Variables y Operadores

¿Qué son las variables?

Las variables son espacios de memoria que se utilizan para almacenar datos en un programa. En la mayoría de los lenguajes de programación, las variables tienen un nombre y un tipo de datos que define el tipo de información que pueden almacenar, como números, texto, valores booleanos, entre otros.

Por ejemplo, si estás creando un programa que registra la temperatura del aire en una habitación, puedes utilizar una variable para almacenar la temperatura actual. De esta manera, el programa puede realizar operaciones utilizando el valor almacenado en la variable.

Las variables son fundamentales para la programación, ya que permiten almacenar y manipular datos de manera dinámica en un programa. A medida que los programas se vuelven más complejos, el uso correcto de las variables se vuelve cada vez más importante para garantizar que el programa funcione correctamente.

```
// Declaración de variable
let  temperaturaActual;


// Asignación de valor
temperaturaActual  =  25;


// Utilización de la variable
console.log(
"La temperatura actual es: "  +  temperaturaActual  +  " grados Celsius"
```

```
);
```

En este ejemplo, se declara una variable llamada `temperaturaActual` utilizando la palabra clave `let`. Luego se le asigna un valor de `25`. Finalmente, se utiliza la variable en una operación de concatenación de strings para imprimir un mensaje en la consola con el valor almacenado en la variable.

Null y Undefined

En JavaScript, una variable puede ser `null` o `undefined`. `null` se utiliza para indicar que una variable no tiene valor y ha sido intencionalmente asignada a este valor. `undefined`, por otro lado, indica que una variable no ha sido inicializada o que se ha declarado pero no se le ha asignado ningún valor.

```
let variable1 = null;

let variable2;


console.log(variable1); // null

console.log(variable2); // undefined
```

En el ejemplo anterior, la variable `variable1` se ha asignado explícitamente a `null`, mientras que la variable `variable2` no ha sido inicializada y, por lo tanto, tiene el valor `undefined`.

Es importante tener en cuenta que `null` es un valor asignable, mientras que `undefined` no lo es. Puedes comprobar si una variable es `null` o `undefined` utilizando operadores de igualdad:

```
console.log(variable1 === null); // true
```

```
console.log(variable2 === undefined); // true
```

En el ejemplo anterior, se utiliza el operador de igualdad (`===`) para comparar las variables con `null` y `undefined` . La primera comparación devuelve `true` , indicando que `variable1` es igual a `null` . La segunda comparación también devuelve `true` , indicando que `variable2` es igual a `undefined` .

NaN (Not a Number)

NaN es un valor especial en JavaScript que representa “No es un número”. Se produce cuando se intenta realizar una operación matemática inválida, como dividir entre cero o realizar una operación aritmética con valores no numéricos.

```
const resultado = 10 / "Hola";

console.log(resultado); // NaN
```

En el ejemplo anterior, se intenta dividir el número 10 entre la cadena de texto “Hola”. Como esto es una operación no válida, el resultado es NaN.

NaN es un valor especial de tipo numérico, por lo que se puede utilizar para detectar operaciones matemáticas inválidas en un programa. Puedes comprobar si una variable es NaN utilizando la función `isNaN()` :

```
const resultado = 10 / "Hola";

console.log(isNaN(resultado)); // true
```

En este caso, `isNaN(resultado)` devuelve `true` , indicando que el resultado es NaN.

Es importante tener en cuenta que NaN es contagioso. Esto significa que cualquier operación que involucre NaN también dará como resultado NaN.

Const, Let y Var

En JavaScript, hay tres formas de declarar variables: `const`, `let` y `var`.

`const` se utiliza para declarar variables que no cambiarán de valor en todo el programa. Una vez que se le asigna un valor a una variable `const`, no se puede cambiar.

```
const PI = 3.14;

console.log(PI); // 3.14


// Intento de reasignación (genera un error)

PI = 3.14159;
```

En este ejemplo, `PI` se declara como una constante con un valor de `3.14`. Al intentar reasignarle un nuevo valor, se producirá un error, ya que las variables `const` no pueden cambiar de valor.

`let` se utiliza para declarar variables que pueden cambiar su valor en cualquier momento durante la ejecución del programa.

```
let contador = 0;

console.log(contador); // 0


contador = contador + 1;

console.log(contador); // 1
```

En este ejemplo, se declara una variable `contador` y se le asigna el valor inicial de `0`. Luego, se incrementa su valor en `1` y se muestra por consola el nuevo valor.

`var` es la forma más antigua de declarar variables en JavaScript. A diferencia de `const` y `let`, `var` tiene un ámbito de función o global. Esto significa que una variable declarada con `var` estará disponible en toda la función o en todo el programa, dependiendo de dónde se declare.

¿Que son los Operadores?

Los operadores son elementos clave en JavaScript que nos permiten realizar diversas operaciones y manipulaciones en los valores. Los operadores se utilizan para realizar cálculos matemáticos, comparaciones, asignaciones y operaciones lógicas. Son fundamentales en la programación, ya que nos brindan la capacidad de interactuar y transformar datos de manera dinámica.

Importancia de los Operadores

Los operadores desempeñan un papel crucial en la escritura de programas en JavaScript. Aquí se presentan algunas razones por las que los operadores son importantes:

1. **Cálculos matemáticos:** Los operadores aritméticos como suma, resta, multiplicación y división nos permiten realizar cálculos matemáticos esenciales. Estos cálculos son fundamentales para resolver problemas y realizar tareas numéricas en la programación.

2. **Comparaciones y toma de decisiones:** Los operadores de comparación nos permiten comparar valores y tomar decisiones en función de los resultados. Esto es especialmente útil para realizar pruebas lógicas y controlar el flujo de ejecución en un programa. Por ejemplo, podemos determinar si un número es mayor que otro, si dos valores son iguales o si una condición se cumple.

3. **Asignación de valores:** Los operadores de asignación nos permiten asignar valores a variables. Esto es esencial para almacenar datos y realizar operaciones posteriores con esos valores. Podemos actualizar el contenido de una variable, incrementar su valor, asignar valores condicionales y más.

4. **Operaciones lógicas:** Los operadores lógicos nos permiten combinar valores booleanos y realizar operaciones lógicas como AND, OR y NOT. Estas operaciones son esenciales para evaluar múltiples condiciones y tomar decisiones basadas en ellas.

5. **Manipulación de cadenas de texto:** Los operadores en JavaScript también se utilizan para manipular cadenas de texto. Por ejemplo, el operador de concatenación (+) se utiliza para unir dos o más cadenas de texto y formar una cadena más larga.

Tipos de Operadores

JavaScript ofrece una amplia gama de operadores que cubren diferentes áreas y tipos de operaciones. Algunos tipos de operadores comunes son:

- **Operadores aritméticos:** realizan operaciones matemáticas, como suma, resta, multiplicación, división y módulo.

- **Operadores de comparación:** comparan valores y devuelven un resultado booleano, como igualdad, desigualdad, mayor que, menor que, entre otros.

- **Operadores lógicos:** permiten combinar valores booleanos y realizar operaciones lógicas, como AND, OR y NOT.

- **Operadores de asignación:** asignan valores a variables y pueden realizar operaciones junto con la asignación, como +=, -=, *=, entre otros.

- **Operadores de incremento y decremento:** se utilizan para aumentar o disminuir el valor de una variable en una unidad.

- **Operadores de concatenación:** se utilizan para unir cadenas de texto y formar una cadena más larga.

Es importante comprender y utilizar correctamente los operadores en JavaScript, ya que nos brindan herramientas poderosas para trabajar con datos y controlar el flujo de un

programa. Un buen dominio de los operadores nos permitirá escribir código más eficiente, conciso y expresivo.

Aquí tienes las tablas convertidas a listas en formato JavaScript:

Tipos de Operadores:

- Aritméticos:
  - `+`, `-`, `*`, `/`, `%`: Realizan cálculos matemáticos.
- Comparación:
  - `==`, `!=`, `>`, `<`, `>=`, `<=`: Comparan valores y devuelven un resultado booleano.
- Lógicos:
  - `&&`, `||`, `!`: Permiten combinar valores booleanos y realizar operaciones lógicas.
- Asignación:
  - `=`, `+=`, `-=`, `*=`, `/=`: Asignan valores a variables y realizan operaciones junto con la asignación.
- Incremento/Decremento:
  - `++`, `--`: Aumentan o disminuyen el valor de una variable en una unidad.
- Concatenación:
  - `+`: Unen cadenas de texto y forman una cadena más larga.