# (Notes) Var, Let and Const

## Variable Shadowing

In JavaScript, variable shadowing occurs when a variable with the same name as a variable in a higher scope is declared in a lower scope. It's a common practice but can lead to confusion if not used carefully because the inner variable will "shadow" the outer variable. In first example, let a inside the if block shadows the outer let a, and they are two separate variables despite having the same name.

## Illegal Shadowing

This occurs when trying to shadow a variable using var within the same scope where that variable is already defined using let or const. In second example, var b = "Bye"; is illegal shadowing because b is already declared using let in the same scope.

## Hoisting

In JavaScript, hoisting is a behavior where variable and function declarations are moved to the top of their containing scope during the compilation phase. However, only the declarations are hoisted, not the initializations or assignments. In third example, console.log(a); will result in undefined because the variable a is hoisted to the top but not initialized until later in the code (var a = 10;).

## Temporal Dead Zone (TDZ)

TDZ is a specific behavior related to variables declared using let and const. It refers to the period between the start of the block scope and the actual declaration of the variable. During the TDZ, accessing the variable will result in a ReferenceError. In fourth example, trying to log a, b, and c before their respective declarations will result in ReferenceError because they are in the TDZ until they are declared.