1. **var:** Tiene un ámbito de función. Si se declara dentro de una función, solo es accesible dentro de esa función. Sin embargo, si se declara fuera, es accesible en todo el script. Además, var ignora los bloques (if, for, etc.), lo que puede llevar a confusiones. El problema de var es la capacidad de redeclarar variables, que puede llevar a confusiones y errores sutiles en el código, ya que no hay advertencias cuando se vuelve a declarar una variable existente

**let:** Tiene un ámbito de bloque, lo que significa que solo es accesible dentro del bloque donde fue declarado. Esto permite un mejor control sobre el alcance de las variables y evita errores inesperados

1. **Hoisting** es un comportamiento en JavaScript donde las declaraciones de variables y funciones se mueven al principio del ámbito en el que se encuentran, antes de que se ejecuten otras líneas de código en el mismo ámbito.

* Hoisting permite usar variables y funciones antes de ser declaradas.
* Las declaraciones con var son elevadas e inicializadas como undefined.
* Las declaraciones con let y const son elevadas, pero no inicializadas, lo que puede llevar a errores si se intenta acceder a ellas prematuramente.

Ejemplos en código:

saludar(); // “Hola mundo”

function saludar() {

console.log (“Hola mundo”);

}

console.log(nombre); // undefined

var nombre = “Joaquin”;

console.log(nombre); // “Joaquin”