let, var and const - ES6

let

- 'let' allows you to declare variables that are mutable, meaning their values can be reassigned.

- It's perfect for situations where you need to update or reassign a variable's value within its scope.

- Used in loops and conditionals

```
let count = 10;
count = 20; // Valid: count can be reassigned
console.log(count) //Outputs: 20
```

Const

• 'const' declares variables that are immutable - once assigned, their values cannot be changed.

• This provides a sense of security, ensuring that critical values remain constant throughout your code.

• Used when defining values like configuration settings, mathematical constants, or references to objects.

```
const pi = 3.14;
pi=10; //pi cant be reassigned
//Error - Assignment to constant variable.
```

Var

• Variables declared with var are function- scoped or globally scoped, but they are not block-scoped.

• Var variables can be re-declared and updated within their scope.

• Since var doesn't have block scope, it can lead to unintended issues when used in loops and conditionals.

```
function example(){
    if (true){
      var x = 10;
    }
    console.log(x); //Outputs: 10
}
```

Aim for Immutability

- While 'let' and 'const' offer distinct advantages, embracing immutability whenever possible can lead to more predictable and bug-resistant code.

- Favor 'const' for values that shouldn't change and leverage techniques like object.freeze) for immutable objects.

FOLLOW