**LOOP STATEMENTS**

Loop statements are the primary mechanism for telling a JavaScript interpreter to execute statements again and again until a specified condition is met. JavaScript supports following looping statements :

* for loop
* for/in a loop (explained later)
* while loop
* do…while loop

Most loops have a counter variable which is initialised before the loop starts and then it is tested as part of the condition (expression)evaluated before every iteration of the loop. Finally, the counter variable is incremented or updated at the end of the loop body just before the condition is evaluated again.

**For**

The for loop consists of three optional expressions separated by semicolon, followed by a block of statements executed in the loop. Loop statements executed repeatedly again and again until the condition is false. The for loop is used when we know in advance how many times the script code should run.

The Syntax is

for ([initial-expression];[condition];[increment-expression])

{

statements

}

**Parameters**

Initial-expression- is used to initialise a counter variable.

Condition- If condition evaluates to true, the statements are executed.

Incr-expression- to increment the counter variable.

Example

Let us say we want to print numbers from 0 to 10. The code will be as follows.

<script>

for (let i=0; i<10; i++)

{

document.write(i + "<br/>");

}

</script>

Take note, you can include an html <br/> tag so that the output is displayed line by line.

We could also modify the loop increment if for instance we wanted to print 0, 2, 4, 6, 8. The code will be as shown below.

<script>

for (let i=0; i<10; i+=2)

{

document.write(i+"<br/>");

}

</script>

We can also have a countdown loop. In this instance, we will have to modify our code as below.

<script>

for (let i=10; i>0; i-=2)

{

document.write(i+"<br/>");

}

</script>

The output of the above code will be :

10  
8  
6  
4  
2

**Implementing a break**

We can implement a break which causes the loop to terminate once a certain condition has been met. For instance in the previous example, we may want to stop executing the loop once sum is 49. the code will look as below:

<script>

document.write("<h2>7's Multiplication Table</h2>")

for (let i=1; i<=10; i++)

{

let sum= i \* 7

document.write(i + " x " + " 7 " + " = " + sum +"<br/>");

if (sum==49)

break;

}

</script>

**Implementing a continue**

The continue statement skips the statement following it and executes the loop with next iteration. It is used along with an if statement inside while, do-while, for, or label statements. The code below will print everything except the value 49.

<script>

document.write("<h2>7's Multiplication Table</h2>")

for (let i=1; i<=10; i++)

{

let sum= i \* 7

if (sum==49)

continue;

document.write(i + " x " + " 7 " + " = " + sum +"<br/>");

}

</script>

**Task 1**

Write a JavaScript for loop that will iterate from 1 to 20. For each iteration, it will check if the current number is odd or even, and display a message to the screen.

Sample Output :   
"1 is odd"   
"2 is even"

task1.js

**Task 2:(This has been covered under conditionals)**

Create a function called calculateGrade() which takes an array of numbers (subject marks). This function should work out the average grade, and depending on the average, return one of the following:

* 1. 1-59: F
  2. 60-69: D
  3. 70-79: C
  4. 80-89: B
  5. 90-100: A

**Task 3:**

Create a function called sum() which takes a limit argument. The function should return the sum of all the multiples of 3 and 5 between 0 and the limit.

**Task 4**

Create a function called showPrimes() which takes a limit argument. The function should return all the prime numbers up to the limit. \*Note: a prime number can only be divided by 1 and itself.