ...
14     //console.log(indice)
15
16     //Esta línea imprime cada elemento del arreglo gracias a la variable indice
17     //console.log(arreglo[indice])
18
19     console.log(`La posición ${indice} contiene al elemento ${arreglo[indice]}`)
20
21 }
22 */
23 var arregloDosDimensiones =[  
24     [9, 'falcao', 35, 'monaco'],
25     [10, 'james', 30, 'sao pablo'],
26     [11, 'cuadrado', 32, 'inter'],
27     [1, 'ospina', 35, 'alnasar']
28 ]
29
30 console.log(arregloDosDimensiones)
```

Al imprimir se muestra el contenido del arreglo, fíjese en los índices del arreglo principal:

ⓘ DevTools is now available in Spanish!

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. A red arrow points from the text 'Ahora, fíjese en los índices de los arreglos hijos:' to the index '0' of the first inner array, which is highlighted with a red box. The console output is as follows:

```
▼ (4) [Array(4), Array(4), Array(4), Array(4)] i ciclos.js:30
  ▶ 0: (4) [9, 'falcao', 35, 'monaco']
  ▶ 1: (4) [10, 'james', 30, 'sao pablo']
  ▶ 2: (4) [11, 'cuadrado', 32, 'inter']
  ▶ 3: (4) [1, 'ospina', 35, 'alnasar']
    length: 4
  ▶ [[Prototype]]: Array(0)
```

Ahora, fíjese en los índices de los arreglos hijos:

The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. A red box highlights the index '0' of the first inner array. Another red box highlights the index '0' of the first element within that inner array. The console output is as follows:

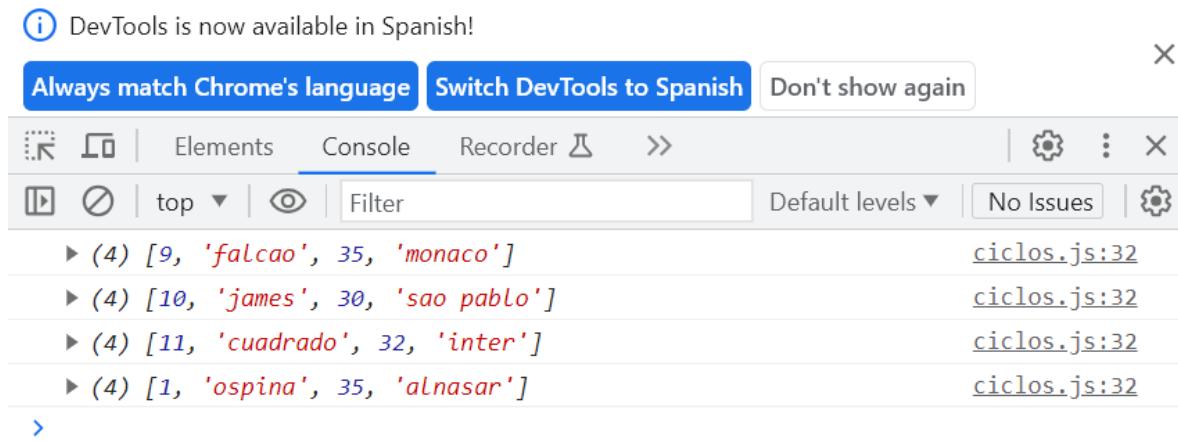
```
ⓘ DevTools is now available in Spanish!
Always match Chrome's language Switch DevTools to Spanish Don't show again
X
Console Elements Recorder >
Default levels No Issues
ciclos.js:30
▼ (4) [Array(4), Array(4), Array(4), Array(4)] i
  ▼ 0: Array(4)
    0: 9
    1: "falcao"
    2: 35
    3: "monaco"
    length: 4
    ▶ [[Prototype]]: Array(0)
  ▶ 1: (4) [10, 'james', 30, 'sao pablo']
  ▶ 2: (4) [11, 'cuadrado', 32, 'inter']
  ▶ 3: (4) [1, 'ospina', 35, 'alnasar']
    length: 4
  ▶ [[Prototype]]: Array(0)
```

Para imprimir uno a uno los elementos del arreglo principal, es necesario usar un ciclo:

```

23 var arregloDosDimensiones =[
24   [9, 'falcao', 35, 'monaco'],
25   [10, 'james', 30, 'sao pablo'],
26   [11, 'cuadrado', 32, 'inter'],
27   [1, 'ospina', 35, 'alnasar']
28 ]
29
30 for(var indice = 0; indice < arregloDosDimensiones.length; indice++){
31   console.log(arregloDosDimensiones[indice])
32 }
33
34 }
```

Resultado:



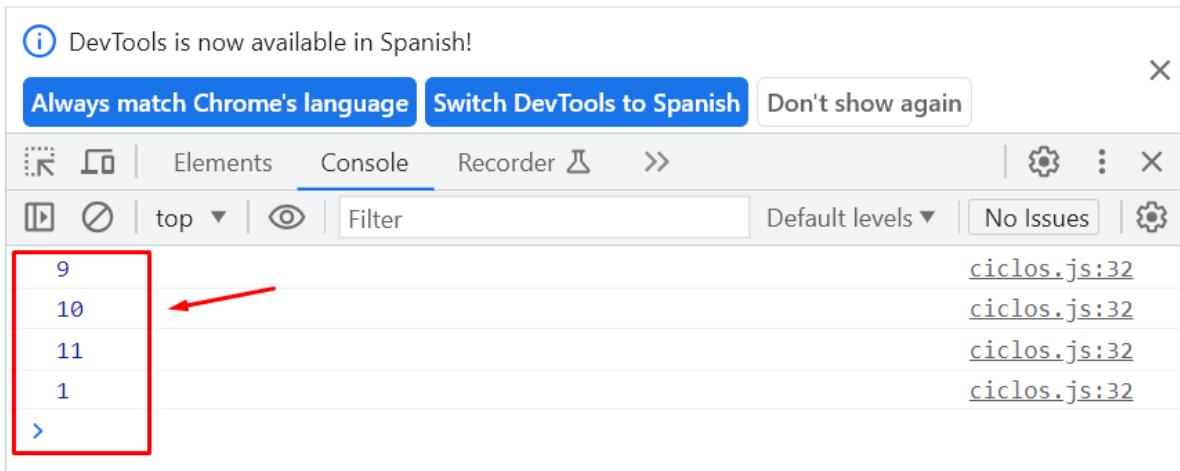
En este ejemplo, el arreglo principal contiene la información de unos jugadores de futbol, primer dato de cada arreglo HIJO corresponde al número de la CAMISETA, el segundo dato corresponde al NOMBRE del jugador, el tercer dato corresponde a la EDAD del jugador, y el cuarto dato corresponde al CLUB donde juega.

Si quisiera imprimir el número de las camisetas de los jugadores hay que tener en cuenta que este dato se encuentra en la posición cero (0) de cada arreglo HIJO:

```

30 for(var indice = 0; indice < arregloDosDimensiones.length; indice++){
31
32   console.log(arregloDosDimensiones[indice][0])
33 }
34 }
```

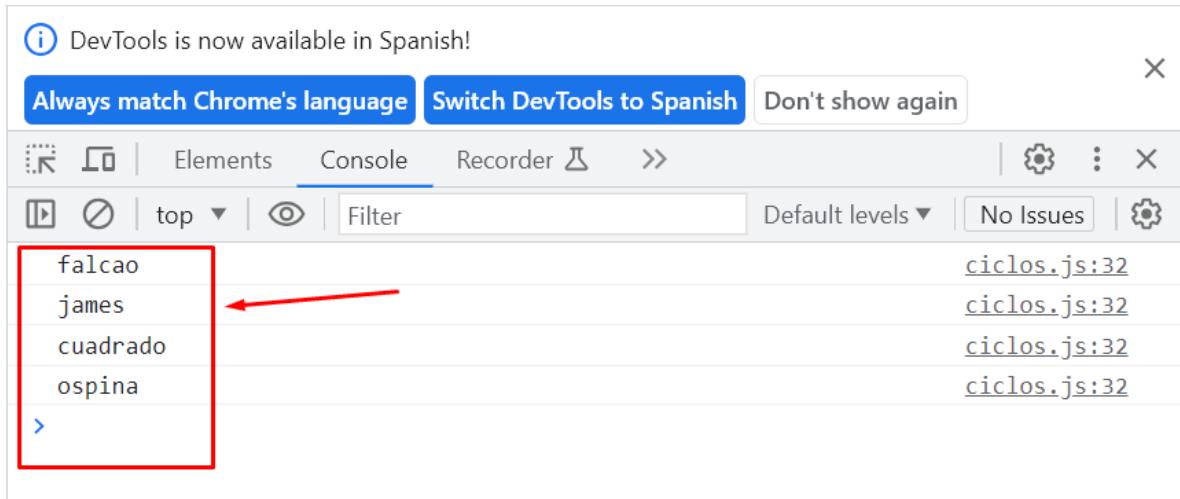
Resultado:



Si quisiera imprimir los nombres de los jugadores se utilizaría el índice uno (1):

```
for(var indice = 0; indice < arregloDosDimensiones.length; indice++){  
    console.log(arregloDosDimensiones[indice][1])  
}
```

Resultado:



### Construcción de una tabla con ciclo FOR

En archivo HTML:

```

14     <table border="1">
15         <thead>
16             <tr>
17                 <th>números</th>
18             </tr>
19         </thead>
20         <tbody id="tabla">
21             <!-- Aquí van las filas construidas desde JavaScript -->
22         </tbody>
23     </table>

```

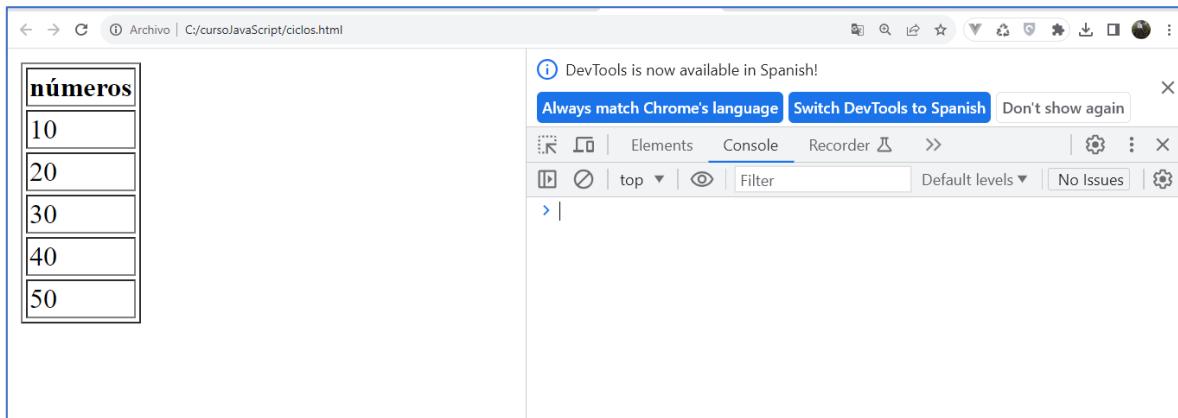
En el archivo JS:

```

34 //Defino arreglo
35 var arreglo = [10, 20, 30, 40, 50]
36 //Defino una cadena de texto vacía
37 var cadena = ""
38 for(var indice = 0; indice < arreglo.length; indice++){
39     //En el ciclo, construyo una fila para la tabla y se va
        concatenando
40     cadena = cadena + `<tr><td>${arreglo[indice]}</td></tr>`
41 }
43 //Enviar las filas construidas al cuerpo de la tabla con id tabla
44 document.getElementById('tabla').innerHTML = cadena

```

Resultado:



## Ejercicios de práctica:

Instructor Jorge E. Andrade C.  
 jandradec@sena.edu.co  
 Centro Agropecuario La Granja  
 Regional Tolima

1. Usando ciclos FOR, pedir al usuario que digite 5 números para guardarlos en un arreglo, después debe imprimir el contenido del arreglo en una tabla usando `document.getElementById('tabla')`
2. Crear un programa que pida al usuario un número del 1 al 9 y muestre su tabla de multiplicar con el CICLO FOR, si el usuario digita un número diferente, se debe mostrar por defecto la tabla del 1. Para pedir el número al usuario haga uso de un **prompt**.
3. Usando ciclos FOR, pedir al usuario que digite 10 números para guardarlos en un arreglo y después organizar en otros dos arreglos los números pares e impares respectivamente.
4. Usando ciclos FOR y una matriz, pedir al usuario que digite 3 lenguajes de programación que conozca y que califique de 1 a 10 el conocimiento que tiene sobre ellos. El arreglo debe parecerse a lo siguiente:

```
> lenguajes
< ▼ (3) [Array(2), Array(2), Array(2)] ⓘ
  ► 0: (2) ['javascript', 8]
  ► 1: (2) ['php', 9]
  ► 2: (2) ['java', 5]
    length: 3
  ► [[Prototype]]: Array(0)
>
```

## Bucle WHILE (Ciclo Mientras)

```
js > JS ciclos.js > ...
36 var contador = 0 ←
37
38 while(contador < 5){
39
40     console.log(contador)
41
42     contador++
43
44 }
```

ⓘ DevTools is now available in Spanish!

Always match Chrome's language  Switch DevTools to Spanish  Don't show again

Elements Console Recorder  Filter Default levels  No Issues

```
0 ciclos.js:40
1 ciclos.js:40
2 ciclos.js:40
3 ciclos.js:40
4 ciclos.js:40
```

## Ejercicio de práctica:

1. Crear un programa que pida el usuario un número del 1 al 9 y muestre su tabla de multiplicar con el CICLO WHILE, si el usuario digita un número diferente, se debe mostrar por defecto la tabla del 1. Para pedir el número al usuario haga uso de un **prompt**.

## Funciones y eventos:

Una función en programación es un bloque de código o un conjunto de varias líneas de código que realizan una tarea específica y que al ejecutarse retorna o no un valor.

Para mayor comprensión observar el video **Funciones en programación** (<https://www.youtube.com/watch?v=MiH3pbP4EFc>) del autor canal en YouTube **Fernando Herrera**. El video estará disponible en el LMS. De igual modo el instructor lo compartirá a través de WhatsApp.

**FUNCIONES**

Una función es un conjunto de líneas de código que realizan una tarea específica y pueden retornar algo.

```
function saludar () {  
    //  
}  
function saludar ( ) {  
    //  
}  
saludar ();
```

HOLA

PARÁMETRO

Funciones en programación

4447 vistas • 16 may. 2018

164 5 COMPARTIR GUARDAR ...

## Proyectos:

### Calculadora

Ejercicio:

Crear dos archivos, uno para el html y el otro para el JS así:

The screenshot shows a code editor interface with the following details:

- Explorer View:** Shows a folder named "CURSOJAVASCRIPT" containing "img" (with "logoSena.png"), "js" (with "calculadora.js", "ciclos.js", "condicionales.js", "dom.js", "operadores.js", and "variables.js"), and "html" (with "calculadora.html", "ciclos.html", and "index.html").
- Editor View:** The file "calculadora.html" is open. The code is as follows:

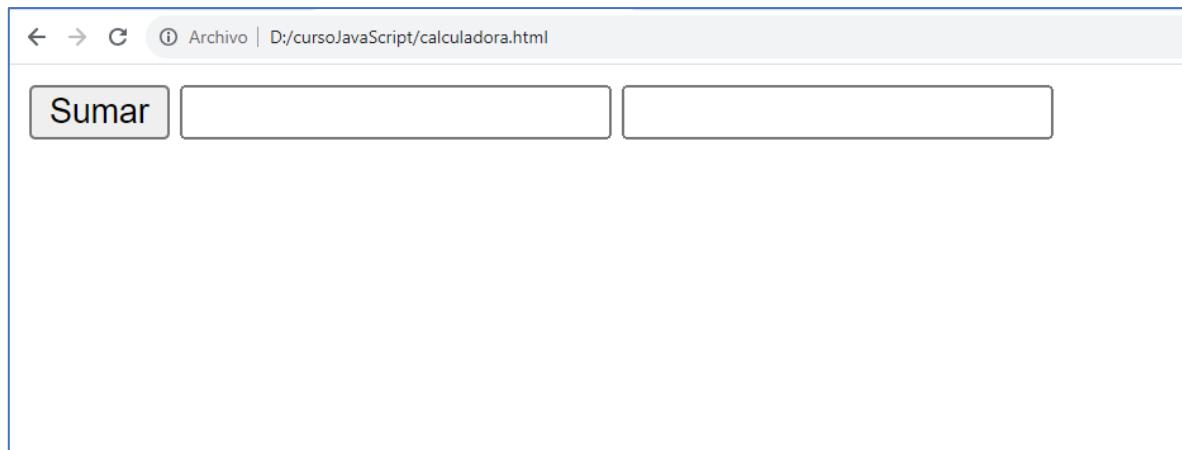
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <script defer text="javascript" src="js/calculadora.js"></script>
<title>Document</title>
</head>
<body>
</body>
</html>
```

- Toolbars and Status Bar:** The status bar at the top right shows "cursоСJavaScript".

Editar el HTML:

```
calculadora.html X JS calculadora.js
calculadora.html > html > body
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6
7      <script defer text="javascript" src="js/calculadora.js"></script>
8
9      <title>Document</title>
10 </head>
11 <body>
12
13     <button>Sumar</button>
14     <input type="number" id="numero1">
15     <input type="number" id="numero2">
16
17     <h1 id="resultado"></h1>
18
19 </body>
20 </html>
```

Resultado:



```
calculadora.html | JS calculadora.js X
js > JS calculadora.js > sumar
1 function sumar(){
2     num1 = document.getElementById('numero1').value
3     num2 = document.getElementById('numero2').value
4
5     resultado = num1 + num2
6
7     document.getElementById('resultado').innerText = resultado
8 }
```

```
calculadora.html X | JS calculadora.js
calculadora.html > html > body > button
1 <html lang="en">
2 <head>
3     <meta charset="UTF-8">
4     <meta name="viewport" content="width=device-width, initial-scale=1.0">
5
6     <script defer text="javascript" src="js/calculadora.js"></script>
7
8     <title>Document</title>
9 </head>
10 <body>
11
12     <button onclick="sumar()">Sumar</button>
13
14     <input type="number" id="numero1">
15     <input type="number" id="numero2">
16
17     <h1 id="resultado"></h1>
18
19 </body>
20 </html>
```

← → ⌛ Archivo | D:/cursoJavaScript/calculadora.html

Sumar 2 4

24

Recuerde que para obtener el resultado correcto debe cambiar el tipo de dato, en este caso se recibe un texto y para convertir a número se usa la instrucción parseInt:

```
calculadora.html JS calculadora.js X
js > JS calculadora.js > sumar
1   function sumar(){
2     num1 = document.getElementById('numero1').value
3     num2 = document.getElementById('numero2').value
4
5     resultado = parseInt(num1) + parseInt(num2)
6
7     document.getElementById('resultado').innerText = resultado
8 }
```

Resultado:

← → ⌛ Archivo | D:/cursoJavaScript/calculadora.html

Sumar 3 6

9

## Ejercicio de práctica:

Instructor Jorge E. Andrade C.  
jandradec@sena.edu.co  
Centro Agropecuario La Granja  
Regional Tolima

Crear tres (3) botones más para las operaciones matemáticas básicas que faltan (Resta, Multiplicación y División). Usar el mismo div para mostrar el resultado y los mismos input para obtener los números.

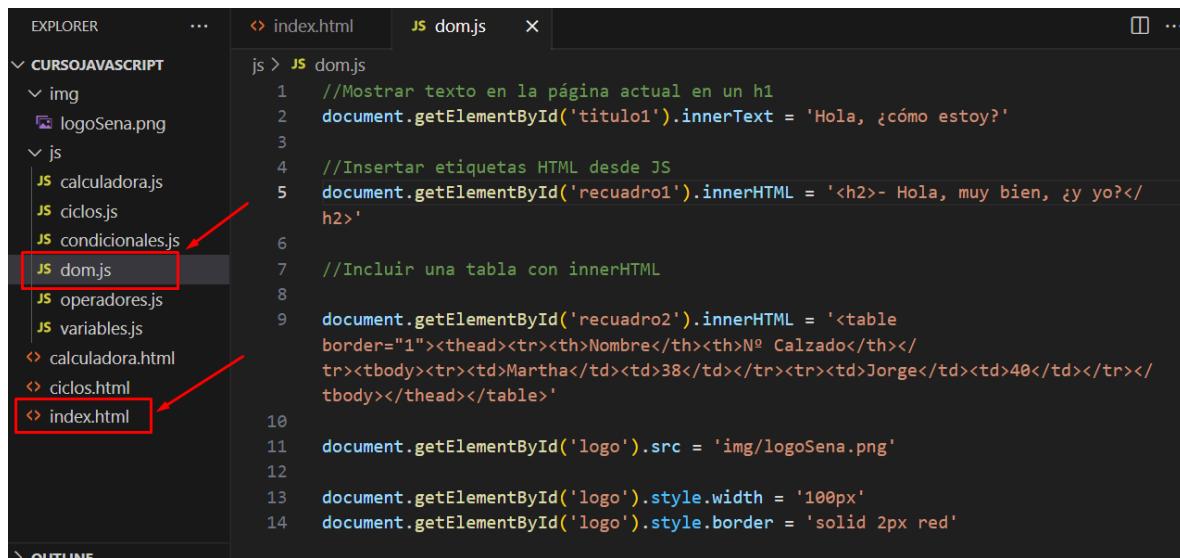
## Eventos

En Javascript existe un concepto llamado evento, que no es más que una notificación de que alguna característica interesante acaba de ocurrir, generalmente relacionada con el usuario que navega por la página.

Dichas características pueden ser muy variadas:

- Click de ratón del usuario sobre un elemento de la página
- Pulso de una tecla específica del teclado
- Reproducción de un archivo de audio/video
- Scroll de ratón sobre un elemento de la página
- El usuario ha activado la opción «Imprimir página»

Vuelva abrir los archivos index.html y dom.js



```
//Mostrar texto en la página actual en un h1
document.getElementById('titulo1').innerText = 'Hola, ¿cómo estoy?'
//Insertar etiquetas HTML desde JS
document.getElementById('recuadro1').innerHTML = '<h2>- Hola, muy bien, ¿y yo?</h2>'
//Incluir una tabla con innerHTML
document.getElementById('recuadro2').innerHTML = '<table border="1"><thead><tr><th>Nombre</th><th>Nº Calzado</th></tr></thead><tbody><tr><td>Martha</td><td>38</td></tr><tr><td>Jorge</td><td>40</td></tr></tbody></table>'

document.getElementById('logo').src = 'img/logoSena.png'
document.getElementById('logo').style.width = '100px'
document.getElementById('logo').style.border = 'solid 2px red'
```

# Hola, ¿cómo estoy?

- Hola, muy bien, ¿y yo?

Nombre	Nº Calzado
Martha	38
Jorge	40



**Evento onclick:** al dar un clic sobre un elemento

```
index.html x js dom.js
index.html > html > body > div#calculadora
19     <div id="recuadro2">
20
21     </div>
22
23     <img src="" id="logo">
24
25     <div id="calculadora">
26
27     </div>
28
29     <button onclick="cambiarColor('red')>Borde Rojo</button>
30     <button onclick="cambiarColor('blue')>Borde Azul</button>
31
32 </body>
33 </html>
```

### Función cambiarColor()

```
index.html x js dom.js x
js > js dom.js > cambiarColor
8
9     document.getElementById('recuadro2').innerHTML = '<table
border="1"><thead><tr><th>Nombre</th><th>Nº Calzado</th></tr>
<tbody><tr><td>Martha</td><td>38</td></tr><tr><td>Jorge</td><td>40</td></tr>
</tbody></table>'
10
11    document.getElementById('logo').src = 'img/logoSena.png'
12
13    document.getElementById('logo').style.width = '100px'
14    document.getElementById('logo').style.border = 'solid 2px red'
15
16    function cambiarColor(color)[
17        document.getElementById('logo').style.border = `solid 2px ${color}`
18    ]
19
20 ]
```

Resultado:

← → C Archivo | D:/cursoJavaScript/index.html

# Hola, ¿cómo estoy?

- Hola, muy bien, ¿y yo?

Nombre	Nº Calzado
Martha	38
Jorge	40

Borde Rojo      Borde Azul

Cambiar Logo:

```

index.html X JS dom.js

index.html > html > body > button
26
27     </div>
28
29     <button onclick="cambiarColor('red')>Borde Rojo</button>
30     <button onclick="cambiarColor('blue')>Borde Azul</button>
31
32     <!-- Botones para cambiar logo -->
33     <button onclick="cambiarLogo(1)>Logo 1</button>
34     <button onclick="cambiarLogo(2)>Logo 2</button>
35
36 </body>
37 </html>

```

Buscar en Google un logo diferente y guardarlo en la carpeta de imágenes del proyecto:



Instructor Jorge E. Andrade C.  
jandradec@sena.edu.co  
Centro Agropecuario La Granja  
Regional Tolima

The screenshot shows a code editor with two tabs: 'index.html' and 'JS dom.js'. The 'JS dom.js' tab is active and contains the following code:

```
js > JS dom.js > cambiarLogo
16 function cambiarColor(color){
17
18     document.getElementById('logo').style.border = `solid 2px ${color}`
19
20 }
21
22 function cambiarLogo(imagen){
23
24     if(imagen == 1){
25
26         document.getElementById('logo').src = 'img/logoSena.png'
27
28     }else if(imagen == 2){
29
30         document.getElementById('logo').src = 'img/logonegro.png'
31
32     }
33
34 }
```

A red rectangular box highlights the code from line 22 to line 34. Two red arrows point from the text 'document.getElementById('logo').src =' to the string 'img/logoSena.png' at line 26 and 'img/logonegro.png' at line 30.

Resultado:



Archivo |

D:/cursoJavaScript/index.html

# Hola, ¿cómo estoy?

- Hola, muy bien, ¿y yo?

Nombre	Nº Calzado
Martha	38
Jorge	40



Borde Rojo

Borde Azul

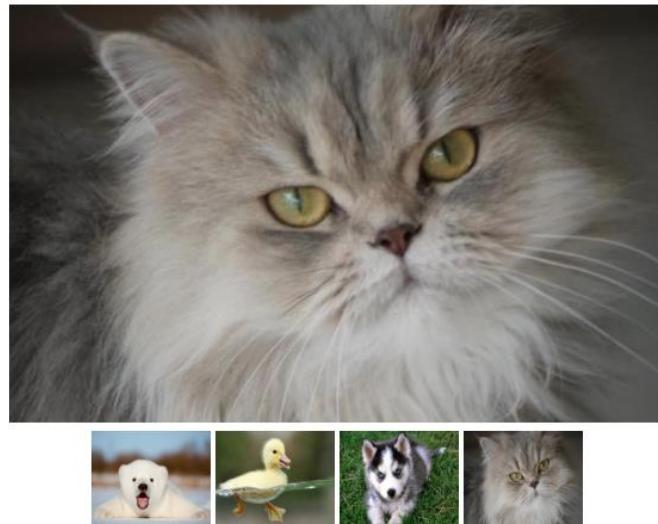
Logo 1

Logo 2

## Galería de imágenes

## Ejercicio de práctica:

1. El instructor muestra un ejemplo de una galería de fotos interactiva, la cual ejecuta una acción cuando el usuario da clic en una imagen. El aprendiz debe generar el código de la galería, puede ser de la temática deseada, no necesariamente los animales.



## Gato Misingo



## Pato Aparato

2. Incluir botones de siguiente y atrás:



**Oso Cariñoso**



3. Consultar la descripción de los siguientes eventos:
  - a. Ondblclick
  - b. Onkeydown
  - c. Onkeyup
  - d. Onload
  - e. Onmousedown
  - f. Onmousemove
  - g. Onmouseout
  - h. Onmouseover
  - i. Onsubmit
4. Consultar el funcionamiento de la instrucción addEventListener y presentar un ejemplo