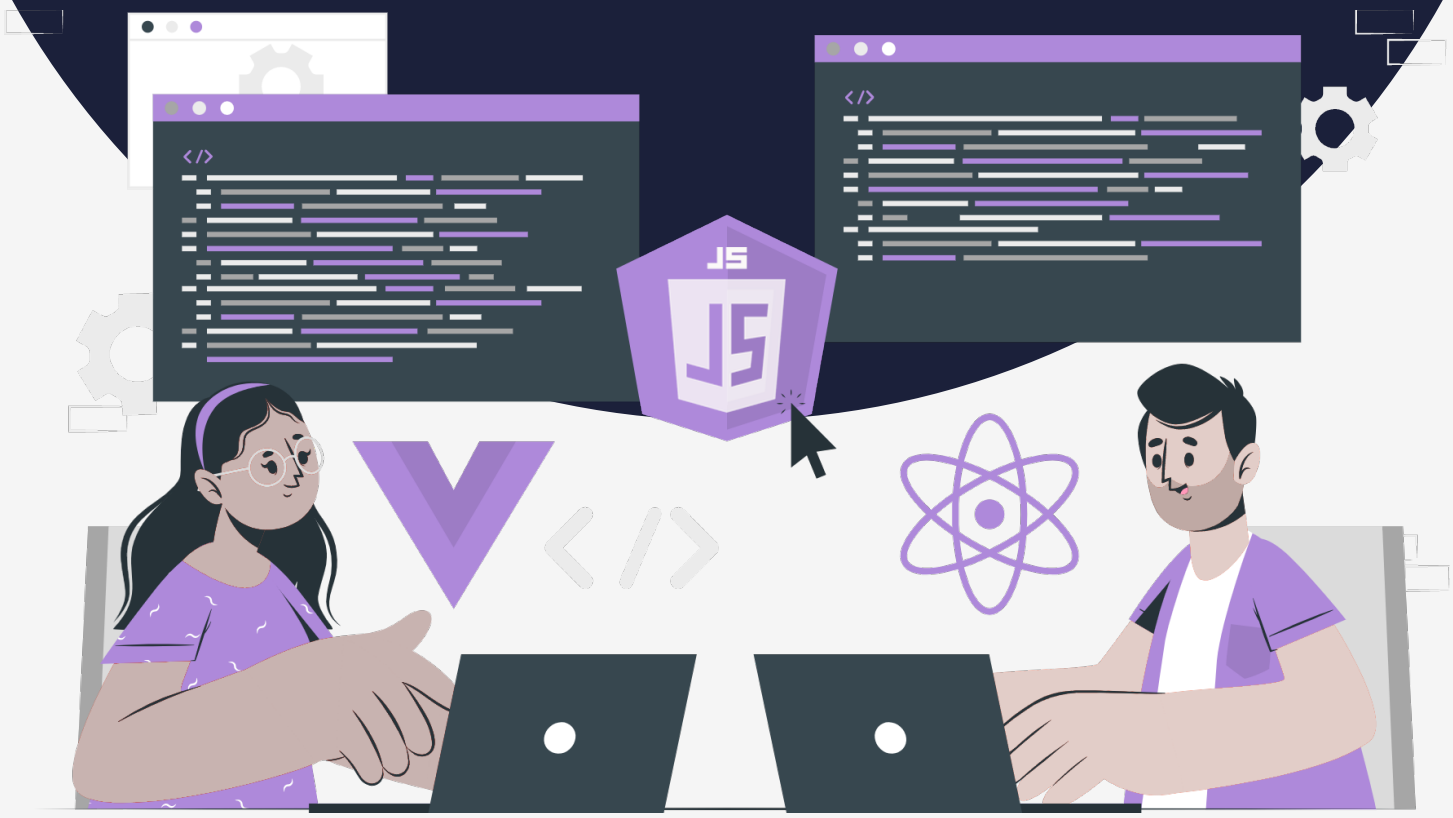
Lesson:

What are higher-order functions



PW SKILLS

List of content :

What are higher-order functions?

The functions that use only primitives (pre-defined datatypes) or objects as arguments, and only return primitives or objects are named first-order functions.

However, functions are treated as first-class citizens in JavaScript which means that functions can be

- assigned to different variables,

```
const hello = function() {
  return 'Hello!'
};
hello();
```

- passed as arguments to different functions

```
function useFunction(func) {
  return func();
}
useFunction(function () { return 7 });
```

- returned from different functions.

```
function returnFunction() {
  return function() { return 7 };
}
const exFunc = returnFunction();
exFunc();
```

Higher-order functions are, in fact, functions that accept another function as an argument or return another function.

In the above examples, useFunction() is a higher-order function because it accepts a function as an argument. Also, returnFunction() is a higher-order function because it returns another function.

```
function calculatorFunction(operation, initialValue, numbers) {
  let total = initialValue;
  for (const number of numbers) {
    total = operation(total, number);
  }
  return total;
}
function sum(n1, n2) {
  return n1 + n2;
}
function multiply(n1, n2) {
  return n1 * n2;
}
calculatorFunction(sum, 0, [1, 3, 4]);     // 8
calculatorFunction(multiply, 1, [1, 3, 4]); // 12
```

Explanation:
Here, calculatorFunction(operation, initialValue, numbers) is a higher-order function as it accepts a function as the first argument.

sum() is the function that describes the addition operation. calculatorFunction(sum, 0, [1, 3, 4]) is using sum() function to perform the sum of numbers.

Similarly, multiply() describes the multiplication operation. calculatorFunction(multiply, 1, [1, 3, 4]) is using multiply() function to perform the product of numbers.