

## Motivation for the Course

There are lot of benefits of using JavaScript in Web Technologies. JavaScript is an interpreted Web development language. It need not be compiled so no compiler is required. The browser interprets JavaScript as its HTML tags. The syntax of JavaScript is too easy. Any person can learn it very easily and use it to develop dynamic website. JavaScript is an case-based language. It means that different code segment is executed when certain event occurs. For example, a code segment may execute when the user clicks a button or makes a move in the mouse over an object etc. JavaScript provides all capabilities of a procedural language. It provides condition checking, loops and branching facilities that can be executed in a web page. JavaScript is platform independent language. Any JavaScript code is executed on different types of hardware a JavaScript program written for.

JavaScript can be used to validate the input. JavaScript can used to create different buttons with interesting mouse rollover effects. It makes browsing more interesting and attractive. JavaScript can be used to create popup windows. These windows are normally used to display important announcements, offers and news etc. JavaScript can be used to create dynamic contents in a website. Different HTML tags can be generated based on the user input etc. JavaScript can be used to interest with the user. The input entered by the user can be processed and proper message can be displayed to the user. The interactive capabilities of a website make it more interesting and productive for the users.

## Objective of the Course

The main intention of the workshop is to provide exposure to, and awareness of, JavaScript, making it feasible for beginners to do innovative projects. we hope that your interest in and *passion* for innovation will be ignited by this workshop.

It is suggested that participants form like-minded groups so they can continue working in future on Web development and other related topics and do projects, by studying the literature, brainstorming about problems and best possible solutions.

*After each exercise, we will provide a brief discussion of the potential applications of the concept covered in the exercise, and suggest some sample post-lab exercises for problem solving. We will also provide a few references related to the concept, so users can explore the concept further on their own.*

## **Prerequisites for the course**

- Knowledge in operating a Computer
- Basic concepts in HTML and C programming

## **What can you learn from the book?**

- Become familiar with JavaScript.
- Understand the Basic syntax for all the concepts and execute programs on your own.
- Create your own web page
- Create a game using JavaScript

# Table of Contents

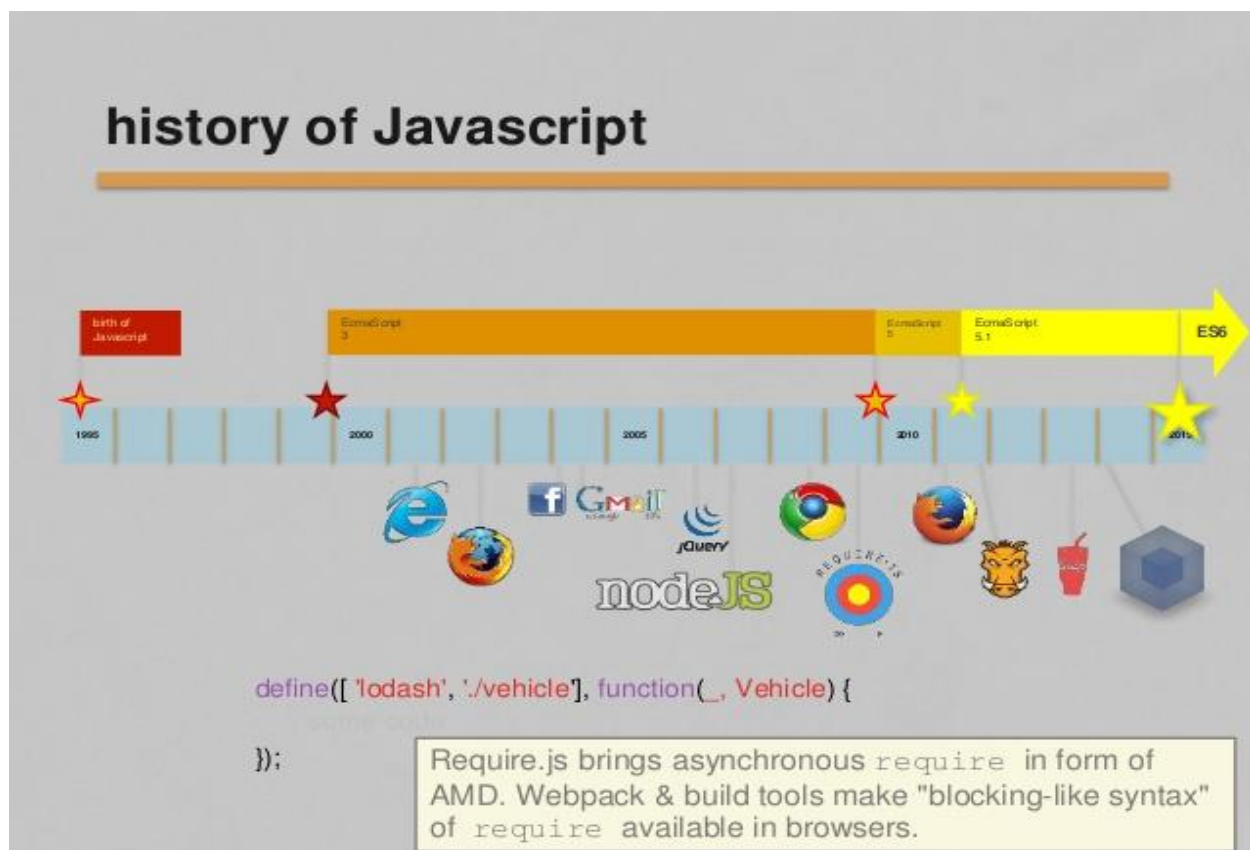
<b>Topic– 1: Introduction to JavaScript.....</b>	<b>6</b>
1.1. Introduction.....	6
1.2. History.....	6
1.3. Development Tools.....	7
<b>Topic – 2: Programming.....</b>	<b>10</b>
2.1 Basic Syntax.....	10
2.2 Hello World.....	10
2.3 White Space.....	11
2.4 Semicolon.....	12
2.5 Case Sensitive.....	12
2.6 External File.....	12
2.7 Expressions.....	13
2.8 Comments.....	14
<b>Topic– 3: Data types.....</b>	<b>16</b>
3.1 Literals.....	16
3.2 Variable.....	16
3.3 Scope of variable.....	17
3.4 Naming Variable.....	17
3.5 Keywords.....	18
<b>Topic – 4: Operators.....</b>	<b>21</b>
4.1 Arithmetic.....	21
4.2 Comparison.....	25
4.3 Logical.....	27
4.4 Bitwise.....	28
4.5 Assignment.....	30
4.6 Miscellaneous.....	32
<b>Topic–5: Conditional Statements.....</b>	<b>37</b>
5.1 if.....	38
5.2 if...else.....	40
5.3 if...else if.....	42
5.4 Switch case.....	44
<b>Topic–6: Looping Statements.....</b>	<b>49</b>
6.1 For.....	49
6.2 While.....	52
6.3 Do While.....	54
6.4 Function.....	56
<b>Topic–7: Creating Webpage.....</b>	<b>59</b>
<b>Topic–8: Games.....</b>	<b>63</b>

# 1. INTRODUCTION TO JAVASCRIPT

## 1.1 INTRODUCTION:

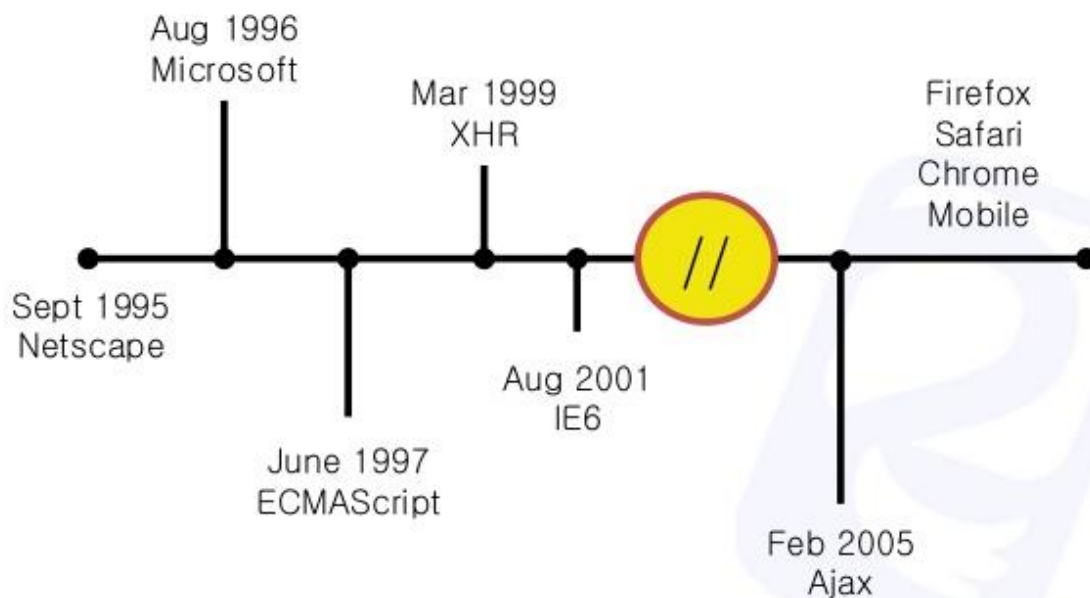
JavaScript is a web development programming language. It is one of the web development programming languages. The others are HTML and CSS. These three languages are combined to build the web applications and web pages. Without JavaScript, if the web page is designed only with HTML and CSS, the web page will be like a poster in the wall, it can be colorful, attractive but at the end of the day, the user can only read the page, he/she cannot perform other dynamic functions. Initially, it only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases.

## 1.2 HISTORY:



Netscape Communications recruited Brenden Eich in 1995, with the aim of embedding Scheme programming language into Netscape Navigator. Brenden Eich created the language JavaScript in May 1995. Then it was developed under the name of 'MOCHA'. Later it was named as 'LIVESCRIPT' in its beta version. In order to compete with Microsoft whose programming language 'JAVA' was highly popular, and since the language live script has similar syntax to java, the language live script was renamed as 'JAVASCRIPT'.

## History of Javascript



36

[http://channel9.msdn.com/Events/Build/2012/4-000?utm\\_source=javascriptweekly](http://channel9.msdn.com/Events/Build/2012/4-000?utm_source=javascriptweekly)

### 1.3 DEVELOPMENT TOOLS:

JavaScript does not require any expensive development tools. We can even start with a simple text editor such as Notepad. Some of the development tools are as follows:

- Microsoft Frontpage
- Macromedia Dreamweaver MX
- Macromedia Homesite 5
- Sublime Text
- Gulp
- NPM

- Webpack
- Browserify
- Eslint
- Jasmin



## QUESTIONS



1. Who developed the JavaScript? In what name it was developed first?
2. What is JavaScript?
3. How is Java different from JavaScript?
4. Which is the recently used platform for Web Development?
5. What was JavaScript called in its beta version of development?

## References

1. <https://en.wikipedia.org/wiki/JavaScript>
2. <https://www.geeksforgeeks.org/javascript-tutorial/>
3. [https://www.tutorialspoint.com/javascript/javascript\\_overview.htm](https://www.tutorialspoint.com/javascript/javascript_overview.htm)
4. Obama writes his first line of code in JavaScript-[https://www.huffingtonpost.in/2014/12/09/obama-code\\_n\\_6294036.html](https://www.huffingtonpost.in/2014/12/09/obama-code_n_6294036.html)

## 2. PROGRAMMING

### 2.1 BASIC SYNTAX:

In JavaScript, the programming is represented by the word 'script'. In the html programming language, the JavaScript is placed between the script tags such as

```
<html>

<body>

<script>

.....

//JS CODE

.....

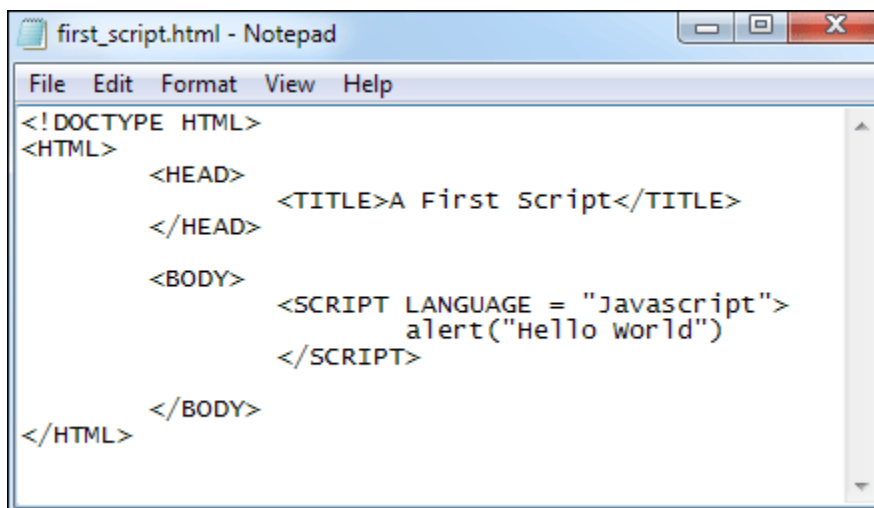
</script>

</body>

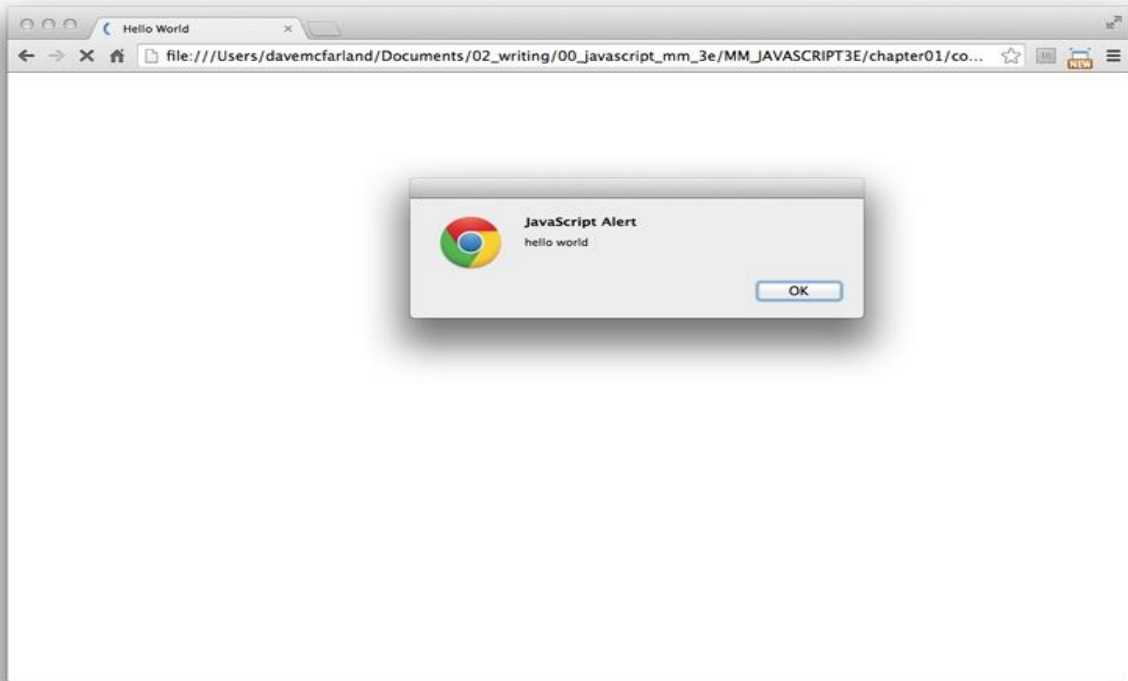
</html>
```

The JavaScript can also be used in head section of the HTML in between script tags.

Let us write the Hello world program in JavaScript.



When we save this file with extension as '.html' and open the file, the output will be displayed in web browser as shown below.



You just did your first JavaScript program.

## 2.2 WHITE SPACE:

The White spaces and Line breaks are not considered by the decoder/compiler. It only takes the code we give in. So, we can freely use the white spaces and line breaks wherever we want, in order to indent the code and make it easy to read.

## 2.3 SEMICOLONS:

In JavaScript, the semicolons are used just as in C, C++, Java, etc. However, these semicolons can be omitted if we write all our statements in a new line such as

```
<script>  
var p = 52  
var q = 4  
</script>
```



Semicolons must be used when the code is in line as follows:

```
<script>  
Var p = 52; var q = 4;  
</script>
```

It is a good practice to use semicolons always in the code.

## 2.4 CASE SENSITIVITY:

JavaScript is a Case Sensitive language. Case Sensitivity means the programming language should be written with a predefined way of consistent capitalization for variables, keywords, etc.

## 2.5 EXTERNAL FILE:

When we work in JavaScript, the more advanced it goes, we need to include classes and other necessary functions. We can also include the JavaScript from an external file in local storage to HTML code such as

```
<html>  
<head>  
<title>External File</title>  
</head>
```

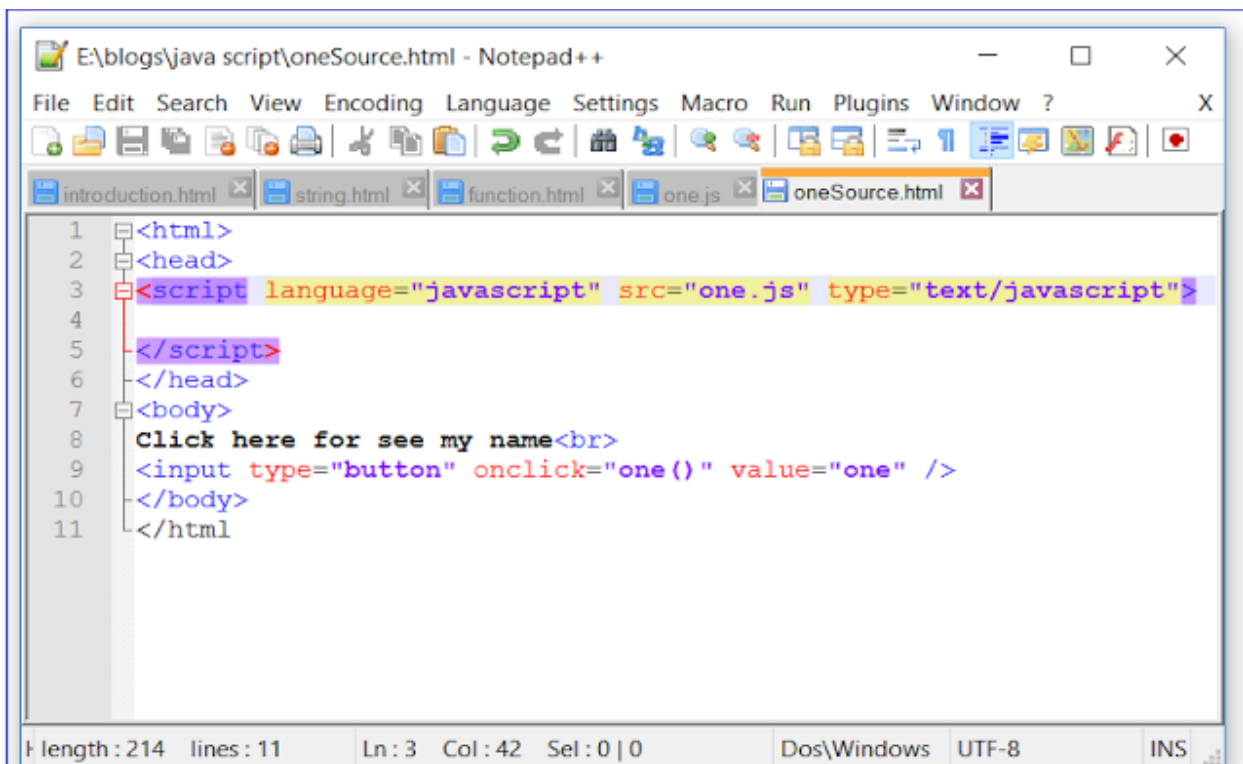
<body>

<script type = "text/javascript" src = "filename.js" ></script>

</body>

</html>

To use the external file in HTML, the JavaScript must be written in file and it should be saved with the extension **‘.js’**. The file name should be used to open the file as mentioned above. For example:



```
1 <html>
2 <head>
3 <script language="javascript" src="one.js" type="text/javascript">
4
5 </script>
6 </head>
7 <body>
8 Click here for see my name<br>
9 <input type="button" onclick="one()" value="one" />
10 </body>
11 </html>
```

See the highlighted section of code in the above program. Here the file name for the external JavaScript file is ‘one’ and it is saved with the extension ‘.js’.

## 2.6 EXPRESSIONS:

An expression is a term which combines variables, values and operators, which computes to a value. The computation is called an evaluation.

For example, 4 \* 5 evaluates to 20:

4 \* 5

Expressions can also have variable values:

```
x * 10
```

The values can be of various types, such as numbers and strings.

For example, "Tharun" + " " + "Chelladurai", evaluates to "Tharun Chelladurai"

## 2.7 COMMENTS:

Not all JavaScript statements are "executed". Code after double slashes `//` or between `/*` and `*/` is treated as a comment. Comments are ignored, and will not be executed

```
var x= 5; //I will be executed
```

```
// var x = 6; I will NOT be executed
```

## **POST-LAB EXERCISE**

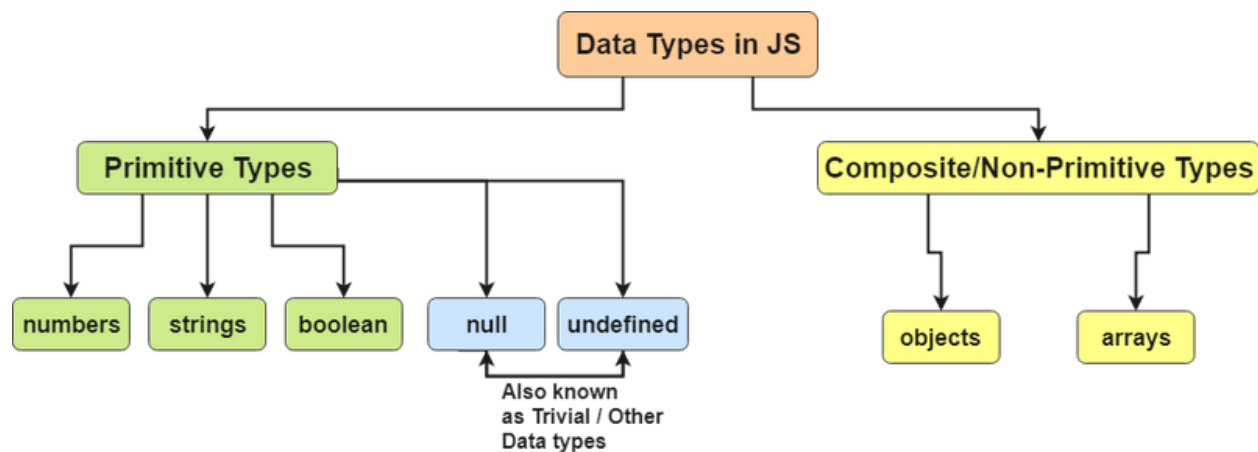
1. Write a JavaScript program to alert your Name?
2. Write an expression in JavaScript, such that final result is your age?
3. Add an external file into the HTML language, which contains the JavaScript code to run the print your friends name.
4. What are the comments? Why are they used?

## **References**

1. Hello world Program-[https://www.homeandlearn.co.uk/javascript/first\\_script.html](https://www.homeandlearn.co.uk/javascript/first_script.html)
2. [https://www.w3schools.com/js/js\\_where.asp](https://www.w3schools.com/js/js_where.asp)
3. Hello World Program Output-<https://www.oreilly.com/library/view/javascript-jquery/9781491948583/ch01.html>
4. External file-<https://medium.com/@chamzz/use-external-javascript-file-d2667f857fe8>

### 3. DATA TYPES

Datatypes are the types of values that can be symbolized and controlled in a programming language. JavaScript consists of primitive data types and composite/reference datatypes. They are



Fixed values are called literals. Variable values are called variables.

#### 3.1 LITERALS:

The most important rules for writing fixed values are:

Numbers are written with or without decimals:

124

309.42

Strings are text, written within double or single quotes:

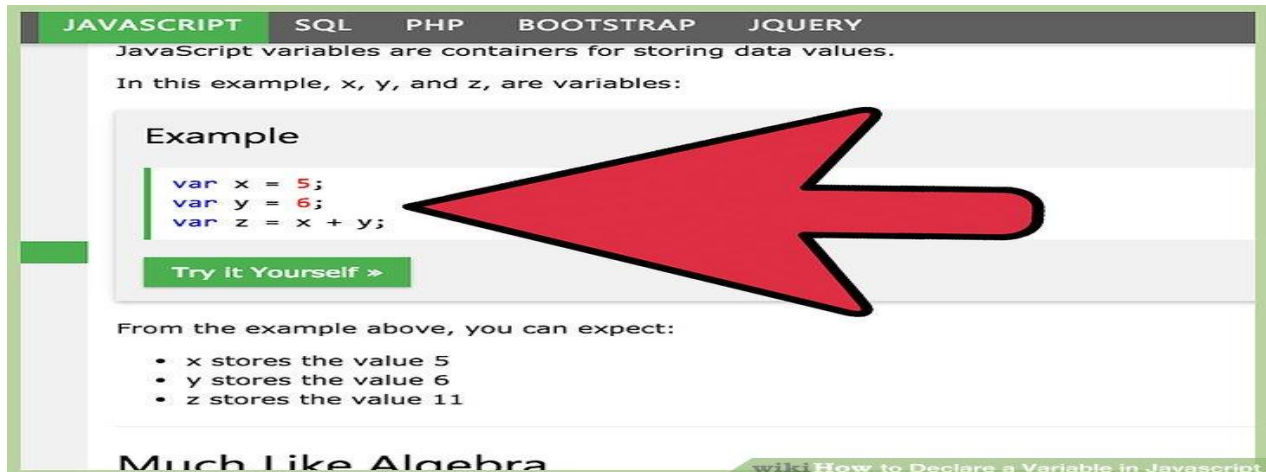
"Taj Mahal"

'Taj Mahal'

#### 3.2 VARIABLES:

In a programming language, variables are used to store data values. JavaScript uses the **'var'** keyword to declare variables. An equal to sign is used to assign values to variables.





**3.3 SCOPE OF VARIABLE:** The scope of a variable is the boundary for the working area for the variable in a program. There are two scopes in JavaScript. They are as follows:

**LOCAL** - The variable works within the function, where they are declared. They will work only when the function is called. They will terminate, when the function loop is ended or when the function is ended. So, we can say that the lifetime for the variable is the lifetime of the function.

**GLOBAL** - Here, the variables will work till the whole program is completed. The variable is accessible by all parts of the programs like classes, functions, etc. The lifetime of the variable is the lifetime of the program.

```
<script type="text/javascript">
var myVar = "global"; // Declare a global variable
function checkscope()
{
var myVar = "local"; // Declare a local variable
alert(myVar);
}
</script>
```

**VARIABLE**

The above code gives the output as follows:

local

### 3.4 NAMING VARIABLES:

There are certain rules in naming a variable. They are as follows:

- The variable names must not be the reserved keywords. The keywords are the ones which have specific function which is already predefined in the programming language. When we use the keywords as variable names, the compiler gets confused that whether the user is calling the value in the variable or the keyword and throws an error.
- The variables must not begin with a number. They can begin with a character or an underscore. For example, 365sum is invalid, while sum and \_356sum are valid.
- We know that JavaScript is case sensitive, therefore the variables such as 'Raj' and 'raj' are different and they refer to two different variables.

### 3.5 RESERVED WORDS / KEYWORDS:

The keywords in JavaScript language are as follows:

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try

const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

Each of the above words have their own functions.

## POST-LAB EXERCISE

1. Write a JavaScript code to declare a variable to store a 5digit number.
2. Write a JavaScript code to declare a variable of Local and Global Scope and write your observations.
3. Write a JavaScript using the keyword 'abstract'
4. Write the JavaScript code using the keyword 'static'.
5. What is the purpose the keyword 'short'?

## References

1. [https://www.tutorialspoint.com/javascript/javascript\\_variables.htm](https://www.tutorialspoint.com/javascript/javascript_variables.htm)
2. [https://www.w3schools.com/js/js\\_syntax.asp](https://www.w3schools.com/js/js_syntax.asp)
3. Datatypes-<https://simplesnippets.tech/javascript-variables-data-types/>
4. Variable-<https://www.wikihow.com/Declare-a-Variable-in-Javascript>

## 4. OPERATORS

JavaScript uses operators to compute values and perform mathematical/logical operations. The various operators in JavaScript are:

- Arithmetic operators
- Comparison operators
- Logical operators
- Bitwise operators
- Assignment operators
- Miscellaneous operators

### 4.1 ARITHMETIC OPERATORS:

The various arithmetic operators and their functions are given below.

#### ADDITION:

The addition operator **‘+’**, is used to add two operands.

For example:

```
var a = 2; var b = 3;
```

```
var c=a+b;
```

(or)

```
2+3;
```

The addition operator also works for strings.

For example:

```
var a = 'Vaidhya'; var b = 'prakash';
```

```
var c = a+b;
```

The value in variable c is “Vaidhyaprakash”.

#### SUBTRACTION:

The subtraction operator **‘-’**, is used to subtract two operands.

For example:

```
var a = 9; var b =6;
```

```
var c = a-b;
```

(or)

```
var c=9-6
```

#### MULTIPLICATION:

The multiplication operator **'\*'**, is used to multiply two operands.

For example:

```
var a=3; var b=3;
```

```
alert(a*b);
```

The above code will give a popup message as **'9'**.

### **DIVISION:**

The division operator **'/'**, divides the two operands and will give the quotient as the output.

For example:

```
var a=3; var b=3;
```

```
var c=a/b;
```

The value **'1'** will be stored in the variable c.

### **MODULUS:**

The modulus operator **'%'**, divides two operands and will give the remainder of the division as output.

For example:

```
var a = 3; var b=3;
```

```
var c=a/b;
```

The value **'0'** will be stored in the variable c.

### **INCREMENT:**

The increment operator **'++'**, is used to increment the value in a variable by 1. There are two types of increment operators.

#### **POST-INCREMENT:**

For example:

```
var p = 10;
```

```
document.write(p++);
```

The output will be 10. Here the value 10 is assigned to p at this step and it is incremented by 1 at the next step.

#### **PRE-INCREMENT:**

For example:

```
var p = 10;  
document.write(++p);
```

The output will be 11. Here the value is incremented and assigned to p at this single step.

### **DECREMENT:**

The decrement operator is **--**. It is used to decrement the value by 1. Here also there are two types. They are:

#### **PRE-DECREMENT:**

For example:

```
var q = 2;  
document.write(--p);
```

The output of the above code is '1'. The value 2 is decremented by 1 and assigned to the variable q.

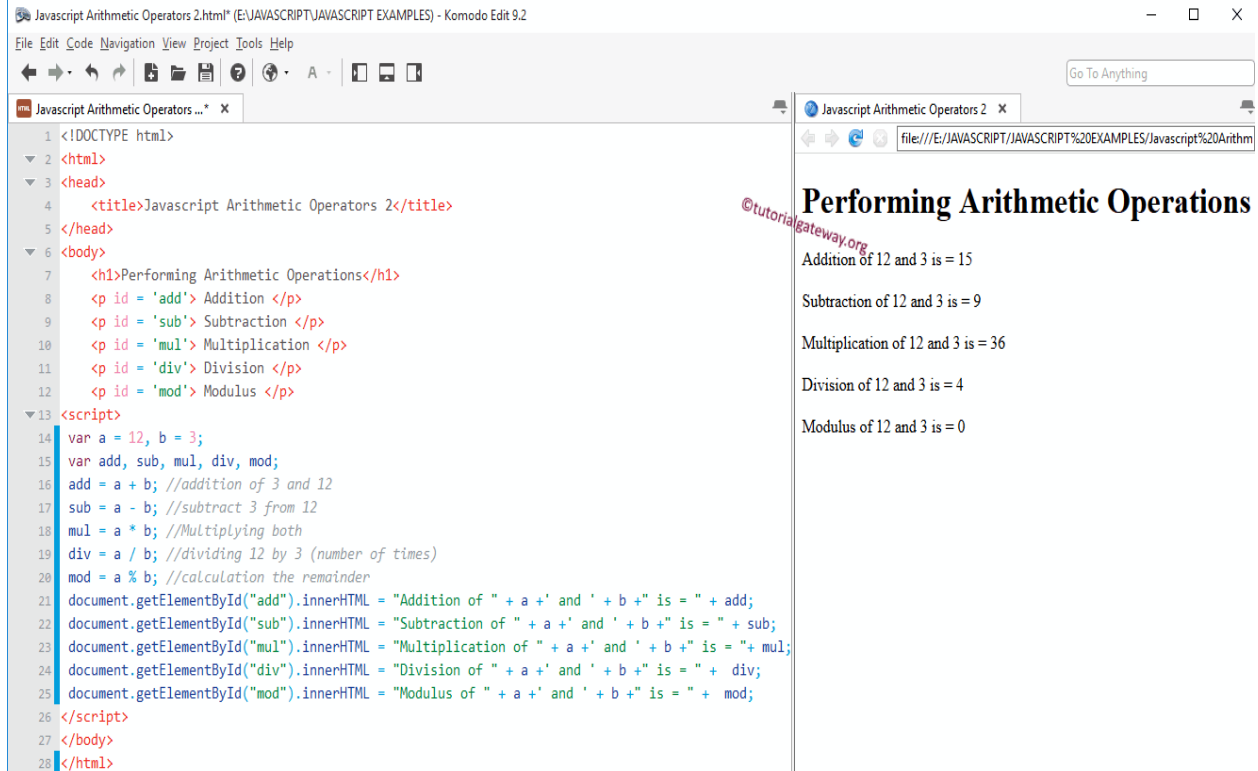
#### **POST-DECEMENT:**

For example:

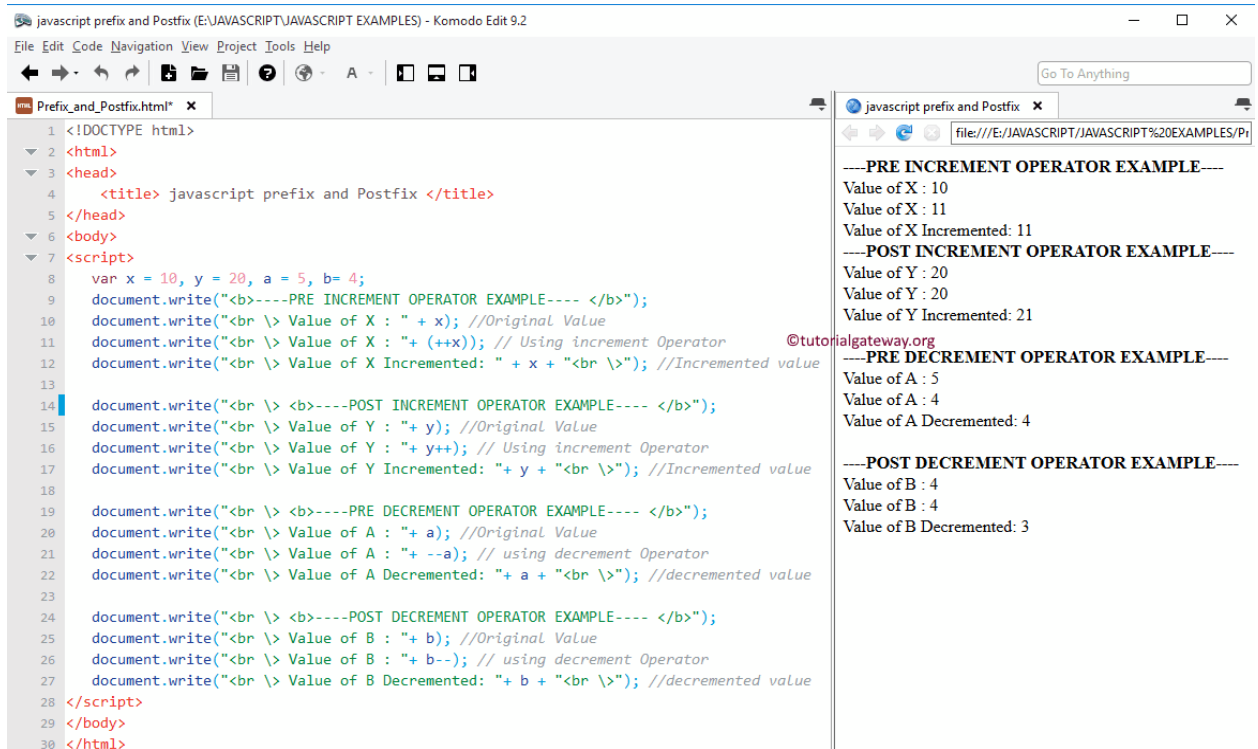
```
var q = 2;  
document.write(p--);
```

The output for the above code is '2'. The value 2 is assigned to the variable q in this step and then it is decremented by 1 in the next step.

The following image shows the code for the arithmetic operators, excluding increment and decrement operators.



The following image clearly shows the code and output for the increment and decrement operators.





## 4.2 COMPARISON OPERATORS:

The comparison operators are used to compare the values of the two operands.

### **EQUAL:**

**'=='** is the equal comparison operator. It is used to check whether two operands have the same value or not. If they have same values then it will return as 'true' otherwise it will return as 'false'.

For example: a=2; b=5;

(a==b);

### **OUTPUT:**

false

### **NOT EQUAL:**

**'!='** is the not equal operator. It is used to make sure that whether the two operands have the different values or not. If the operands have the same values then it will return 'false' otherwise it will return as 'true'.

For example: a=2; b=5;

(a!=b);

### **OUTPUT:**

true

### **GREATER THAN:**

**'>'** is the greater than operator.

Consider a>b;

It will check whether the value in operand 'a' is greater than 'b' or not. If 'a' is greater it will return 'true', otherwise it returns as 'false'.

For example: a=5; b=3;

a>b

### **OUTPUT:**

true

### **LESSER THAN:**

**'<'** is the lesser than operator.

Consider a<b;

It will check whether the value in operand 'a' is lesser than 'b' or not. If 'a' is lesser it will return 'true', otherwise it returns as 'false'.

For example: a=9; b=99;  
a<b;

**OUTPUT:**

true

**GREATER THAN OR EQUAL:**

**'>='** is the greater than or equal operator.

Consider a>=b;

It will check whether the value in operand 'a' is greater than 'b' or equal to 'b'. If it satisfies any one of the two conditions it will return 'true', otherwise it returns as 'false'.

For example: a=2; b=2;  
a>=b;

**OUTPUT:**

true

Even if the value of 'a' is greater than 2, the output will remain unchanged.

**LESSER THAN OR EQUAL:**

**'<='** is the lesser than or equal operator.

Consider a<=b;

It will check whether the value in operand 'a' is lesser than 'b' or equal to 'b'. If it satisfies any one of the two conditions it will return 'true', otherwise it returns as 'false'.

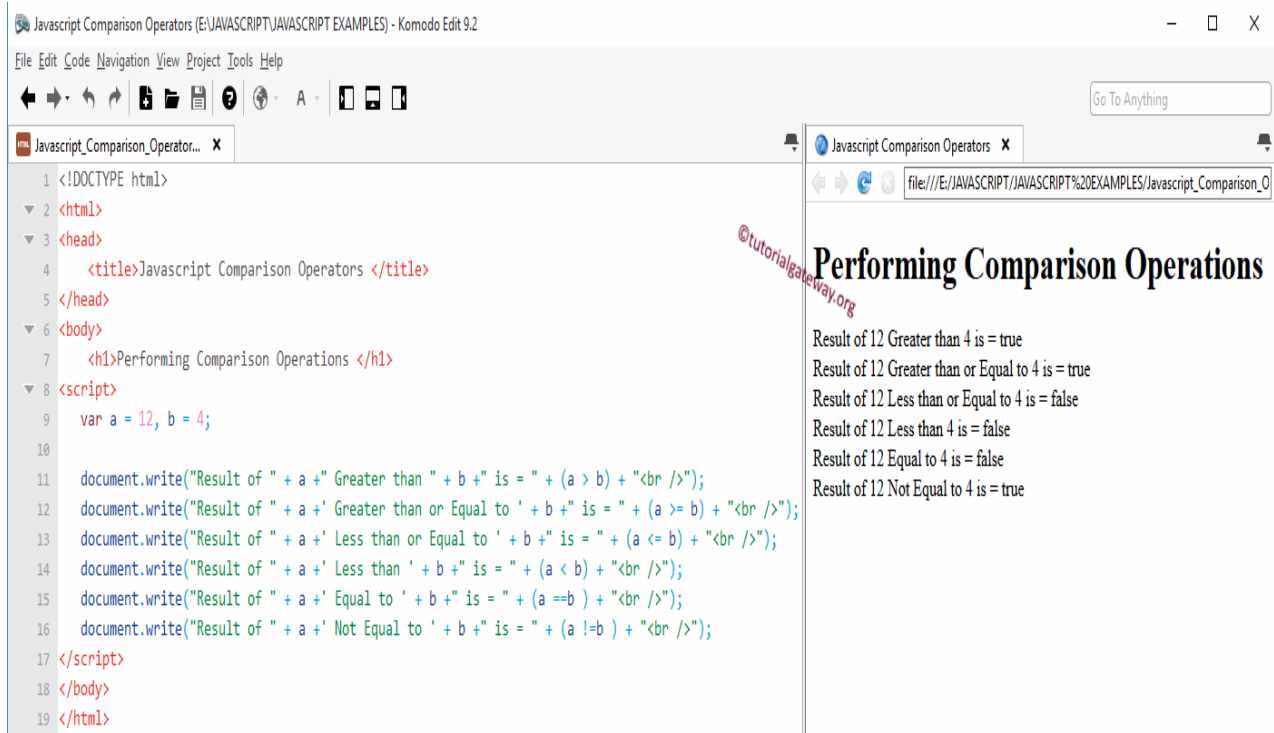
For example: a=7; b=54;  
a<=b;

**OUTPUT:**

true

Even if the value is equal to the value in b or lesser, the output will remain unchanged.

The following image shows the code and the output for all the comparison operators.



### 4.3 LOGICAL OPERATORS:

JavaScript contains logical operators. As the name suggests, they are logical and return the output as 'true' or 'false'. They are also called as Relational operators. They are:

#### LOGICAL AND:

'&&' is the logical AND operator. It is a binary operator. It will return 'true' if both the operands have some value other than zero.

For example: a=1; b=5;

a&&b;

#### OUTPUT:

true

#### LOGICAL OR:

'||' is the logical OR operator. It is also a binary operator. It will return 'true' if any one of the two operands have some value other than zero.

For example: a=0; b=1;

a||b;

**OUTPUT:**

true

**LOGICAL NOT:**

**‘!’** is the logical not operator. It is a unary operator. It will return ‘true’ if the operand has the value ‘false’. It will return ‘false’ if the operand has the value ‘true’.

For example: a=1

!a;

**OUTPUT:**

0

**4.4 BITWISE OPERATOR:**

The bitwise operator in the JavaScript performs operations in each of the bits of the operand value i.e. the operand value is converted into binary number and then the operation is performed in the binary bits and the resultant binary value is converted into the decimal value and is given as output.

**BITWISE AND:**

**‘&’** is the bitwise and operator. Here the operand value is converted into the binary number, then the addition operation is performed in the two binary numbers on every bit and then the resultant binary value is converted into the decimal value which is the output.

For example: a=2; b=3;

a&b;

**OUTPUT:**

2

**BITWISE OR:**

**‘|’** is the bitwise OR operator. The operand value is converted into the binary value and OR operation is performed between the two operands and the resultant binary value is converted into the decimal value which is the output.

For example: a=2; b=3;

a|b;

**OUTPUT:**

**BITWISE NOT:**

'~' is the bitwise not operator. It is a unary operator. It accepts a decimal number as a input, it converts the decimal number into the binary number and then does logical Not operation on every bits of the binary number and then converts the resultant binary number into the decimal number which is the output.

For example: b=3;

~b;

**OUTPUT:**

-4

**BITWISE XOR:**

'^' is the bitwise XOR operator. It is a binary operator. It converts the decimal number into the binary number and then XOR operation is performed in the two binary numbers bit by bit and the resultant is converted into the decimal number which is the output.

XOR operation is the logical operation which gives true (1) if one of the values is true and the way it differs from the OR operator is that, both of the values should not be true. If both the inputs have true values, then the output is false.

For example: a=2; b=3;

a^b;

**OUTPUT:**

1

**LEFT SHIFT:**

'<<' is the left shift operator. It moves all the bits in its first operand to the left by the number of spaces specified by the second operand and replaces the empty spaces with zeroes.

For example: a=2;

a<<1;

**OUTPUT:**

4

## RIGHT SHIFT:

'>>' is the right shift operator. The left operand's value is moved right by the number of bits specified by the right operand.

For example: a=2;

a>>1;

## OUTPUT:

1

## RIGHT SHIFT WITH ZERO:

'>>>' is the right shift with zero. This operator is just like the >> operator, except that the bits shifted in on the left are always zero.

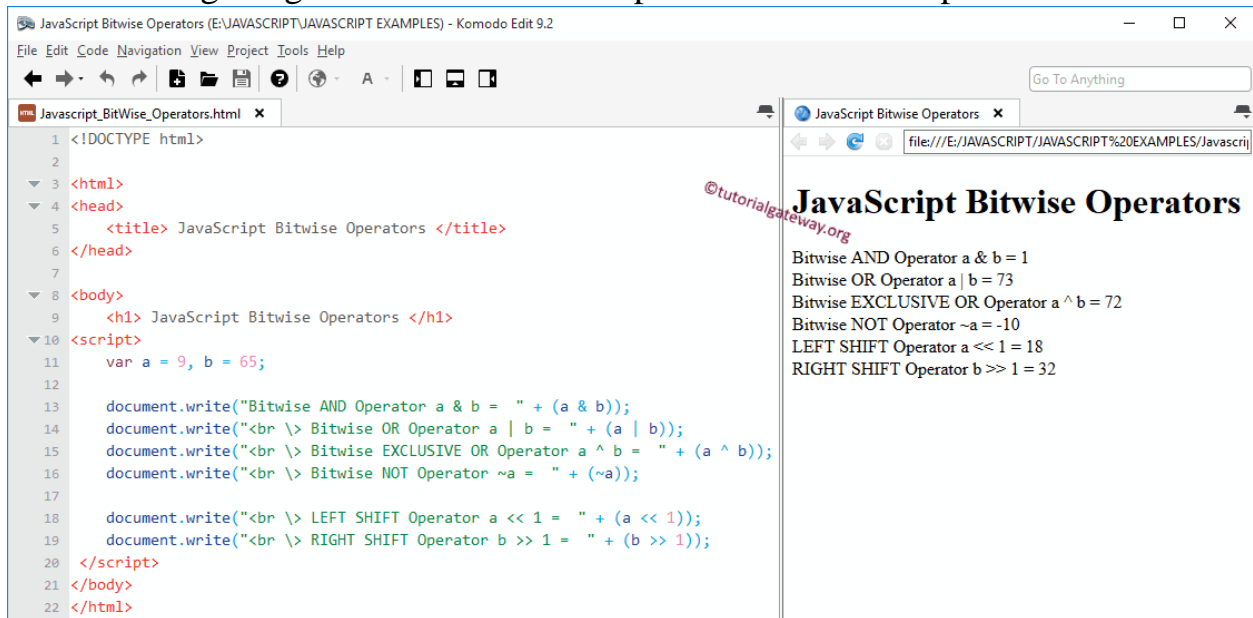
For example: a=2;

a>>>1;

## OUTPUT:

1

The following image has the code and output for the Bitwise operators.



## 4.5 ASSIGNMENT OPERATORS:

JavaScript provides the following assignment operators.

### SIMPLE ASSIGNMENT:

Assigns values from the right-side operand to the left side operand

**Ex:**  $C = A + B$  will assign the value of  $A + B$  into  $C$

### **ADD AND ASSIGNMENT:**

It adds two operands and assigns the result to the left operand.

**Ex:**  $C += A$  is equivalent to  $C = C + A$

### **SUBTRACT AND ASSIGNMENT:**

It subtracts two operands and assigns the result to the left operand.

**Ex:**  $C -= A$  is equivalent to  $C = C - A$

### **MULTIPLY AND ASSIGNMENT:**

It multiplies the right operand with the left operand and assigns the result to the left operand.

**Ex:**  $C *= A$  is equivalent to  $C = C * A$

### **DIVIDE AND ASSIGNMENT:**

It divides the left operand with the right operand and assigns the result to the left operand.

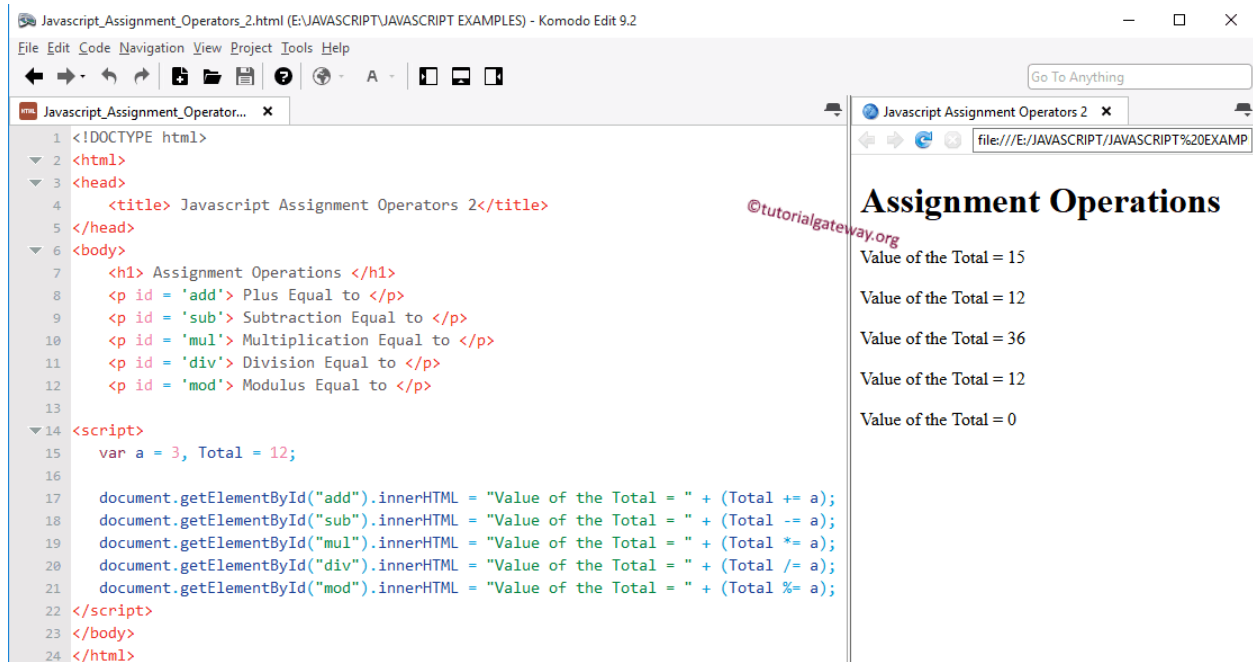
**Ex:**  $C /= A$  is equivalent to  $C = C / A$

### **MODULES AND ASSIGNMENT:**

It takes modulus using two operands and assigns the result to the left operand.

**Ex:**  $C \%= A$  is equivalent to  $C = C \% A$

The execution of the assignment operators and their output is shown as follows:



## 4.6 MISCELLANEOUS OPERATOR:

In JavaScript there are two miscellaneous operators. They are

- Conditional operator
- Typeof operator

### CONDITIONAL OPERATOR:

**'? :**' is the conditional operator. It operates on three operands and hence it is called Ternary operator.

“(condition)? true: false;” is the general syntax.

For example: a=54; b=65;

```
c = (a>b)?a:b;
document.write(c);
```

### OUTPUT:

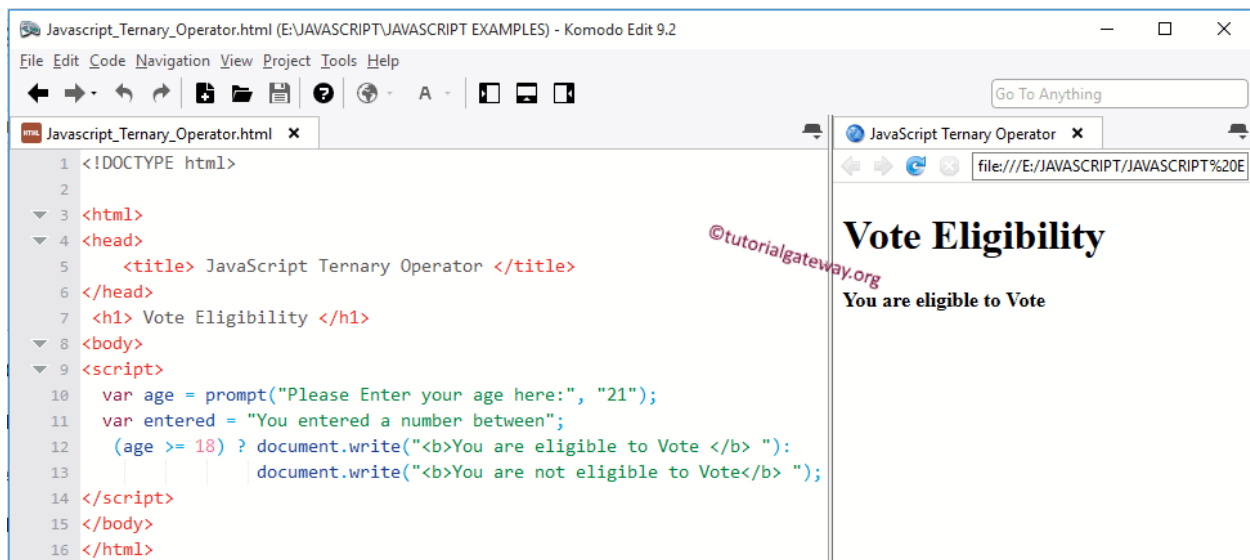
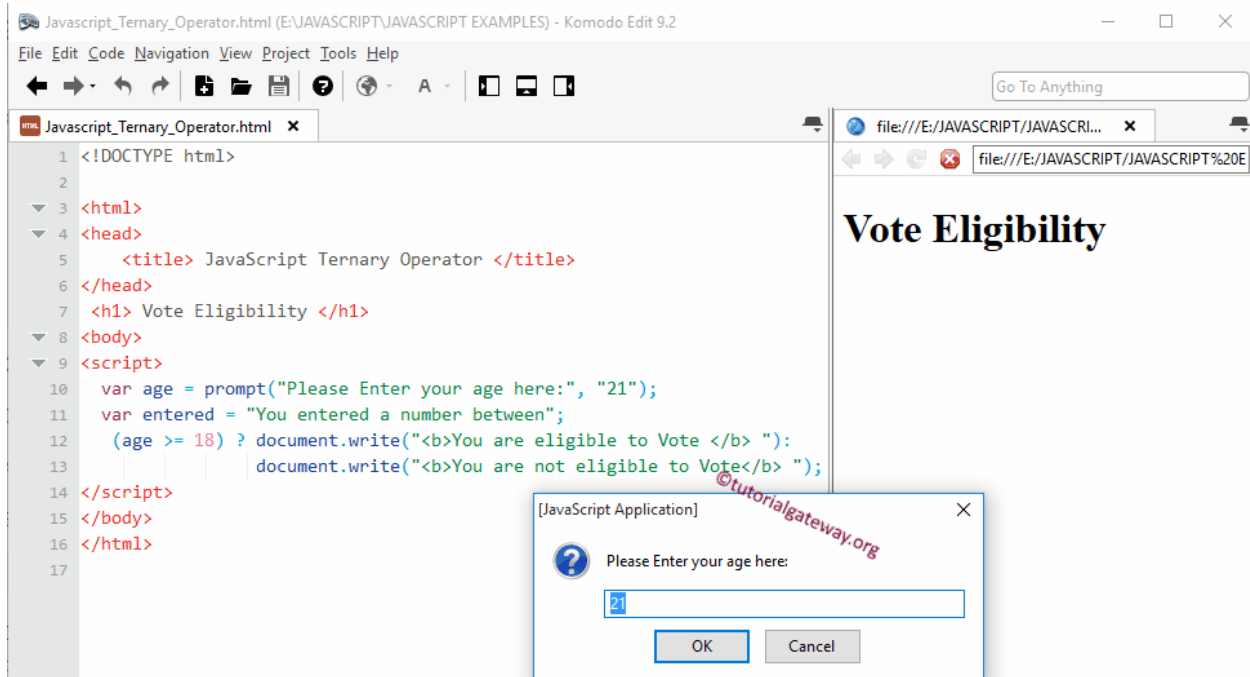
65

In the above example, the condition is a>b, if the condition is true, then the statement next to question mark will be executed, else if the condition is not true, then the statement next to the colon will be executed.



Here the  $a > b$  condition is not true, therefore the value of 'b' is assigned to the operand c.

Let us consider the situation of voting, if a person is above 18 years old, he is eligible to vote, otherwise he is not eligible to vote. This can be done by using conditional operator as follows.



## TYPEOF OPERATOR:

The **typeof** operator is a unary operator that is placed before its single operand, which can be of any type. Its value is a string indicating the data type of the operand.

The *typeof* operator evaluates to "number", "string", or "Boolean", based on its operand value and returns true or false based on the evaluation.

Here is a table of the return values for the **typeof** Operator.

Type	String Returned by typeof
Number	"number"
String	"string"
Boolean	"boolean"
Object	"object"
Function	"function"
Undefined	"undefined"
Null	"object"

For example:

```
var a = 10;  
var b= (typeof a == "string")? "a is string": "a is number";  
document.write(b);
```

## OUTPUT:

a is number

## CODE

## OUTPUT

**JavaScript typeof**

The typeof operator returns the type of a variable or an expression.

```
string
string
string
number
number
number
number
number
```

```
<html>
<body>
<h2>JavaScript typeof</h2>
<p>The typeof operator returns the type of a variable
or an expression.</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML=
typeof "" + "<br>" +
typeof "Sweta" + "<br>" +
typeof "Sweta Kumari" + "<br>" +
typeof 0 + "<br>" +
typeof 314 + "<br>" +
typeof 3.14 + "<br>" +
typeof (3) + "<br>" +
typeof (3 + 4);
</script>
</body>
</html>
```

The above example code uses the typeof operator to find the datatypes of the inputs as strings and numbers.

## POST-LAB EXERCISES

1. Write a JavaScript code to explain the Increment and Decrement operators.
2. Write a JavaScript program to find the datatype of the following inputs.

```
var a=2;
```

```
var b=true;
```

```
var c="Ram";
```

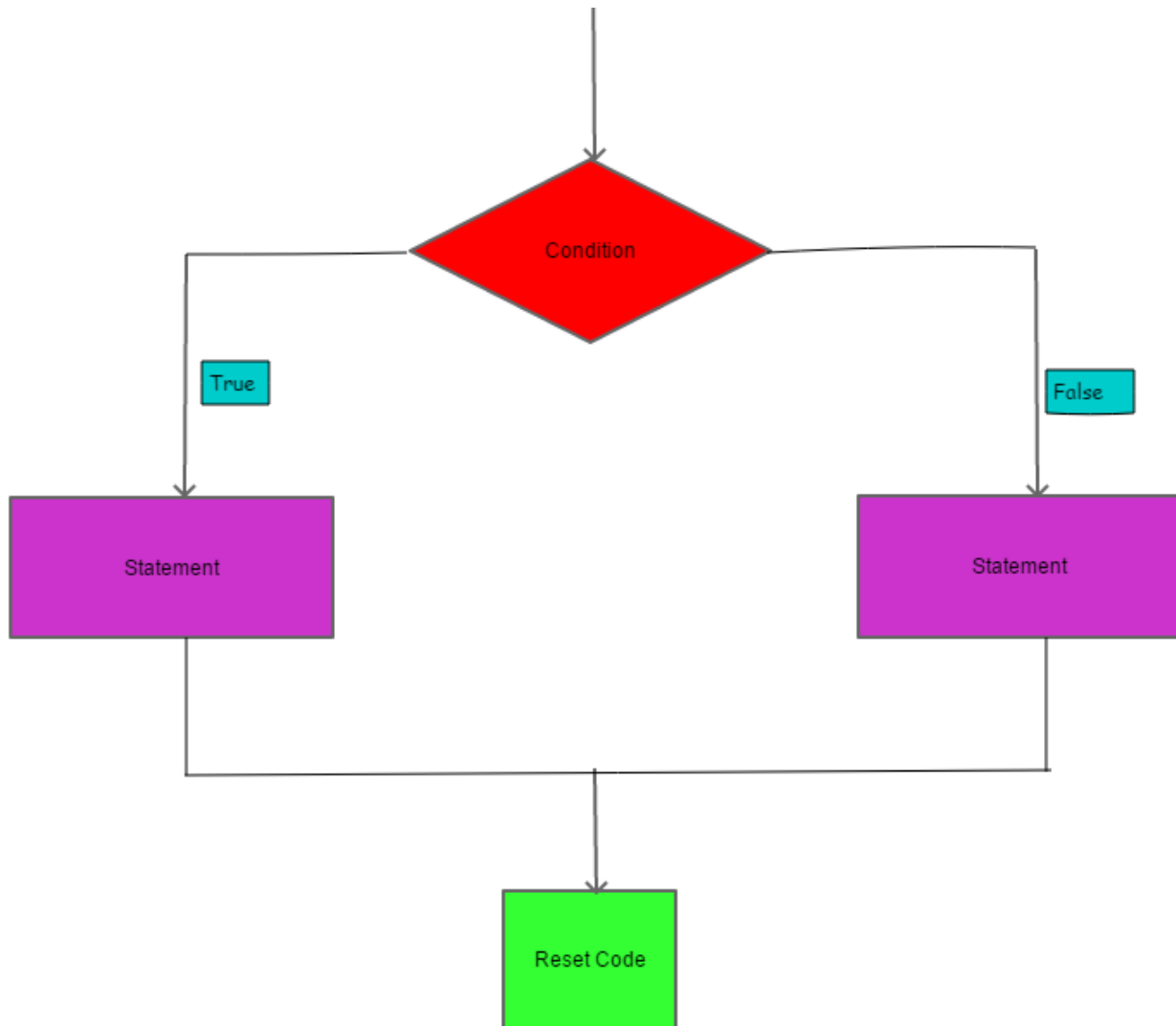
3. Write a JavaScript code using the Left shift operator.
4. Write a JavaScript code using the bitwise Not operator.
5. Write the JavaScript code for Logical And operator?

## References

1. [https://www.tutorialspoint.com/javascript/javascript\\_operators.htm](https://www.tutorialspoint.com/javascript/javascript_operators.htm)
2. Output for Operators-<https://www.tutorialgateway.org/javascript/>
3. Type of Operator-<https://unacademy.com/lesson/javascript-type-of-operator-in-hindi/CFBP1VSZ>

## 5. CONDITIONAL STATEMENTS

The conditional statements are the ones which are used in the situation where based on a condition, decision is made, an operation is performed.



JavaScript supports the following conditional statements.

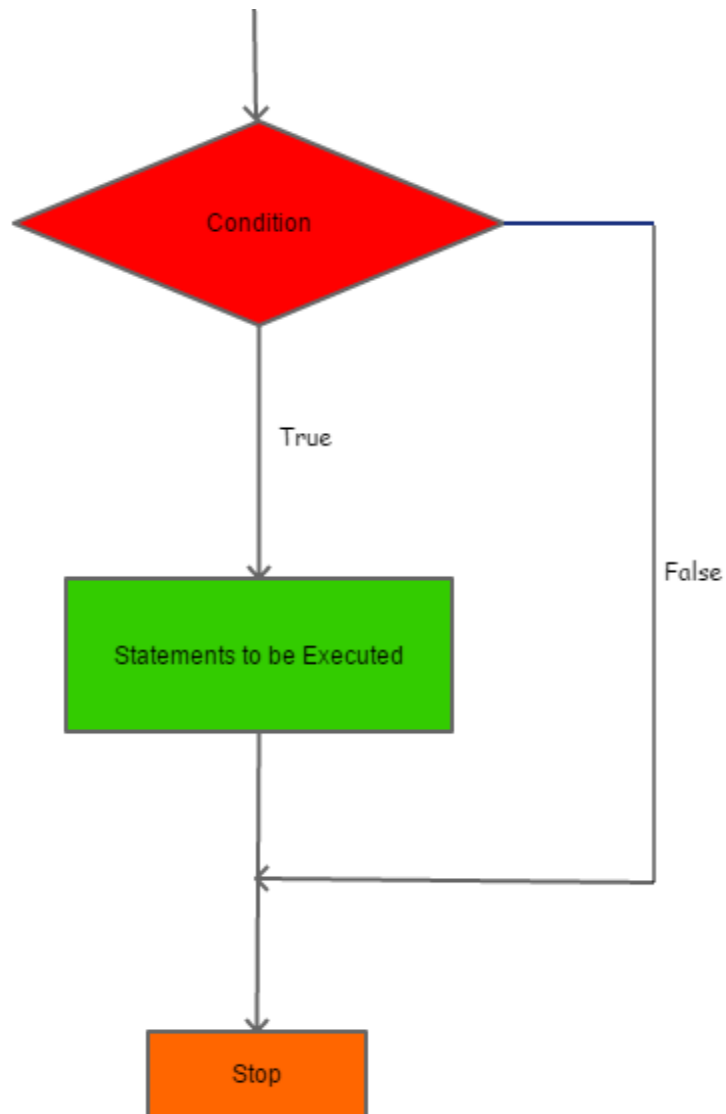
- if statement
- if... else statement
- if... else if... statement
- switch case

## 5.1 if statement:

The **if** statement checks for a condition and if it is satisfied, then it evaluates to true, then the steps of operation inside the if block will be executed. If the condition gets failed then the result of the condition is false. If the condition returns false then operations in the if block are not executed and the control goes to the next step.

### FLOWCHART:

The flowchart for the if statement is as follows:



## SYNTAX:

The syntax for the if statement is as follows:

if (condition)

{

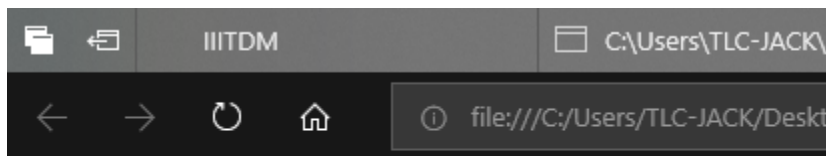
Statement(s) to be executed if condition is true

}

## PROGRAM:

```
1 <!DOCTYPE HTML>
2 <HTML>
3 <BODY>
4 <SCRIPT> //This program explains the if statement
5 var a=prompt("Enter any Number","");//The variable 'a' is declared and gets input from
  user,prompt command will display a pop up window and gets the user input
6 if(a>0){document.write("This is a Natural Number");};//The condition a>0 is checked and it
  evaluates true or false based on the user input,if it is satisfied the statements in the if
  block are executed
7 </SCRIPT>
8 </BODY>
9 </HTML>
```

## OUTPUT:



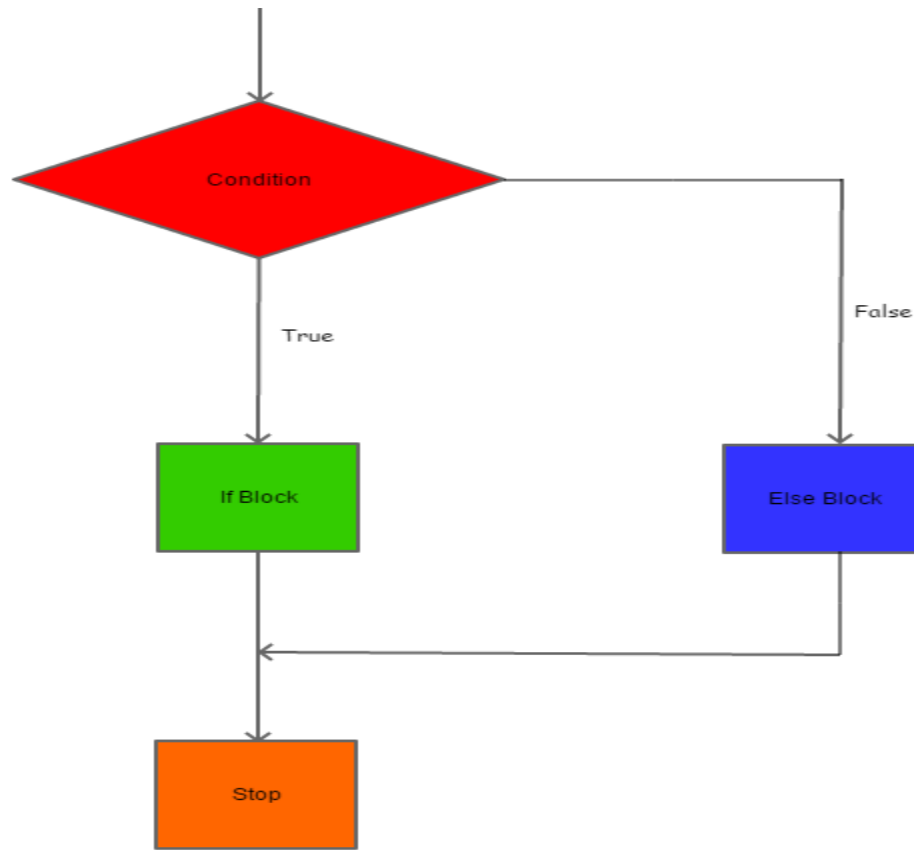
This is a Natural Number

In the above code the prompt command gets the input by displaying the popup window and asks the user to “Enter any number”. Let us consider that the user gives the input as 5. The input 5 is greater than zero. Therefore, the output is printed as “This is a Natural number”.

## 5.2 if.... else statement:

In the **if... else** statement there will be a condition given and if the inputs satisfy the condition then a set of operations will be performed and if the condition is not satisfied then a set of operation is performed.

### FLOWCHART:



### SYNTAX:

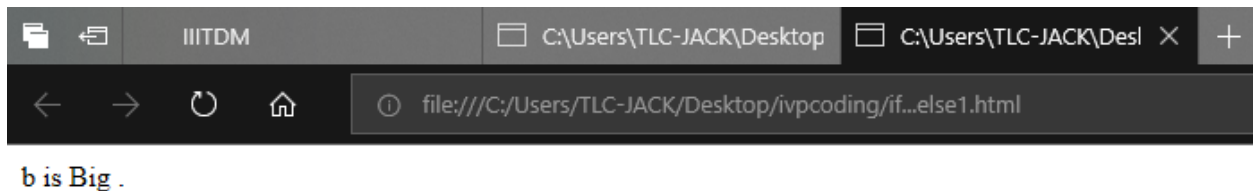
```
if (condition)
{
    Statement(s) to be executed if condition is true
}
else
{
    Statement(s) to be executed if condition is false
}
```



## PROGRAM:

```
1 |<!DOCTYPE HTML>
2 ▼ <HTML>
3 ▼ <BODY>
4 ▼ <SCRIPT>//This program explains the working of if else statement
5 var a=5;//The variable 'a' is declared and initialized to 5
6 var b=6;//The variable 'b' is declared and initialized to 6
7 if(a>b){document.write("a is Big");};//The condition a>b is checked and as it is not true
   the control passes to the else statement
8 else{document.write("b is Big");};//The statements in the else block are executed as the
   if condition was not satisfied
9 </SCRIPT>
10 </BODY>
11 </HTML>
```

## OUTPUT:

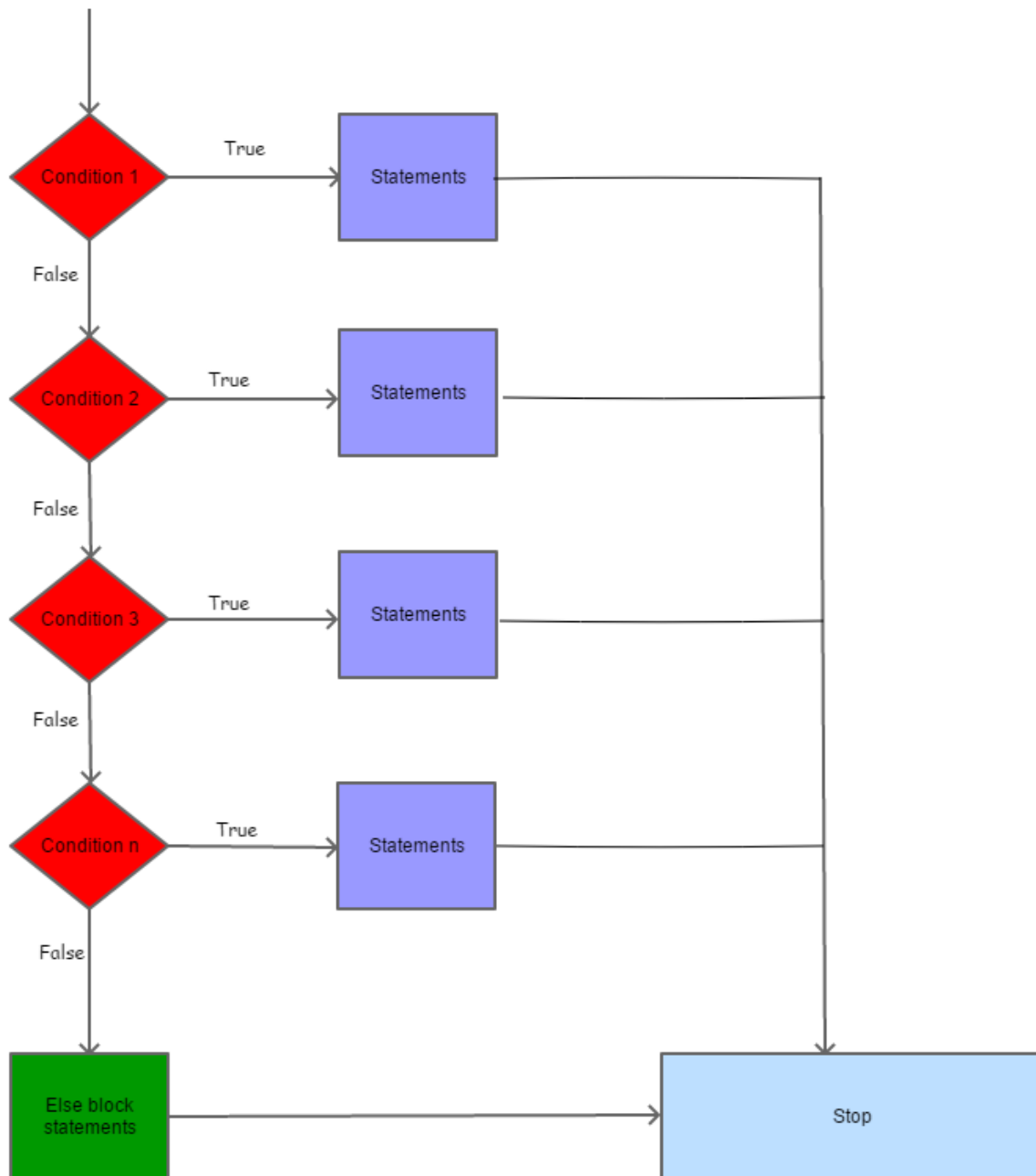


In the above program, the value in a is not bigger than the value in b, therefore the operations in the else loop is evaluated.

### 5.3 if.... else if statement:

The **if.... else if** statement is used when there are more than two possibilities. There will be multiple conditions, each condition will be checked and whichever condition gets satisfied, the statements in its block will be executed. If a condition is evaluated to be true than, the conditions after it won't be checked and they will be skipped.

## FLOWCHART:



## **SYNTAX:**

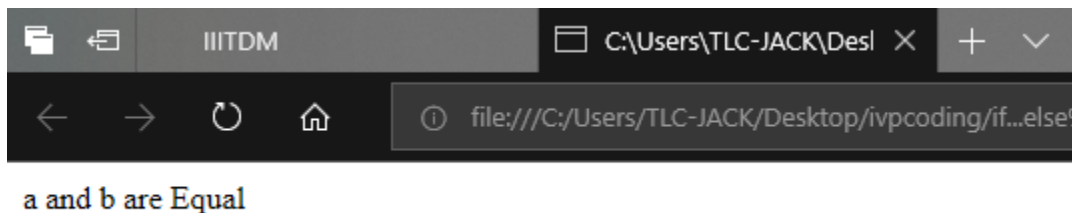
```
if (condition 1)
{
    Statement(s) to be executed if condition is true

}
else if(condition 2)
{
    Statement(s) to be executed if condition is true
}
else if(condition 3)
{
    Statement(s) to be executed if condition is true
}
    .
    .
    .
else if(condition n)
{
    Statement(s) to be executed if condition is true
}
```

## PROGRAM:

```
1 |<!DOCTYPE HTML>
2 ▾ <HTML>
3 ▾ <BODY>
4 ▾ <SCRIPT> //This is a Program to explain if..else if statement
5 var a=4; //The variable a is declared and its value is assigned as 4
6 var b=4.0; //The variable b is declared and its value is assigned as 4.0
7 if(a>b){document.write("a is Big");} //The condition a>b is checked and it is not
   satisfied so the control passes to else if
8 else if(b>a){document.write("b is Big");} //The condition b>a is checked and it is
   also not satisfied so the control passes to next else if statement
9 else if(a==b){document.write(" a and b are Equal");} //The Condition a==b is checked
   and it is satisfied so the statements in this else if statement is executed.
10 </SCRIPT>
11 </BODY>
12 </HTML>
```

## OUTPUT:



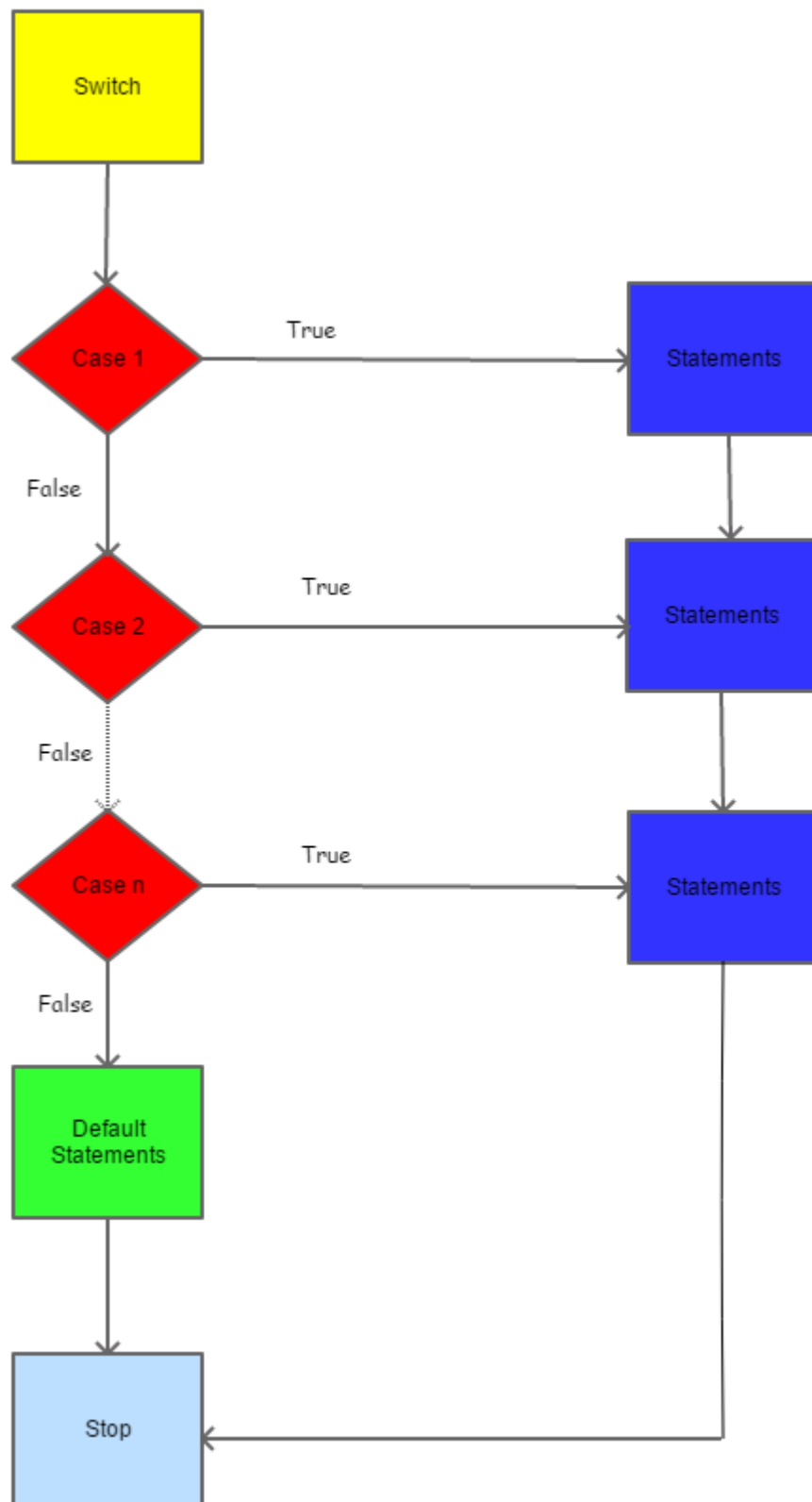
In the above program the first condition in the if statement is checked, it is not satisfied. Now the next condition in the else if is checked, it is also not satisfied. Then the third condition is checked, it is satisfied therefore, the statements in it are executed.

## 5.4 SWITCH CASE:

When there is more than one possibility available for a single expression, then the switch case statement is used. The case which satisfies the condition is executed.

## FLOWCHART:

The flowchart for the switch case statement is as follows:



## SYNTAX:

```
switch (expression)
{
    case value_1: statement_1;
                break;
    case value_2: statement_2;
                break;
    case value_3: statement_3;
                break;
    default: default_statement;
}
```

## PROGRAM:

```
1 <html>
2 <body>
3 <script type = "text/javascript"> //This program explains the Switch case statement
4 <!--
5     var grade = 'A'; //The variable grade is declared and the assigned the value 'A'
6     document.write("Entering switch block<br />"); //The statement is printed
7     switch (grade) {
8         case 'A': document.write("Good job<br />"); //This case is satisfied and the statements in this
9                 case are printed.
10                break; //The control passes out of the switch statement by the break command.
11
12                case 'B': document.write("Pretty good<br />");
13                break;
14
15                case 'C': document.write("Passed<br />");
16                break;
17
18                case 'D': document.write("Not so good<br />");
19                break;
20
21                case 'F': document.write("Failed<br />");
22                break;
23
24                default: document.write("Unknown grade<br />")
25            }
26            document.write("Exiting switch block"); //The statement is printed.
27        //-->
28    </script>
29    <p>Set the variable to different value and then try...</p>
30 </body>
</html>
```

## OUTPUT:



Entering switch block  
Good job  
Exiting switch block

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

In the above code the grade is 'A' therefore the case 'A' is printed.

## POST-LAB EXERCISES

1. Write a JavaScript program to check whether a number is prime or not.
2. Write a JavaScript program to check whether a given number is even.
3. Write a JavaScript program to use the prompt key to get input from the user.
4. Write a JavaScript program to check whether a given number is multiple of 2,3,4,6,7.
5. Write a JavaScript program to check whether a given number is a Square number.

## References

1. <https://www.tutorialspoint.com/javascript/javascript.htm>
2. Try out the Exercise - [https://www.w3schools.com/js/js\\_exercises.asp](https://www.w3schools.com/js/js_exercises.asp)



## 6. LOOPING STATEMENTS

The looping statements are used when a set of expressions are to be performed repeatedly until a certain condition is true. On every iteration the, the condition is checked, when the condition evaluates to be false the looping stops and the control passes out of the loop.

The looping statements that are offered by JavaScript are as follows:

- ❖ for loop
- ❖ while loop
- ❖ do while loop

### 6.1 For loop:

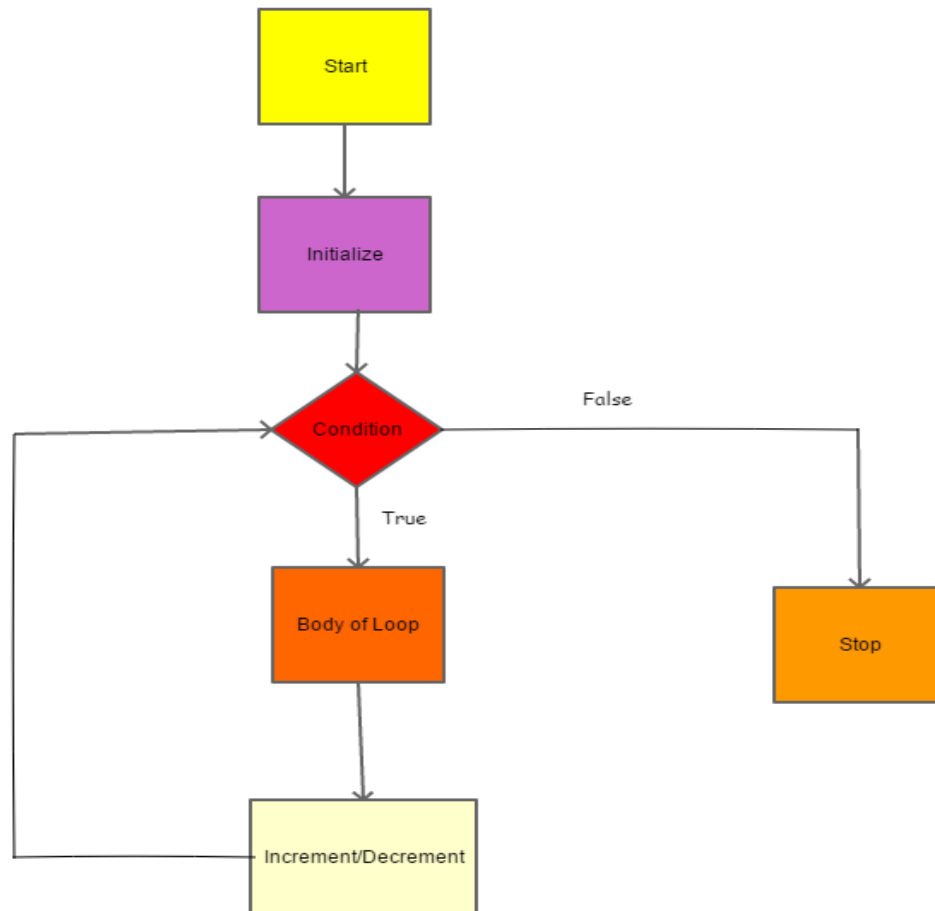
The **for** loop consists of three segments. They are initialization, condition, iteration statement.

**Initialization-** a variable is assigned a value.

**Condition-** the condition is checked on every iteration, if it results in true statements in for loop is executed, if it results in false, then the control jumps to the next statement in the loop without going inside the loop.

**Iteration statement-** it contains the increment or the decrement operator.

## FLOWCHART:



## SYNTAX:

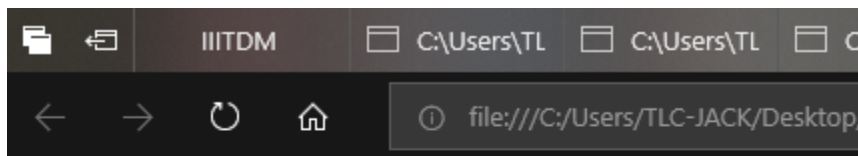
```
for (initialization; test condition; iteration statement)
{
    Statement(s) to be executed if test condition is true
}
```

## PROGRAM:

Look at the sample program given below.

```
1 <!DOCTYPE HTML>
2 <HTML>
3 <BODY>
4 <SCRIPT>//This program illustrates the working of the for loop
5 var i;//The variable 'i' is initialized
6 document.write("We are printing numbers from 1-5" + "<br/>");//The statement is printed
7 for (i=1;i<=5;i++){document.write(i + "<br/>");};//First the initialization is done, then
8 //the condition is checked and the control goes into the loop when the condition is true.
9 //The statements in the loop are executed once and the incrementation process takes
10 //place.Now again the condition is checked if its true the loop body is executed.
11 //This is repeated again and again until the condition is not satisfied. When The
12 //condition is not satisfied the control goes out of the loop,here it occurs when i=6 in
13 //the 7th iteration.
14 document.write("Thus,We printed the numbers from 1-5");//The statement is printed
15 </SCRIPT>
16 </BODY>
17 </HTML>
```

## OUTPUT:



We are printing numbers from 1-5

1  
2  
3  
4  
5

Thus,We printed the numbers from 1-5

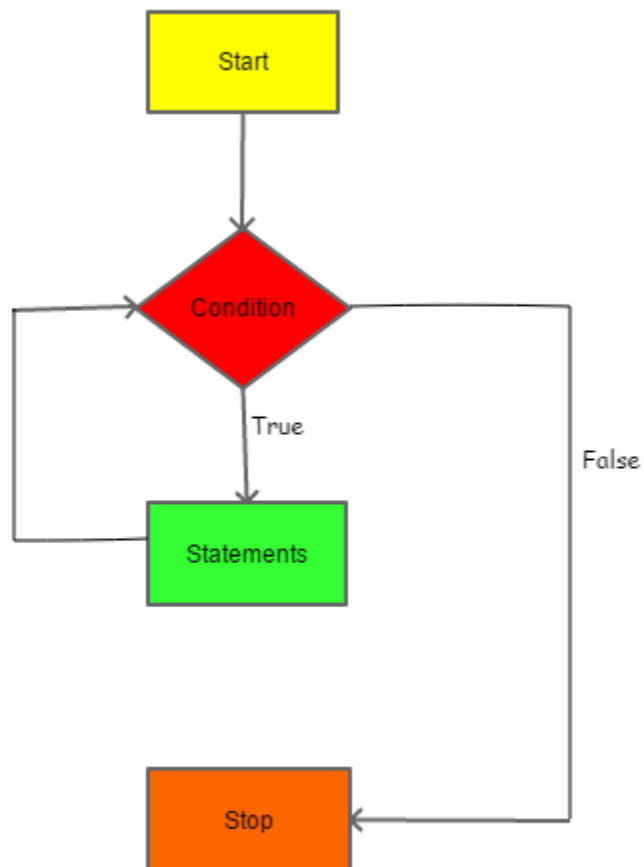
The above program initializes the value of variable 'i' in the for loop as 1. Now, the condition is tested and the condition is satisfied the loop statements are executed. This is the first iteration. After the first iteration is completed the value of 'i' is incremented as given in the iteration statement. Now again the condition is checked and it is true then the loop statements are executed. This is the second iteration. The

above process is repeated until the condition is not satisfied. When the condition is not satisfied, the control gets out of the loop and the for loop is executed.

## 6.2 WHILE LOOP:

**while** loop is the most basic loop structure in JavaScript. The function is the same, but this loop won't have the initial three segments as the 'for loop'. The condition is checked, if it is true the looping statements are executed and after execution of the looping statements the condition is checked and this process repeated until the condition is not satisfied.

### FLOWCHART:



### SYNTAX:

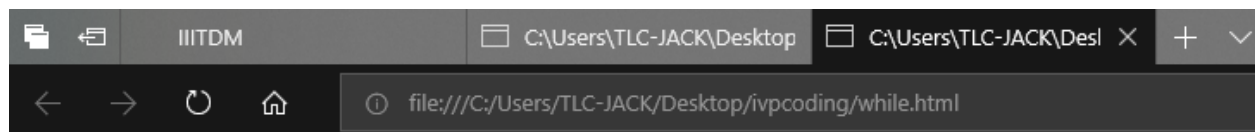
while (condition)

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

## PROGRAM:

```
1  <!DOCTYPE HTML>  
2  <HTML>  
3  <BODY>  
4  <SCRIPT>//This program explains the While loop  
5  var i=1;//The variable 'a' is declared and initialized to 1.  
6  document.write("We are printing the numbers from 1-5" + "<br/>");//The statement is  
   printed  
7  while(i<=5){document.write(i + "<br/>");i++;};//The condition i<=5 is checked and if  
   it satisfied the loop body is executed.  
8  //Then again the condition is checked and the loop body is executed if the condition  
   is true.  
9  //The control goes out of the loop when the condition evaluates to false,here when  
   i=6.  
10 document.write("Thus, We printed the numbers from 1-5 using while loop");//The  
   statement is printed  
11 </SCRIPT>  
12 </BODY>  
13 </HTML>
```

## OUTPUT:



We are printing the numbers from 1-5

1  
2  
3  
4  
5

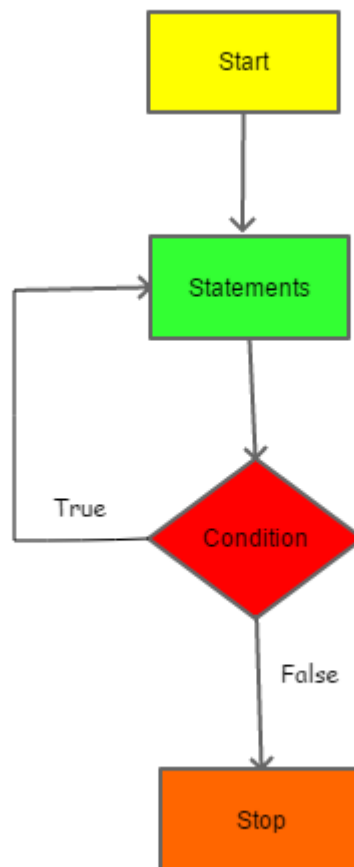
Thus, We printed the numbers from 1-5 using while loop

In the above program the, the number of iterations is 5. On the sixth iteration the 'i' value is changed to 6, the condition results in false and the control goes out of the loop.

### 6.3 DO WHILE LOOP:

The **do while** loop is slightly different from the above two loops. In the above two loops the condition is checked first and then the loop statements are executed, while in this loop the condition is placed below the looping statements and the loop is executed first and then the condition is checked. So, the loop executes its first iteration by default, even if the condition is false.

#### FLOWCHART:



#### SYNTAX:

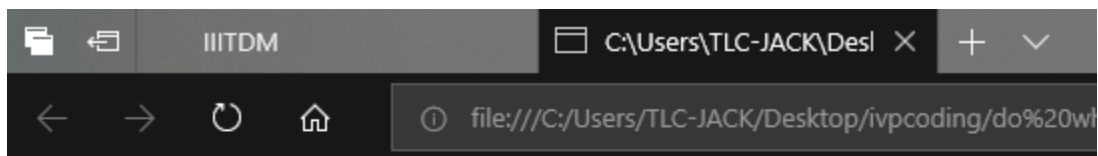
```
do
{
    Statement(s) to be executed if test condition is true
}
```

while(condition)

## PROGRAM:

```
1 <!DOCTYPE HTML>
2 <HTML>
3 <BODY>
4 <SCRIPT>//This is a Sample program for Do while loop
5 var i=1;//"i" an variable is declared and initialized its value to '1'.
6 document.write("We are printing the numbers from 1-5" + "<br/>");//The
  statement is printed
7 do{document.write(i + "<br/>");i++;}while(i<=5);//The value of i is printed
  till 'i' is lesser than or equal to 5
8 document.write("Thus, We printed the numbers from 1-5 using do while
  loop");//The statement is printed
9 </SCRIPT>
10 </BODY>
11 </HTML>
```

## OUTPUT:



We are printing the numbers from 1-5

1  
2  
3  
4  
5

Thus, We printed the numbers from 1-5 using do while loop

In the above program, if the value of 'i' was initially 6, the loop statements will be executed once and the number 6 will be printed once. When the condition is checked, it will result in false and then the control will go out of the loop.

## 6.4 FUNCTION:

The **function** is used in JavaScript program, where a set of codes need to be repeated for a certain number of times in various time, not at the same time. The time in which they are repeated is based on the requirement. Function provides the user the reusability of code. The function statements are executed when it is called in the program. The function will not be executed if it is not even called in the program.

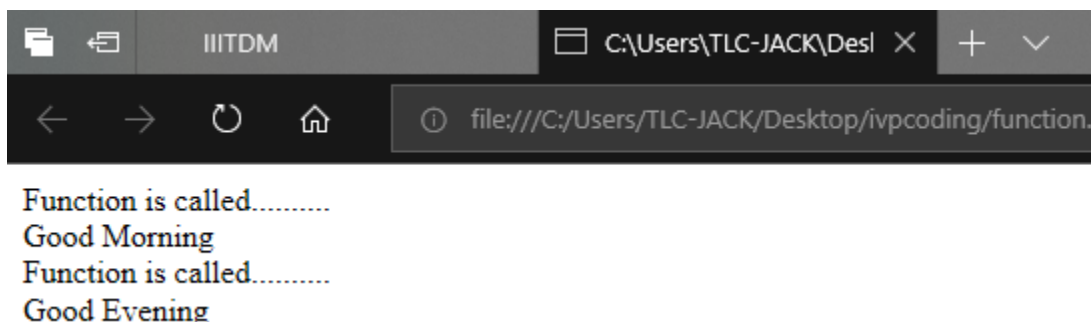
### SYNTAX:

```
function function_name(parameter-list)
{
    statements
}
```

### PROGRAM:

```
1 <!DOCTYPE HTML>
2 <HTML>
3 <BODY>
4 <SCRIPT> //This is a sample program for understanding the functions in
   JavaScript
5 function Morning(){document.write("Good Morning" + "<br/>");} //The function
   'Morning' is defined
6 function Evening(){document.write("Good Evening" + "<br/>");} //The function
   'Evening' is defined
7 document.write("Function is called....." + "<br/>"); //The statement is
   printed
8 Morning(); //The function Morning is called and its statements are executed
9 document.write("Function is called....." + "<br/>"); //The Statement is
   printed
10 Evening(); //The function Evening is called and its statements are executed
11 </SCRIPT>
12 </BODY>
13 </HTML>
```

### OUTPUT:





## POST-LAB EXERCISES

1. Write a JavaScript program to print the numbers from 1-10 in descending order.
2. Write a JavaScript program to display the Factorial of the given number.
3. Write a JavaScript program to display the sum of numbers from 1 to n, where n is the input from the user.
4. Write a JavaScript program to print a Fibonacci series.
5. Write a JavaScript program to print the following output.

1

1 2

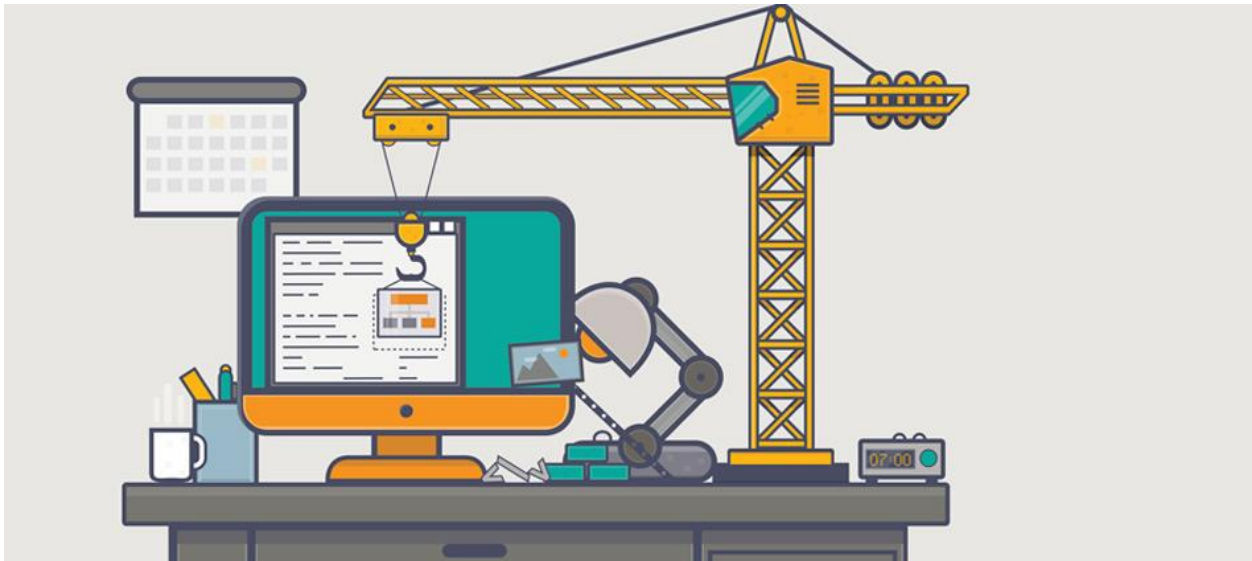
1 2 3

1 2 3 4

## References

1. <https://www.tutorialspoint.com/javascript/javascript.htm>
2. Cheat Sheet for JS-<https://htmlcheatsheet.com/js/>

## 7. CREATING A WEBPAGE



Now, we are going to create our webpage.

A web page is any page on an internet website. Everything we created above is a webpage, if it is in internet. The Webpage can be created by using HTML, CSS and JavaScript. The collection of Web pages is called Website.

Web browsers will frequently have to access multiple web resource elements, such as reading style sheets, scripts, and images, while presenting each web page.

Now let us get into creating a web page. Get ready to design your own web page.

Let us design a web page to show our Fortune.

**STEP-1:** Create an array of Qualities. An array can be created as follows:

```
var array_name=['value1', 'value2', 'value3',....., 'value_n'];
```

**STEP-2:** Use `Math.random()` function to generate a random number and use `Math.round()` function to round off the integer value into the random number.

**STEP-3:** Use a Function and enclose all the above coding inside that function.

**STEP-4:** Call the function in the Body of the HTML, when the button ‘Fortune is pressed’.

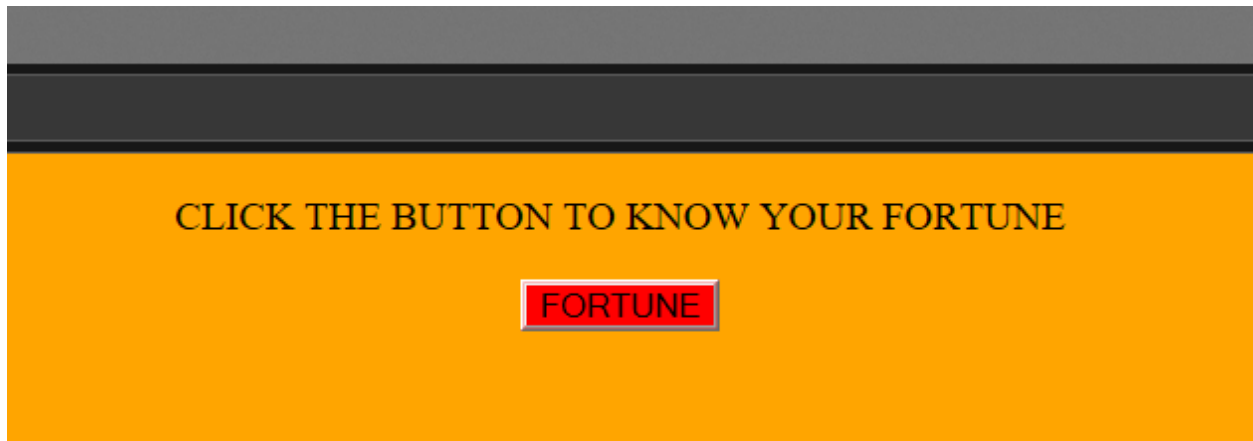
**STEP-5:** The button can be created as follows:

```
<input type =“button” value=“Fortune”/>
```

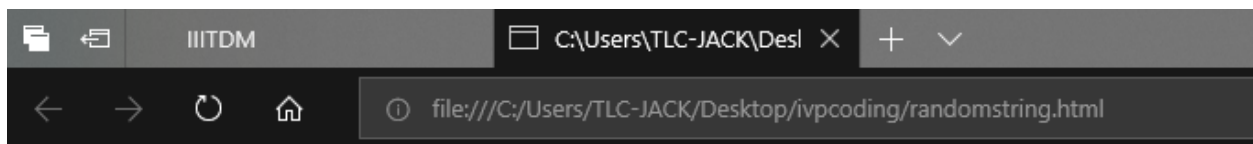
## PROGRAM:

```
1 <!DOCTYPE HTML>
2 <HTML><!--This program is to create a Fortune Web page-->
3 <HEAD>
4 <TITLE>GAME</TITLE><!--The title of the webpage is Game-->
5 <STYLE>
6 INPUT[TYPE="BUTTON"]{BACKGROUND-COLOR:RED;}
7 </STYLE><!--The button with color red is created-->
8 </HEAD>
9 <BODY STYLE="BACKGROUND-COLOR:ORANGE";>
10 <!--The Background color of the whole page is made orange-->
11 <P><CENTER>CLICK THE BUTTON TO KNOW YOUR FORTUNE</CENTER></P>
12 <!--The text is printed-->
13 <CENTER><INPUT TYPE="BUTTON" VALUE="FORTUNE" ONCLICK="task() ;"/></CENTER>
14 <!--The button is made to appear in centre of the web page-->
15 <!--The function task() is called when the button Fortune is clicked-->
16 <SCRIPT>//The function task is declared
17 function task(){
18     var random = Math.random( ); //the variable random is initialized to Math.random function which randomly
    gives a number
19     document.write("<BR/>" + "Your Lucky Number is " + Math.round(random*10) + "<br/>" );
20     //The number is made as whole number and lesser than 10
21     var myArray=['GOOD','EMOTIONALY
    OPEN','HONEST','POSITIVE','CARETAKER','KIND','CONFIDENT','HAPPY','KING','SUCCESSFUL'];
22     //An array 'myArray' is created with 10 personalities
23     var rand = myArray[Math.round(Math.random() * myArray.length)];
24     //A variable named 'rand' will randomly select a personality from the array using Math.round() and
    math.random() function.
25     document.write("You will be " + rand + " in your life ");//The statement is printed
26 };
27 </SCRIPT>
28 </BODY>
29 </HTML>
```

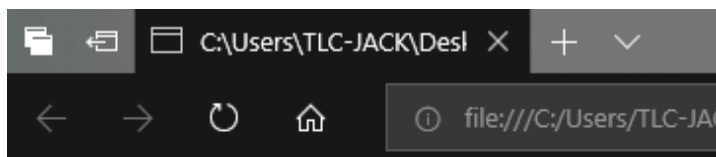
## OUTPUT:



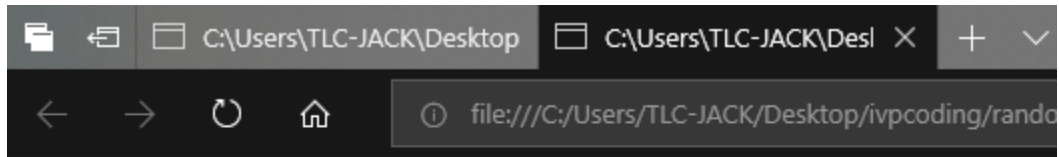
The screen appears as above, when we click the 'Fortune' button it tells our fortune as follows:



Your Lucky Number : 4  
Your will be KIND in your life



Your Lucky Number : 7  
Your will be HAPPY in your life



Your Lucky Number is 9  
Your will be HONEST in your life

## **POST-LAB EXERCISE**

- 1.What is the use of the center tag in the above code?
- 2.What is the use of the paragraph tag in the code?
- 3.What is the use of the align Text command in JavaScript?
- 4.How the background color can be changed?
- 5.Write a JavaScript code to using for in loop?

## **References**

1. [https://en.wikipedia.org/wiki/Web\\_page](https://en.wikipedia.org/wiki/Web_page)
- 2.Try out the exercise-<https://exercism.io/tracks/javascript/exercises>

## 8. GAME



Here, are few games for you to try it out. Go have some Fun.

### 8.1 LIFETIME CALCULATOR:

C:\Users\TLC-JACK\Desktop

C:\Users\TLC-JACK\Desl

×

+

▼

←

→

↺

🏠

① file:///C:/Users/TLC-JACK/Desktop/ivpcoding/age.html

Please enter your Date of Birth::  Example: (november 1,1966)

Start Timer

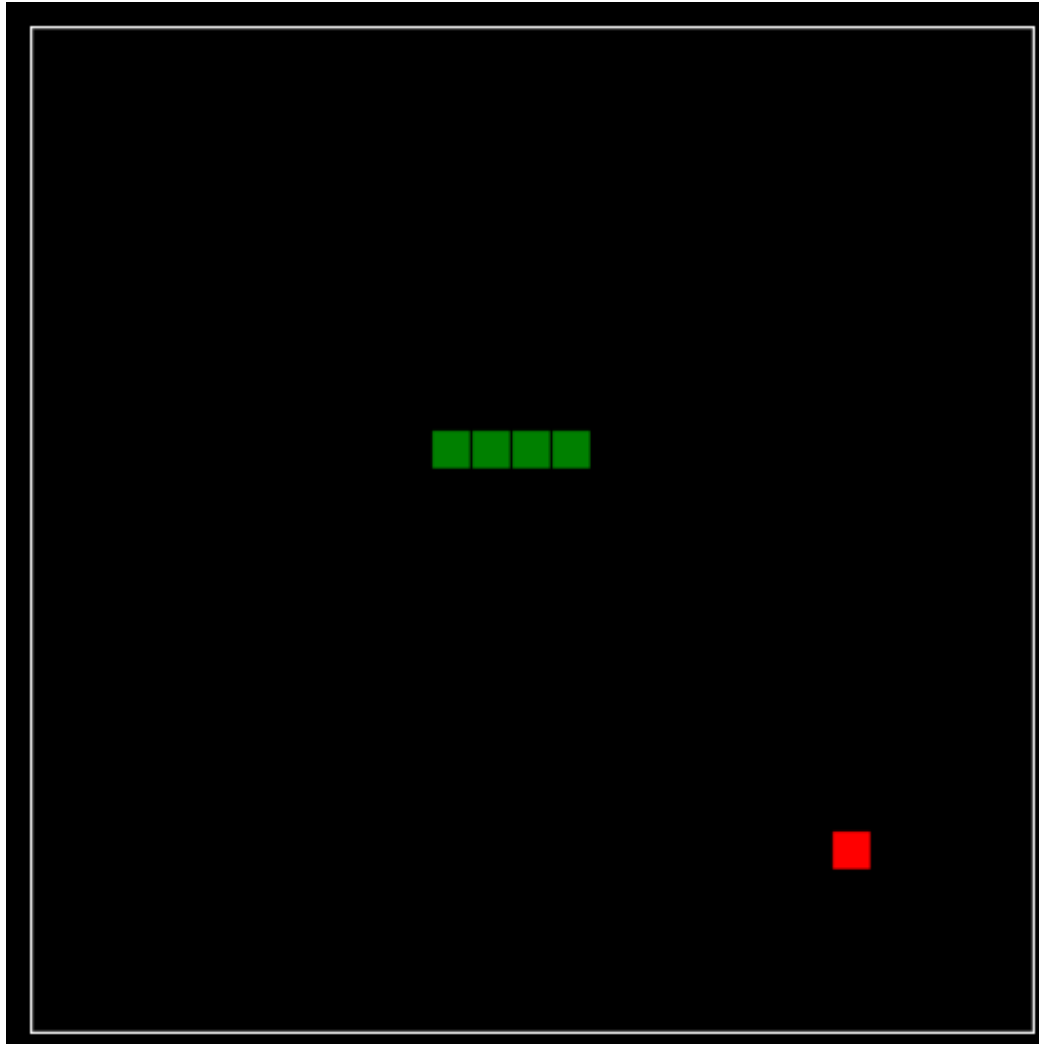
Reset Age

You are days old:

Plus hours old:

Plus minutes old:

## 8.2 SNAKE GAME:



The Source codes for the above output can be found in the reference links given below.



# References

- 1.Lifetime Calculator- <http://www.javascriptkit.com/script/cut8.shtml>
- 2.Snake Game-<https://gist.github.com/straker/ff00b4b49669ad3dec890306d348adc4>
- 3.180 websites in 180 days, try out the programs and have fun-  
<https://jenniferdewalt.com/>