Day 5: Let

JavaScript ES6 Let

Here's a useful discussion related to the *let* function and a comparison with the usual *var* function.

Let

The function *let* allows you to declare variables that are limited in scope to the block, statement, or expression on which it is used.
This is unlike the *var* keyword, which defines a variable globally or locally to an entire function regardless of block scope.

Scoping Rules

Variables declared by *let* have the block, in which they are defined, as their scope, as well as any contained subblocks.

SAMPLE CODE

```
function varScope() {
  var num1 = 31;
  if (num1 % 2 == 1) {
    var num1 = 32;  // same variable, num.
    console.log(num1);  // 32
  }
  console.log(num1);  // 32
}

function letScope() {
  let num2 = 31;
  if (num2 % 2 == 1) {
    let num2 = 32;  // different variable, num.
    console.log(num2);  // 32
  }
  console.log(num2);  // 31
}

console.log("Output using varScope() :");
varScope();
console.log("");
console.log("Output using letScope() :");
letScope();
```

OUTPUT

```
Output using varScope() :
32
32

Output using letScope() :
32
31
```

Errors with let

Redeclaring the same variable within the same function or block scope raises an *Error*.

SAMPLE CODE

```
let a = 1;
let a = 2;
```

OUTPUT

```
Identifier 'a' has already been declared
```

Task

We have declared a global variable `index`. Your task is to edit the given code so that the value of the global variable doesn't change.