**Hoisting in JavaScript**

* **What is Hoisting?**
  + Hoisting is a JavaScript behavior where variable declarations are moved to the top of their scope during the compilation1 phase.
  + However, only the declarations are hoisted, not their initializations.
* **Variable Declarations (using var)**
  + var declarations are hoisted to the top of their function scope.

JavaScript

console.log(x); // Output: undefined (x is declared but not initialized)

var x = 10;

console.log(x); // Output: 10

* **Function Declarations**
  + Function declarations are also hoisted to the top of their scope.

JavaScript

myFunction(); // Output: "Hello from function"

function myFunction() {

console.log("Hello from function");

}

* **let and const Declarations**
  + let and const declarations are not hoisted, unlike var.
  + Attempting to use a let or const variable before its declaration will result in a ReferenceError.

JavaScript

console.log(y); // Output: ReferenceError: Cannot access 'y' before initialization

let y = 20;

* **Consequences of Hoisting**
  + Can lead to unexpected behavior if not understood.
  + Can make debugging more challenging.
* **Best Practices**
  + Declare variables at the beginning of their scope for better readability and to avoid potential pitfalls related to hoisting.
  + Use let and const instead of var whenever possible to avoid hoisting-related issues and improve code clarity.

**In Summary**

Hoisting is a JavaScript behavior that can sometimes lead to unexpected results. Understanding how hoisting works is important for writing correct and maintainable JavaScript code. By using let and const and declaring variables at the beginning of their scope, you can minimize the potential issues related to hoisting.