JavaScript Statements and Loops

if statement

The **if** statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.

Syntax

The syntax for a basic if statement is as follows −

```
if (expression)
{
  Statement(s) to be executed if expression is true
}
```

Here a JavaScript expression is evaluated. If the resulting value is true, the given statement(s) are executed. If the expression is false, then no statement would be not executed. Most of the times, you will use comparison operators while making decisions.

Example

```
<html>
  <body>

    <script type="text/javascript">

        var age = 20;

        if( age > 18 )
       {
          document.write("Qualifies for driving");
        }

    </script>


  </body>
</html>
```

if...else statement:

The **'if...else'** statement is the next form of control statement that allows JavaScript to execute statements in a more controlled way.

Syntax

```
if (expression)
{
Statement(s) to be executed if expression is true
}
Else
{
Statement(s) to be executed if expression is false
}
```

Here JavaScript expression is evaluated. If the resulting value is true, the given statement(s) in the ‘if’ block, are executed. If the expression is false, then the given statement(s) in the else block are executed.

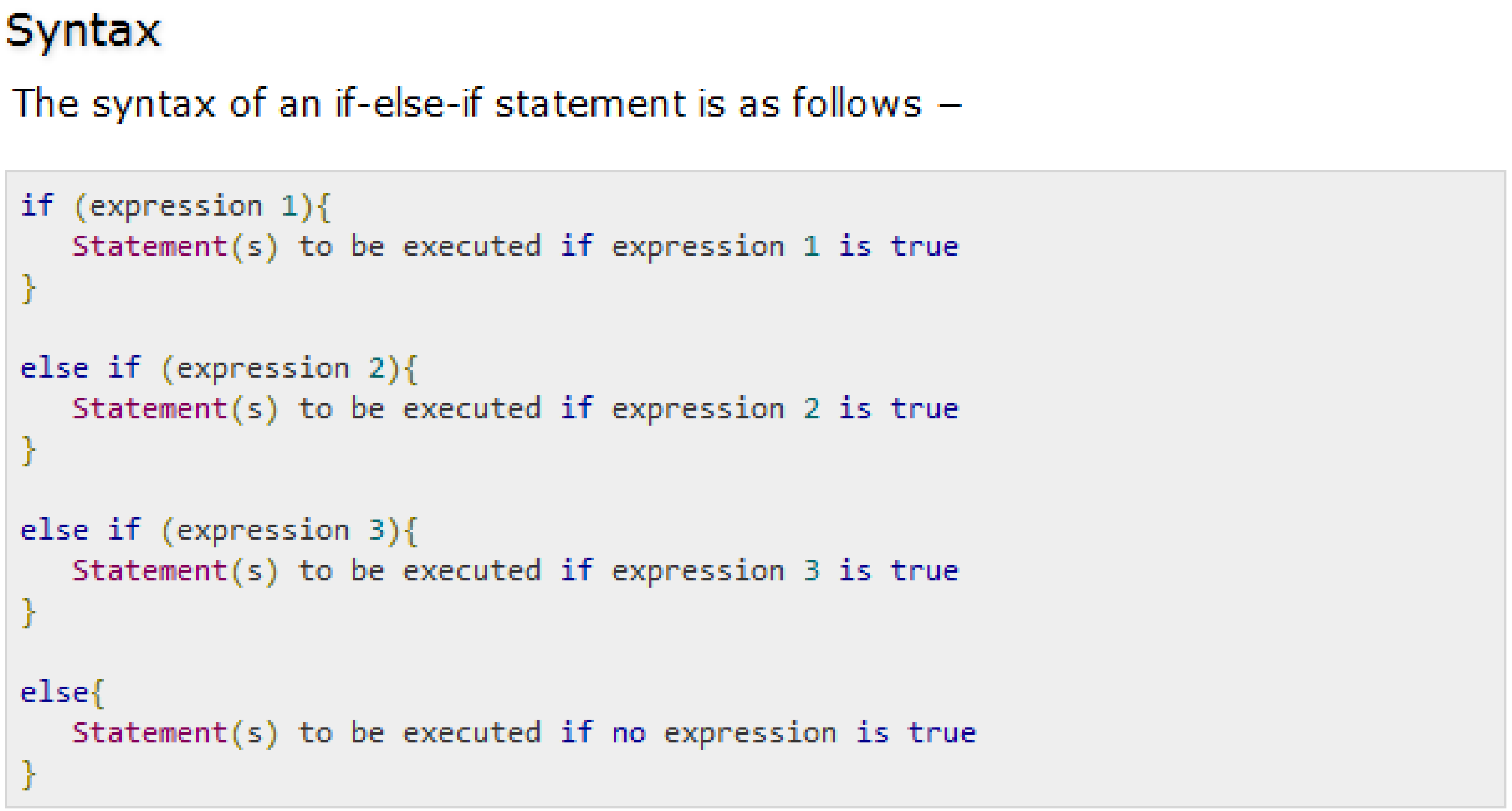
Example of if-else

```
<html> <body>
<script type="text/javascript">

var age = 15;

 if( age > 18 )
{
document.write("<b>Qualifies for driving</b>");
 }
else
{
 document.write("<b>Does not qualify for driving</b>");
}
</script>
</body>
 </html>
```

if...else if... Statement

The **if...else if...** statement is an advanced form of **if…else** that allows JavaScript to make a correct decision out of several conditions.

syntax

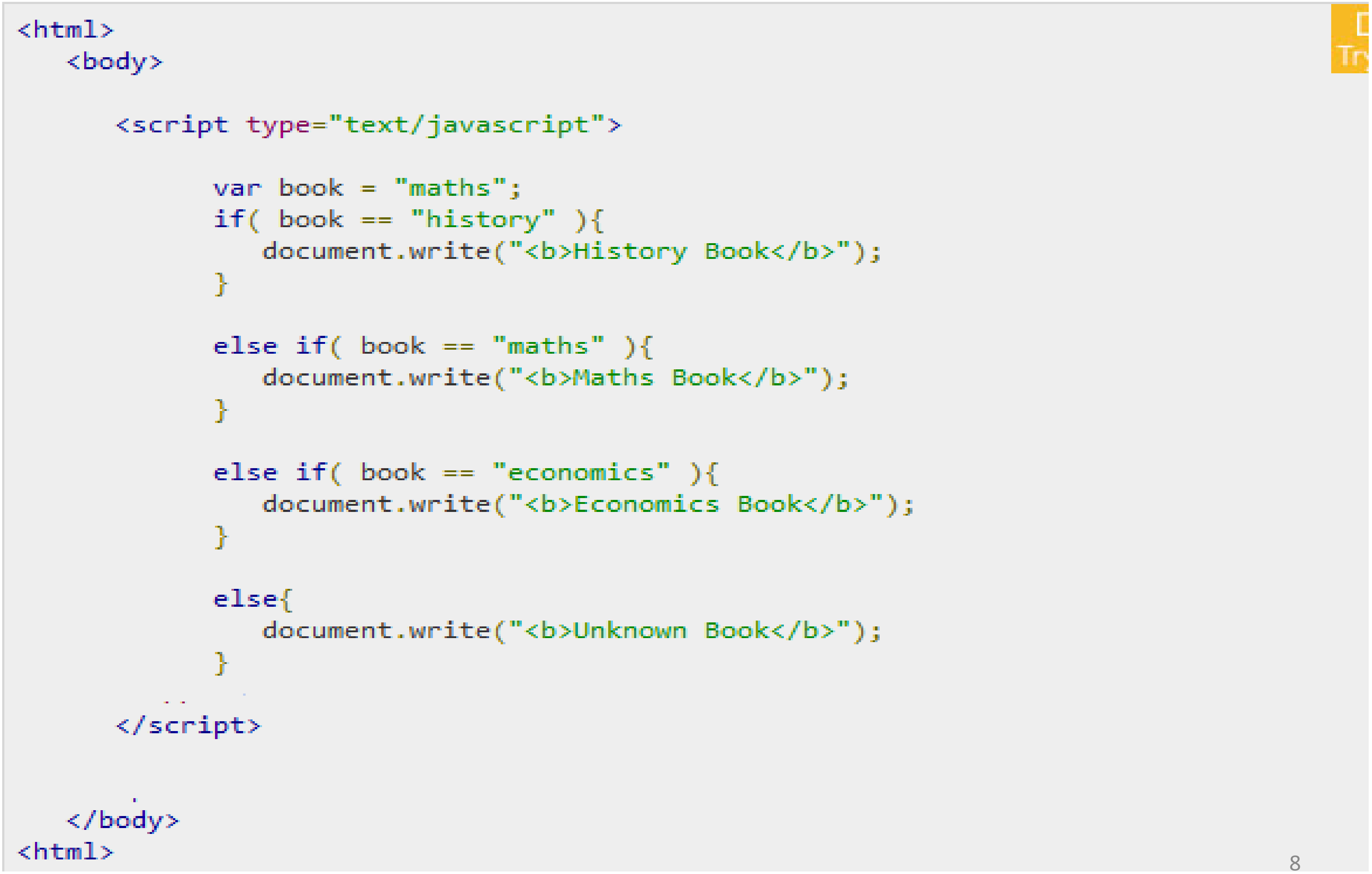


Syntax

The syntax of an if-else-if statement is as follows –

```
if (expression 1){
   Statement(s) to be executed if expression 1 is true
}

else if (expression 2){
   Statement(s) to be executed if expression 2 is true
}

else if (expression 3){
   Statement(s) to be executed if expression 3 is true
}

else{
   Statement(s) to be executed if no expression is true
}
```

Example if-else-if



```
<html>
   <body>

      <script type="text/javascript">

            var book = "maths";
            if( book == "history" ){
               document.write("<b>History Book</b>");
            }

            else if( book == "maths" ){
               document.write("<b>Maths Book</b>");
            }

            else if( book == "economics" ){
               document.write("<b>Economics Book</b>");
            }

            else{
               document.write("<b>Unknown Book</b>");
            }

      </script>


   </body>
<html>
```

Practice question

Q. WAP to accept a number from user using prompt box.

Then check whether the number is even or odd

solution

```
<html><body>

   <script>

      var n = prompt("enter the number", "write here");

      if(n==0)
     {
        document.write("number is zero");
      }
      else if(n%2==0)
     {
     document.write("even");
      }

    else
     {
     document.write("odd");
     }

   </script>  </body>
```

Practice question

WAP to accept age of user (with help of prompt box) and check whether the user is eligible for driving or not?

solution

```
<html>
  <body>

    <script type="text/javascript">

        var age = prompt("enter age", "write here");

        if( age > 18 )
        {
          document.write("Qualifies for driving");
        }

        else
       {
       document.write("not qualified for driving");
        }

    </script>
    <body> </html>
```

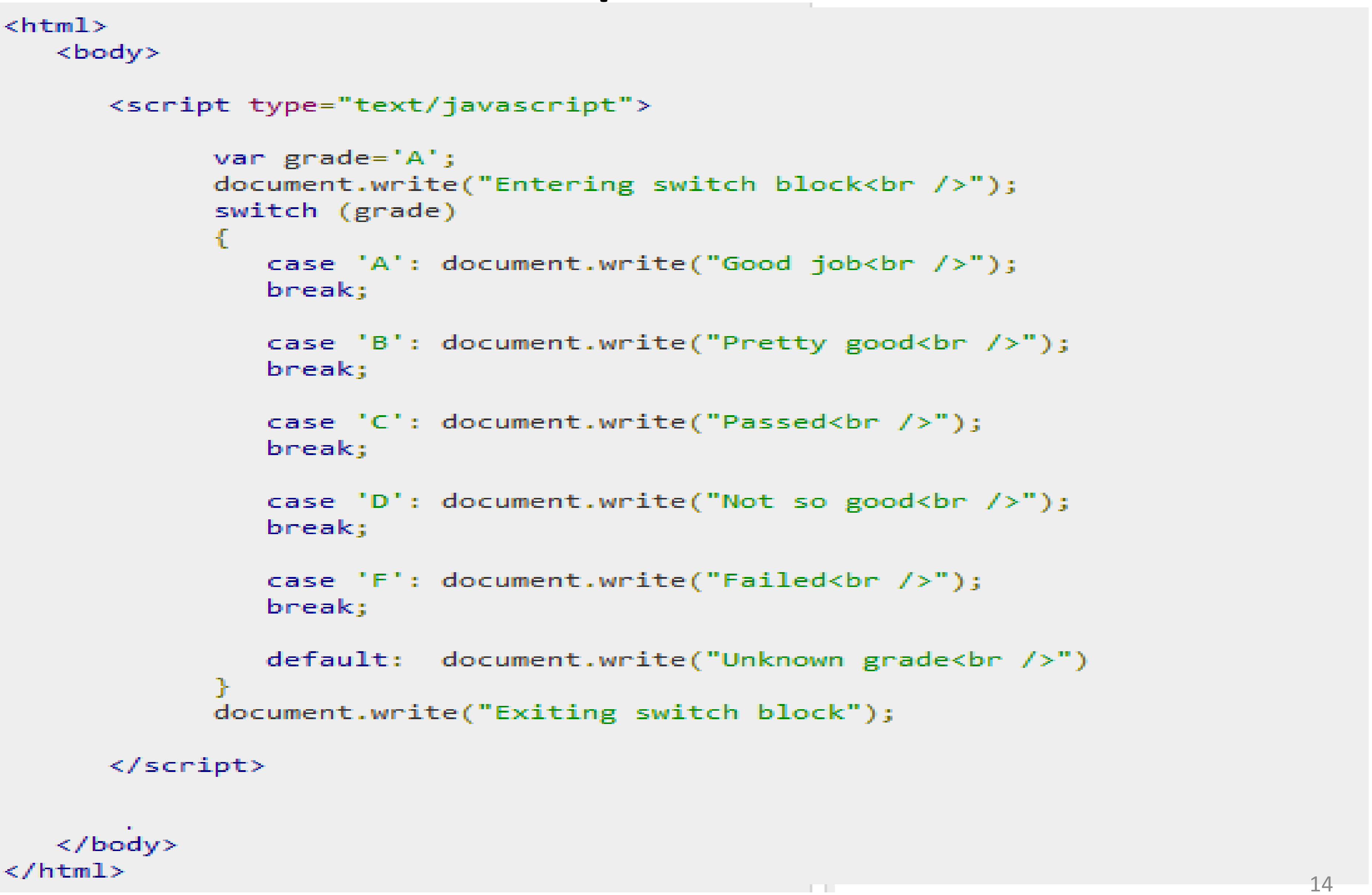
switch

Syntax

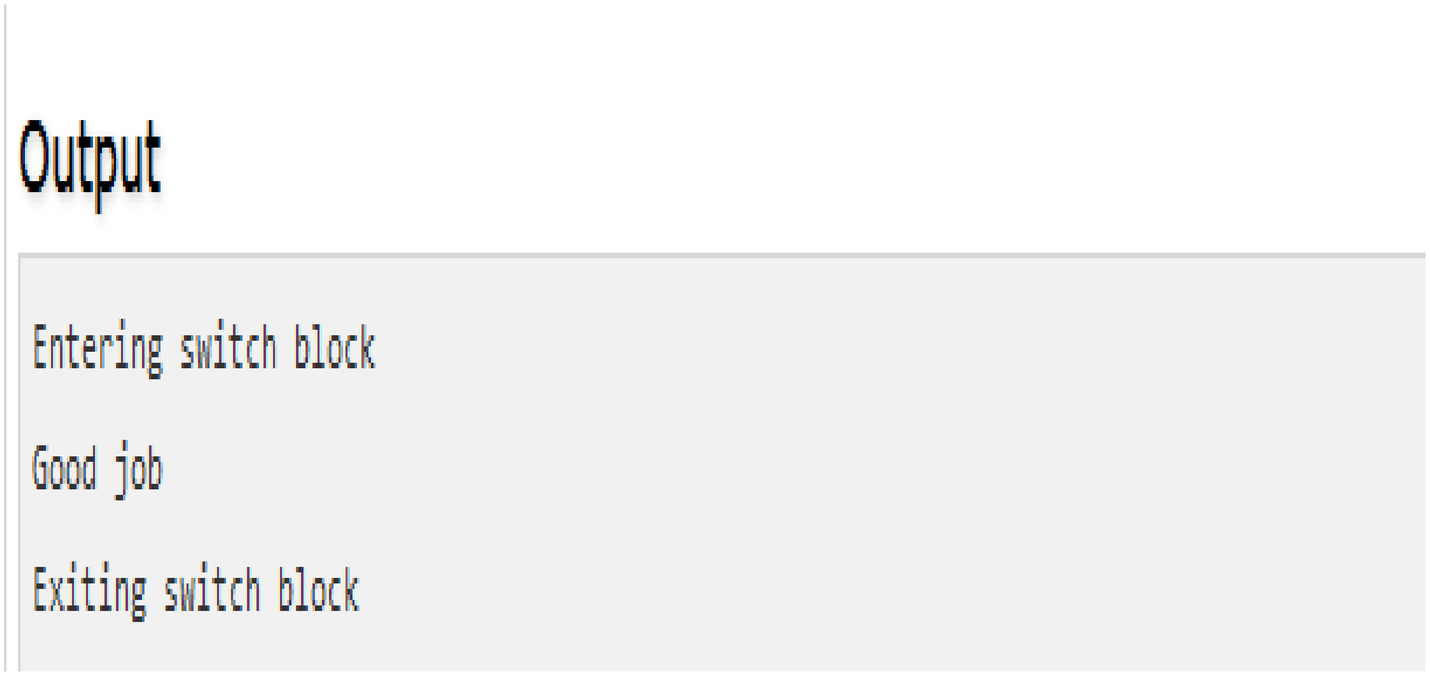
The objective of a **switch** statement is to give an expression to evaluate and several different statements to execute based on the value of the expression. The interpreter checks each **case** against the value of the expression until a match is found. If nothing matches, a **default** condition will be used.

```
switch (expression)
{
   case condition 1: statement(s)
   break;

   case condition 2: statement(s)
   break;
   ...

   case condition n: statement(s)
   break;

   default: statement(s)
}
```

Example of switch



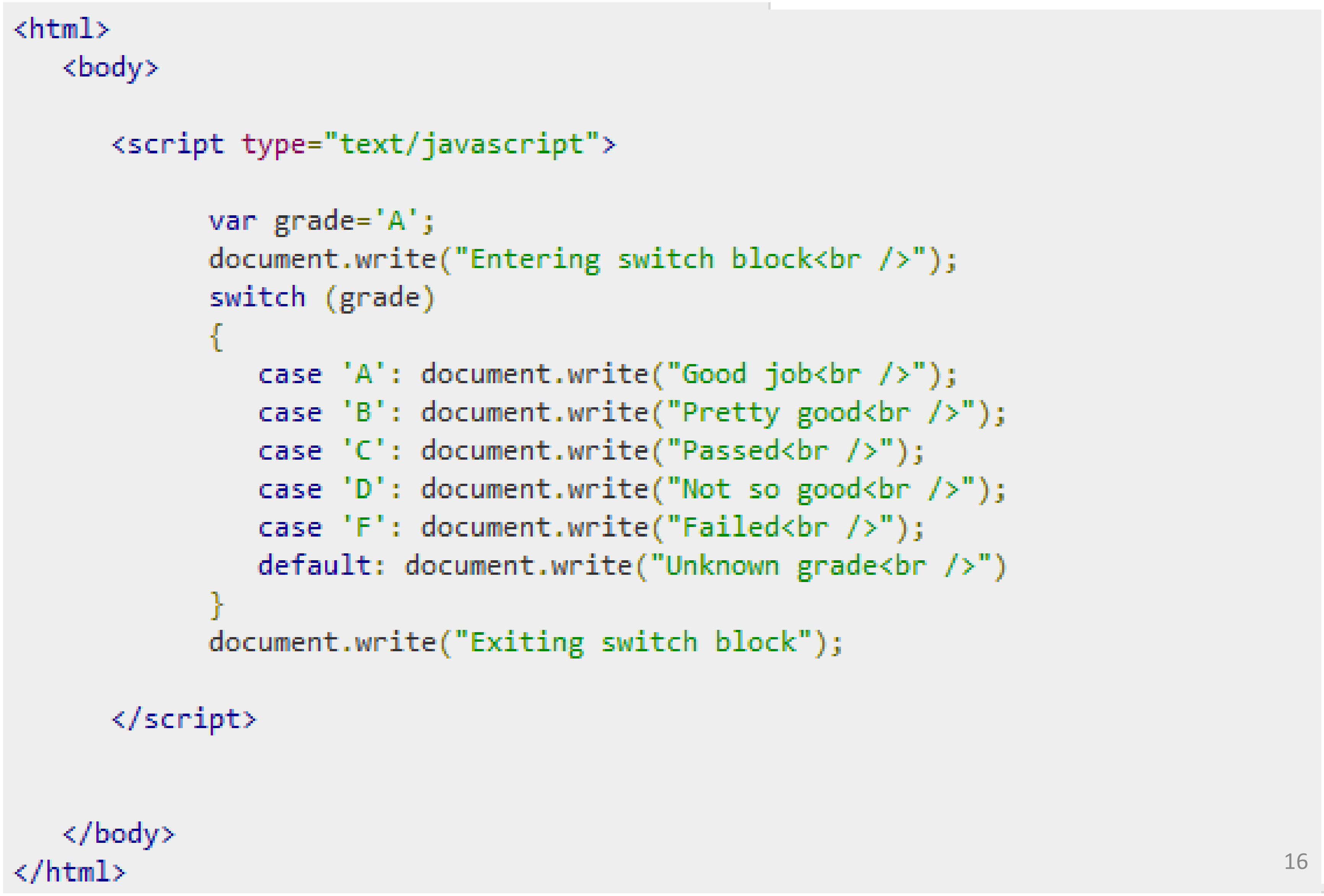
```
<html>
   <body>

      <script type="text/javascript">

            var grade='A';
            document.write("Entering switch block<br />");
            switch (grade)
            {
               case 'A': document.write("Good job<br />");
               break;

               case 'B': document.write("Pretty good<br />");
               break;

               case 'C': document.write("Passed<br />");
               break;

               case 'D': document.write("Not so good<br />");
               break;

               case 'F': document.write("Failed<br />");
               break;

               default:  document.write("Unknown grade<br />")
            }
            document.write("Exiting switch block");

      </script>


   </body>
</html>
```



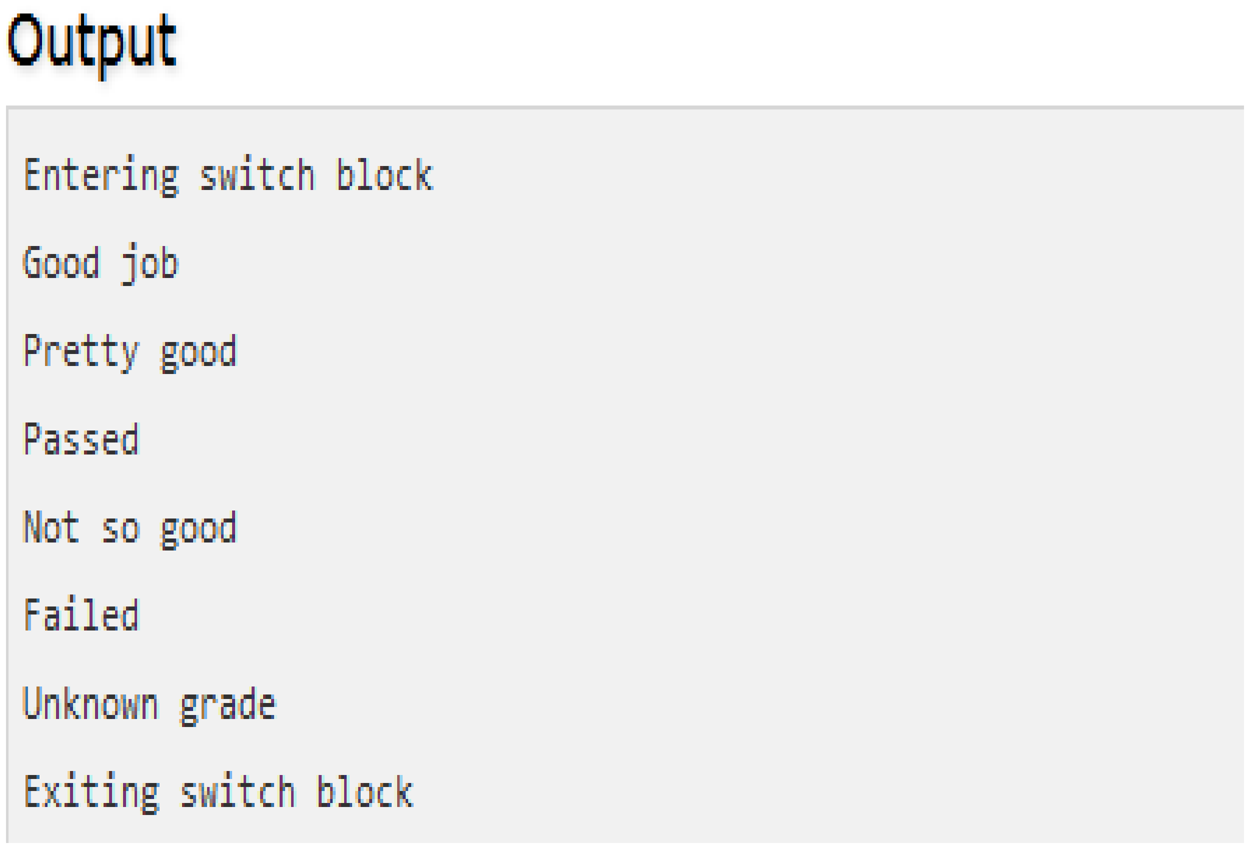
Output

```
Entering switch block

Good job

Exiting switch block
```

When you use switch without “break”



```
<html>
   <body>

      <script type="text/javascript">

            var grade='A';
            document.write("Entering switch block<br />");
            switch (grade)
            {
               case 'A': document.write("Good job<br />");
               case 'B': document.write("Pretty good<br />");
               case 'C': document.write("Passed<br />");
               case 'D': document.write("Not so good<br />");
               case 'F': document.write("Failed<br />");
               default: document.write("Unknown grade<br />")
            }
            document.write("Exiting switch block");

      </script>


   </body>
</html>
```



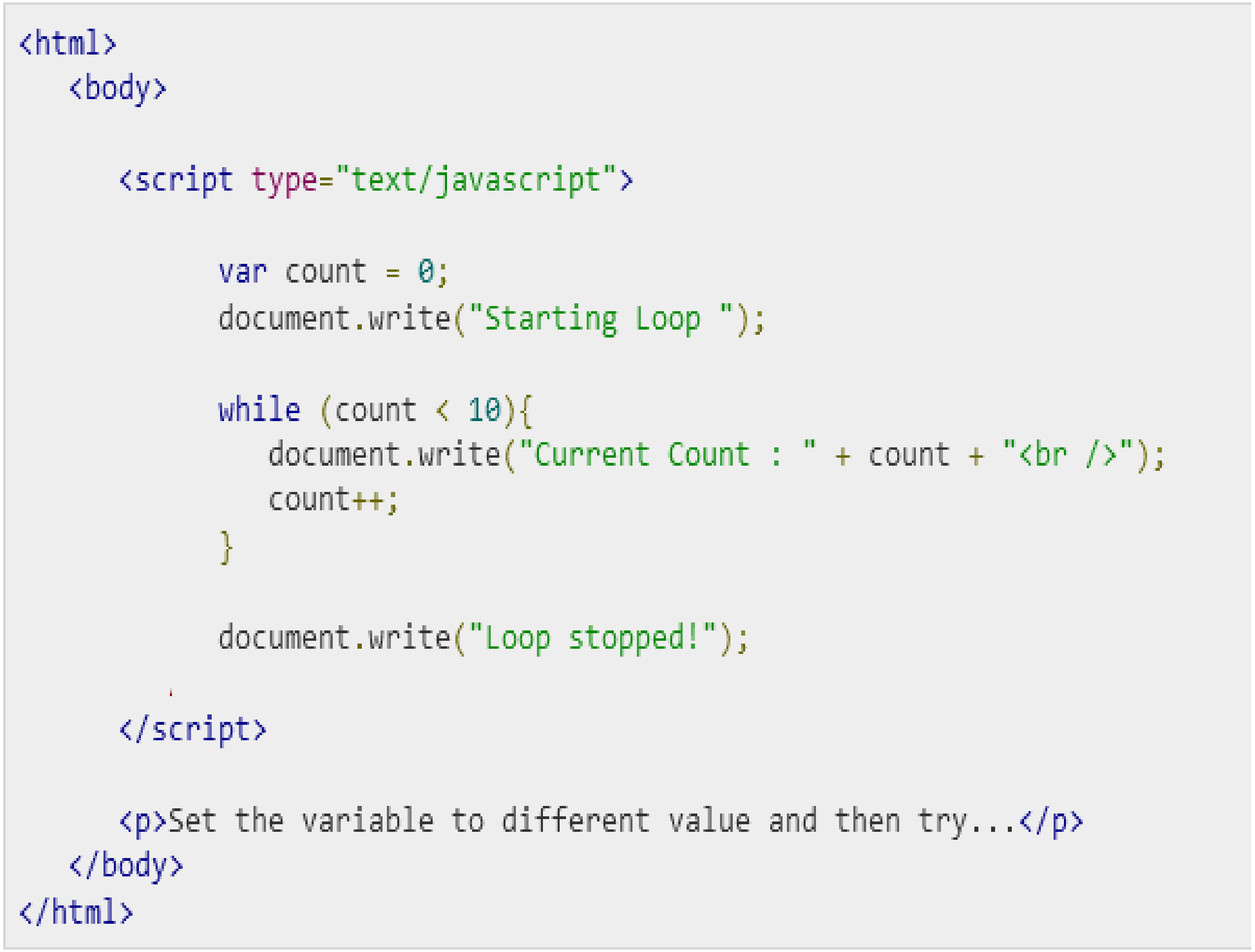
Output

```
Entering switch block
Good job
Pretty good
Passed
Not so good
Failed
Unknown grade
Exiting switch block
```

JavaScript - While Loops

- **Syntax**
- The syntax of **while loop** in JavaScript is as follows –

```
while (expression)
{
 Statement(s) to be executed if expression is
true
 }
```



```
<html>
   <body>

      <script type="text/javascript">

         var count = 0;
         document.write("Starting Loop ");

         while (count < 10){
            document.write("Current Count : " + count + "<br />");
            count++;
         }

         document.write("Loop stopped!");

      </script>

      <p>Set the variable to different value and then try...</p>
   </body>
</html>
```

Output

```
Starting Loop
Current Count : 0
Current Count : 1
Current Count : 2
Current Count : 3
Current Count : 4
Current Count : 5
Current Count : 6
Current Count : 7
Current Count : 8
Current Count : 9
Loop stopped!
Set the variable to different value and then try...
```

Do-while

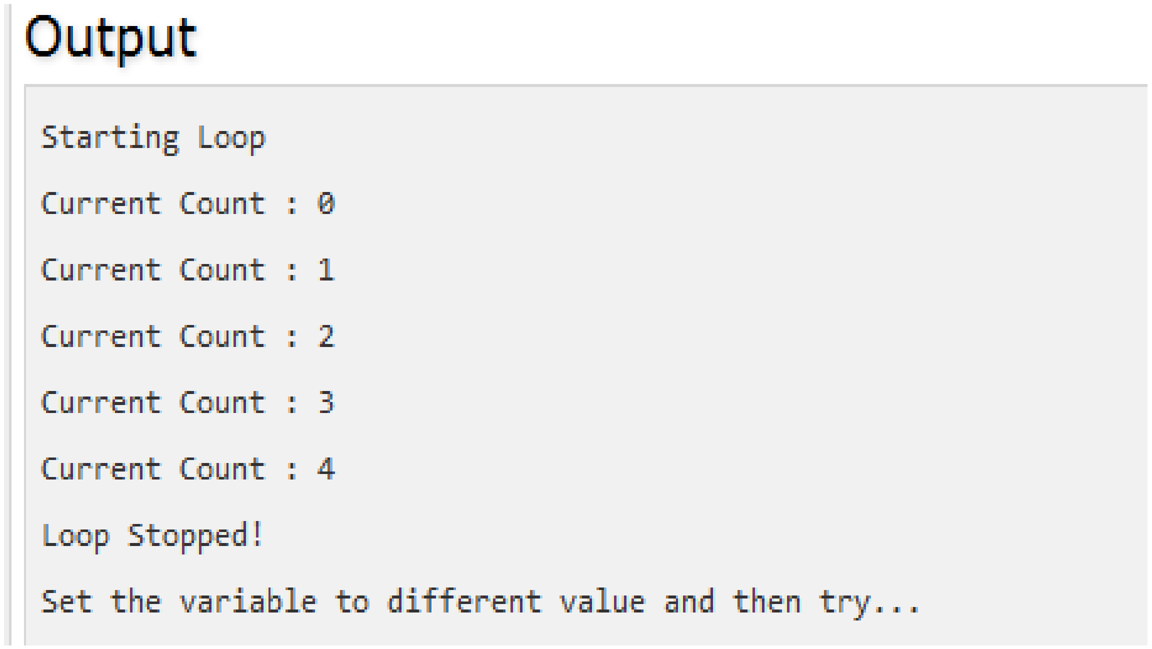
Syntax

The syntax for **do-while** loop in JavaScript is as follows –

```
do
{
 Statement(s) to be executed;
}
while (expression);
```

Example do-while

```
<html>
   <body>

      <script type="text/javascript">

            var count = 0;

            document.write("Starting Loop" + "<br />");
            do{
               document.write("Current Count : " + count + "<br />");
               count++;
            }

            while (count < 5);
            document.write ("Loop stopped!");

      </script>

      <p>Set the variable to different value and then try...</p>
   </body>
</html>
```

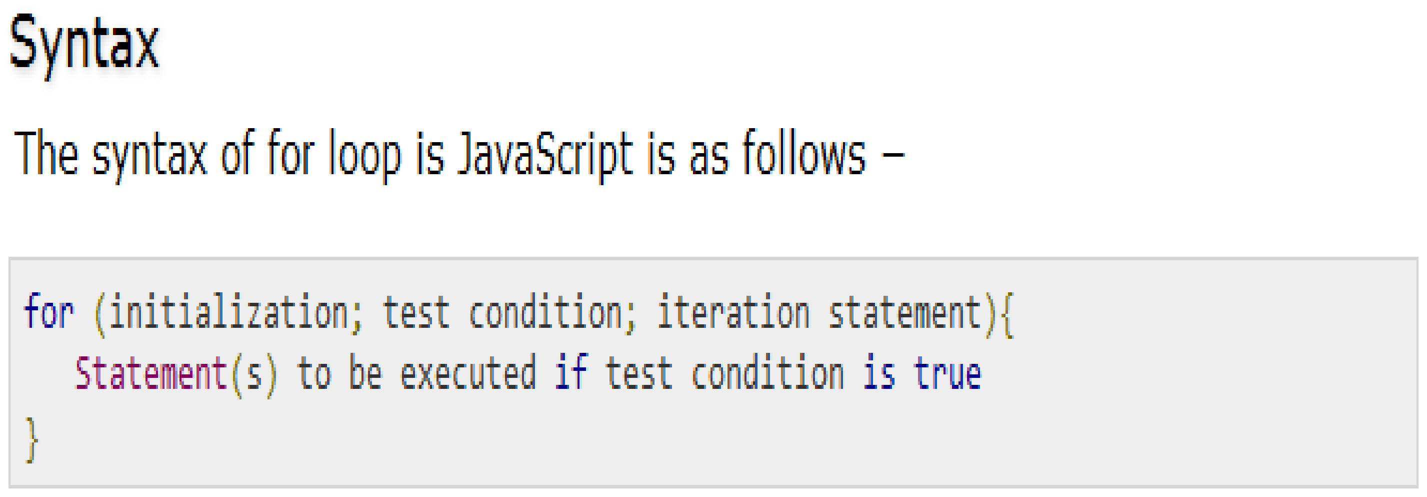


Output

```
Starting Loop

Current Count : 0

Current Count : 1

Current Count : 2

Current Count : 3

Current Count : 4

Loop Stopped!

Set the variable to different value and then try...
```

For Loop

- The '**for**' loop is the most compact form of looping. It includes the following three important parts −
- The **loop initialization** where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.
- The **test statement** which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.
- The **iteration statement** where you can increase or decrease your counter.

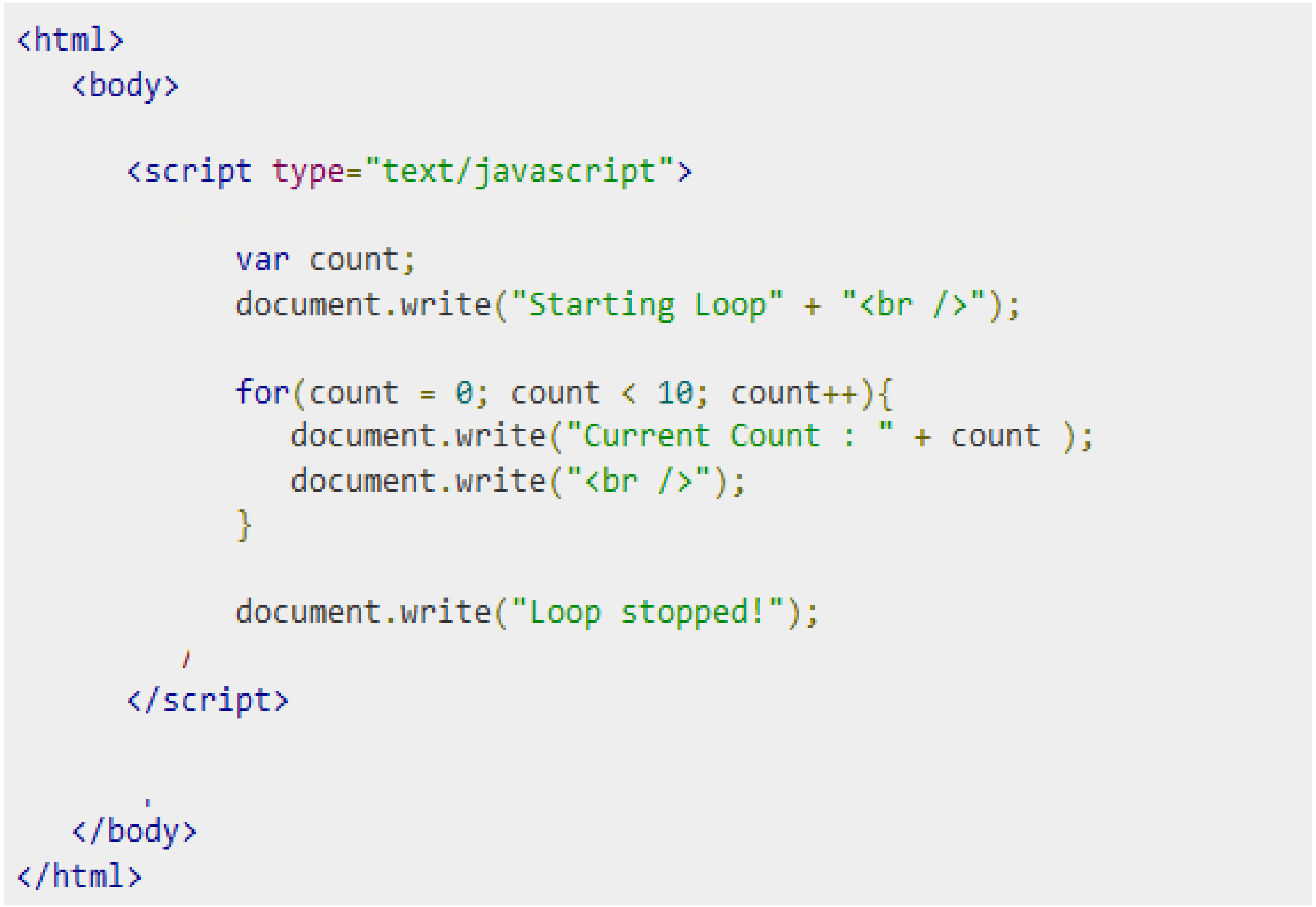


Syntax

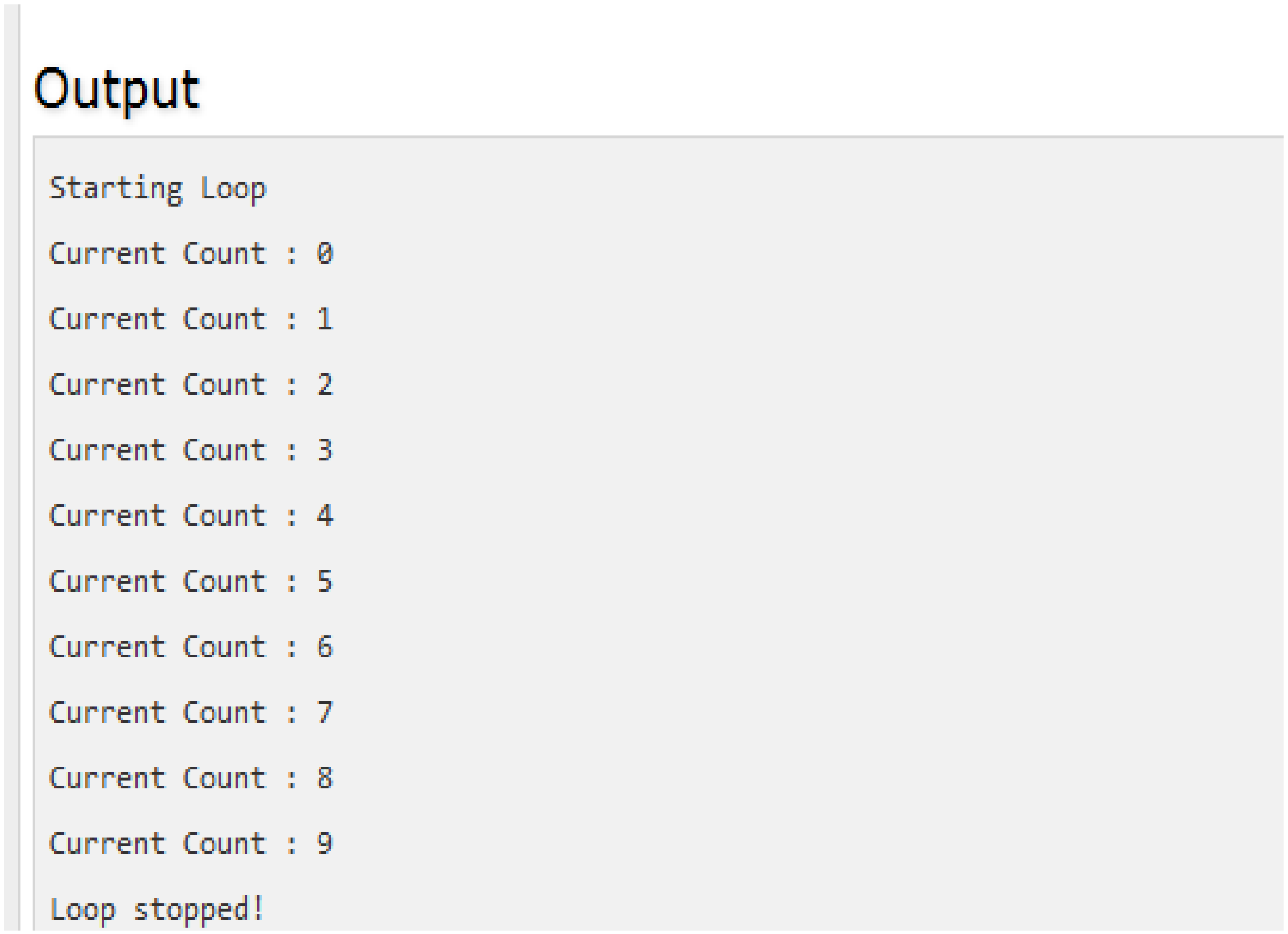
The syntax of for loop is JavaScript is as follows –

```
for (initialization; test condition; iteration statement){
   Statement(s) to be executed if test condition is true
}
```

example



```
<html>
   <body>

      <script type="text/javascript">

            var count;
            document.write("Starting Loop" + "<br />");

            for(count = 0; count < 10; count++){
               document.write("Current Count : " + count );
               document.write("<br />");
            }

            document.write("Loop stopped!");

      </script>


   </body>
</html>
```



Output

```
Starting Loop

Current Count : 0

Current Count : 1

Current Count : 2

Current Count : 3

Current Count : 4

Current Count : 5

Current Count : 6

Current Count : 7

Current Count : 8

Current Count : 9

Loop stopped!
```

JavaScript *for...in* loop

- The **for...in** loop is used to loop through an object's properties.
- As we have not discussed Objects yet, you may not feel comfortable with this loop. But once you understand how objects behave in JavaScript, you will find this loop very useful.

Syntax

```
for (variablename in object)
{ statement or block to execute }
```

Example

Try the following example to implement 'for-in' loop. It prints the web browser's **Navigator** object.

```
<html>
   <body>

      <script type="text/javascript">

            var aProperty;
            document.write("Navigator Object Properties<br /> ");

            for (aProperty in navigator) {
               document.write(aProperty);
               document.write("<br />");
            }
            document.write ("Exiting from the loop!");

      </script>


   </body>
</html>
```

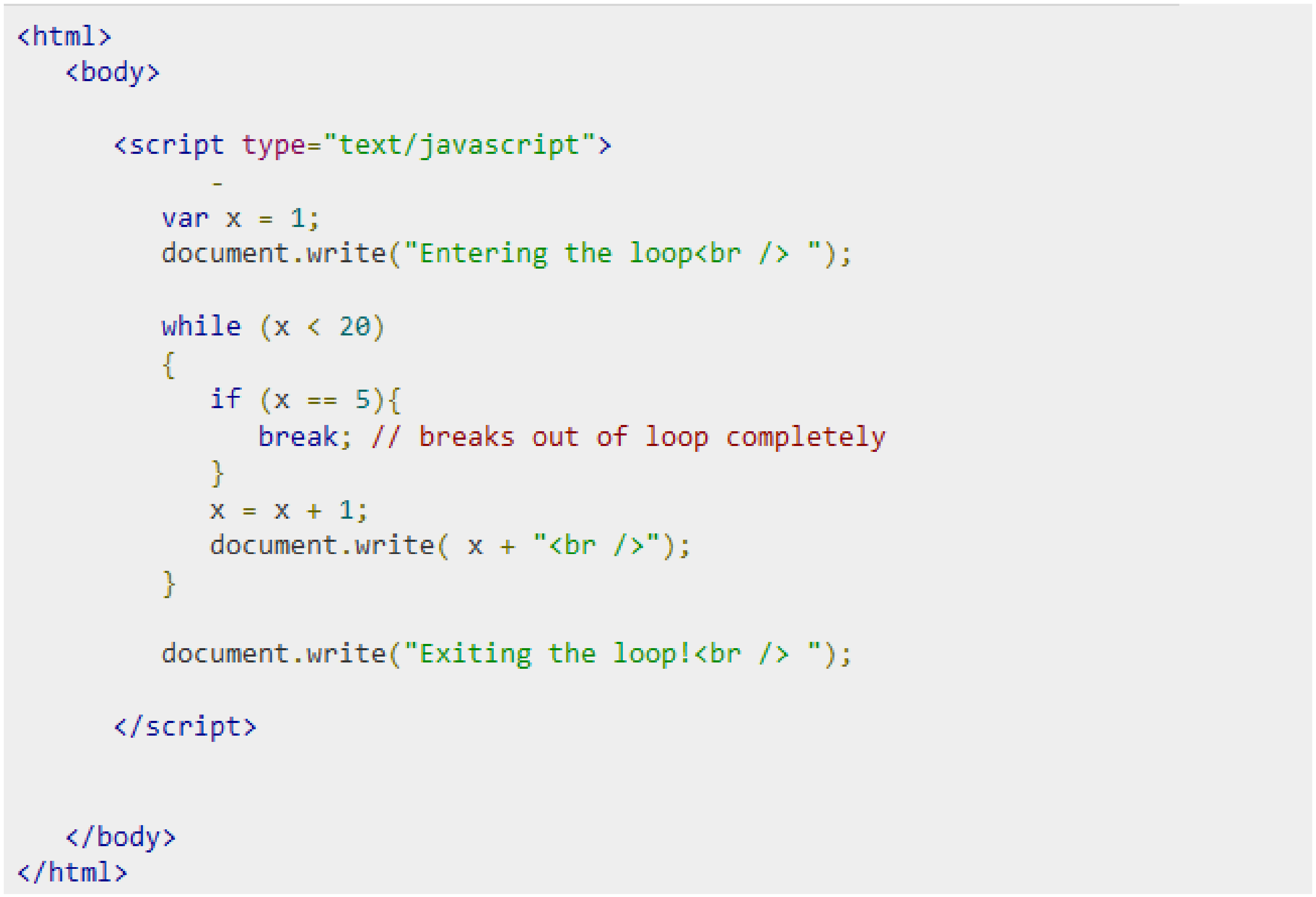
Output

```
Navigator Object Properties
serviceWorker
webkitPersistentStorage
webkitTemporaryStorage
geolocation
doNotTrack
onLine
languages
language
userAgent
product
platform
appVersion
appName
appCodeName
hardwareConcurrency
maxTouchPoints
vendorSub
```

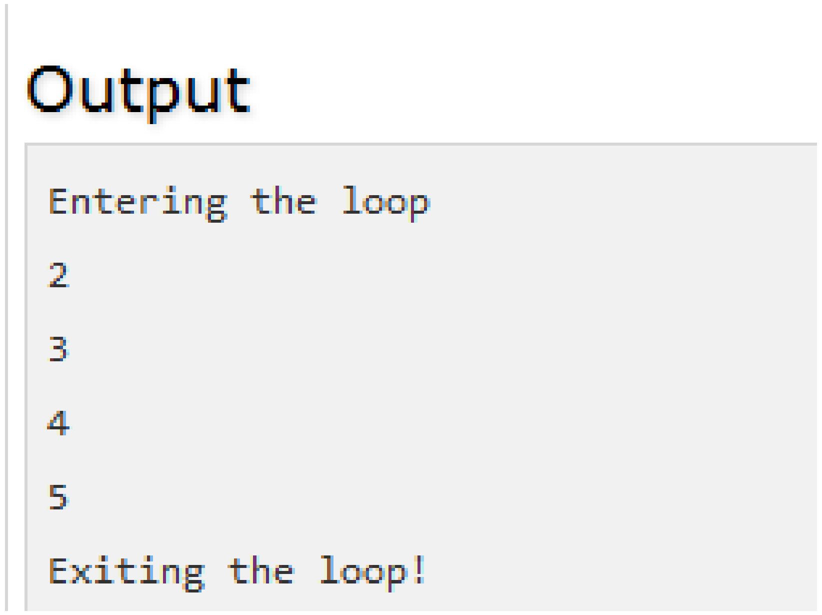
The break Statement

- The **break** statement, which was briefly introduced with the *switch* statement, is used to exit a loop early, breaking out of the enclosing curly braces.

Example of break



```
<html>
   <body>

      <script type="text/javascript">
            -
         var x = 1;
         document.write("Entering the loop<br /> ");

         while (x < 20)
         {
            if (x == 5){
               break; // breaks out of loop completely
            }
            x = x + 1;
            document.write( x + "<br />");
         }

         document.write("Exiting the loop!<br /> ");

      </script>


   </body>
</html>
```



Output

```
Entering the loop

2

3

4

5

Exiting the loop!
```

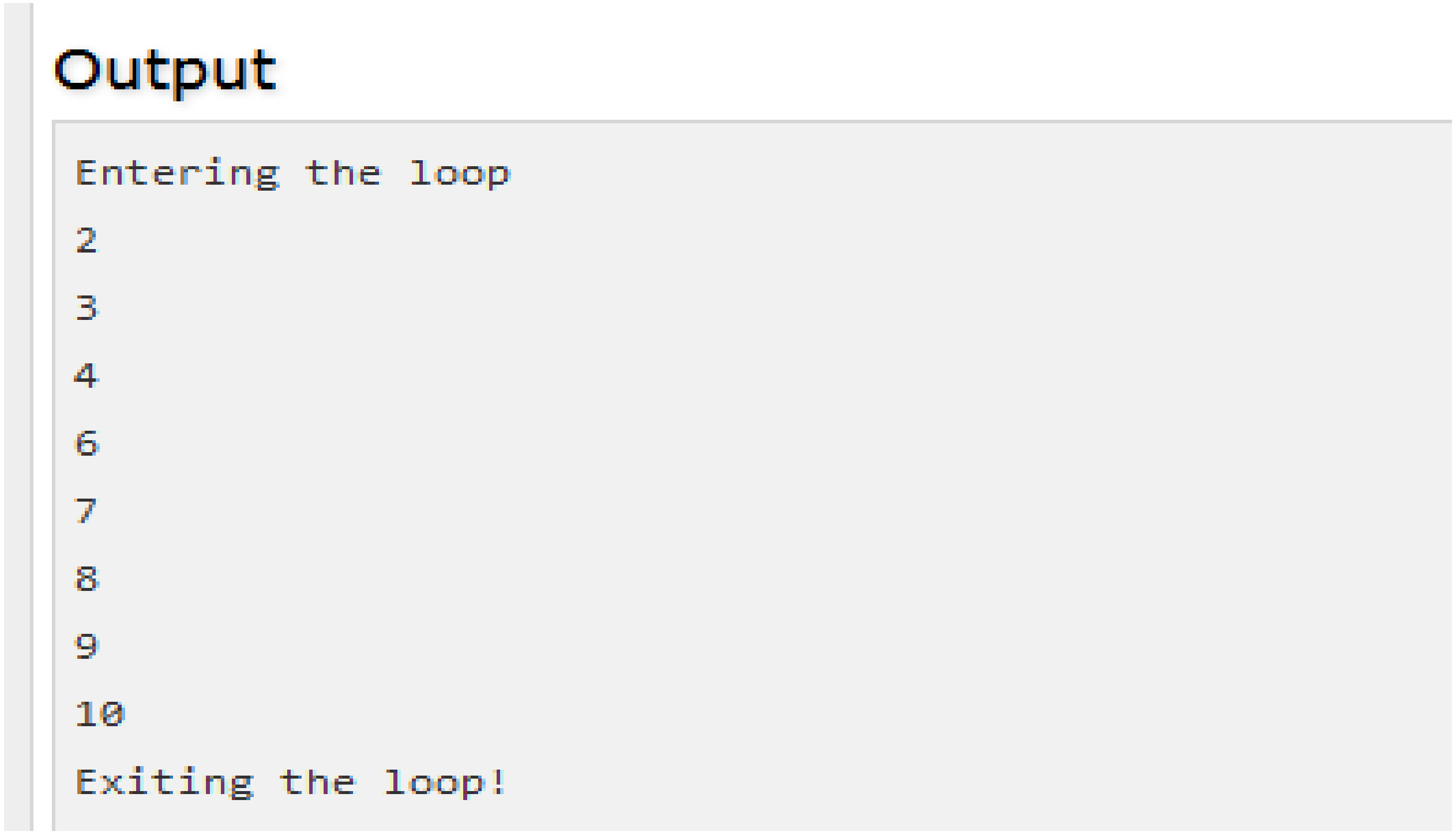
The continue Statement

- The **continue** statement tells the interpreter to immediately start the next iteration of the loop and skip the remaining code block. When a **continue** statement is encountered, the program flow moves to the loop check expression immediately and if the condition remains true, then it starts the next iteration, otherwise the control comes out of the loop.

Example

This example illustrates the use of a **continue** statement with a while loop. Notice how the **continue** statement is used to skip printing when the index held in variable **x** reaches 5 −

```
<html>
   <body>

      <script type="text/javascript">

            var x = 1;
            document.write("Entering the loop<br /> ");

            while (x < 10)
            {
               x = x + 1;

               if (x == 5){
                  continue; // skip rest of the loop body
               }
               document.write( x + "<br />");
            }

            document.write("Exiting the loop!<br /> ");

      </script>


   </body>
</html>
```



Output

```
Entering the loop
2
3
4
6
7
8
9
10
Exiting the loop!
```