

**Nombre del Aprendiz: Bastian Tenorio**

**Taller de JavaScript 1**

**Nombre del instructor: Daniel Martínez Payán.**

**Servicio Nacional de Aprendizaje (SENA)**

**Centro de Biotecnología Industrial.**

**2025**

Contenido

[Introducción 2](#_Toc191745493)

[Contexto 2](#_Toc191745494)

[¿Qué es JavaScript? 3](#_Toc191745495)

[Bibliografía 17](#_Toc191745496)

[Ilustración 1. Función matematica 7](file:///C:\Users\Administrador\Downloads\Informe%20Taller%20Algoritmia.docx#_Toc191647860)

# Introducción

Aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

# Contexto

Aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

# ¿Qué es JavaScript?

## Preguntas de comprensión

1. ¿Qué es JavaScript y para qué se utiliza?

    JavaScript es un lenguaje de programación que se utiliza principalmente para darle interactividad y dinamismo a las páginas, se utiliza tanto en el navegador como en el servidor.

2. ¿Cuáles son las características principales de JavaScript?

    - Es un lenguaje interpretado y de tipado dinámico.

    - permite programar de manera orientada a objetos usando prototipos.

    - Es flexible y se utiliza tanto en el lado del cliente como en el servidor.

    - soporta programación asincrónica.

3. ¿Cuál es la diferencia entre `var` y `let` en JavaScript?

    - var tiene ámbito global así que si se declara dentro de un bloque aun así será accesible fuera de este mismo lo que puede generar problemas si no se utiliza correctamente.

    - let tiene ámbito de bloque, lo que limita su uso solamente al bloque en donde se declara y ayuda a evitar conflictos y errores.

4. ¿Por qué es importante la sensibilidad a mayúsculas y minúsculas en

JavaScript?

    Eso es porque JavaScript distingue entre mayúsculas y minúsculas, lo que significa que nombre, Nombre y NOMBRE son distintas. es importarte para evitar errores al referenciar variables.

5. ¿Qué tipos de datos se pueden almacenar en una variable en JavaScript?

    -Números (enteros y decimales)

    -BigInt (se agregó recientemente al lenguaje para representar enteros de longitud arbitraria)

    -Cadenas de texto (strings)

    -Booleanos (true o false)

    -Objetos (los objetos se utilizan para almacenar colecciones de datos y entidades más complejas)

    -Arreglos (arrays)

    -Funciones

    -Valores especiales como null y undefined (El significado de undefined es “valor no asignado”)

6. ¿Cómo se declara un objeto en JavaScript?

    Podemos crear un objeto usando las llaves {…} con una lista opcional de propiedades. Una propiedad es un par “key:value” o “clave:valor”, donde key es un string y value puede ser cualquier cosa.

    ej: let user = {     // un objeto

        nombre: "John",  // En la clave "nombre" se almacena el valor "John"

        edad: 30        // En la clave "edad" se almacena el valor 30

        };

7. ¿Qué es la concatenación de cadenas y cómo se realiza en JavaScript?

    Es el proceso de unir varias cadenas de texto en una sola.

8. ¿Cuáles son las diferencias entre usar `+`, `,` y template literals (Comillas invertidas) para concatenar?

    +: Une cadenas y variables, creando una sola cadena.

    ,: Se usa en funciones como console.log para imprimir varios valores, pero no concatena cadenas, sino que imprime valores separados por un espacio.

    Template literals (comillas invertidas): Permiten incluir variables y expresiones de forma más legible usando la sintaxis ${variable}

9. ¿Cómo funciona la función `prompt` en JavaScript?

    La función prompt muestra una pequeña ventana en el navegador pidiéndole al usuario que ingrese información. La ejecución del código se detiene hasta que el usuario escribe algo o cancela la acción, y el valor ingresado se guarda en una variable.

10.¿Qué tipo de datos devuelve `prompt`?

    Aunque el usuario escriba un número, prompt siempre devuelve un string (una cadena de texto).

11.¿Cuál es la diferencia entre `let` y `const`?

    -let se utiliza para declarar variables cuyo valor puede cambiar con el tiempo.

    -const se utiliza para declarar constantes, es decir, valores que no cambiarán una vez asignados.

12.¿Por qué no se puede cambiar el valor de una constante?

    Porque al definir una constante con const, se está estableciendo que ese valor permanecerá fijo durante toda la ejecución del programa, lo que evita modificaciones accidentales.

13.¿Qué son los operadores matemáticos y cuáles son algunos ejemplos?

    Los operadores matemáticos son símbolos que te permiten hacer cálculos básicos. Algunos ejemplos son:

    Suma: +

    Resta: -

    Multiplicación: \*

    División: /

    Módulo: % (devuelve el resto de una división)

14.¿Cómo se realiza la potenciación en JavaScript?

    La potenciación se realiza usando el operador \*\*. Por ejemplo:

    let resultado = 2 \*\* 3; // 2 elevado a 3 es igual a 8

    También puedes usar la función Math.pow(2, 3) para obtener el mismo resultado.

15. ¿Qué son las condicionales y cómo funcionan en JavaScript?

    Las condicionales son estructuras que permiten que el programa tome decisiones. Según se cumpla o no una condición, se ejecuta un bloque de código u otro.

16. ¿Cuál es la sintaxis básica de una declaración `if`?

    if (condición) {

    // código a ejecutar si la condición es verdadera

    }

    También puedes añadir else para definir qué hacer cuando la condición no se cumple.

17. ¿Qué funciones matemáticas están disponibles en la clase Math?

    La clase Math de JavaScript ofrece varias funciones como:

    -Math.abs() devuelve el valor absoluto de un número.

    -Math.pow() eleva un número a la potencia indicada.

    -Math.sqrt() para la raíz cuadrada.

    -Math.sin(), Math.cos(), Math.tan() devuelven el seno, coseno, y tangente de un ángulo dado en radianes.

    -Math.toRadians() Convierte un ángulo en grados a radianes.

    -Math.toDegrees() Convierte un ángulo en radianes a grados.

    -Math.floor() para redondear hacia abajo.

    -Math.ceil() para redondear hacia arriba.

    -Math.round() para redondear al entero más cercano.

    -Math.max() y Math.min() para obtener el valor máximo o mínimo de un conjunto de números.

18. ¿Cómo se utiliza `Math.random()`?

    La función Math.random() genera un número aleatorio entre 0 (incluido) y 1 (excluido). Si necesitas un número en un rango específico, puedes multiplicar el resultado y, en algunos casos, redondearlo.

19. ¿Qué es un array y cómo se declara en JavaScript?

    Un array es como una lista ordenada de elementos. Se declara utilizando corchetes []. Por ejemplo:

    let colores = ["rojo", "verde", "azul"];

    Esto crea un array con tres elementos.

20. ¿Cómo se accede a los elementos de un array?

    Referenciar el nombre del arreglo, seguido del número de índice del elemento que queremos acceder, entre corchetes. Cada elemento del array tiene un índice que comienza en 0. Así, para acceder al primer elemento de colores se pone como ejemplo:

    let primerColor = colores[0]; // "rojo"

\*/

/\*Preguntas de Análisis

21. ¿Cómo se diferencia JavaScript del lado del cliente y del lado del servidor?

    Del lado del cliente: Se ejecuta en el navegador del usuario. Es el responsable de la interactividad en la página (por ejemplo, mostrar menús, validar formularios o animaciones).

    Del lado del servidor: Se ejecuta en un servidor (por ejemplo, usando Node.js) y se encarga de la lógica detrás de escena, como el manejo de bases de datos o la autenticación de usuarios.

22. ¿Qué versiones de ECMAScript son compatibles con los navegadores

modernos?

    La mayoría de los navegadores actuales soportan ECMAScript 6 (ES6) y versiones posteriores.

23. ¿Qué problemas podrían surgir al re-declarar variables con `var`?

    Se podría sobrescribir un valor sin querer o tener variables accesibles en lugares del código donde no se espera.

24. ¿Cuál es la importancia de los tipos de datos en la programación?

    Los tipos de datos le indican al programa cómo debe tratar la información. Saber si un valor es un número, un texto o un valor verdadero/falso es importante para realizar operaciones correctas.

25. ¿En qué situaciones sería más conveniente usar template literals?

    Son ideales cuando necesitas construir cadenas de texto que incluyan variables o expresiones también al utilizar cadenas de caracteres de más de una línea.

26. ¿Cómo manejarías los casos en que el usuario no ingresa datos?

    Se comprueba si el valor ingresado está vacío o es nulo antes de continuar. Si no se recibe ningún dato, mostraría un mensaje pidiendo que se complete el campo.

27. ¿En qué situaciones sería más apropiado usar `const` en lugar de `let`?

    Se usa const cuando se sabe que el valor de la variable no debe cambiar durante la ejecución del programa, como configuraciones, URLs o constantes matemáticas, porque ayuda a prevenir modificaciones accidentales.

28. ¿Qué ocurre si intentas realizar una operación matemática con un string?

    Si el string contiene números, JavaScript intentará convertirlo a número y realizar la operación. Pero si el string no representa un número, el resultado será NaN (Not a Number), lo que indica que la operación no se pudo completar correctamente.

29. ¿Cómo puedes anidar condicionales y por qué sería útil?

    Anidar condicionales significa poner una estructura if dentro de otra. Es útil cuando necesitas evaluar varias condiciones de forma secuencial. Por ejemplo, podrías primero verificar si un usuario está registrado y, dentro de esa comprobación, comprobar si tiene permiso para acceder a cierto contenido.

30. ¿Por qué es útil redondear números en ciertas aplicaciones?

    Redondear números ayuda a simplificar resultados y hacerlos más legibles. Es especialmente útil en casos como cálculos de precios, puntuaciones o medidas, donde tener demasiados decimales puede resultar innecesario.

31. ¿Cuáles son las ventajas de usar arrays en lugar de variables individuales?

    Los arrays te permiten agrupar múltiples elementos en una sola variable, lo que facilita el manejo de datos. En lugar de crear muchas variables separadas para cada elemento, con un array puedes recorrer, buscar o modificar los elementos de manera más rapida y con menos código.

\*/

// Actividades de Codificación.

// 32.Crea una nueva variable usando `let` y asígnale tu nombre. Luego imprímelo en la página.

let nombre = 'Bastian';

document.writeln(nombre, '<br>');

// 33. Crea un objeto que represente a un estudiante con propiedades como nombre, edad y curso.

let estudiante= {

    nombre: 'daniel',

    edad: 17,

    curso: 'ADSO'

}

// 34. Crea una cadena que incluya tu nombre y tu hobby utilizando template literals.

let hobby = 'GYM';

let mensaje = `hola soy ${nombre} y mi hobby es ${hobby}.`;

// 35. Pide al usuario que ingrese su comida favorita y muéstralo en la página.

let comida;

comida=prompt('Ingrese su comida favorita: ');

document.writeln(comida, '<br>')

// 36. Declara una constante para el número pi y muéstrala en la página.

const pi = Math.PI;

document.writeln(pi, '<br>')

// 37. Pide al usuario que ingrese dos números y muestra la suma, resta, multiplicación y división.

let n1, n2;

n1= Number(prompt('Ingrese el primer número:'))

n2=Number(prompt('Ingrese el segundo número:'))

document.writeln(`Suma: ${n1 + n2}<br>`)

document.writeln(`Resta: ${n1 - n2}<br>`)

document.writeln(`Multiplicación: ${n1 \* n2}<br>`)

document.writeln(`División: ${n2 !== 0 ? (n1 / n2) : "No se puede dividir por 0"} <br>`)

// 38. Crea un programa que pida la edad del usuario y determine si es mayor o menor de edad.

let edad;

edad=prompt('Ingrese su edad:')

if (edad >=18) {

    document.writeln('Es mayor de edad')

}else{

    document.writeln('Es menor de edad')

}

document.write('<br>');

// 39. Crea un programa que genere un número aleatorio entre 1 y 100 y muéstralo en la página.

let numAleatorio = (Math.random()\*100) + 1;

document.writeln(`Numero Aleatorio: ${numAleatorio}`)

document.write('<br>');

// 40. Crea un array de tus tres frutas favoritas y muéstralas en la página.

let frutas =['Manzana','Pera','Mango'];

document.writeln(`Frutas: ${frutas.join(", ")}<br>`);

// Analiza y Explica

// 41. ¿Qué errores contiene este código? ¿Cuál es la salida esperada? Explica por qué.

document.writeln('HOLA MUNDO <br>');

// document.writeln(HOLA MUNDO);

// El error en el código es que HOLA MUNDO se usa sin comillas, lo que hace que JavaScript lo interprete como una variable. Si no está definido, generará un ReferenceError . La salida esperada es un error, a menos que HOLA MUNDO esté previamente definida, en cuyo caso mostraría su valor.

// 42. ¿Qué valor se mostrará en la página? ¿Es correcto re-declarar una variable con `var`? Explica tu respuesta.

var Nombre; // se declara una variable

Nombre='gala';

// Se inicializa la variable

var Nombre = 'juan';

// Re-declaracion de variable

document.writeln(Nombre)

document.writeln('<br>')

// La salida será "juan" porque en JavaScript se permite redeclarar variables con var. Primero se asigna "Santiago", pero luego se vuelve a declarar y se reasigna "juan", sobrescribiendo el valor anterior. Aunque es válido, no es recomendable usar vardebido a posibles confusiones; es mejor let o const.

// 43. ¿Cuál será el resultado de esta línea? ¿Por qué ocurre este comportamiento? ¿Cómo se podría corregir?

let persona = {nombre: "julian", edad: "42", estatura: 1.74, genero: "Masculino"};

// document.writeln(persona.edad + 5);

/\*El resultado de document.writeln(persona.edad + 5);será "285" en lugar de 33, porque edad está almacenada como una cadena ("28") , y al sumarle 5, JavaScript la trata como una concatenación en lugar de una suma numérica. Para corregirlo, se puede convertir edadun número con parseInt(persona.edad)o Number(persona.edad), así: document.writeln(Number(persona.edad) + 5);. \*/

document.writeln(Number(persona.edad) + 5);

document.writeln(parseInt(persona.edad) + 5);

document.writeln('<br>')

// 44. Pregunta: ¿Hay alguna diferencia en la salida entre las dos líneas? ¿Qué pasaría si apellido fuera null?

let NOMBRE = "nancy";

let apellido = "estupiñan";

document.writeln("Nombre y apellido: " + NOMBRE + ' ' + apellido);

document.writeln('<br>')

document.writeln(`Nombre: ${NOMBRE} Apellidos: ${apellido}`);

document.writeln('<br>')

/\*La posición del apellido difiere en cada formato; Además, si apellido es null, ambos mostrarán ese valor en lugares distintos, la primera línea mostraría "Nombre y apellido: nancy null", mientras que la segunda línea con interpolación de cadenas mostraría "Nombre: nancy Apellidos: null". \*/

// 45. Si el usuario no ingresa nada en el prompt de edad, ¿qué valor se mostrará en la página? ¿Cómo podrías manejar este caso?

nombre = prompt("Escriba su nombre...");

edad = prompt("Digite su edad...");

document.writeln("Nombre: " + nombre);

document.writeln('<br>')

document.writeln("Edad: " + edad);

/\*Al usar prompt para capturar la edad, si el usuario no ingresa nada se obtiene una cadena vacía (""), se recomienda manejarlo asignando un valor predeterminado (por ejemplo, edad = prompt("Digite su edad...") || "No especificado";) o convertir la entrada a número y validar que sea un valor correcto. \*/

// 46. ¿Qué error ocurrirá al ejecutar este código? Explica por qué no se puede re-asignar un valor a una constante.

const Precio = 500.00;

Precio = 300.00; // Intento de re-asignación

document.writeln("Precio: ", Precio);

// Se obtiene un TypeError: Assignment to constant variable, porque en JavaScript las constantes (const) no pueden cambiar de valor una vez declaradas.

// 47. ¿Qué valor se mostrará como resultado de la suma? ¿Cómo se puede corregir para que realice una suma numérica?

let a = prompt("Ingrese un número:");

let b = prompt("Ingrese otro número:");

let sum = a + b;

document.writeln("La suma es: " + sum);

// si el usuario ingresa “3” y “4”, el resultado será “34” porque ambas entradas se tratan como cadenas y se concatenan. Para corregirlo y obtener una suma numérica, se deben convertir los valores a número:

/\*

let a = Number(prompt("Ingrese un número:"));

let b = Number(prompt("Ingrese otro número:"));

let sum = a + b;

document.writeln("La suma es: " + sum);

Así, si el usuario ingresa “3” y “4”, el resultado será 7.

\*/

// 48. ¿Qué pasará si el usuario ingresa un valor no numérico para la edad? ¿Cómo puedes mejorar el manejo de este caso?

/\*

let nombre = prompt(Ingrese su nombre...);

let edad = prompt(Ingrese su edad);

edad = parseInt(edad);

if (edad >= 18) {

    document.write(nombre, ` eres mayor de edad`);

} else if (edad < 18) {

    document.write(nombre, ` eres menor de edad`);

} else {

    document.write(No ingresaste tu edad);

}

\*/

// si el usuario ingresa un valor no numérico para la edad, parseInt(edad) devolverá NaN, lo que puede causar un comportamiento inesperado. Para solucionarlo, se debe validar si edad es un número antes de usarla:

/\*

let nombre = prompt(Ingrese su nombre...);

let edad = parseInt(prompt(Ingrese su edad));

if (isNaN(edad)) {

    document.write(No ingresaste una edad válida);

} else if (edad >= 18) {

    document.write(${nombre}, eres mayor de edad);

} else {

    document.write(${nombre}, eres menor de edad);

}

 \*/

// 49. ¿Es correcto declarar la variable `precio` dos veces? ¿Qué implicaciones tiene esto en el código?

var precio = Math.round(399.53);

document.writeln('Precio redondeado: ', precio, '<br>');

var precio = Math.round(399.43);

document.writeln('Precio redondeado: ', precio, '<br>');

/\*

el código declara la variable precio dos veces con var, lo cual es válido en JavaScript pero puede causar confusión porque var permite redeclaración dentro del mismo ámbito. Esto se evita usando let o const, según sea necesario

\*/

// 50. ¿Qué ocurrirá al intentar acceder al índice 3? ¿Cómo puedes evitar este error?

let Frutas = ["Manzanas", "Peras", "Mandarinas"];

document.write(frutas[3]); // Accediendo a un índice fuera de rango

/\*el código intenta acceder al índice 3 en un array con solo tres elementos (0, 1 y 2), lo que devuelve undefined. Para evitar este error, se debe verificar que el índice exista antes de acceder:

let frutas = ["Manzanas", "Peras", "Mandarinas"];

let indice = 3;

if (indice < frutas.length) {

    document.write(frutas[indice]);

} else {

    document.write("El índice está fuera de rango");

} \*/

(Martínez Allende, García Monroy, & Linares González, 2022)

AAAAAAAAAAAAAAAAAA

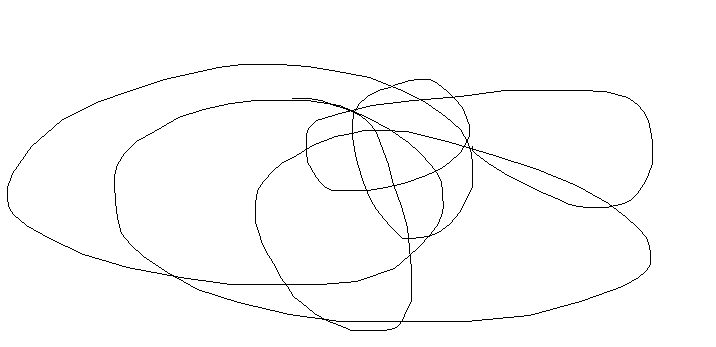


Ilustración 1. Función matematica

Fuente: elaboración propia

jjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjjj

|  |  |  |
| --- | --- | --- |
| ***Diferencias entre los ciclos, loops o bucles Repetir y Mientras*** | | |
| ***Característica*** | ***Ciclo Repetir*** | ***Ciclo Mientras*** |
| Ejecución Inicial | Siempre se ejecuta al menos 1 vez | Puede no ejecutarse |
| Verificación de Condición | Al final | Al principio |

# Bibliografía

Martínez Allende, L., García Monroy, A. I., & Linares González, E. E. (2022). El juego, estrategia pedagógica en la enseñanza de la programación y elaboración de algoritmos. *Revista Iberoamericana para la Investigación y el Desarrollo Educativo - RIDE, 13*(25), 21. doi:org/10.23913/ride.v13i25.1267