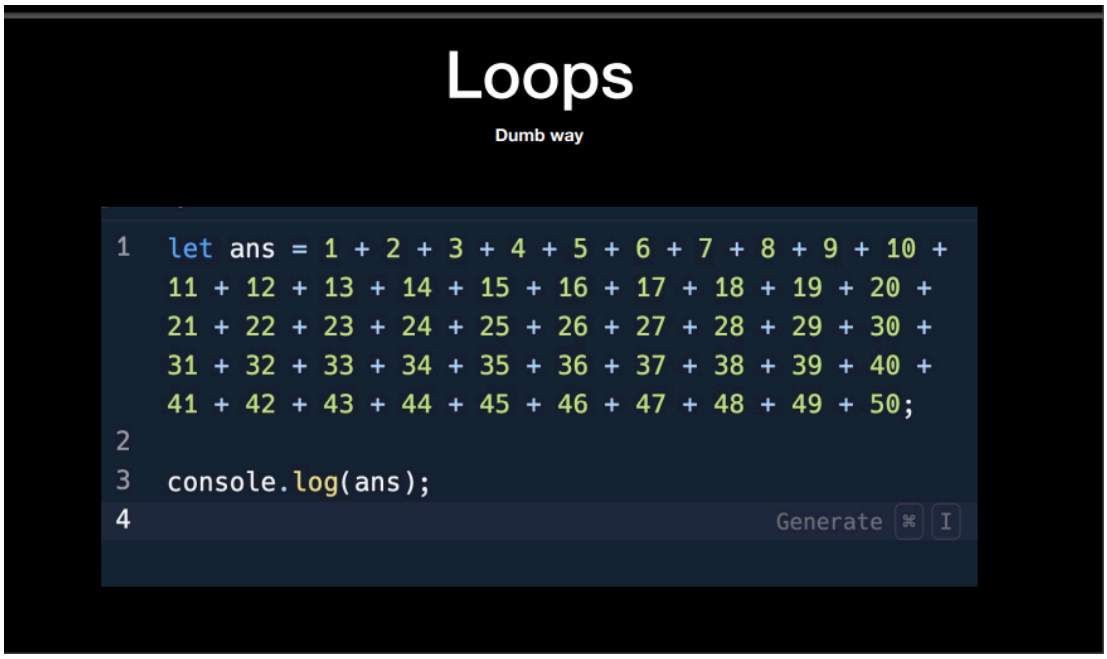
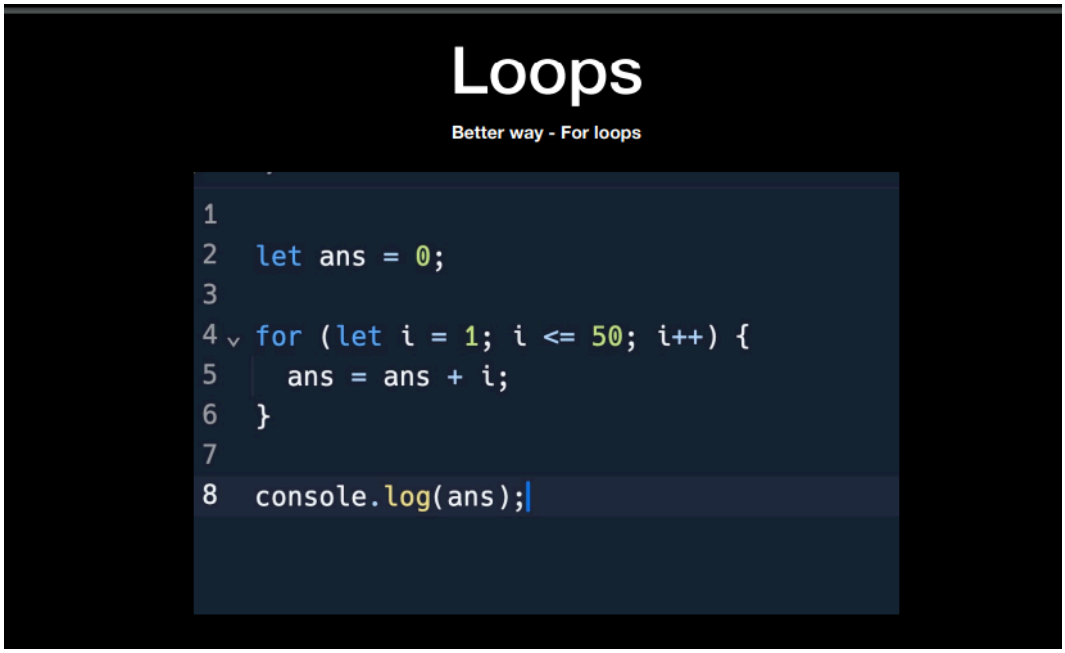
Loop, Functions, and callbacks

Calculating the sum from 1 to 50



Lot of code will be required to write from 500 or 5000

When we have to use repeated logic then we should use loop.



In for loop, the whole line doesn't immediately run,

Initialisation take place first and only once(let i =1)

The comparison section runs (i<=50) if this condition is true we will loop in when the condition is not met we will break out of for loop (to line 8)

Then control reaches line 5 (ans = ans +i) , after line 5(body of for loop) is executed it goes to i++ (updation section) then it checks the condition, and then either loop body or terminate .

To visualize your code

http://latentflip.com/loupe

Functions

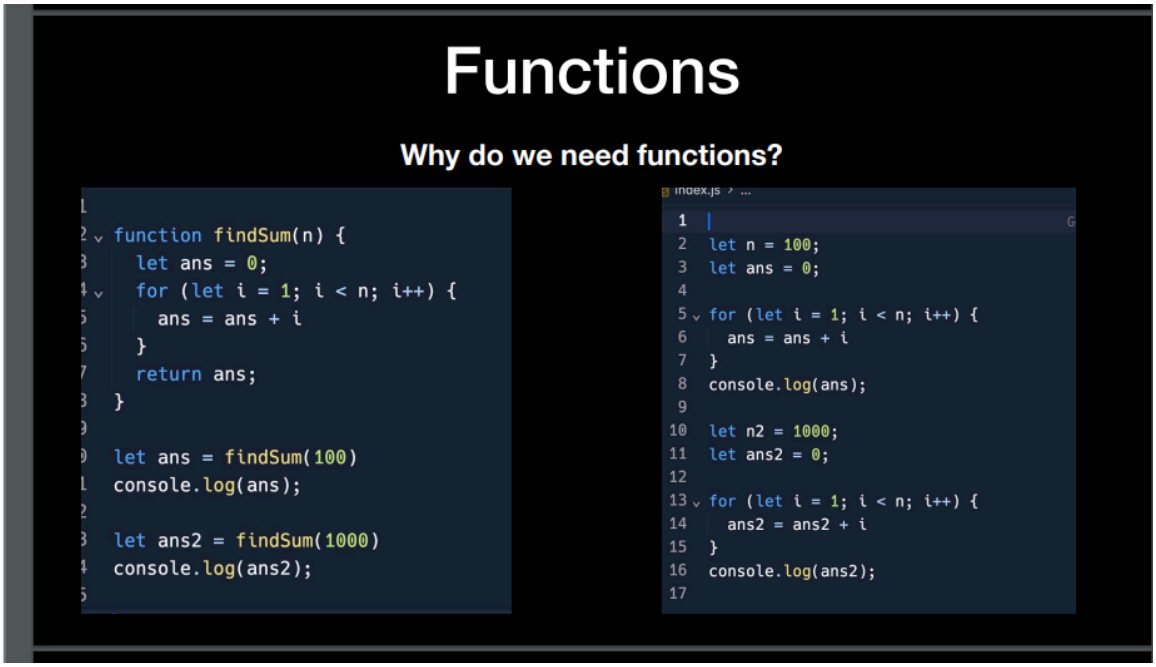
A function in JavaScript is a set of statements that performs a task or calculates a value it should take some input and return an output where there is some obvious relationship between the input and the output.

Example:

```
function sum (a,b){
    return a+b;
}
```

console.log(sum(5,6));

Why do we need functions?



Here without function, we can also do the same operations.

We need functions to prevent us from repeating the same logic

We violate **DRY** principle when we don't use functions.

Callback Functions

Basically calling function as argument

- Can we call one function inside another function ?

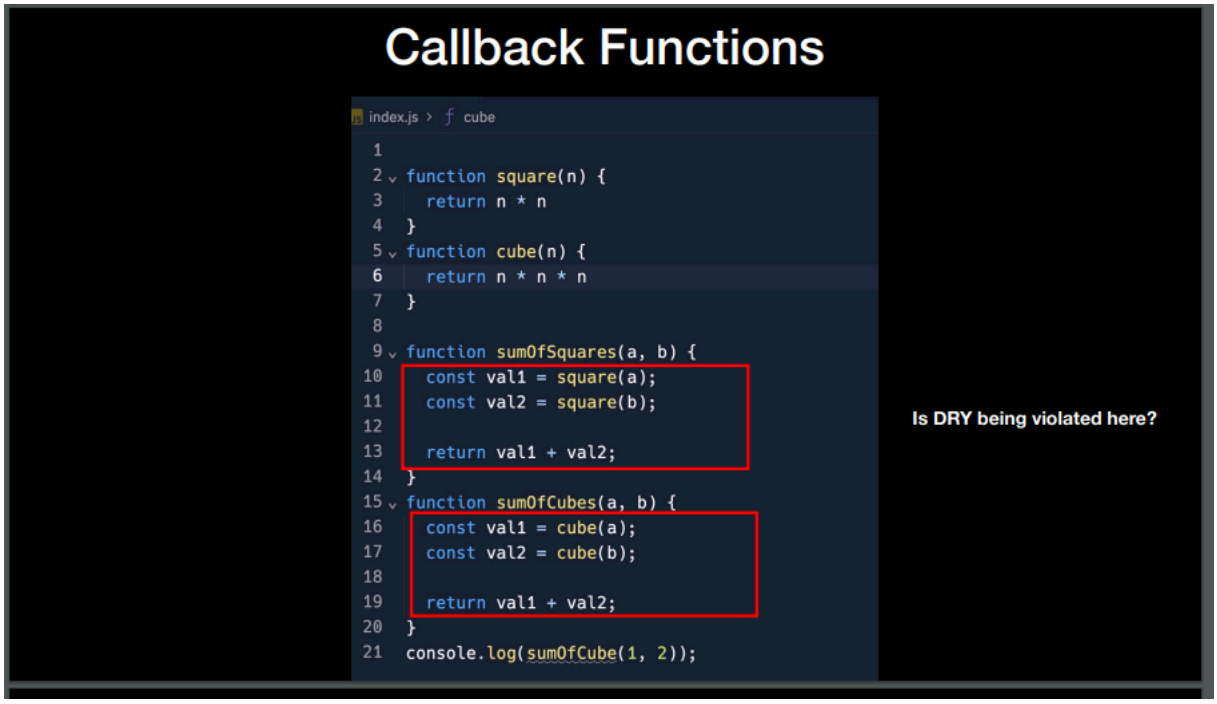
  Yes!

Now we want sum of cubes and sum of squares

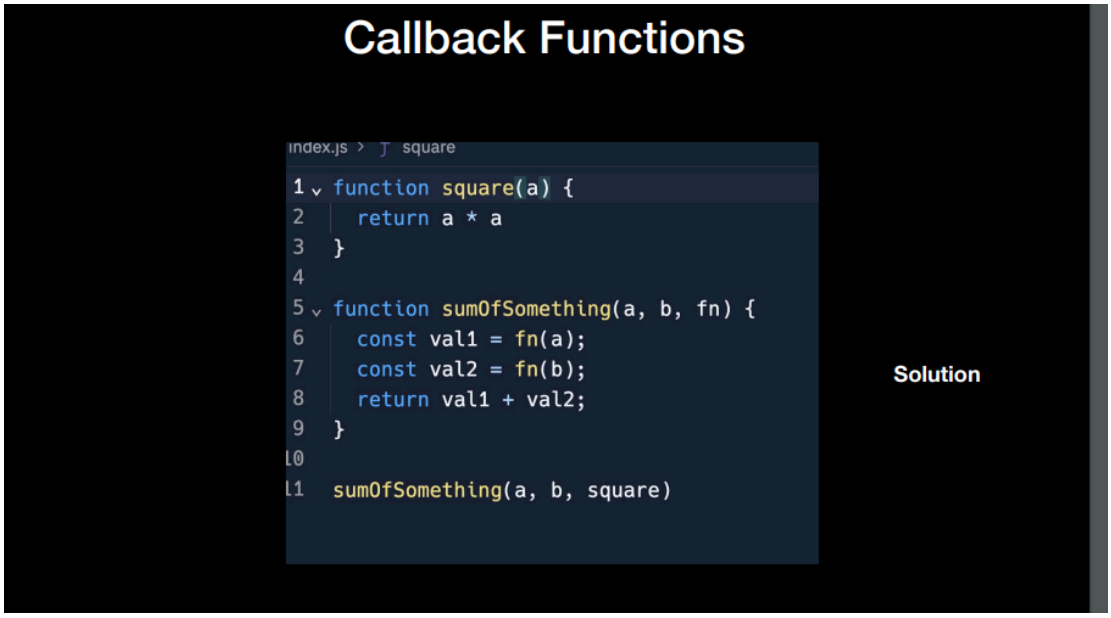
```
function sq(n){
    return n*n;
}
function cube(n){
    return n*n*n;
}
function sumOfSq(a,b){
    return sq(a)+sq(b);
}
function sumOfCube(a,b){
    return cube(a)+cube(b);
```

```
}
const ans = sumOfSq(2,3);
console.log(ans);
console.log(sumOfCube(2,3));
```

Is **DRY** being violated?



Yes, Here comes the callback in the picture!



```
function sq(n){
    return n*n;
}
function cube(n){
    return n*n*n;
}
function sumOfSomething(a,b,callback){
    console.log(callback);
    return callback(a)+callback(b);
}
const ans = sumOfSomething(2,3,sq);
console.log(ans);
console.log(sumOfSomething(2,3,cube));
```

[Function: sq]

13

[Function: cube]

35

Anonymous function:

```
function sumOfSomething(a,b,callback){
        console.log(callback);
        return callback(a)+callback(b);
    }

const ans = sumOfSomething(2,3,function(n){
    return n*n;
});
console.log(ans);
```

[Function (anonymous)]

13