

# BLOQUE DIDÁCTICO B: MANEJO DE LA SINTAXIS DEL LENGUAJE



Román-Herrera, J.C.

## DESARROLLO WEB EN ENTORNOS CLIENTE



# TEMARIO DEL BLOQUE DIDÁCTICO B

1. Variables.
2. Tipos de datos.
3. Asignaciones.
4. Operadores.
- 5. Comentarios al código.**
6. Sentencias.
7. Decisiones.
8. Bucles.



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)



## OBJETIVOS DEL BLOQUE B

- Dominar la sintaxis básica del lenguaje.
- Comprender y utilizar distintos tipos de variables y operadores presentes en el lenguaje JavaScript.
- Conocer las diferentes sentencias condicionales de JavaScript y saber realizar operaciones complejas con ellas.

## 5. COMENTARIOS AL CÓDIGO



Comentar el código es una tarea documental **IMPORTANTÍSIMA**

Su uso es indispensable en trabajos en grupo, pero abusar de ellos es algo no recomendable...

INFO ABOUT RIGHTS

2110019392312

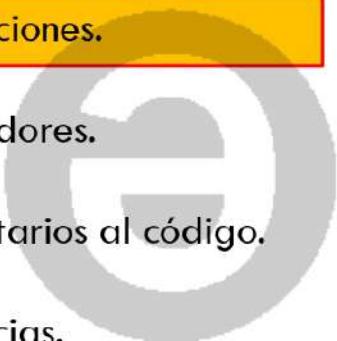
En java Script podemos usar dos técnicas:

[www.safecreative.org/work](http://www.safecreative.org/work)

- // comentario en una linea
- /\* comentario en varias lineas \*/

# TEMARIO DEL BLOQUE DIDÁCTICO B

1. Variables.
2. Tipos de datos.
3. Asignaciones.
4. Operadores.
5. Comentarios al código.
6. Sentencias.
7. Decisiones.
8. Bucles.



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org](http://www.safecreative.org)



## OBJETIVOS DEL BLOQUE B

- Dominar la sintaxis básica del lenguaje.
- Comprender y utilizar distintos tipos de variables y operadores presentes en el lenguaje JavaScript.
- Conocer las diferentes sentencias condicionales de JavaScript y saber realizar operaciones complejas con ellas.

# 1. Y 3. VARIABLES Y ASIGNACIÓN



## Palabras reservadas en JavaScript

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

¿Para qué sirven? → Sirven para cargar datos en memoria

Podemos declarar variables en JavaScript mediante:

- **let** → Declara una variable local de bloque editable
- **var** → Declara una variable global ó de bloque editable
- **const** → Declara una variable actual no editable

¿Cuáles son los criterios para definir variables? (reglas)

- Tienen que empezar por:
  - Letra
  - Guion bajo
  - Símbolo de dólar
- No puede coincidir con palabras reservadas

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)



# 1. Y 3. VARIABLES Y ASIGNACIÓN



Solución

Crea un archivo HTML con JS. Define tres variables (X, Y, Z) mediante los métodos let, var y const. Asignales tres valores diferentes, por ejemplo, 1, 2 y 3. Reproduce por consola las tres variables.

INFO ABOUT RIGHTS

2110019392312

www.safecreative.org

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <script>
      var X = 1;
      let Y = 2;
      const Z = 3;
      console.log(X,Y,Z);
    </script>
  </body>
</html>
```

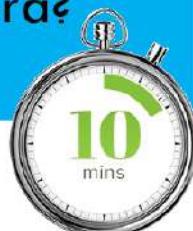
# 1. Y 3. VARIABLES Y ASIGNACIÓN



Solución

Aprovecha el código anterior, y define debajo de Z, un bloque, y dentro de él crea tres nuevas variables(x, y, z) mediante el uso de los tres métodos aprendidos.

Ahora saca por consola desde dentro del bloque las variables iniciales, y las que has acabado de crear, ¿qué ha sucedido? ¿y si ahora las sacas desde fuera? ¿qué sucede?



```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <script>
      var X = 1;
      let Y = 2;
      const Z = 3;
      {
        var x = 4;
        let y = 5;
        const z = 6;
        console.log(X,Y,Z);
        console.log(x,y,z);
      }
    </script>
  </body>
</html>
```

Se reproducen bien

```
<!DOCTYPE html>
<html>
  <head></head>
  <body>
    <script>
      var X = 1;
      let Y = 2;
      const Z = 3;
      {
        var x = 4;
        let y = 5;
        const z = 6;
      }
      console.log(X,Y,Z);
      console.log(x,y,z);
    </script>
  </body>
</html>
```

Solo se reproducen las variables globales (X,Y,Z)

# 1. Y 3. VARIABLES Y ASIGNACIÓN



¿Cuántos tipos de variables existen? → Dos tipos en función del nivel de acceso.

Podemos encontrarnos los siguientes tipos de variables:

- Locales → Definidas dentro de un bloque de código, solo existen dentro de ahí
- Globales → Definidas fuera de un bloque de código, por lo que existen siempre

Ejemplo:

A screenshot of a browser's developer tools, specifically the Console tab. The URL bar shows "www.safecreative.org/work". The console output is as follows:

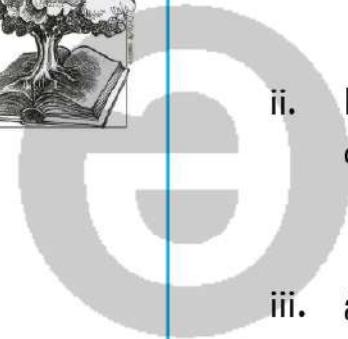
```
> let cuenta = 4; //variable global
  if (cuenta>3){ let saludo="hola amigos"; /*Variable local*/
    console.log(saludo);} else {let saludo="adios amigos"; /*Variable local*/
    console.log(saludo);}
  hola amigos
< undefined
> |
```

The code defines a global variable "cuenta" and an inner variable "saludo" which is only accessible within its own scope. The output shows the value of "saludo" as "hola amigos".

# 1. Y 3. VARIABLES Y ASIGNACIÓN

Realiza por parejas las siguientes operaciones por consola:

- i. Declarar la variable “dato” de tipo texto con la instrucción let y cuyo contenido sea “Soy una variable”
- ii. Prueba a declarar de nuevo “dato” de tipo texto con la instrucción let y cuyo contenido sea ahora “20” ¿Qué ha sucedido? (se tienen que declarar i e ii a la vez)  
**2110019392312**
- iii. ¿Cómo lo harías para cambiarle el valor?  
[www.safecreative.org/work](http://www.safecreative.org/work)





# 1. Y 3. VARIABLES Y ASIGNACIÓN



**SOLUCIÓN**

Realiza por parejas las siguientes operaciones por consola:

- i. Declarar la variable “dato” con la instrucción let y cuyo contenido sea “Soy una variable”  
`let dato;`  
`dato = “Soy una variable”`
- ii. Prueba a declarar de nuevo “dato” con la instrucción let y cuyo contenido sea ahora “20” ¿Qué ha sucedido?  
`let dato;`  
`dato = “Soy una variable”`  
`let dato;`  
`dato = “20”`

**Da un error porque no lo puedo definir dos veces la misma variable**
- iii. ¿Cómo lo harías para cambiarle el valor?  
`let dato;`  
`dato = “Soy una variable”`  
`dato = “20”`

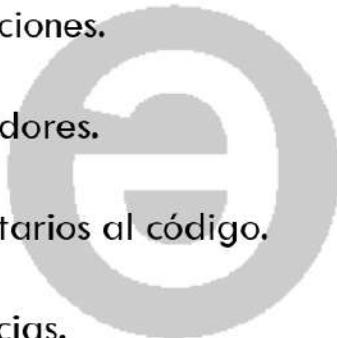
INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# TEMARIO DEL BLOQUE DIDÁCTICO B

1. Variables.
2. Tipos de datos.
3. Asignaciones.
4. Operadores.
5. Comentarios al código.
6. Sentencias.
7. Decisiones.
8. Bucles.



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org](http://www.safecreative.org)



## OBJETIVOS DEL BLOQUE B

- Dominar la sintaxis básica del lenguaje.
- Comprender y utilizar distintos tipos de variables y operadores presentes en el lenguaje JavaScript.
- Conocer las diferentes sentencias condicionales de JavaScript y saber realizar operaciones complejas con ellas.

## 2. TIPOS DE DATOS

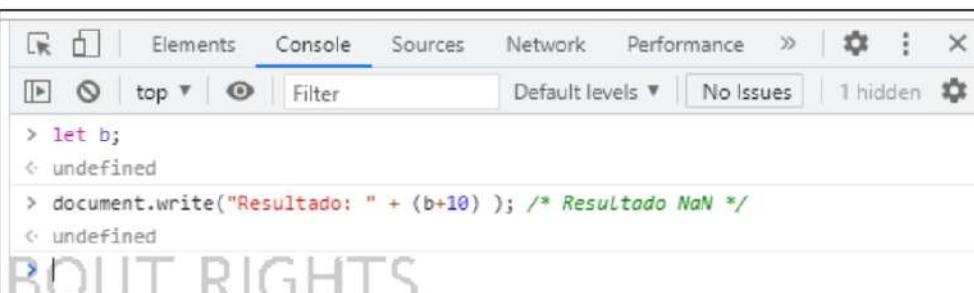
¿Qué tipos de datos existen en JavaScript?



- **!** → Para aquellas variables que **no tienen valor asignado**.
- **null** → Para **valores desconocidos**.
- **boolean** → Para **datos lógicos** con valor True False.
- **number** → Para datos **enteros y decimal**.
- **string** → Para **cadenas de texto**.
- **Symbol** → Para datos **inmutables y únicos**.
- **bigint** → Para números **enteros y de longitud arbitraria**.
- **object** → Para estructuras de datos **complejas**.

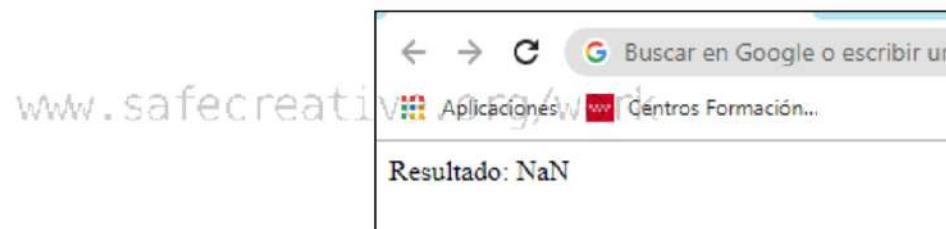
## 2. TIPOS DE DATOS

Aquellas **variables** que **no se ha asignado valor** son los **undefined**

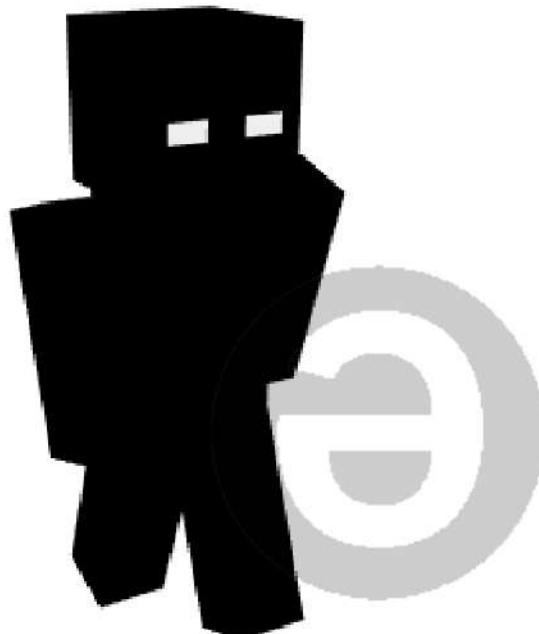


```
let b;
undefined
document.write("Resultado: " + (b+10)); /* Resultado NaN */
undefined
NaN
```

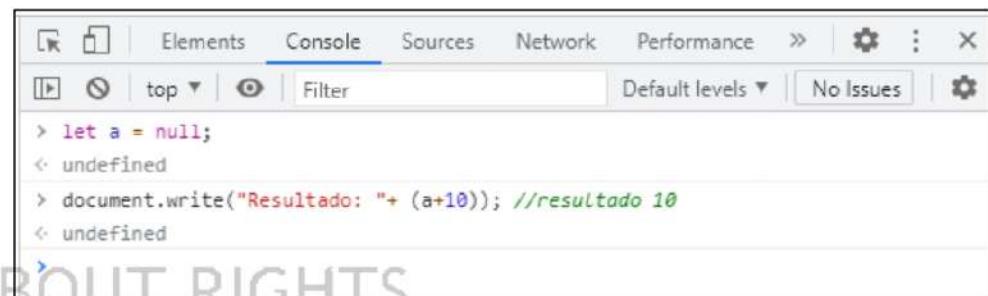
2110019392312



## 2. TIPOS DE DATOS



Aquellas **variables** que representan un valor **nulo** o **vacío** son los **null**



```
Elements Console Sources Network Performance > | X
top ▾ Filter Default levels ▾ No Issues | X
let a = null;
undefined
> document.write("Resultado: "+ (a+10)); //resultado 10
undefined
```

INFO ABOUT RIGHTS

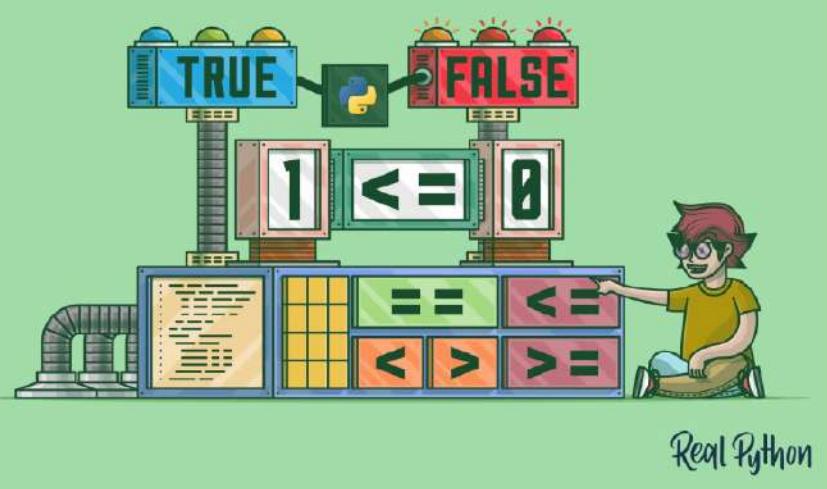
2110019392312



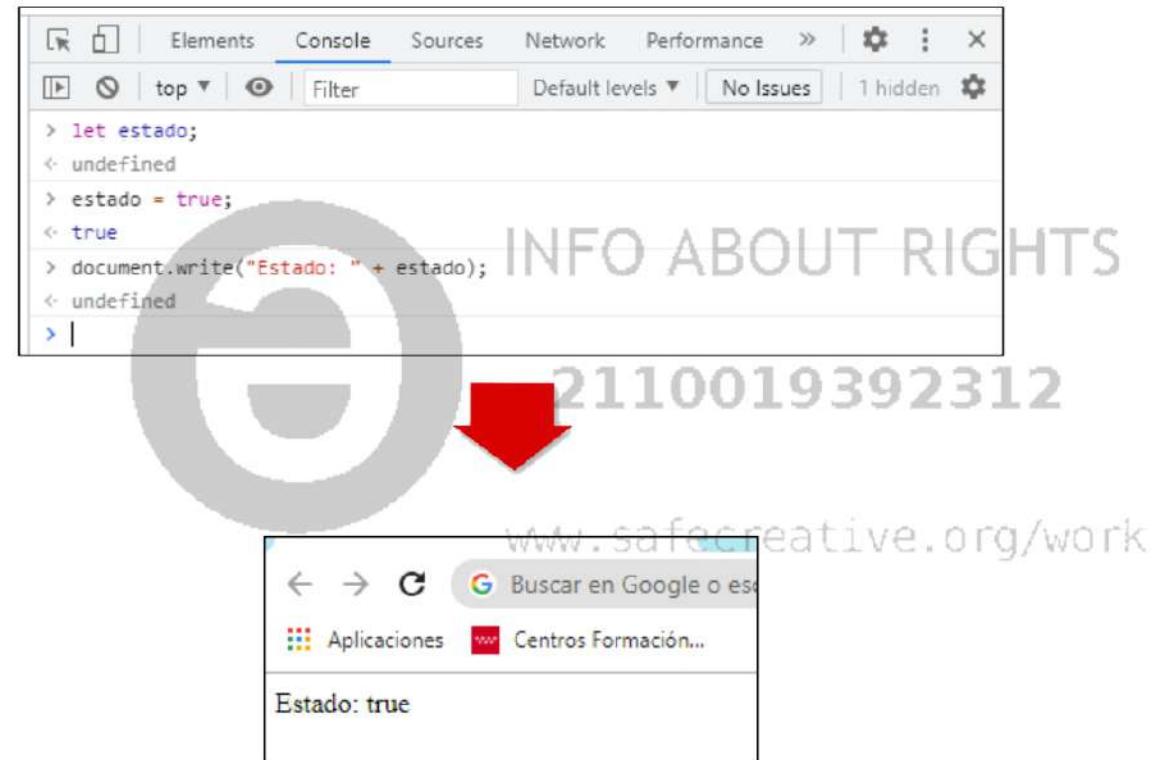
https://www.google.com/search? ← → C google.com/s

Resultado: 10

## 2. TIPOS DE DATOS



Aquellas **variables** que **solo** pueden tener **si** ó **no** son los **boolean**



INFO ABOUT RIGHTS  
2110019392312

Console

```
> let estado;
< undefined
> estado = true;
< true
> document.write("Estado: " + estado);
< undefined
> |
```

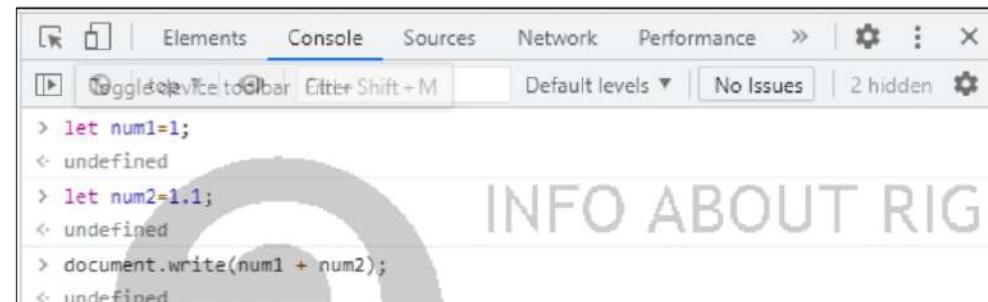
www.safecreative.org/work

Estado: true

## 2. TIPOS DE DATOS



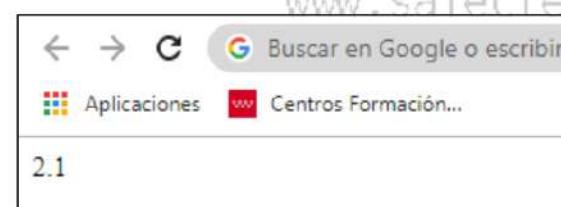
Aquellas **variables** que representan **números enteros o decimales** son las **number**



The screenshot shows a browser's developer tools open to the 'Console' tab. The code entered is:

```
> let num1=1;  
< undefined  
> let num2=1.1;  
< undefined  
> document.write(num1 + num2);  
< undefined
```

A large red arrow points downwards from the console area to a search bar at the bottom of the slide, which contains the URL [www.safecreative.org/work](http://www.safecreative.org/work).



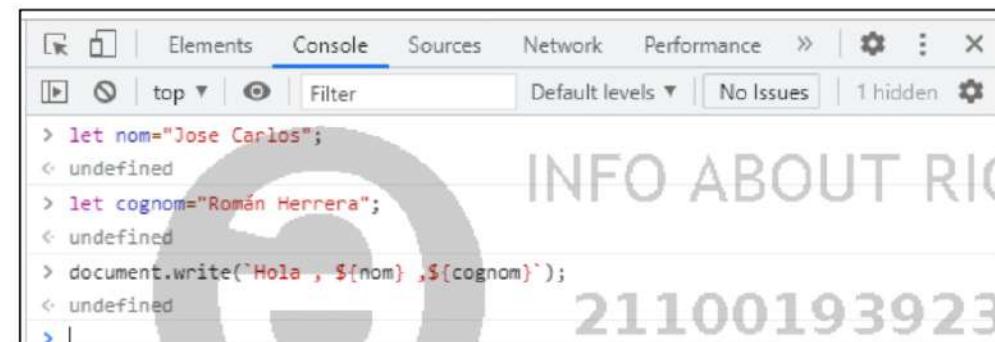
## 2. TIPOS DE DATOS

The lovely Gang night may song long shines  
 Weathered and farwell my heart was beating  
 The rosebush on the morrow the violet beautiful  
 The artist, evening song our love new life  
 To belinda holds her moist so long farewell  
 Now I leave this little hut where my beloved lives  
 Walking now with oiled steps throught the leaves  
 Luna shines throughfut bush and ask zephyr per patre  
 And the high trees bearing her shed incense on the track  
 How beautiful the coolness of this lovely summer night!  
 How the sol fills with happiness in this full place of god!  
 I can scarcely grasp the bliss, yet Heaven, I could shun  
 A thousand nights like this if my darling granted one  
 The lovely Gang night may song long shines  
 Weathered and farwell my heart was beating  
 The rosebush on the morrow the violet beautiful  
 The artist, evening song our love new life  
 To belinda holds her moist so long farewell  
 Now I leave this little hut where my beloved lives  
 Walking now with oiled steps throught the leaves  
 Luna shines throughfut bush and ask zephyr per patre  
 And the high trees bearing her shed incense on the track  
 How beautiful the coolness of this lovely summer night!  
 How the sol fills with happiness in this full place of god!  
 I can scarcely grasp the bliss, yet Heaven, I could shun  
 A thousand nights like this if my darling granted one

Aquellas **variables** que contienen **texto** son los **strings**

Si queremos hacer uso de **variables** que contienen **texto** usaremos la estructura:

**`\${nombre\_variable}`**



```

Elements  Console  Sources  Network  Performance  >  ⚙  :  X
top  Filter  Default levels  No Issues  1 hidden  ⚙
> let nom="Jose Carlos";
< undefined
> let cognom="Román Herrera";
< undefined
> document.write(`Hola , ${nom} , ${cognom}`);
< undefined
>
  
```

INFO ABOUT RIGHTS  
2110019392312



## 2. TIPOS DE DATOS

The lovely Gang night may song long shines  
 Wextened and farwell my heart was beating  
 The rosebush on the morrow the violet beautiful  
 The artist, evening song our love new life  
 To delings holds him morst so long farewell  
 Now I leave this little hut where my beloved lives  
 Walking now with oiled steps througth the leaves  
 Luna shines throught bush and ask zephyr per patre  
 And the high trees bearing her shed incense on the track  
 How beautiful the coolness of this lovely summer night!  
 How the air fills with happiness in this full place of god!  
 I can scarcely grasp the bliss, yet Heaven, I could shun  
 A thousand nights like this if my darling granted one  
 The lovely Gang night may song long shines  
 Wextened and farwell my heart was beating  
 The rosebush on the morrow the violet beautiful  
 The artist, evening song our love new life  
 To delings holds him morst so long farewell  
 Now I leave this little hut where my beloved lives  
 Walking now with oiled steps througth the leaves  
 Luna shines throught bush and ask zephyr per patre  
 And the high trees bearing her shed incense on the track  
 How beautiful the coolness of this lovely summer night!  
 How the air fills with happiness in this full place of god!  
 I can scarcely grasp the bliss, yet Heaven, I could shun  
 A thousand nights like this if my darling granted one

Dentro de los **strings** existen una serie de **caracteres especiales**:

- **\n** → Salto de línea.
- **\r** → Retorno de carro.
- **\t** → Tabulador.
- **\\"** → Carácter \.
- **\'** → Carácter '.
- **\"** → Carácter ".
- **\xnn** → Carácter Unicode de 2 dígitos en **hexadecimal**.
- **\unnnn** → Carácter Unicode de 4 dígitos en **hexadecimal**.

## 2. TIPOS DE DATOS



Los **strings** se manejan en JavaScript mediante el empleo de:

- Comillas dobles → “
- Comilla simple → ‘
- Comilla Invertida → `

Si quiero **sacar el valor de una variable** usar solo **comillas invertidas (`)** y dentro del texto tendré que referenciar a la variable mediante la secuencia **`\${mi\_variable}`**

Ejemplo:

[www.safecreative.org/work](http://www.safecreative.org/work)

```
> let nombre = "Jose Carlos";
  console.log(`hola ${nombre}`);
```

```
hola Jose Carlos
```

```
< undefined
```

## 2. TIPOS DE DATOS



El uso de **strings** suele combinarse con el cálculo, siendo totalmente compatible

Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Uso strings</title>
</head>
<body>
  <script>
    var s1=9;
    var s2=10;
    document.write(`El resultado de sumar S1=${s1} y S2=${s2} es igual a ${s1+s2}`);
  </script>
</body>
</html>
```



Archivo |

C:/Users/jcrh1/Desktop/javascript/strings.html

El resultado de sumar S1=9 y S2=10 es igual a 19

## 2. TIPOS DE DATOS

Text Operations:  
Preprocessing

JavaScript dispone de una serie de métodos que permiten operar con texto:

- `.localeCompare` → Compara strings sin tener en cuenta mayúsculas o minúsculas.
- `.length` → Devuelve la cantidad de caracteres de un texto o string.
- `.charAt` → Extrae el carácter de la posición.
- `.toUpperCase` → Convierte a mayúsculas.
- `.toLowerCase` → Convierte a minúsculas.
- `.indexOf` → Devuelve la primera posición del texto.
- `.lastIndexOf` → Devuelve la última posición del texto a buscar.
- `.endsWith` → Devuelve true si el texto acaba en los caracteres introducidos.
- `.startsWith` → Devuelve true si el texto comienza con los caracteres introducidos.
- `.replace` → Reemplaza un texto por otro.
- `.trim` → Quita espacios en blanco a izquierda y derecha del texto.
- `.slice` → Extrae el texto entre las posiciones introducidas (valores negativos valen).
- `.substring` → Extrae el texto entre las posiciones (sin negativos).
- `.substr` → Extrae el texto en una cantidad de caracteres que definimos.
- `.split` → Divide el texto en un array por un carácter

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.localeCompare`

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Ejemplo Métodos Strings</title>
</head>
<body>
  <script>
    var texto1="jose";
    var texto2="Jose";
    document.write(texto1.localeCompare(texto2));
  </script>
</body>
</html>
```

← → ⌂ Archivo | C:/Users/jcrh1/Desktop/javascript/

Compara el string de `texto1` y `texto2`, pudiendo devolver:

- 0 → Si son exactamente iguales
- 1 → Si el 2º texto es mayor que el 1º
- -1 → Si el 2º texto es menor que el 1º

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método **.length**

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto="La vaca Lola, tiene cola";
        document.write(texto.length);
    </script>
</body>
</html>
```

Los **espacios** se cuentan así como las **comas**

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método **.charAt**

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto="Celia";
        document.write(texto.charAt(1));
    </script>
</body>
</html>
```

El **primer carácter** comienza a numerarse con **0**

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.toUpperCase` y `.toLowerCase`

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var nom="albERTO";
        document.write(`El alumno ${nom.toLowerCase()} debiera ir así: ${nom.toUpperCase()}`);
    </script>
</body>
</html>
```



El alumno alberto debiera ir asi:ALBERTO

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.indexOf` y del método `.lastIndexOf`

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto="El cielo es azul, y solo azul es el cielo";
        document.write(`La palabra azul está en ${texto.indexOf("azul")}) y en ${texto.lastIndexOf("azul")})`);
    </script>
</body>
</html>
```

← → ⌂ Archivo | C:/Users/

La palabra azul está en 12 y en 25

El primer carácter comienza a numerarse con 0

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.startsWith` y del método `.endsWith`

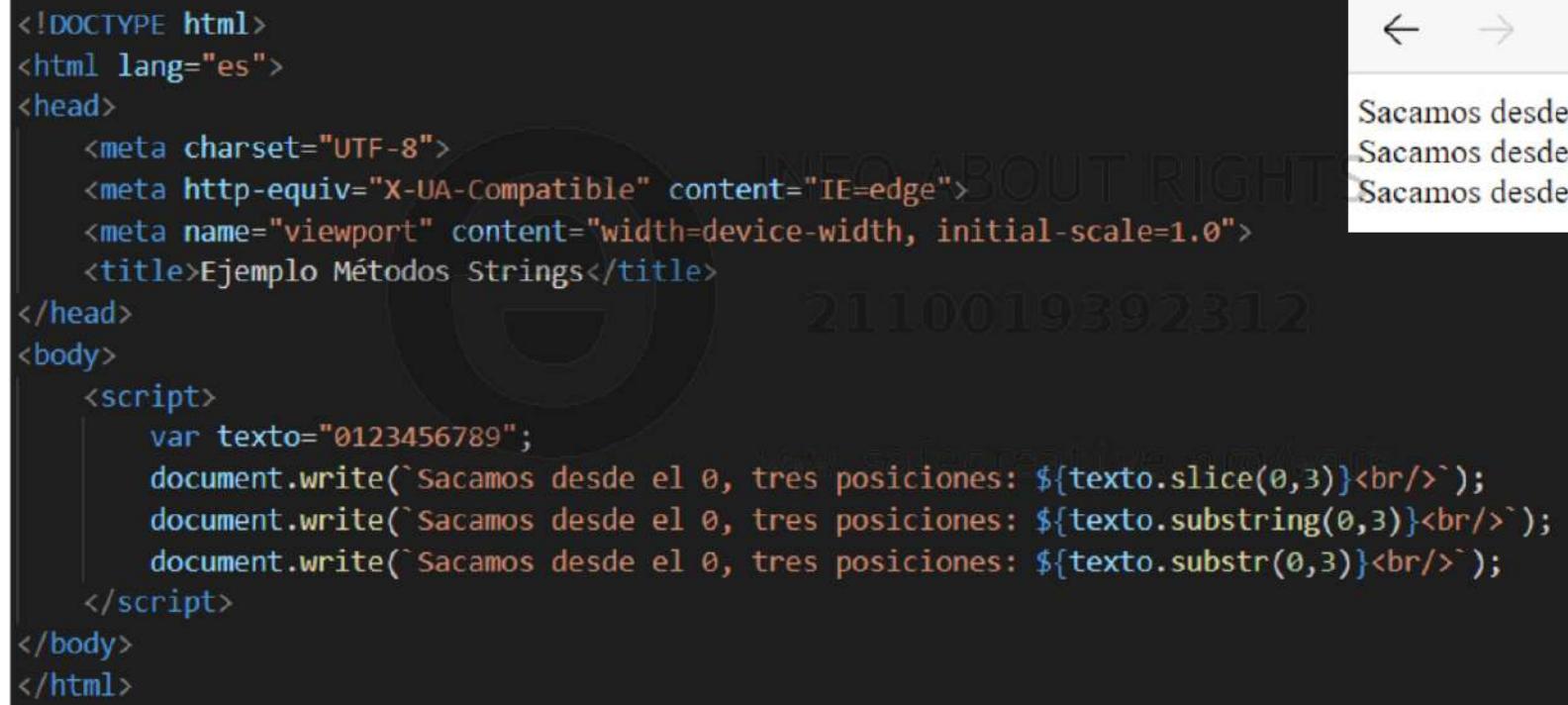
```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto="El cielo es azul, y solo azul es el cielo";
        document.write(`¿El texto comienza por "el"?: ${texto.startsWith("El")}<br/>`);
        document.write(`¿El texto termina por "cielo"?: ${texto.endsWith("cielo")}<br/>`);
    </script>
</body>
</html>
```

← → ⌂ Archivo | C:/Users/jcrh1/Desktop,

¿El texto comienza por "el"?: true  
¿El texto termina por "cielo"?: true

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.slice`, `.substring` y del método `.substr`



```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto="0123456789";
        document.write(`Sacamos desde el 0, tres posiciones: ${texto.slice(0,3)}<br/>`);
        document.write(`Sacamos desde el 0, tres posiciones: ${texto.substring(0,3)}<br/>`);
        document.write(`Sacamos desde el 0, tres posiciones: ${texto.substr(0,3)}<br/>`);
    </script>
</body>
</html>
```

← → ⌂ Archivo | C:/Users/jcrh1/

Sacamos desde el 0, tres posiciones: 012  
Sacamos desde el 0, tres posiciones: 012  
Sacamos desde el 0, tres posiciones: 012

## 2. TIPOS DE DATOS

Vamos a ver un ejemplo del método `.trim` y del método `.split`

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo Métodos Strings</title>
</head>
<body>
    <script>
        var texto=" 0 1 2 3 4 5 6 7 8 9 ";
        document.write(texto.trim().split(" "));
    </script>
</body>
</html>
```



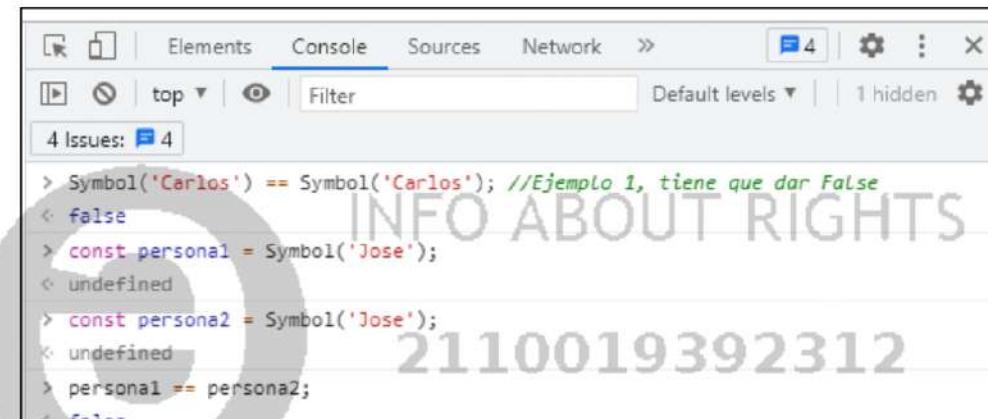
0,1,2,3,4,5,6,7,8,9

## 2. TIPOS DE DATOS



Aquellas **variables** que guardan **identificadores únicos** para los objetos que sirven para identificarlos son los **Symbol**

Estos identificadores se encuentran ocultos a cualquier otro código



```
Elements Console Sources Network > 4 | Filter Default levels | 1 hidden
4 Issues: 4
> Symbol('Carlos') == Symbol('Carlos'); //Ejemplo 1, tiene que dar False
< false
> const personal = Symbol('Jose');
< undefined
> const persona2 = Symbol('Jose');
< undefined
> personal == persona2;
< false
```

www.safecreative.org/work

## 2. TIPOS DE DATOS



**BigInt** es un tipo de numérico que provee soporte a enteros de tamaño arbitrario

Comprendidos entre  $2^{53}$  y -1

INFO ABOUT RIGHTS

Se consigue:

- agregando una n al final del número entero
- llamando a la función **BigInt**

www.safecreative.org/work  
Las operaciones quedan sujetas a:

- Operaciones entre variables del mismo tipo

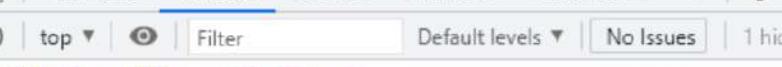
## 2. TIPOS DE DATOS



## INFO ABOUT RIGHTS

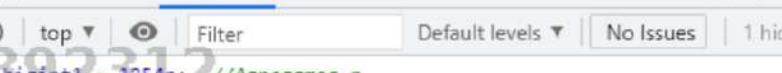
```
> let bigint1 = 1254n;  
> let bigint2 = BigInt(
```

[www.safecreative.org/work](http://www.safecreative.org/work)



The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The console output is as follows:

```
> let bigint1 = 1254n; //Agregamos n
  let bigint2 = BigInt(1254); //Usando La funcion
< undefined
> document.write(bigint1 + bigint2); //Permite operar entre ellos
< undefined
```



The screenshot shows the browser's developer tools open to the 'Console' tab. The URL in the address bar is 'https://www.creative.org/work'. The console displays the following code and its execution results:

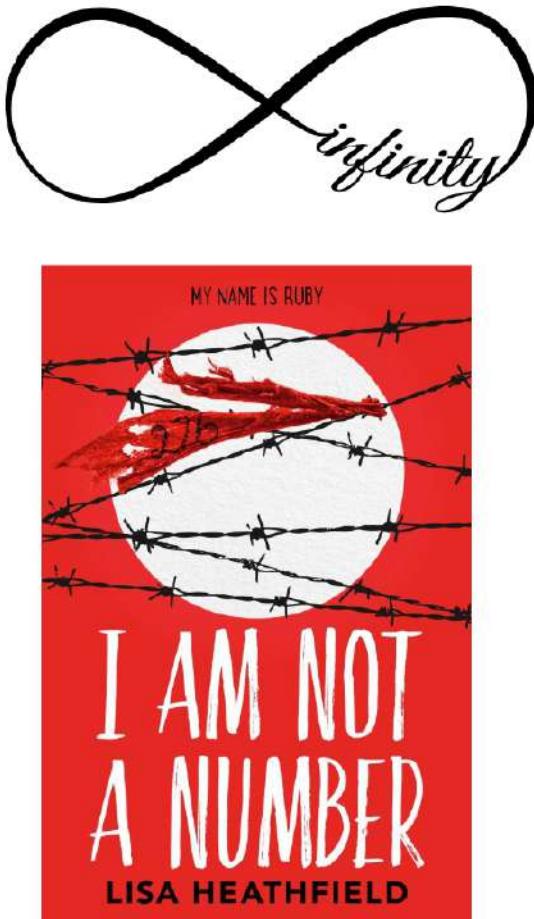
```
> let bigint1 = 1254n; //Agregamos n
let bigint2 = BigInt(1254); //Usando la funcion
< undefined
> document.write(bigint1 / 7n); //Permite dividir pero sin decimal
< undefined
>
```



The screenshot shows the Chrome DevTools interface with the 'Console' tab selected. The console output is as follows:

```
> let bigint1 = 1254n; //Agregamos n
let bigint2 = BigInt(1254); //Usando la funcion
< undefined
> document.write(bigint1 / 7); //No podemos operar entre distintos tipos
✖ > Uncaught TypeError: Cannot mix BigInt and other types, use explicit conversions
      at <anonymous>:1:24
```

## 2. TIPOS DE DATOS



**Infinity** es un **número especial** que nos proporciona JavaScript para el **cálculo simbólico**

INFO ABOUT RIGHTS

Se consigue:

- $1/0$
- $\text{Infinity}$
- $\text{Infinity}/k$  ó  $\text{Infinity} \cdot k$  ó  $\text{Infinity} + k$  ó  $\text{Infinity} - k$

En el caso de las **indeterminaciones matemáticas**, JavaScript nos permite manejarlo mediante el **`Nan`** ó **`Not a Number`**

Se consigue:

$$\frac{\infty}{\infty} \quad \infty - \infty \quad \frac{0}{0} \quad 1^\infty \quad 0 \cdot \infty \quad 0^0 ; \quad \infty^0$$

## 2. TIPOS DE DATOS



Entrega

Entrega: B

Formato entrega: .pdf

Margen: (vertical: 2) + (horizontal: 3)

Fuente: Calibri / Arial - 11 ptos - Justificado

Interlineado: 1,5 ptos

Realiza el siguiente ejercicio de **investigación**, y **contesta de manera argumentada**:

- i. Declara una variable llamada var1 que contenga 5 en formato carácter
- ii. Declara una variable llamada var2 que contenga 10 en formato numérico
- iii. Suma var1 y var2, ¿qué ha sucedido? ¿por qué?
- iv. Resta var1 menos var2, ¿qué ha sucedido? ¿por qué?
- v. Expresa el apartado 3 y el apartado 4 de nuevo en forma de alerta



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

Fecha entrega: 30/09/21

JOSE-CARLOS-ROMAN-HERRERA.Entrega-B.pdf

# TEMARIO DEL BLOQUE DIDÁCTICO B

1. Variables.
2. Tipos de datos.
3. Asignaciones.
4. Operadores.
5. Comentarios al código.
6. Sentencias.
7. Decisiones.
8. Bucles.

INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)



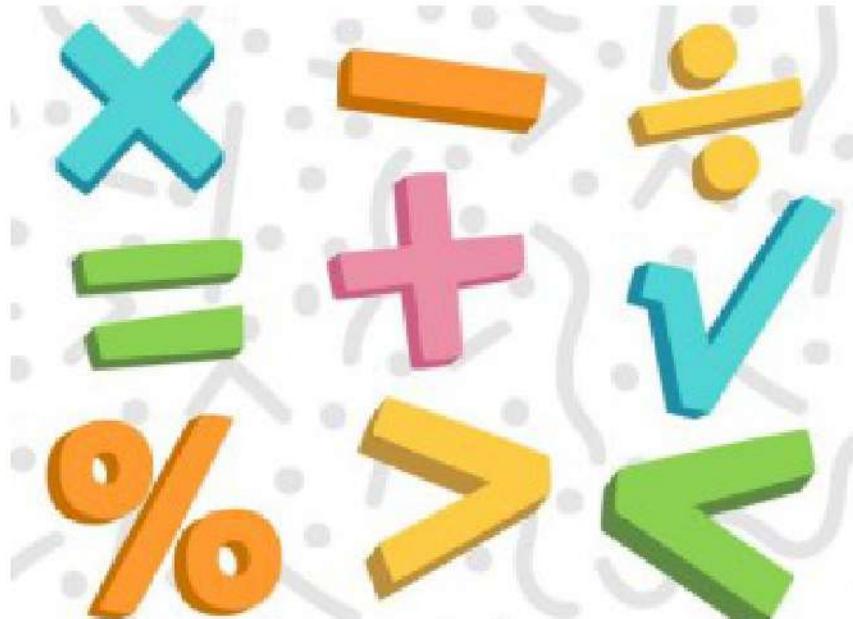
## OBJETIVOS DEL BLOQUE B

- Dominar la sintaxis básica del lenguaje.
- Comprender y utilizar distintos tipos de variables y operadores presentes en el lenguaje JavaScript.
- Conocer las diferentes sentencias condicionales de JavaScript y saber realizar operaciones complejas con ellas.

## 4. OPERADORES

¿Qué nos permiten hacer los operadores?

- {
- Combinar diferentes valores en una expresión
  - Manipular el valor de las variables
  - Realizar operaciones matemáticas con sus valores
  - Comparar diferentes valores
- }



Las **operadores** se clasifican **6 grupos** según su **manera funcional**:

- ✓ Operadores **aritméticos**
- ✓ Operadores **lógicos**
- ✓ Operadores de **comparación**
- ✓ Operadores **bit a bit**
- ✓ Operadores de **asignación**
- ✓ Operadores **especiales**

INFO ABOUT RIGHTS

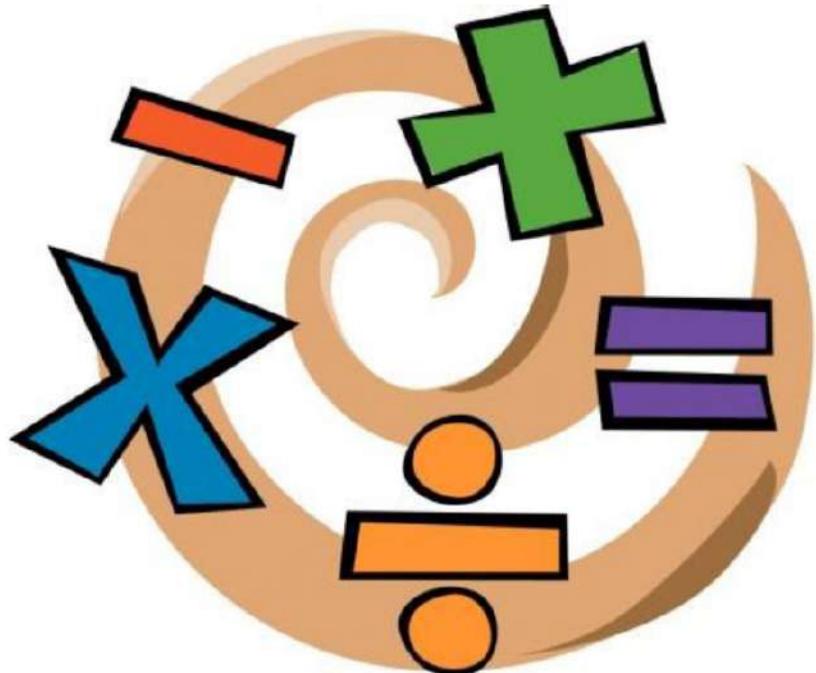
2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# 4. OPERADORES

Operadores **Aritméticos**

¿qué hacen? → Encargados de realizar las **operaciones matemáticas**



Estos son **8**:

- ✓ + → **Suma** valores, concatena cadenas alfanuméricas
- ✓ - → **Resta** valores
- ✓ \* → **Producto** de valores
- ✓ / → **Cociente** de valores
- ✓ \*\* → **Potenciación** de valores
- ✓ % → **Resto** de la división
- ✓ ++ → **Aumento** en una **unidad** la variable
- ✓ -- → **Decremento** en una **unidad** la variable

INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

## 4. OPERADORES



**SOLUCIÓN**

Realiza las siguientes operaciones:

- i. Declarar la variable “dato1” de tipo texto con la instrucción let y cuyo contenido sea “10”

`let dato1="10";`

- ii. Declarar la variable “dato2” de tipo numérico con la instrucción let y cuyo contenido sea 2

`let dato2=2;`

- iii. Prueba a realizar la siguiente instrucción: `dato1=+dato1` ¿Qué ha sucedido? ¿Qué tipo de variable esa ahora?

`Se ha cambiado el tipo de dato` contenido en la variable `dato1`

- iv. Calcula el cociente entre `dato1` y `dato2`

`let cociente=null;`

`Cociente = dato1 / dato2`

## 4. OPERADORES

Operadores **lógicos**

¿qué hacen? → Encargados de definir la **toma de decisiones**

Estos son **3**:

- ✓ **AND (&&)** → Devuelve **TRUE** cuando las dos comparaciones son ciertas  
Devuelve **FALSE** cuando alguna de las dos comparaciones es falsa
- ✓ **OR (||)** → Devuelve **TRUE** cuando alguna de las comparaciones es verdadera  
En caso contrario devuelve **FALSE**
- ✓ **NOT (!)** → Devuelve **SIEMPRE EL VALOR CONTRARIO**



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

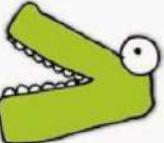
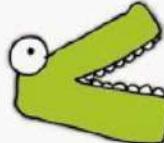
# 4. OPERADORES

Operadores **lógicos**

OPERADOR	DESCRIPCIÓN	EJEMPLO	RESULTADO
&&	AND	TRUE && TRUE TRUE && FALSE FALSE && TRUE FALSE && FALSE	TRUE FALSE FALSE FALSE
	OR	TRUE    TRUE TRUE    FALSE FALSE    TRUE FALSE    FALSE	TRUE TRUE TRUE FALSE
!	NOT	TRUE FALSE	FALSE TRUE

# 4. OPERADORES

Operadores de **bitwise**

		
greater than	less than	equal to

¿qué hacen? → Encargados de realizar las **comparaciones**

Estos son **8**:

- ✓ **>** → Devuelve **TRUE** si el operador **izq** es **mayor** que el **dech**
- ✓ **<** → Devuelve **TRUE** si el operador **izq** es **menor** que el **dech**
- ✓ **>=** → Devuelve **TRUE** si el operador **izq** es **mayor o igual** que el **dech**
- ✓ **<=** → Devuelve **TRUE** si el operador **izq** es **menor o igual** que el **dech**
- ✓ **==** → Devuelve **TRUE** si el operador **izq** es **igual** que el **dech**
- ✓ **!=** → Devuelve **TRUE** si el operador **izq** es **distinto** que el **dech**
- ✓ **====** → Devuelve **TRUE** si el operador **izq** es **igual y mismo tipo** que el **dech**
- ✓ **!==** → Devuelve **TRUE** si el operador **izq** **NO** es **igual y del mismo tipo** que el **dech**

# 4. OPERADORES

¿qué hacen? → Se utilizan para **asignar** un valor a una variable

Estos son 12:

- ✓ Asignación → `=`
- ✓ De adición → `+=`
- ✓ De sustracción → `-=`
- ✓ De multiplicación → `*=`
- ✓ De división → `/=`
- ✓ De resto → `%=`
- ✓ De desplazamiento a la izquierda → `<<=`
- ✓ De desplazamiento a la derecha → `>>=`
- ✓ De desplazamiento a la derecha sin signo → `>>>=`
- ✓ Asignación AND bit a bit → `&=`
- ✓ Asignación XOR bit a bit → `^=`
- ✓ Asignación OR bit a bit → `|=`



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# 4. OPERADORES

## Operadores **asignación**

OPERADOR	DESCRIPCIÓN	EJEMPLO	RESULTADO
$X=Y$	Asigna a X el valor de Y	$X=1$	$X=1$
$X+=Y$	Suma con asignación $X=X+Y$	$X+=1$	$X=X+1$
$X-=Y$	Resta con asignación $X=X-Y$	$X-=1$	$X=X-1$
$X*=Y$	Multiplicación con asignación $X=X*Y$	$X*=1$	$X=X*1$
$X/=Y$	División con asignación $X=X/Y$	$X/=1$	$X=X/1$
$X\%=Y$	Módulo con asignación $X=X\%Y$	$X\%=1$	$X=X\%1$
$X<<=Y$	Asigna a X el resultado de la operación $X = X<<Y$	$X=8<<1$	$X=18$
$X>>=Y$	Asigna a X el resultado de la operación $X = X>>Y$	$X=12<<2$	$X=3$
$X\&=Y$	Asigna a X el resultado del producto lógico $X = X\&Y$	$X=12\&9$	$X=8$
$X^=Y$	Asigna a X el resultado del producto lógico $X = X^Y$	$X=12^8$	$X=4$
$X =Y$	Asigna a X el resultado del producto lógico $X = X Y$	$X = 12  8$	$X=12$

## 4. OPERADORES

Operadores **especiales**



Son **7**:

- ✓ **delete** → Borra un objeto o su propiedad | borra elemento de una matriz
- ✓ **in** → Devuelve **TRUE** si la propiedad está en el objeto
- ✓ **instanceof** → Devuelve **TRUE** si el objeto es de una propiedad determinada
- ✓ **new** → Sirve para **crear una instancia** en un objeto
- ✓ **this** → Palabra clave que refiere al **objeto actual**
- ✓ **typeof** → Sirve para **conocer el tipo** de un objeto
- ✓ **void** → Operador que especifica una expresión que **será evaluada** sin devolver ningún valor

# 4. OPERADORES



Entrega

Entrega: C

Formato entrega: .pdf

Margen: (vertical: 2) + (horizontal: 3)

Fuente: Calibri / Arial - 11 ptos - Justificado

Interlineado: 1,5 ptos

Realiza el siguiente ejercicio de **investigación**, acerca de los operadores **BITWISE**.

Intenta al menos contestar a las siguientes preguntas:

- ✓ ¿Qué son?
- ✓ ¿Cuántos son?
- ✓ ¿Para qué se usan?

Pon un ejemplo de cada uno en JavaScript



INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

JOSE-CARLOS-ROMAN-HERRERA.Entrega-C.pdf



# TEMARIO DEL BLOQUE DIDÁCTICO B

- 
1. Variables.
  2. Tipos de datos.
  3. Asignaciones.
  4. Operadores.
  5. Comentarios al código.
  6. Sentencias.
  7. Decisiones.
  8. Bucles.

INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)



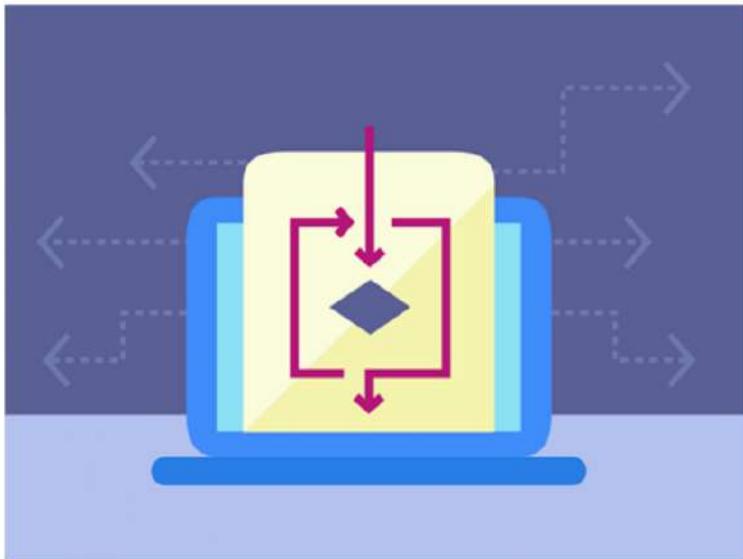
## OBJETIVOS DEL BLOQUE B

- Dominar la sintaxis básica del lenguaje.
- Comprender y utilizar distintos tipos de variables y operadores presentes en el lenguaje JavaScript.
- Conocer las diferentes sentencias condicionales de JavaScript y saber realizar operaciones complejas con ellas.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

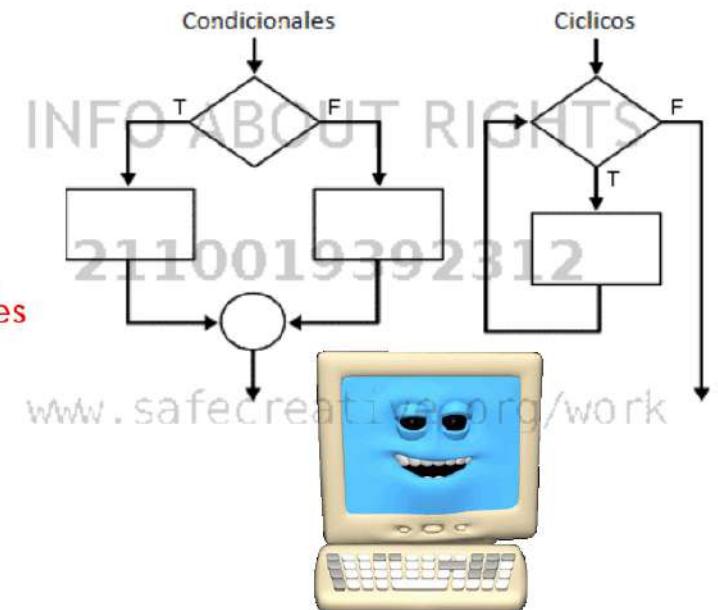
¿Qué son las **estructuras de control**?

→ son las sentencias que nos permiten dinamizar el programa mediante los **operadores lógicos, aritméticos y de comparación**



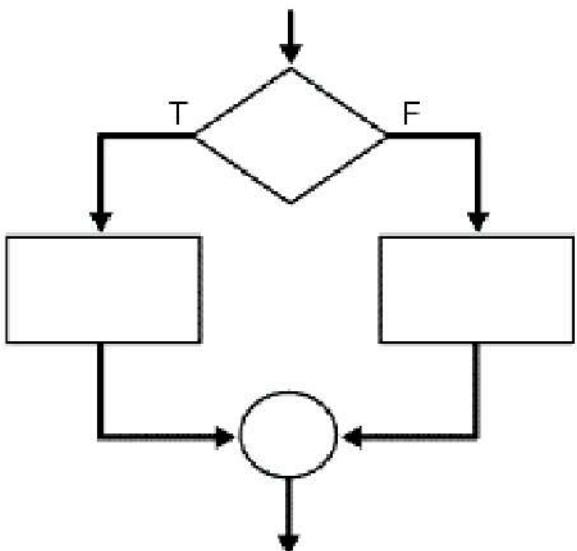
Estos son:

- ✓ If else → Son las **estructuras condicionales**
  - ✓ for
  - ✓ while
  - ✓ break  
continue
  - ✓ switch
  - ✓ try  
catch
- Son los **bucles**



# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Estructuras Condicionales



Sintaxis: `if(condición) {código_en_caso_verdadero} else {código_en_caso_falso}`

```

if(condición_0) {código}
else if (condición_1) {código} ... else if (condición_n) {código}
else{código}
  
```



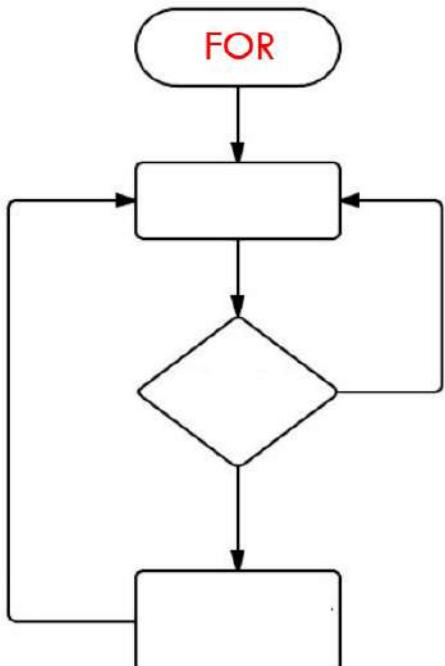
```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>Condicional If Else</title>
6      </head>
7      <body>
8          <script>
9              let nota=null;
10             nota = prompt('Introduce tú nota del examen (0-10): ');
11             if (nota>=5){
12                 alert("Calificación de: Apto");
13             } else {alert("Calificación de: Suspenso");}
14         </script>
15     </body>
16 </html>
  
```

Desarrolla una página en HTML que contenga un SCRIPT que pida al usuario introducir la nota obtenida en su examen de acceso a la universidad, y en caso de ser mayor o igual a cinco que emane una alerta que diga que la calificación es Aprobado, en caso contrario suspenso.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Bucles o Control de Flujo



Sintaxis: `for(valor inicial; condición; actualización) {instrucciones_a_ejecutar}`

`for(variable in objeto) {expresión}` (Arrays)

`for(variable of elemento) {expresión}` (Matrices, cadenas, mapas..)



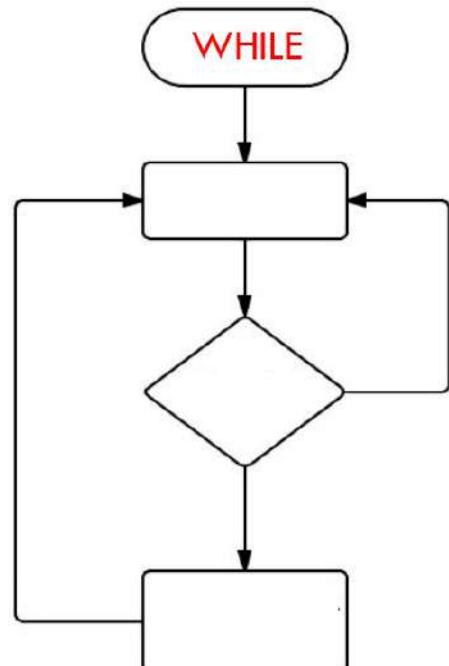
```

1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>Bucles: FOR</title>
6      </head>
7      <body>
8          <script>
9              let cont=0;
10             let desc=10;
11
12             for (cont,desc; cont<11;cont++,desc--){
13                 document.write("Contador: "+cont+" | Descontador: "+desc+"<br/>");
14             }
15         </script>
16     </body>
17 </html>
  
```

Desarrolla una página en HTML que contenga un SCRIPT con dos variables, una inicializada en 0 y la otra en 10. Desarrolla un script con un solo bucle for, que maneje estas variables y pase una unidad a la otra variable y por consiguiente se lo reste a la variable que cede valor.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Bucles o Control de Flujo



Sintaxis: **while(condición) {instrucciones\_a\_ejecutar}**

**do{instrucciones\_a\_ejecutar} while (condición)**

(Más control)

```

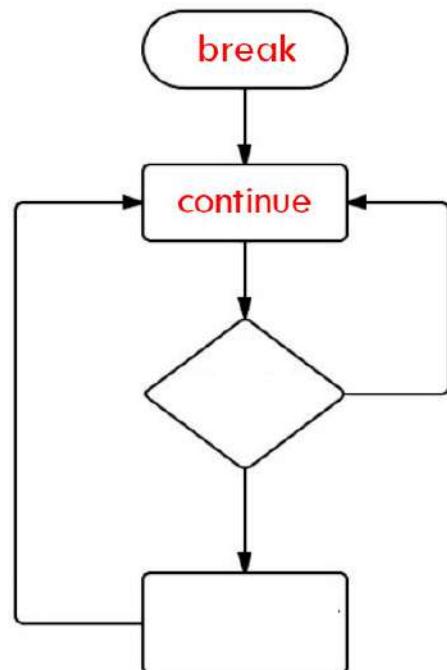
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>Bucles: WHILE</title>
6      </head>
7      <body>
8          <script>
9              function numeroAleatorio(min, max) {
10                  return Math.round(Math.random() * (max - min) + min);
11              }
12              numero_aleatorio = numeroAleatorio(1, 10);
13              let numero;
14              do {
15                  numero = prompt("Introduzca un numero del 1 al 10: ");
16              }
17              while(numero_aleatorio!=Number(numero));
18              alert("¡¡Has adivinado el numero!!")
19          </script>
20      </body>
21  </html>
  
```



Desarrolla una página en HTML que contenga un SCRIPT que pida que introduzcas un número del 1 al 10, y después compruebe ese número que has elegido con otro número que ha pensado el programa (número random) y si es correcto que salga una alerta diciendo que has adivinado el número.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Bucles o Control de Flujo



Sintaxis:

- break** → Rompe el while o el for (asociado dentro de if del for/while)
- continue** → Hace que no termine la ejecución del bucle



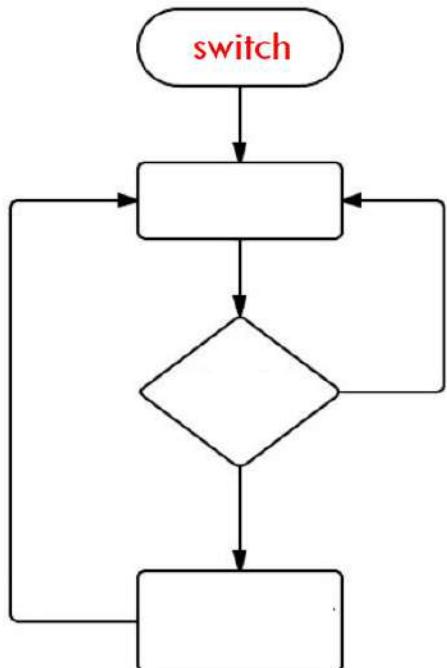
```

1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Bucles: CONTINUE</title>
6  </head>
7  <body>
8  <script>
9
10 let numero=0;
11 for(numero=0; numero<101; numero++){
12     if(numero/2 == parseInt(numero/2)){continue}
13     document.write(numero+"<br/>");
14 }
15 </script>
16 </body>
17 </html>
  
```

Desarrolla una página en HTML que contenga un SCRIPT que genere todos los números del 0 al 100 y solo aquellos que sean impares van a ser visualizados

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Bucles o Control de Flujo



Sintaxis:

```

switch (expresión) {
    case Valor1: código a ejecutar si coincide; break;
    case Valor2: código a ejecutar si coincide; break;
    ...
    case ValorN: código a ejecutar si coincide; break;
    default: código; break;
}
  
```

```

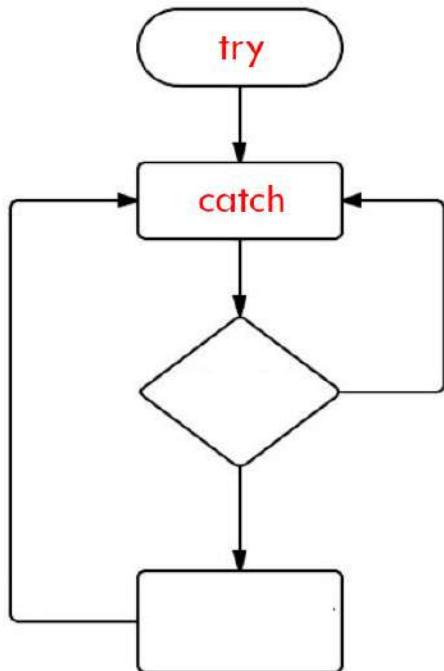
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <meta charset="UTF-8">
5          <title>Bucles: SWITCH</title>
6      </head>
7      <body>
8          <script>
9              let numero=null;
10             numero = Number(prompt("Introduce tu código de trabajador: "))
11             switch(numero){
12                 case 1: document.write("Eres médico"); break;
13                 case 2: document.write("Eres enfermera"); break;
14                 case 3: document.write("Eres auxiliar"); break;
15                 default: document.write("No trabajas aquí");
16             }
17         </script>
18     </body>
19 </html>
  
```



Desarrolla una página en HTML que contenga un SCRIPT que pida al usuario el código de su puesto de trabajo en un hospital, en función de este deberá sacar su profesión por fuera. Recuerda que 1 es médico, 2 enfermera, 3 auxiliar.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES

## Bucles o Control de Flujo



Sintaxis: `try {código} catch(error){ Gestión_del_error }`

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4  <meta charset="UTF-8">
5  <title>Bucles: TRY CATCH</title>
6  </head>
7  <body>
8  <script>
9
10 let var2=2;
11 try{
12     document.write(var1+var2);
13 }
14 catch(error){
15     document.write("Se ha producido un error")
16 }
17 </script>
18 </body>
19 </html>
  
```



Desarrolla una página en HTML que contenga un SCRIPT que el cual pida realizar la suma de dos variables var1 (10) y var2 (2), pero para poder generar el error no declares var1.

# 6. Y 7. Y 8 SENTENCIAS Y DECISIONES Y BUCLES



## Actividad Calificable

Desarrolla el siguiente programa y realiza el correspondiente informe justificado que explique su funcionamiento, y/o su desarrollo.

Haz una página web que implemente un juego de encontrar un número aleatorio bajo las siguientes premisas:

- ✓ La página calculará un número del 1 al 100
- ✓ Luego preguntará al usuario por el número
- ✓ Si el usuario escribe algo que no es un número se indica el error y se vuelve a pedir el número.
- ✓ Si no, le dice si el número escrito por el usuario es menor o mayor y vuelve a preguntar cuál es.
- ✓ Si se cancela cualquier cuadro el juego termina indicando que se canceló el juego.
- ✓ Al final, si se ha finalizado correctamente el juego se indica el número de intentos.
- ✓ Se permite volver a jugar al usuario mediante un cuadro de confirmación.

Ejercicio N°: 02

Formato entrega: **.doc o .docx + .js y/.html**

Fecha entrega: Viernes 01/11/ 2021

Margen: (vertical: 2) + (horizontal: 3)

Fuente: Calibri / Arial - 11 ptos - Justificado

Interlineado: 1,5 ptos

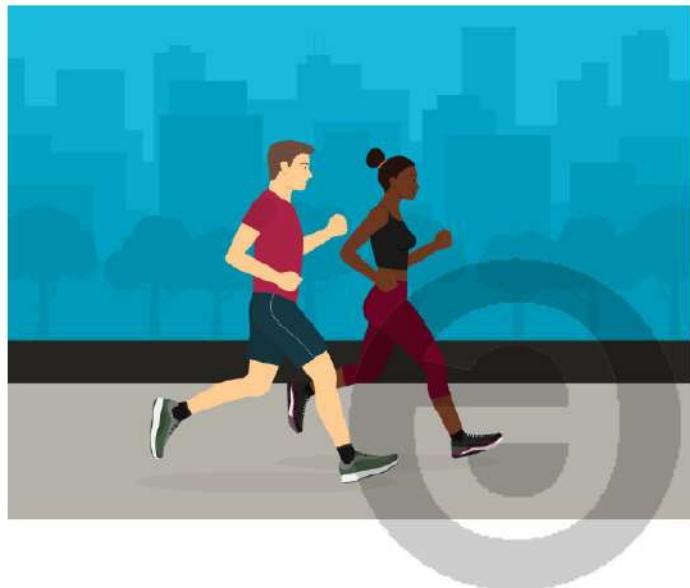
Anexo → Numerar fotografías o capturas de pantalla.

JOSE-CARLOS-ROMAN-HERRERA.Ejercicio-02.docx

# EJERCICIOS



# PROPUESTA EJERCICIO DE CLASE NÚMERO 01



Realiza una página en HTML5, que contenga un título h1 que diga: “Estos son los números primos entre el 1 y 100”. Implementa un SCRIPT que calcule los números primos entre 1 y 100

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

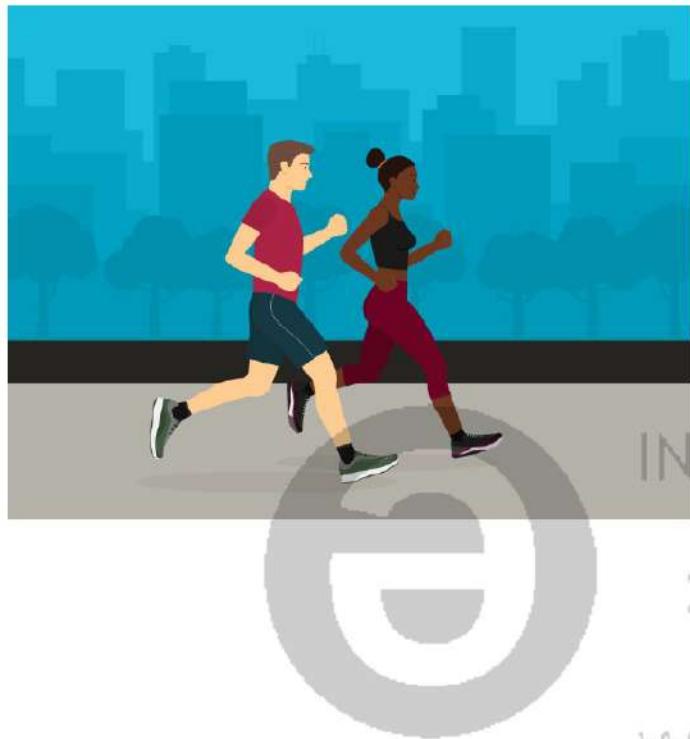
# PROUESTA EJERCICIO DE CLASE NÚMERO 01

## Solución

Realiza una página en HTML5, que contenga un título h1 que diga: “Estos son los números primos entre el 1 y 100”. Implementa un SCRIPT que calcule los números primos entre 1 y 100

```
<!DOCTYPE html><html><head><title>Números primos</title></head>
<body>
    <script language=javascript>
var i,j,primo;
document.write('1<br>');
document.write('2<br>');
document.write('3<br>');
for(i=4;i<=100;i++)
{
    primo=0;
    for(j=2;j<i;j++)
    {
        if(i%j==0) primo=1;
    }
    if (primo==0) {document.write(i); document.write('<br>')}
}
</script>
</body></html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 02



Realiza una página en HTML5, que contenga un título h1 que diga: “Intentos dado”. Implementa un SCRIPT que calcule el número de intentos necesarios para que salga el valor de 5 en una tirada de un dado no trucado.

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# PROUESTA EJERCICIO DE CLASE NÚMERO 02

## Solución

Realiza una página en HTML5, que contenga un título h1 que diga: “Intentos dado”.

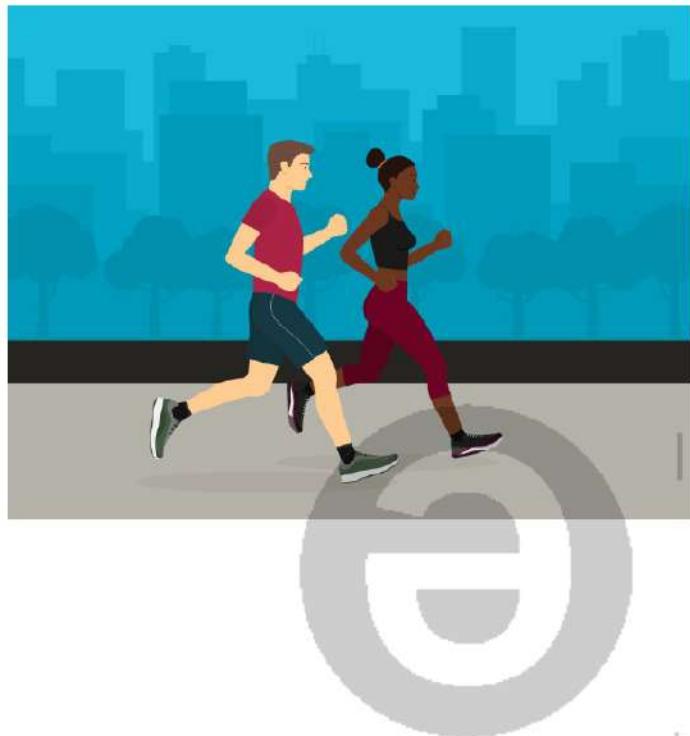
Implementa un SCRIPT que calcule el número de intentos necesarios para que salga el valor de 5 en una tirada.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Intentos dado</title>
</head>
<body>
    <h1>Intentos dado</h1>
    <script type="text/javascript">
        var n = 0; // Contador
        var x = 0; // Valor dado

        function aleatorio(min, max) {return Math.floor(Math.random() * (max - min)) + min;
        }
        while ( x!=5) {
            x = aleatorio(0,7);
            n++;
            document.write("Intento: "+n+" - Valor dado: "+x+'<br/>')
        }

    </script>
</body>
</html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 03



Realiza una página en HTML5, que contenga un título h1 que diga: “Números de la sucesión de Fibonacci menores a trescientos”. Implementa un SCRIPT que calcule los números de la sucesión de Fibonacci menores a trescientos.

INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# PROUESTA EJERCICIO DE CLASE NÚMERO 03

## Solución

Realiza una página en HTML5, que contenga un título h1 que diga: “Números de la sucesión de Fibonacci menores a trescientos”. Implementa un SCRIPT que calcule los números de la sucesión de Fibonacci menores a trescientos.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>Números de Fibonacci menores a trescientos</title>
</head>
<body>
    <h1>Números de Fibonacci menores a trescientos</h1>
    <script type="text/javascript">
        var x = 0; // Primer valor
        var y = 1; // Segundo valor
        var z = null; // Tercer valor
        do {
            document.write(y+'<br/>');
            z = x+y;
            x = y;
            y = z;
        } while(y<300)
    </script>
</body>
</html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 04



Realiza una página en HTML5, que contenga un título h1 que diga: “Divisible por siete”. Implementa un SCRIPT que vaya sacando número aleatorio entre el 1 y el 500, cuya condición de parada consista que hayan aparecido al menos siete veces un número divisible por siete.

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# PROUESTA EJERCICIO DE CLASE NÚMERO 04

## Solución

Realiza una página en HTML5, que contenga un título h1 que diga: “Divisible por siete”. Implementa un SCRIPT que vaya sacando número aleatorio entre el 1 y el 500, cuya condición de parada consista que hayan aparecido al menos siete veces un número divisible por siete.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Divisible por siete</title>
</head>
<body>
    <h1>Divisible por siete</h1>
    <script lang="javascript">
        var condicion = false; //condicion de parada
        var n=null; //numero aleatorio
        var i=null; //contador
        function numaleatorio(min,max){return Math.floor((Math.random() * (max-min)) +min);}

        while(condicion==false && i<=7) {
            n = parseInt(numaleatorio(1,501));
            document.write(n+"<br>");
            if(n%7==0){
                if(i==7){condicion=true} else{i++}
            }
        }
    </script>
</body>
</html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 05



Realiza una página en HTML5, que contenga un título h1 que diga: “Mi tabla”. Implementa un SCRIPT que pida cuantas filas y columnas quieras crear. [INFO ABOUT RIGHTS](#)



2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# PROUESTA EJERCICIO DE CLASE NÚMERO 05

Solución

Realiza una página en HTML5, que contenga un título h1 que diga: "Mi tabla". Implementa un SCRIPT que pida cuantas filas y columnas quieras crear.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Mi tabla</title>
    <style type="text/css">
        table{border-collapse: collapse; width: 100%;}
        td{border: 1px solid black}
    </style>
</head>
<body>
<h1>Mi tabla</h1>
<script type="text/javascript">
    var f = null;
    f = Number(prompt("¿Cuantas filas?: "));
    var c = null;
    c = Number(prompt("¿Cuantas columnas?: "));
    document.write("<table>");
    for (let i=1; i<=f; i++){
        document.write("<tr>");
        for(let j=1; j<=c; j++){document.write("<td>&ampnbsp</td>");}
        document.write("</tr>");
    }
    document.write("</table>");
</script>
</body>
</html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 06



Realiza una página en HTML5, que contenga un título h1 que diga: “Factorial”. Implementa un SCRIPT que pida un número y te saque el factorial de dicho número por pantalla.

INFO ABOUT RIGHTS

2110019392312

[www.safecreative.org/work](http://www.safecreative.org/work)

# PROUESTA EJERCICIO DE CLASE NÚMERO 06

Solución

Realiza una página en HTML5, que contenga un título h1 que diga: “Factorial”. Implementa un SCRIPT que pida un número y te saque el factorial de dicho número por pantalla.

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Factorial</title>
</head>
<body>
    <h1>Factorial</h1>
    <script type="text/javascript">
        var n = Number(prompt("Dame número: "));
        var parada=false;
        var c = n;
        var f = 1;

        while(parada==false){
            f = n*f;
            n=n-1;
            if(n==0){parada=true;}
        }
        document.write("<p>El factorial de "+c+" es: "+f+"</p>");
    </script>
</body>
</html>
```

# PROPUESTA EJERCICIO DE CLASE NÚMERO 07



Realiza una aplicación web en HTML5, que muestre 2000 cuadrados de color aleatorio de 50 x 50 píxeles, y cuya posición sea aleatoria en pantalla.

INFO ABOUT RIGHTS



# PROUESTA EJERCICIO DE CLASE NÚMERO 07

Solución

Realiza una aplicación web en HTML5, que muestre 2000 cuadrados de color aleatorio de 50 x 50 pixeles, y cuya posición sea aleatoria en pantalla.

```
1  <!DOCTYPE html>
2  <html lang="es">
3  <head>
4      <title>Cuadrados aleatorios</title>
5      <style>
6          div{
7              position:fixed;
8              width:50px;
9              height:50px;
10         }
11     </style>
12 </head>
13 <body>
14 <script>
15 const NUM CUADROS=2000;
16 for(let i=1;i<=NUM CUADROS;i++){
17     let rojo=parseInt(Math.random()*255);
18     let verde=parseInt(Math.random()*255);
19     let azul=parseInt(Math.random()*255);
20     let left=(Math.random()*100)+"%";
21     let top=(Math.random()*100)+"%";
22     document.write(
23         `<div style='background-color:`+
24         `rgb(${rojo},${verde},${azul});`+
25         `left:${left};top:${top}`+`></div>`+
26     );
27 }
28 </script>
29 </body>
30 </html>
```

Román-Herrera, J.C.

