Introduction to JavaScript

- Basically, Java is a full programming language developed by Sun Microsystems with formal structure.
- But, JavaScript is a scripting language developed by Netscape that is used to modify web pages.
- The JavaScript must be written in the HTML document between container tag <SCRIPT>….</SCRIPT>
- Special Attribute For <script> tag is, type="text/javascript".

Use to JavaScript

- JavaScript provides programming tool to HTML designer.
- JavaScript can put dynamic text in HTML page.
- JavaScript can react to events.
- JavaScript can read and write HTML elements.
- JavaScript can be used to validate data.
- JavaScript can be used to detect the visitor's browse.
- JavaScript can be used to create cookies.

Features of JavaScript

- **JavaScript is case sensitive**
  - JavaScript is case sensitive so the functions which have lowercase should be written like that.
  - If we want to use write() function of JavaScript then we cannot use it as Write().
  - In JavaScript variables are also case sensitive so variable "a" is not same as "A".
- **The Symbols must be closed if opened**
  - In JavaScript, symbols like ({" ' , Such symbols if used then it must closed like, ' " }).
- **White space is ignored**
  - White space are space, tab and new line.
  - These all kind of white spaces are ignored by the JavaScript.
- **Comments**
  - We can put comments in JavaScript
    - Single Line "//" slash is used.
    - For multi line comment /* */
- **Ending statements with semicolon**
  - In JavaScript the semicolons are optional.
  - But if we want to put new statement in same line we can use **Semicolon.**

- **Example of Semicolon..**
    - document.write("Hello"); document.write("Test");

- **Script Tag in HEAD section.**
    - The script tag is used in head section.
    - Mostly User Defined Function will paste in head section
    - Syntax:

```
<head>
    <script type="text/javascript">
        ……
    </script>
</head>
```

- **Script Tag in BODY section.**
    - To display something in the document page we can use Script Tag in Body Section.
    - Syntax:

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

- **Insert Special Characters**
    - We can insert special characters like " ' \ by giving backslash (****) in document.
        - **For Example,** document.write("Hello! \'Friends\'")
    - Output :

      Hello 'Friends'

Variables

- Human mind has memory cells inside the brain to remembering various values.
- Similar to human brain computer has also memory Cells.
    - Each memory cell in the computer has unique address.
    - It is not possible to remember the address of each memory cells.
    - That's why cell is introduced with a name which is known as **"Variable".**

Variable Declaration

- To declare a variable in JavaScript we can use following syntax.

Syntax : var <variable name> = value;

Example : var a=10;

- Here, the **var** is a keyword for defining the variable.
- But this **var** keyword is not compulsory.
- A variable can defined without using **var**. a=10;

Rules For Variable Naming

- The first letter of variable-name should be alphabet. Like, s1
- The variable-name can consist of digits or alphabets.
- There should be no space between the variable-name.
- No special symbols other than underscore(_) used.
- Variable-name cannot be a keyword.
- The max-length of variable-name is 31 characters.

Life Time of Variable

- **Local Variable**
  - When the variable is declared inside of the function, the lifetime of that variable is for that function only.
  - It cannot be accessed outside the function and so such variable is know as local variable.

- **Global variable**
  - When the variable is declared outside of the function, the lifetime of the variable is for all the functions.
  - We can use that variable for each function in the same page. It is called as global variable.

JavaScript Operator & Operand

- **What is Operator?**
  - The symbols which are used to perform mathematical or logical operation are known as operator.
  - Like,
    - +, -, *, /, &&, ||, !=, <, >, etc.
- **What is Operand?**
  - Operand means the variable or value which is used before and after the operator.
  - Like,
    - A+B (Here, **A** and **B** are operand and **+** is operator)

JavaScript Operator

- In JavaScript there are different types of operators available which are given below.
    1. **Arithmetic Operator**
    2. **Relational Operator**
    3. **Logical Operator**
    4. **Assignment Operator**
    5. **String Operator**

- **Arithmetic Operator:**
    - Arithmetic operators are used when any mathematical operation is to be done.

Sign	Introduction
+	Used for addition
-	Used for subtraction
*	Used for multiplication
/	Used for division
%	Used for finding the remainder. It is known as modulus operator.

- **Logical Operator:**
    - When more than two conditions are to be checked at the same time, we can use logical.

Sign	Introduction
&&	And operator. Returns true if all the condition are true
||	Or operator. Returns true if any of the conditions is true.
!	Not operator. It reverses the condition. That means if the result of the condition is true then it converts into false.

- **Relational Operator:**
    - When variables are related to some value at that the relational operator is used.

Sign	Introduction
==	This operator is used to compare the two variables or the values to check the equality. It checks only the value and not the data type.
===	This operator is used for comparing the equality of the variables or the values to check the equality. It will check value and data types also.
!=	This operator is for not equality. It will return true if the values are not equal regardless the data type.
!==	This operator is also for not equality. It will compare two values and data types too.
<	Used for less than
>	Used for greater than

>=	Used for greater than equal to
<=	Used for less than equal to

- **Assignment Operator:**
  - There are different assignment operators as given.

Sign	Introduction
=	Assigning the value a=5
*=	a*=10 is similar to a=a*10
/=	a/=10 is similar to a=a/10
+=	a+=10 is similar to a=a+10
-=	a-=10 is similar to a=a-10
%=	a%=10 is similar to a=a%10

- **String Operator:**
  - The operator + is used for the concatenation of the two strings and so + is also known as String Operator.
  - **Syntax** : document.write("Text"+varList);
  - **Purpose** : To writing output to a page.
  - **Example** : document.write("Monarch");
    document.write(10+10);
  - Note : We can put required HTML tags inside inverted commas.

Conditional Statements

- While we working with JavaScript we can use such kind of control structure to get particular output with branching.

1. **If (Statements) :**
   - To check the given condition. If condition is TURE then it execute given statements.
   - **Syntax** :
```
if(condition)
{
        Statements when condition is true;
}
```
   - **Purpose**: It will process only if the condition is true otherwise nothing.
   - **Example** :
```
<script type="text/javascript">
        var a=5;
        if(a==1)
        {
                document.write("It is 1");
        }
</script>
```

- **Design a web page using JavaScript display as given. (Simple if…)**

****

A=10

B=20

B is Greater

****

- **If…Else :**

Syntax :

```
if(condition)
{
        Statements when condition is true;
}
else
{
        Statements when condition is false;
}
```

Purpose :

- It will process first section if the condition is true otherwise it executes the second section if the condition is false.

Example :

```
<script type="text/javascript">
        var age=19
        document.write("<br>Age ="+age);
        if(age>18){
                document.write("<br>You can VOTE");
        }
        else {
                document.write("<br>You cannot VOTE");
        }
</script>
```

- **If…Else if (Ladder If) :**

Syntax :

```
if(condition)
        {   Statements when condition is true;  }
 else if(condition)
        {   Statements when condition is true;  }
 else
        { Statement when all condition is false;}
```

Purpose :

- It will process first section if first condition is true, or it will process second section if second condition is true, otherwise it executes else section if all the condition is false.

Example :

```
<script type="text/javascript">
        var no=25
        document.write("No = "+no+"<br>");
        if(no>0){
                document.write("Number is Positive");
        }else if(no<0){
                document.write("Number is Negative");
        }else{
                document.write("Number is Zero");
            }
</script>
```

- **Nested If :**

Syntax :

```
if(condition)
{
        if(condition)
        {
                When condition is true;
        }
}
else
{
                Statement when all condition is false;
}
```

Purpose :

- Nested means one **if** inside the other **if**. If first condition will be true then it will check the other **if**, both condition will be true then it will execute given statement.

Example :

```
<script type="text/javascript">
        var age=23;
        var gen="M";
        if(age>21)
        {
                if(gen=='M'){
```

```
                document.write("You can Marry");
            }
            else{
                document.write("Sorry! you can not Marry");
            }
        }
</script>
```

2. Switch Case :

- The switch statement causes a particular group of statements to be chosen from several available groups.
- The selection is tasted upon the current value of an expression that is included within the switch statement.
- Switch case is a built in multi-way decision statement.
- It tests value of given variable (or Text Expression) against a list of case values & block of statements associated with it is executed when a match is found.
- The general form of the switch statement is:
- switch(expression)
- Here expression result is an integer value or char type.
- **Syntax :**

```
switch(TxtExpression)
{
    Case TxtExp1:
        Statements;
    Case TxtExp2:
        Statements;
    Case TxtExpN:
        Statements;
    Default:
        Statements;
}
```

- **Purpose** : Switch case statement will provide us easy features of working it will provide selection based switching in group of statements. After completion of group execution we must put a break; statement to stop execution in each group. We can use either integer number or character for switching from list of group.
- **Example** :

```
<script type="text/JavaScript">
switch(“B")
{
        case “R":
                document.write("Your Choice Red")
                break;
        case “G":
                document.write("Your Choice Green")
                break;
        case “B":
                document.write("Your Choice Blue")
                break;
        default:
                document.write("Your Choice is another")
 }
</script>
```

LOOPS

- Loops are used when we want to execute a part for a block of statements for specified number of times or depending on some specified condition.
- The main thing about any of the loop is that it executed for the number of times based on the logical expression (condition) that is specified to the particular loop.
- When loop checks for the condition to execute at that time there are two types of loops available.
    - Entry Control Loop
    - Exit Control Loop

❖ **Entry Control Loop :**

- The loop which is checked before executing the statements of loop is called as Entry Controlled loop.
- This loop is first checked and if it is true then only the statements under that loop are executed and if it is not true then it transfers the control at to the end of the loop. They are as follow.
    1. **For( ) loop**
    2. **While( ) loop**

1. **For…. loop:**

Syntax :

```
for(initialization; condition; increment or decrement)
{
        statements;
}
```

Purpose : To provide a looping while condition will be true we can use for loop.

Initialization : Initialize the value of variable.

Test Condition : Test condition at here.

Increment/decrement: To increase or decrease the value.

Example :

```
<script type="text/javascript">
        var a=1;
        for(a=1;a<=10;a++)
        {
                document.write("<br>"+a);
        }
</script>
```

OUTPUT :

1 2 3 4 5 6 7 8 9 10

2. **While…. loop:**

- **Syntax :**

```
while(test condition)
{
        statements;
}
```

- **Purpose :**
  1. To provide a looping till condition will be true we can use while loop.
  2. While loop will check the condition first.
  3. If the condition will be true then it will execute the statements given in the loop, after once complete the loop, it will again and again check the condition if condition will false then it will auto exit for the loop.
- **Example :**

```
<script type="text/javascript">
        var a=1;
        while(a<=10)
        {
                document.write("<br>"+a);
                a++;
```

```
        }
</script>
```

❖ **Exit Control Loop :**

- The loop which is checked after executing the statements of loop is called as exit controlled loop.
- This loop is not checked in the began of loop but checks after execution of statement.
- While we want to execute any loop at least once at that time we can use this looping statement. They are as follow.
- **Do…while( )**

Syntax :

```
do
{
    statements;
}while(test condition);
```

Purpose :

- To provide a looping till condition will be true we can use do…while loop.
- Do…while loop will check the condition after execution of statements provided in to loop.
- If the condition will be true then it will again execute the statements given in the loop, if condition will false then it will auto exit for the loop.

Example :

```
<script type="text/javascript">
    var a=1;
    do
    {
        document.write("<br>"+a);
        a++;
    } while(a<=10)
</script>
```

✓ Break Statements

Syntax : break;
Purpose : The break command will break the loop.
Example :

```
<script type="text/javascript">
        var i=0;
        for(i=0;i<=10;i++)
        {
                if(i==3){
                        break;
                        document.write("<br>Num. is" +i);
                }
        }
</script>
```

✓ Continue Statements

Syntax : continue;
Purpose : The continue command will break the current loop and continue with the next loop.
Example :

```
<script type="text/javascript">
        for(i=0;i<=10;i++)
        {
                if(i==3){
                        continue;
                        document.write("Num. is" +i);
                }
        }
</script>
```

Dialog Boxes

- ✓ JavaScript provides us the ability to pickup users input or display small amount of text to the user by using dialog boxes.
- ✓ There are different types of dialog boxes as given.
    1. **Alert Dialog Boxes**
    2. **Conform Boxes**
    3. **Prompt Boxes**

1. Alert Dialog Boxes

Syntax : alert(“message”);

Purpose : To display some textual information on web browser window we can use alert dialog box.

Example :
```
<script type="text/javascript">
        var name="Creative Design";
        alert("Welcome to "+name);
</script>
```

2. Confirm Dialog Boxes

Syntax : confirm(“message”);

Purpose : A confirm box is used if we want the user to verify or accept something. A confirm box displays the predefined message and Ok & Cancel button.

Example :
```
<script type="text/javascript">
        if(confirm("Are you Coming?"))
        {
            document.write("Welcome");
        }
        else
        {
            document.write("No Thanks");
        }
</script>
```

3. **Prompt Dialog Boxes**

Syntax : prompt([“Text”],[default value]);

Purpose : To take input a user value we can use this prompt box.

Example :
```
<script type=“text/javascript”>
        var a,b;
        a=prompt("Enter Value for A :",0);
        b=prompt("Enter Value for B :",0);
        document.write("Value of A + B =",parseInt(a)+parseInt(b));
</script>
```

JavaScript Arrays

What is an Array?

- An array is a special variable, which can hold more than one value at a time.

How to create an Array?

Syntax : var array_name = [item1, item2, ...]; **OR** var array_name = new Array(“item1”, “item2”, ...);

Example : var cars = ["Audi", "Volvo", "BMW"]; **OR** var cars = new Array("Audi", "Volvo", "BMW");

How to Access the element of an Array?

- You access an array element by referring to the **index number**.
- Example:

```
var cars = new Array("Audi", "Volvo", "BMW");
document.write(cars[0]);
document.write(cars[1]);
document.write(cars[2]);
        OR
document.write(cars);
```

JavaScript Functions

- There are two types of functions are in JavaScript.
  - Library Functions
  - User Defined Functions

  - **Library Functions**
    - The functions whose definitions are inbuilt in JavaScript is known as JavaScript library functions.
  - **User Defined Functions**
    - Those process which are required again and again those requirement codes in specified format is known as User Defined Functions.
    - Various Kind of UDF.
      - No argument and No Return value.
      - With argument and No Return value.
      - With argument and With Return value.
    - **Syntax :**

```
Function function_name([var list])
{
        Statements…
        [return value]
}
```

1. **No argument and No Return value**
   - In this kind of UDF, there are no argument will pass to the function and no Return value to the collar function.
   - **Example** :

```
<html>
<head>
        <title></title>
        <script type="text/javascript">
                function star()
                {
                        var a;
                        document.write("<br>");
                        for(a=1;a<=70;a++){
                                document.write("*");
                        }
                        document.write("<br>");
                }
        </script>
</head>
```

```
<body>
    <script type="text/javascript">
        star();
        document.write("This is Testing");
        star();
    </script>
</body>
</html>
```

2. With argument and No Return value

- In this kind of UDF, function takes some argument and does not have any return value.
- **Example** :

```
<html>
<head>
    <title></title>
    <script type="text/javascript">
        function star(x)
        {
            var b;
            document.write("<br>");
            for(b=1;b<=x;b++){
                document.write(b);
            }
        }
    </script>
</head>
<body>
    <script type="text/javascript">
        var a;
        for(a=1;a<=5;a++){
            star(a);
        }
    </script>
</body>
</html>
```

- **Output :**

```
1
12
123
1234
12345
```

3. **With argument and with Return value**
   - In this kind of UDF, this type of functions takes some argument and does return the value.
   - When the value is returned by the function it is required that when the function is called the variable should be there to accept the returned value from the function.
   - **Example** :

```
<html>
<head>
        <title></title>
        <script type="text/javascript">
                function add(a, b)
                {
                        var c;
                        c=a+b;
                        return c;
                }
        </script>
</head>
<body>
        <script type="text/javascript">
                var value;
                value=add(5,6);
                document.write(value);
        </script>
</body>
</html>
```

- **Output :**

  11

JavaScript String Function

- **big()**
  - To display text object with BIG look
  - **Syntax :** stringObject.big();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.big());
    document.write("<br>Welcome to Our College");
</script>
```

- **small()**
  - To display text object with Small look
  - **Syntax :** stringObject.small();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.small());
    document.write("<br>Welcome to Our College");
</script>
```

- **bold()**
  - To display text object with Bold look
  - **Syntax :** stringObject.bold();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city.bold());
    document.write("<br>Welcome to Our College");
</script>
```

- **italics()**
  - To display text object with italic look
  - **Syntax :** stringObject.italics();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. italics());
    document.write("<br>Welcome to Our College");
</script>
```

- **strike()**
  - To display text object with strikeline look
  - **Syntax :** stringObject.strike();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
```

```
        document.write("<br>",city. strike());
        document.write("<br>Welcome to Our College");
    </script>
```

- **fontcolor()**
  - To display text object with color full font.
  - Here, value of color can be specified Hexa code used in HTML or the name of color.
  - **Syntax :** stringObject.fontcolor();
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. fontcolor('red'));
    document.write("<br>Welcome to Our College");
</script>
```

- **fontsize()**
  - To display text object with defined size.
  - **Syntax :** stringObject.fontsize(*integer value 1 to 7*);
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. fontsize(6));
    document.write("<br>Welcome to Our College");
</script>
```

- **link()**
  - To display link for given path/URL.
  - **Syntax :** stringObject.link(*string URL*);
  - **Example :**

```
<script type="text/javascript">
    var city="surat"
    document.write("<br><center>Hellooo…!World…");
    document.write("<br>",city. link("image/demo.jpg"));
    document.write("<br>Welcome to Our College");
</script>
```

- **length()**
  - This function will count number of character in the string and integer value.
  - **Syntax :** stringObject.length();
  - **Example :**

```
<script type="text/javascript">
    var nm="Good Morning "
    size=nm.length;
    document.write("<br>Size is ",size);
</script>
```

- **charAt()**
  - To specifies the character at the given position.
  - The counting of the position start from 0.
  - **Syntax :** stringObject.charAt(integer position);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
```

```
        document.write("<br> ",name.charAt(5));
    </script>
```

- **Output :**

  e

❖ **concat()**

- This function is used to join two or more than one string with the original string.
- **Syntax :** stringObject.concat(string1, string2,…);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write("<br> ",name.concat(", Always Creative"));
</script>
```

- **Output :**

  Be Creative, Always Creative

- **Exercise:**
  - Store Surname, Name and Father name in variable FullName using Concat function…

❖ **indexOf()**

- This function returns the position of first occurrence of the specified string value in the given string.
- If it not found string in same case then it will return -1 else it will display the position of that string.
- **Syntax :** stringObject.indexOf(string, integer from index);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.indexOf("C"));
</script>
```

- **Output :**

  3

❖ **lastIndexOf()**

- This function returns the position of last occurrence of the specified string value in the given string.
- If it not found string in same case then it will return -1 else it will display the position of that string.
- **Syntax :** stringObject.lastindexOf(string, integer from index);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.lastIndexOf("C"));
</script>
```

- **Output :**

  10

❖ **match()**

- This function is used to search the given string from string object. If matched then it will display the same name or it will display the 'null' as return.
- **Syntax :** stringObject.match(string search value);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.match("Creative"));
    if(mystr != null){
            document.write("Your String is Match");
    }
```

```
        else{
            document.write("Your String is Not Match");
        }
    </script>
```

- **Output :**

  Your String is Match

❖ replace()

- This function is used to search the given string from string object. If matched then it will display the same name or it will display the 'null' as return.
- **Syntax :** stringObject.replace(string search value);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.replace("Be","Always"));
    document.write(mystr);
</script>
```

- **Output :**

  Always Creative

❖ search()

- This function is used to search value from given string.
- **Syntax :** stringObject.search(string search value);
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.search("Creative"));
    if(mystr >= 0){
        document.write("Text Found");
    }else{
        document.write("Text Not Found");
    }
</script>
```

- **Output :**

  Text Found

❖ slice()

- This function slice string form the given integer starting point to ending point.
- Here, starting point is compulsory but ending point is default end of string.
- **Syntax :** stringObject.slice(int start, int end);
  - int start mean AFTER THAT number to
  - int end mean TILL THAT number.
- **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    document.write(name.slice(0,2));
    document.write("<br>"+name.slice(3,11));
</script>
```

- **Output :**

  Be
  Creative

- **substr()**
  - This function substr will return string form the given start position and it will stop on the length or end of string.
  - The index of first character is 0.
  - **Syntax :** stringObject.substr(start, length);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.substr(1,6));
    document.write(mystr);
</script>
```

  - **Output :**

```
Be Cre
```

- **substring()**
  - This function substring will display the text from starting index to ending index.
  - The index of first character is 0.
  - **Syntax :** stringObject.substring(start, end length);
  - **Example :**

```
<script type="text/javascript">
    var name="Be Creative";
    var mystr = (name.substring(3,8));
    document.write(mystr);
</script>
```

  - **Output :**

```
Creat
```

- **toLowerCase()**
  - This function will convert given string into lowercase.
  - **Syntax :** stringObject.toLowerCase();
  - **Example :**

```
<script type="text/javascript">
    var name="Be CreAtivE";
    document.write (name.toLowerCase());
</script>
```

  - **Output :**

```
be creative
```

- **toUpperCase()**
  - This function will convert given string into lowercase.
  - **Syntax :** stringObject.toUpperCase();
  - **Example :**

```
<script type="text/javascript">
    var name="Be CreAtivE";
    document.write (name.toUpperCase());
</script>
```

  - **Output :**

```
BE CREATIVE
```

JavaScript Math Function

- **abs()**
  - To find given number as absolute value. It will convert negative value into positive..
  - **Syntax :** Math.abs();
  - **Example :**

```
<script type="text/javascript">
    for(var a=-3;a<=3;a++)
    {
        document.write("<br>");
        document.write(Math.abs(a));
    }
</script>
```

  - **Output :**

```
3
2
1
0
1
2
3
```

- **ceil()**
  - To display nearest round up integer value.
  - **Syntax :** Math.ceil(float value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.ceil(3.5));
</script>
```

  - **Output :**

```
4
```

- **floor()**
  - To display nearest round down integer value.
  - **Syntax :** Math.ceil(float value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.floor(3.5));
</script>
```

  - **Output :**

```
3
```

- **pow()**
  - To find the power of the number for the given number and exponential.
  - **Syntax :** Math.pow(number, Exponential);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.pow(3,2));
</script>
```

  - **Output :**

```
9
```

- **random()**
  - This function will return random integer value each time the page is refreshed.
  - Here, the return type is integer.
  - **Syntax :** Math.random();
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.random());
</script>
```

  - **Output :**

```
0.8419394853003848
```

- **max()**
  - This function will return maximum value from given integer values.
  - **Syntax :** Math.max(int val1, int val2, int val3, …….);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.max(25,50,30,5,20));
</script>
```

  - **Output :**

```
50
```

- **min()**
  - This function will return minimum value from given integer values.
  - **Syntax :** Math.min(int val1, int val2, int val3, …….);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.min(25,50,30,5,20));
</script>
```

  - **Output :**

```
5
```

- **round()**
  - This function round the floating value as mathematical rules.
  - **Syntax :** Math.round(Float Value);
  - **Example :**

```
<script type="text/javascript">
    document.write(Math.round(25.579521));
</script>
```

  - **Output :**

```
26
```

JavaScript Date Function

- **date()**
  - This function returns the current date and time of system.
  - **Syntax :** Date();
  - **Example :**

```
<script type="text/javascript">
    document.write(Date());
</script>
```

  - **Output :**

    Sat Jan 16 2021 17:16:57 GMT+0530 (India Standard Time)

- **getDay()**
  - This function get only Day from the date object. In the range of 0 to 6.
  - Where, 0 = Sunday and 6= Saturday
  - **Syntax :** dateObject.getDay();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Day:"+cdate.getDay());
</script>
```

  - **Output :**

    Day : 23

- **getMonth()**
  - This function get only month from the date object. In the range of 0 to 11.
  - Where, 0 = Januray and 11= December
  - **Syntax :** dateObject.getMonth();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Month:"+cdate.getMonth());
</script>
```

  - **Output :**

    Month : 1

- **getFullYear()**
  - The getFullYear() method returns the year of a date as a four digit number:
  - **Syntax :** dateObject.getFullYear();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Year : "+cdate.getFullYear());
</script>
```

  - **Output :**

    Year : 2021

❖ getHours()

- The getHours() method returns the hours of a date as a number (0-23):
- **Syntax :** dateObject.getHours();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Hours : "+cdate.getHours());
</script>
```

- **Output :**

Hours : 15

❖ getMinutes()

- The getMinutes() method returns the minutes of a date as a number (0-59):
- **Syntax :** dateObject.getMinutes();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Minutes : "+cdate.getMinutes());
</script>
```

- **Output :**

Minutes : 15

❖ getSeconds()

- The getSeconds() method returns the Seconds of a date as a number (0-59):
- **Syntax :** dateObject.getSeconds();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Seconds : "+cdate.getSeconds());
</script>
```

- **Output :**

Seconds : 50

❖ getMilliseconds()

- The getMilliseconds() method returns the Milli Seconds of a date as a number (0-999):
- **Syntax :** dateObject.getMilliseconds();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Milli Seconds : "+cdate.getMilliseconds());
</script>
```

- **Output :**

Milli Seconds : 600

❖ getTime()

- The getTime() method returns the number of milliseconds since January 1, 1970:
- **Syntax :** dateObject.getTime();
- **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    document.write("Milliseconds : "+cdate.getTime());
</script>
```

- **Output :** **Milliseconds : 1611395321245**

- **setDate()**
  - The setDate() method sets the day of a date object (1-31):
  - **Syntax :** dateObject.setDate();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setDate(6)
    document.write("Set Date:"+cdate);
</script>
```

  - **Output :**

    Set Date : Wed Jan 06 2021 15:42:02 GMT+0530 (India Standard Time)

- **setMonth()**
  - The setMonth() method sets the month of a date object (0-11):
  - Where, 0 = Januray and 11= December
  - **Syntax :** dateObject.setMonth();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMonth(3);
    document.write("Set Month:"+cdate);
</script>
```

  - **Output :**

    Set Month : Fri Apr 23 2021 15:41:34 GMT+0530 (India Standard Time)

- **setFullYear()**
  - The setFullYear() method sets the year of a date object. In this example to 2020:
  - **Syntax :** dateObject.setFullYear();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setFullYear(2025);
    document.write("Set Year : "+cdate);
</script>
```

  - **Output :**

    Set Year : Thu Jan 23 2025 15:44:19 GMT+0530 (India Standard Time)

- **setHours()**
  - The setHours() method sets the hours of a date object (0-23):
  - **Syntax :** dateObject.setHours();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setHours(21);
    document.write("Set Hours : "+cdate);
</script>
```

  - **Output :**

    Set Hours : Sat Jan 23 2021 21:58:43 GMT+0530 (India Standard Time)

- **setMinutes()**
  - The setMinutes() method sets the minutes of a date object (0-59):
  - **Syntax :** dateObject.setMinutes();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMinutes(35);
    document.write("Set Minutes : "+cdate);
</script>
```

  - **Output :**

    Set Minutes : Sat Jan 23 2021 16:35:48 GMT+0530 (India Standard Time)

- **setSeconds()**
  - The setSeconds() method sets the seconds of a date object (0-59):
  - **Syntax :** dateObject.setSeconds();
  - **Example :**

```
<script type="text/javascript">
    var cdate=new Date();
    cdate.setMinutes(40);
    document.write("Set Seconds : "+cdate);
</script>
```

  - **Output :**

    Set Month : Sat Jan 23 2021 16:11:35 GMT+0530 (India Standard Time)

JavaScript Array Function

- **concat()**
  - This function is used to join two or more then array in a single array.
  - **Syntax :** arrayObject.concat(array1, array2,…);
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    var vege=new Array("Carrot","Radish","Tomoto");
    fruit=fruit.concat(vege);
    document.write("Mix  : ",fruit);
</script>
```

  - **Output :**

    Mix : Banana,Apple,Mango,Carrot,Radish,Tomoto

- **join()**
  - This function is used to Display all the elements of any array with specified separator.
  - **Syntax :** arrayObject.join(string separator);
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    document.write("<br>Fruits  : ",fruit.join(" * * "));
</script>
```

  - **Output :**

    Fruits : Banana * * Apple * * Mango

- **pop()**
  - This function is used to delete the item from the array. Deletion will starting from the end of the existing array element.
  - **Syntax :** arrayObject.pop( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.pop();
    document.write("<br>Fruits  : ",fruit);
</script>
```

  - **Output :**

    Fruits : Banana,Apple

- **push()**
  - This function is used to add the item into array.
  - **Syntax :** arrayObject.push( );
  - **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.push("Orange");
    document.write("<br>Fruits  : ",fruit);
</script>
```

  - **Output :**

    Fruits : Banana,Apple,Mango,Orange

- **reverse()**
  - This function is used to reverses the order of the elements in an array.

- **Syntax :** arrayObject.reverse( );
- **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    var revarray=fruit.reverse();
    document.write("<br>Fruits  : ",revarray);
</script>
```

- **Output :**

Fruits : Mango,Apple,Banana

❖ shift()

- This function is used to remove first element of the array.
- **Syntax :** arrayObject.shift( );
- **Example :**

```
<script type="text/javascript">
    var fruit=new Array("Banana","Apple","Mango");
    fruit.shift();
    document.write("<br>Fruits  : ",fruit);
</script>
```

- **Output :**

Fruits : Apple,Mango

❖ sort()

- The sort() method sorts the items of an array.
- **Syntax :** arrayObject.sort( );
- **Example :**

```
<script type="text/javascript">
    var data=new Array("7","4","2","1","5");
    document.write("<br>Number  : ",data);
    data.sort();
    document.write("<br>Sorted Number  : ",data);
</script>
```

- **Output :**

Number : 7,4,2,1,5
Sorted Number : 1,2,4,5,7

❖ length()

- This function is used to count number of element present in the array.
- **Syntax :** arrayObject.length( );
- **Example :**

```
<script type="text/javascript">
    var data=new Array("7","4","2","1","5");
    var elements=data.length;
    document.write("Number of element :",elements);
</script>
```

- **Output :**

Number of element : 5

JavaScript Events

- **onclick()**
  - This event occurs when some click action is performed on any of the html element.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1>The onclick Event</h1>
	<p>The onclick event is used to trigger a function when an element is clicked on.</p>
	<button onclick="myFunction()">Click me</button>
	<p id="demo"></p>

	<script>
		function myFunction() {
			document.getElementById("demo").innerHTML = "Hello World";
		}
	</script>
</body>
</html>
```

- **ondblclick()**
  - The ondblclick event occurs when the user double-clicks on an element.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1>The onclick Event</h1>
	<p>The onclick event is used to trigger a function when an element is clicked on.</p>
	<button ondblclick="myFunction()">Click me</button>
	<p id="demo"></p>

	<script>
		function myFunction() {
			document.getElementById("demo").innerHTML = "Hello World";
		}
	</script>
</body>
</html>
```

- **onmouseover ()**
  - The onmouseover event occurs when the mouse pointer is moved onto an element, or onto one of its children.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1 onmouseover="myFunction()" >The onmouseover Event</h1>
	<script>
		function myFunction() {
			alert("Hello World");
		}
	</script>
</body>
</html>
```

- **onmouseout ()**
  - The onmouseout event occurs when the mouse pointer is moved out of an element, or out of one of its children.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<h1 onmouseout="myFunction()" >The onmouseout Event</h1>
	<script>
		function myFunction() {
			alert("Hello World");
		}
	</script>
</body>
</html>
```

- **onkeypress ()**
  - The onkeypress event occurs when the user presses a key (on the keyboard).
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<input type="text" onkeypress="myFunction()" id="txtbox">

	<script>
		function myFunction() {
			document.getElementById("txtbox").value.toUppercase();
		}
	</script>
</body>
</html>
```

- **onkeyup()**
  - The onkeyup event occurs when the user releases a key (on the keyboard).
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<input type="text" onkeyup="myFunction()" id="txtbox">

	<script>
		function myFunction() {
			var x = document.getElementById("txtbox");
			x.value = x.value.toUpperCase();
		}
	</script>
</body>
</html>
```

- **onfocus ()**
  - The onfocus event occurs when an element gets focus.
  - The onfocus event is most often used with <input>, <select>, and <a>.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
	<input type="text" onfocus="myFunction()" id="txtbox">
```

```
<script>
    function myFunction() {
        document.getElementById("txtbox").style.background = "yellow";
    }
</script>
</body>
</html>
```

- **onblur()**
  - The onblur event occurs when an object loses focus.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body>
    <input type="text" onblur="myFunction()" id="txtbox">

    <script>
        function myFunction() {
            document.getElementById("txtbox").style.background = "red";
        }
    </script>
</body>
</html>
```

- **onload()**
  - The onload event occurs when an object has been loaded.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body onload=”myFunction()”>
    <h1>Hello World!</h1>
    <script>
        function myFunction() {
            alert("Page is loaded");
        }
    </script>
</body>
</html>
```

- **onchange()**
  - The onchange event occurs when the value of an element has been changed.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body >
    <p>Select a new car from the list.</p>
    <select id="mySelect" onchange="myFunction()">
        <option value="Audi">Audi</option>
        <option value="BMW">BMW</option>
        <option value="Mercedes">Mercedes</option>
        <option value="Volvo">Volvo</option>
    </select>
    <p id="demo"></p>
    <script>
        function myFunction() {
            var x = document.getElementById("mySelect").value;
            document.getElementById("demo").innerHTML = "You selected: " + x;
        }
```

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

- **onsubmit()**
  - The onsubmit event occurs when a form is submitted.
  - **Example :**

```
<!DOCTYPE html>
<html>
<body >
    <form onsubmit="myFunction()">
        Enter name: <input type="text" name="fname">
        <input type="submit" value="Submit">
    </form>

    <script>
        function myFunction() {
            alert("The form was submitted");
        }
    </script>
</body>
</html>
```

- **onreset()**
  - The onreset event occurs when a form is reset.
  - **Example :**

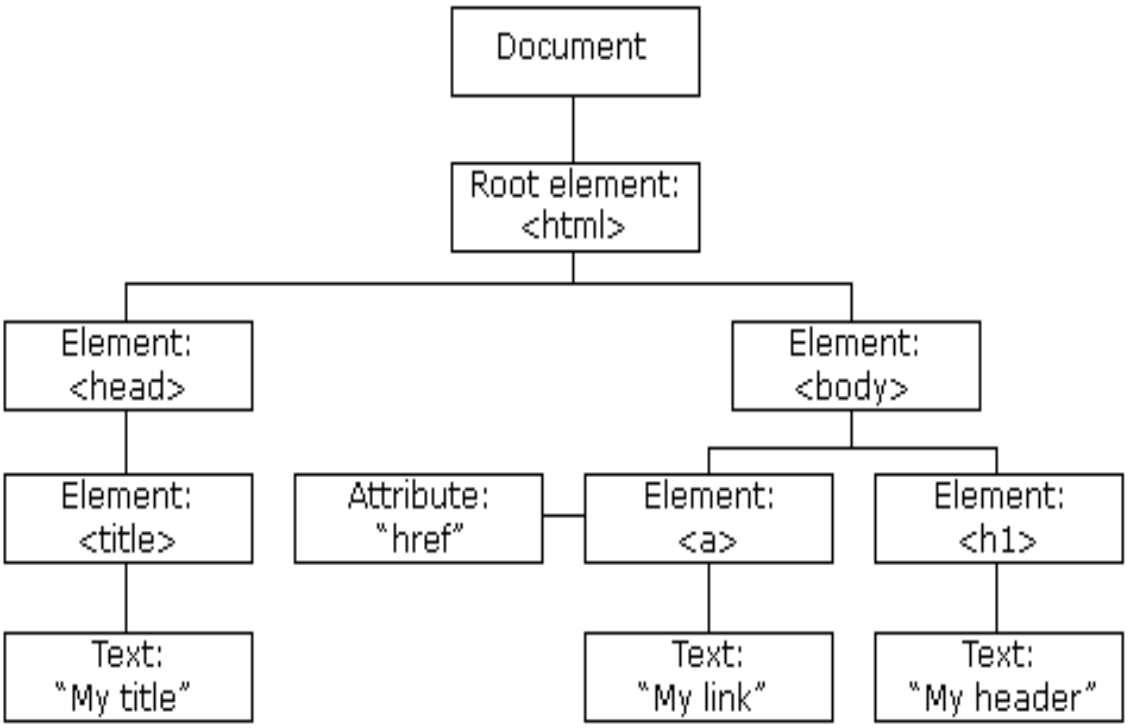
```
<!DOCTYPE html>
<html>
<body >
    <form onreset="myFunction()">
        Enter name: <input type="text" name="fname">
        <input type="reset" value="Reset">
    </form>

    <script>
    function myFunction() {
        alert("The form was Reset");
    }
    </script>
</body>
</html>
```

DOM (Document Object Model) & History Object

- When a web page is loaded, the browser creates a Document Object Model of the page.
- The HTML DOM model is constructed as a tree of Objects.

The HTML DOM Tree of Objects



- ✓ **With the object model, JavaScript gets all the power it needs to create dynamic HTML:**
  - JavaScript can change all the HTML elements in the page
  - JavaScript can change all the HTML attributes in the page
  - JavaScript can change all the CSS styles in the page
  - JavaScript can add new HTML elements and attributes
  - JavaScript can create new HTML events in the page

History Object

- ✓ The history object contains the URLs visited by the user (within a browser window).
- ✓ The history object is part of the window object and is accessed through the window.history property.
- ✓ **Property of JavaScript history object :**
  - **length:** It returns the length of the history URLs visited by user in that session.
  - **back() :** Loads the previous URL in the history list.
  - **forword()** : Loads the next URL in the history list.
  - **go() :** Loads the next URL in the history list.

What is Validation ?

- ✓ Verification that something is correct or conforms to a certain standard.
- ✓ In data collection or data entry, it is the process of ensuring that the data that are entered fall within the accepted boundaries of the application collecting the data.
- ✓ For example, if a program is collecting last names to be entered in a database, the program validates that only letters are entered and not numbers.

JavaScript Form Validation

- ✓ HTML form validation can be done by JavaScript.
- ✓ If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted:
- ✓ Example :

```
<form onsubmit="validation()">
        <table>
                <tr>
                        <td>Name :</td>
                        <td>
                                <input type="text" name="" id="name">
                        </td>
                </tr>
                <tr>
                        <td>
                                <input type="submit" name="">
                        </td>
                </tr>
        </table>
</form>
<script >
        function validation(){
                var name = document.getElementById('name').value;
                if(name == "")
                {
                        alert('Plz Enter Your Name');
                }
        }
</script>
```

JavaScript Email Validation

- ✓ Validating email is a very important point while validating an HTML form.
- ✓ An email is a string (a subset of ASCII characters) separated into two parts by @ symbol. a "personal_info" and a domain, that is personal_info@domain.
- ✓ The length of the personal_info part may be up to 64 characters long and domain name may be up to 253 characters.
- ✓ **Valid Email ID :** info@cdmi.in, example@gmail.com
- ✓ **Invalid Email ID :** info.cdmi.in [@ is not present]
- ✓ Example :

```
<form onsubmit="validation()">
	<table>
		<tr>
			<td>Name :</td>
			<td>
				<input type="text" name="" id="email">
			</td>
		</tr>
		<tr>
			<td>
				<input type="submit" name="">
			</td>
		</tr>
	</table>
</form>
<script >
	function validation(){
		var email = document.getElementById('name').value;
		var emailRegex = /^\w+[A-Z0-9._%-]+@[A-Z0-9.-]+\.[A-Z]{2,4}\b$/i;

		if(emailRegex.test(email) == false)
		{
			alert('Plz Enter Valid Email ID.');
		}
	}
</script>
```