

# Loops, Functions and Callbacks in JS

# Loops

**Calculate sum from 0 to 100**

# Loops

Dumb way

```
1  let ans = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 +  
    11 + 12 + 13 + 14 + 15 + 16 + 17 + 18 + 19 + 20 +  
    21 + 22 + 23 + 24 + 25 + 26 + 27 + 28 + 29 + 30 +  
    31 + 32 + 33 + 34 + 35 + 36 + 37 + 38 + 39 + 40 +  
    41 + 42 + 43 + 44 + 45 + 46 + 47 + 48 + 49 + 50;
```

2

```
3  console.log(ans);
```

4

Generate



# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5     ans = ans + i;
6 }
7
8 console.log(ans);
```

ans = 0

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 0

i = 1



# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 0

i = 1

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 1

i = 1



# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 1

i = 2

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5     ans = ans + i;
6 }
7
8 console.log(ans);
```

ans = 1

i = 2

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5   ans = ans + i;
6 }
7
8 console.log(ans);
```

ans = 3

i = 2

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

**ans = 3**

**i = 3**

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5     ans = ans + i;
6 }
7
8 console.log(ans);
```

**ans = 3**

**i = 3**

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 6

i = 3



# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 6

i = 4

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = 6

i = 4

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5   ans = ans + i;
6 }
7
8 console.log(ans);
```

ans = 10

i = 4

# Loops

Better way - For loops

```
1
2 let ans = 0;
3
4 ✓ for (let i = 1; i <= 50; i++) {
5     ans = ans + i;
6 }
7
8 console.log(ans);
```

ans = very big value

I = 51

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = very big value

I = 51

# Loops

Better way - For loops

```
1
2  let ans = 0;
3
4  for (let i = 1; i <= 50; i++) {
5      ans = ans + i;
6  }
7
8  console.log(ans);
```

ans = very big value

I = 51



# Loops

**Great way to visualise this - <http://latentflip.com/loupe/>**

# Functions

**What is a function?**

**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.**

# Functions

## Syntax?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
```

# Functions

Syntax?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
```

Function keyword

# Functions

Syntax?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
```

Name of fn

# Functions

Syntax?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
```

Arguments



# Functions

Syntax?

```
1
2 ✓ function findSum(n) {
3   let ans = 0;
4   ✓ for (let i = 1; i < n; i++) {
5     ans = ans + i
6   }
7   return ans;
8 }
9
```

Function body

# Functions

## Syntax?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
```

Return value

# Functions

## Another example

```
1  
2 ✓ function sum(a, b) {  
3   |   return a + b;  
4   }  
5
```

# Functions

## How to call a fn?

```
1
2 ✓ function findSum(n) {
3     let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5       ans = ans + i
6     }
7     return ans;
8   }
9
10  let ans = findSum(100)
11  console.log(ans);
```

**Function body**

# Functions

## How to call a fn?

```
1
2 ✓ function findSum(n) {
3   let ans = 0;
4 ✓   for (let i = 1; i < n; i++) {
5     ans = ans + i
6   }
7   return ans;
8 }
9
10 let ans = findSum(100)
11 console.log(ans);
```

**Calling function**

# Functions

**Why do we need functions?**



# Functions

## Why do we need functions?

```
1
2 ✓ function findSum(n) {
3   let ans = 0;
4 ✓ for (let i = 1; i < n; i++) {
5   ans = ans + i
6   }
7   return ans;
8 }
9
10 let ans = findSum(100)
11 console.log(ans);
12
13 let ans2 = findSum(1000)
14 console.log(ans2);
15
```

```
s index.js > ...
1 |
2 let n = 100;
3 let ans = 0;
4
5 ✓ for (let i = 1; i < n; i++) {
6   ans = ans + i
7 }
8 console.log(ans);
9
10 let n2 = 1000;
11 let ans2 = 0;
12
13 ✓ for (let i = 1; i < n; i++) {
14   ans2 = ans2 + i
15 }
16 console.log(ans2);
17
```

# Functions

You are repeating yourself (DRY)

```
1
2 function findSum(n) {
3   let ans = 0;
4   for (let i = 1; i < n; i++) {
5     ans = ans + i
6   }
7   return ans;
8 }
9
10 let ans = findSum(100)
11 console.log(ans);
12
13 let ans2 = findSum(1000)
14 console.log(ans2);
15
```

```
s index.js > ...
1
2 let n = 100;
3 let ans = 0;
4
5 for (let i = 1; i < n; i++) {
6   ans = ans + i
7 }
8 console.log(ans);
9
10 let n2 = 1000;
11 let ans2 = 0;
12
13 for (let i = 1; i < n; i++) {
14   ans2 = ans2 + i
15 }
16 console.log(ans2);
17
```

# Callback Functions

**Step 1 - Can you call one function inside another function?**

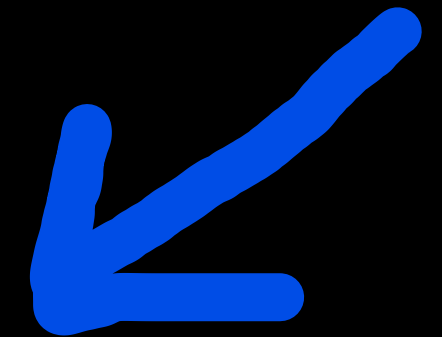


# Callback Functions

**Step 1 - Can you call one function inside another function?**

**Yes**

```
JS index.js > ...  
  
1  // finds the square of the input  
2  function square(n) {  
3    return n * n  
4  }  
5  
6  // finds the sum of the squares of the inputs  
7  function sumOfSquares(a, b) {  
8    const val1 = square(a);  
9    const val2 = square(b);  
10  
11    return val1 + val2;  
12  }  
13  
14 console.log(sumOfSquares(1, 2));
```



# Callback Functions

**Step 1 - Can you call one function inside another function?**

**Yes**

```
JS index.js > ...  
  
1  // finds the square of the input  
2  function square(n) {  
3    return n * n  
4  }  
5  
6  // finds the sum of the squares of the inputs  
7  function sumOfSquares(a, b) {  
8    const val1 = square(a);  
9    const val2 = square(b);  
10  
11    return val1 + val2;  
12  }  
13  
14 console.log(sumOfSquares(1, 2));
```

# Callback Functions

```
js index.js > f cube

1
2 ✓ function square(n) {
3   return n * n
4 }
5 ✓ function cube(n) {
6   return n * n * n
7 }
8
9 ✓ function sumOfSquares(a, b) {
10   const val1 = square(a);
11   const val2 = square(b);
12
13   return val1 + val2;
14 }
15 ✓ function sumOfCubes(a, b) {
16   const val1 = cube(a);
17   const val2 = cube(b);
18
19   return val1 + val2;
20 }
21 console.log(sumOfCube(1, 2));
```



# Callback Functions

```
js index.js > f cube

1
2 ✓ function square(n) {
3   return n * n
4 }
5 ✓ function cube(n) {
6   return n * n * n
7 }
8
9 ✓ function sumOfSquares(a, b) {
10   const val1 = square(a);
11   const val2 = square(b);
12
13   return val1 + val2;
14 }
15 ✓ function sumOfCubes(a, b) {
16   const val1 = cube(a);
17   const val2 = cube(b);
18
19   return val1 + val2;
20 }
21 console.log(sumOfCube(1, 2));
```

Is DRY being violated here?

# Callback Functions

index.js > `square`

```
1 function square(a) {  
2   return a * a  
3 }  
4  
5 function sumOfSomething(a, b, fn) {  
6   const val1 = fn(a);  
7   const val2 = fn(b);  
8   return val1 + val2;  
9 }  
10  
11 sumOfSomething(a, b, square)
```

**Solution**

# Anonymous functions

s index.js > ...

```
1
2 ✓ function sumOfSomething(a, b, fn) {
3   const val1 = fn(a);
4   const val2 = fn(b);
5   return val1 + val2;
6 }
7
8 ✓ sumOfSomething(a, b, function(a) {
9   return a * a
10 })
```

11

Genera