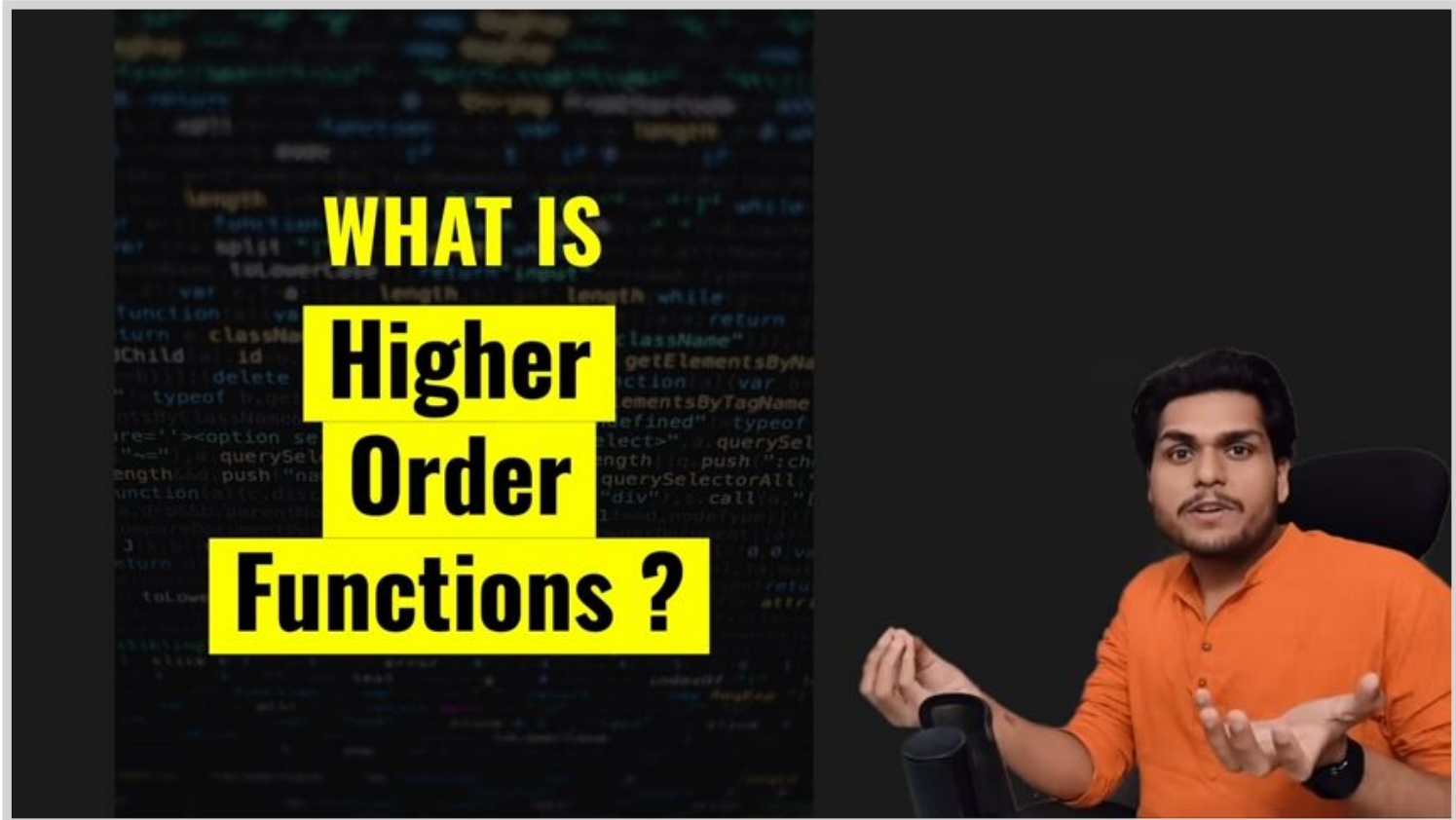


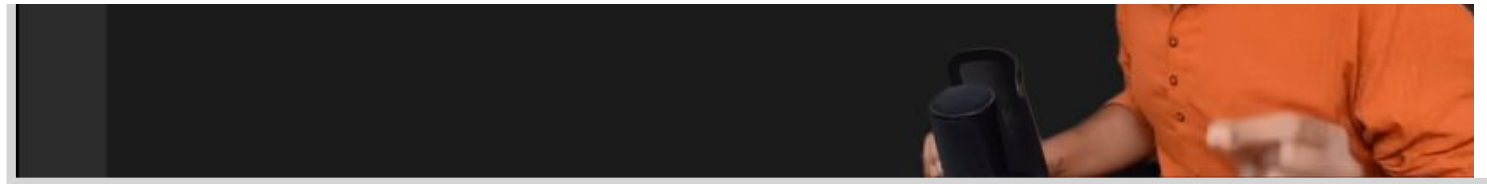
00:38



A function which takes another function as an argument or return a function from it is known as **Higher Order Functions**

01:29





HERE,-> y() is the Higher order funtion & x() is the callBack function

04:00

```
JS index.js x JS higher-order-functions.js
js > JS index.js
1  const radius = [3, 1, 2, 4];
2
3  const calculateArea = function (radius) {
4    const output = [];
5    for (let i = 0; i < radius.length; i++) {
6      output.push(Math.PI * radius[i] * radius[i]);
7    }
8    return output;
9  };
10
```

04:17

```
JS index.js x JS higher-order-functions.js
js > JS index.js
1  const radius = [3, 1, 2, 4];
2
3  const calculateArea = function (radius) {
4    const output = [];
5    for (let i = 0; i < radius.length; i++) {
6      output.push(Math.PI * radius[i] * radius[i]);
7    }
8    return output;
9  };
10
11 console.log(calculateArea(radius));
12
```

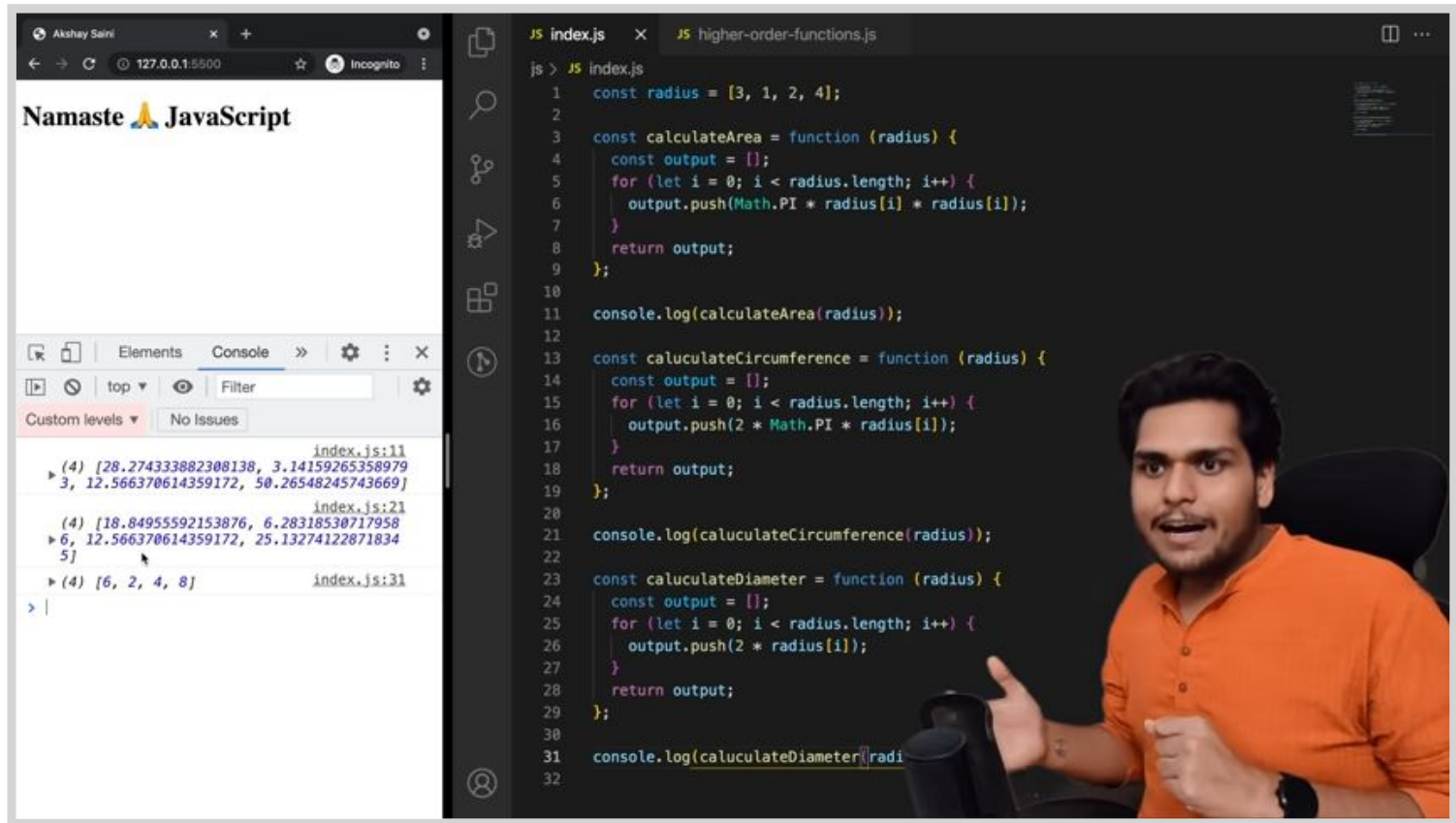
Namaste 🙏 JavaScript

Elements Console >> Filter

Custom levels No Issues

index.js:11
(4) [28.274333882308138, 3.141592653589793, 1.566370614359172, 50.26548245743669]

06:25



The screenshot shows a web browser on the left with the title "Namaste 🙏 JavaScript" and a code editor on the right. The code editor has two tabs: "JS index.js" and "JS higher-order-functions.js". The "JS index.js" tab is active, showing the following code:

```
1 const radius = [3, 1, 2, 4];
2
3 const calculateArea = function (radius) {
4   const output = [];
5   for (let i = 0; i < radius.length; i++) {
6     output.push(Math.PI * radius[i] * radius[i]);
7   }
8   return output;
9 };
10
11 console.log(calculateArea(radius));
12
13 const calculateCircumference = function (radius) {
14   const output = [];
15   for (let i = 0; i < radius.length; i++) {
16     output.push(2 * Math.PI * radius[i]);
17   }
18   return output;
19 };
20
21 console.log(calculateCircumference(radius));
22
23 const calculateDiameter = function (radius) {
24   const output = [];
25   for (let i = 0; i < radius.length; i++) {
26     output.push(2 * radius[i]);
27   }
28   return output;
29 };
30
31 console.log(calculateDiameter(radius));
32
```

The browser's console shows the output of the code:

```
index.js:11 (4) [28.274333882308138, 3.141592653589793, 12.566370614359172, 50.26548245743669]
index.js:21 (4) [18.84955592153876, 6.283185307179586, 12.566370614359172, 25.132741228718345]
index.js:31 (4) [6, 2, 4, 8]
```

the problems here are:-

07:10



The screenshot shows the same web browser and code editor as before. A large yellow text overlay is centered on the screen, reading:

DRY Principle
Don't Repeat Yourself

The code editor shows the same code as before, but the "JS higher-order-functions.js" tab is now active, showing the following code:

```
1 const radius = [3, 1, 2, 4];
2
3 function calculateArea(radius) {
4   const output = [];
5   for (let i = 0; i < radius.length; i++) {
6     output.push(Math.PI * radius[i] * radius[i]);
7   }
8   return output;
9 }
10
11 console.log(calculateArea(radius));
12
13 function calculateCircumference(radius) {
14   const output = [];
15   for (let i = 0; i < radius.length; i++) {
16     output.push(2 * Math.PI * radius[i]);
17   }
18   return output;
19 }
20
21 console.log(calculateCircumference(radius));
22
23 function calculateDiameter(radius) {
24   const output = [];
25   for (let i = 0; i < radius.length; i++) {
26     output.push(2 * radius[i]);
27   }
28   return output;
29 }
30
31 console.log(calculateDiameter(radius));
32
```



17:38

Namaste 🙏 JavaScript

Elements

Console

»

⚙️

⋮

✕

top ▾ 🔍 Filter ⚙️

Custom levels ▾ No Issues

index.js:23

(4) [28.274333882308138, 3.141592653589793, 12.566370614359172, 50.26548245743669]

index.js:25

(4) [28.274333882308138, 3.141592653589793, 12.566370614359172, 50.26548245743669]

>

```
2
3 ✓ const area = function (radius) {
4   return Math.PI * radius * radius;
5 };
6
7 ✓ const circumference = function (radius) {
8   return 2 * Math.PI * radius;
9 };
10
11 ✓ const diameter = function (radius) {
12   return 2 * radius;
13 };
14
15 ✓ const calculate = function (radius, logic) {
16   const output = [];
17   for (let i = 0; i < radius.length; i++) {
18     output.push(logic(radius[i]));
19   }
20   return output;
21 };
22
23 console.log(radius.map(area));
24
25 console.log(calculate(radius, area));
26 // console.log(calculate(radius, circumf
27 // console.log(calculate(radius, diame
28
```

4

const c

5

for (le

6

output

7

}

8

return

9

};

10

11

console.

12

13

const ca

14

const c

15

for (le

16

output

17

}

18

return

19

};

20

