11. Loops, Arrays, and Functions in Javascript

Table of Content

1. Loops in Javascript

In programming, loops are used to repeat a block of code. For example, if you want to show a message 100 times, then you can use a loop.

▼ **Javascript for loop**

The syntax of the `for` loop is:

```
for (initialExpression; condition; updateExpression) {
    // for loop body
}
```

Here,

1. The **initialExpression** initializes and/or declares variables and executes only once.
2. The **condition** is evaluated.
    - If the condition is `false`, the `for` loop is terminated.
    - If the condition is `true`, the block of code inside of the `for` loop is executed.
3. The **updateExpression** updates the value of **initialExpression** when the condition is `true`.

4. The **condition** is evaluated again. This process continues until the condition is `false`.

```
// program to display text 5 times
const n = 5;

// looping from i = 1 to 5
for (let i = 1; i <= n; i++) {
    console.log(`I love JavaScript.`);
}
```

▼ Javascript while loop

The syntax of the `while` loop is:

```
while (condition) {
    // body of loop
}
```

Here,

1. A `while` loop evaluates the **condition** inside the parenthesis `()`.
2. If the **condition** evaluates to `true`, the code inside the `while` loop is executed.
3. The **condition** is evaluated again.
4. This process continues until the **condition** is `false`.
5. When the **condition** evaluates to `false`, the loop stops.

▼ Javascript do-while loop

The syntax of `do...while` loop is:

```
do {
    // body of loop
} while(condition)
```

Here,

1. The body of the loop is executed at first. Then the **condition** is evaluated.
2. If the **condition** evaluates to `true`, the body of the loop inside the `do` statement is executed again.
3. The **condition** is evaluated once again.
4. If the **condition** evaluates to `true`, the body of the loop inside the `do` statement is executed again.
5. This process continues until the **condition** evaluates to `false`. Then the loop stops.

> **Note**: do...while loop is similar to the while loop. The only difference is that in do…while loop, the body of loop is executed at least once.

▼ break and continue

The `break` statement is used to terminate the loop immediately when it is encountered.

The `continue` statement is used to skip the current iteration of the loop and the control flow of the program goes to the next iteration.

▼ Javascript for…in loop

The `for..in` loop in JavaScript allows you to iterate over all property keys of an object.

The syntax of the `for...in` loop is:

```
for (key in object) {
    // body of for...in
}
```

In each iteration of the loop, a key is assigned to the key variable. The loop continues for all object properties.

You can use for…in with Strings, Arrays and Objects in Javascript.

2. Arrays in Javascript

An array is an object that can store multiple values at once. For example,

```
const words = ['hello', 'world', 'welcome'];
```

Here, `words` is an array. The array is storing 3 values.

▼ **Creating an Array:**

- The easiest way to create an array is by using an array literal `[]`. For example,

  ```
  const array1 = ["eat", "sleep"];
  ```

- You can also create an array using JavaScript's `new` keyword.

  ```
  const array2 = new Array("eat", "sleep");
  ```

> **Note**: Array's index starts with 0, not 1.

We can use the `length` property to find the length of the array.

▼ **Some Common Array Methods**

- `push()` adds a new element to the end of an array and returns the new length of an array
- `pop()`**:** removes the last element of an array and returns the removed element
- `forEach()`**:** calls a function for each element
- `sort()`**:** sorts the elements alphabetically in strings and in ascending order
- `includes()`**:** checks if an array contains a specified element
- `indexOf()`**:** searches an element of an array and returns its position
- `splice()`**:** removes or replaces existing elements and/or adds new elements

3. Functions in Javascript

A function is a block of code that performs a specific task. Dividing a complex problem into smaller chunks makes your program easy to understand and reusable.

▼ Declaring a Function

The syntax to declare a function is:

```
function nameOfFunction () {
    // function body
}
```

A function is declared using the `function` keyword.

▼ Calling a Function

```
function nameOfFunction () {
    // function body
}

nameOfFunction(); //calling the function
```

▼ Function Parameters

```
// program to print the text
// declaring a function
function greet(name) {
    console.log("Hello " + name + ":)");
}

function add(a, b) {
    console.log(a + b);
}
```

▼ Function Return

The `return` statement can be used to return the value to a function call.

The `return` statement denotes that the function has ended. Any code after `return` is not executed.

If nothing is returned, the function returns an `undefined` value.

▼ Function Expressions

In Javascript, functions can also be defined as expressions. For example,

```
let x = function (num) { return num * num };
console.log(x(4));

let y = x(3);
console.log(y);
```

Output

```
16
9
```

In the above program, variable x is used to store the function. Here the function is treated as an **expression**. And the function is called using the variable name.

The function above is called an **anonymous function.**

▼ Arrow Functions

The arrow function is one of the features introduced in the ES6 version of JavaScript. It allows you to create functions in a cleaner way compared to regular functions. For example,This function

```
// function expression
let multiply = function(x, y) {
   return x * y;
}
```

can be written as

```
// using arrow functions
let multiply = (x, y) => x * y;
```

using an arrow function.

```
let sum = (a, b) => {
    let result = a + b;
    return result;
}

let result1 = sum(5,7);
```

Assignment

1. Write a JavaScript function that checks whether a passed string is palindrome or not.
2. Write a JavaScript function that returns a passed string with letters in alphabetical order.
3. Write a JavaScript function to calculate the average of marks passed in an array.
4. Learn about various String methods in Javascript: https://www.programiz.com/javascript/library/string
5. Learn about various Math methods in Javascript: https://www.programiz.com/javascript/library/math