```
49:===============:49
====================
-JavaScript is light weight,JIT(just-in-time) compiled programming language.
What is JIT?
-There are 2 types of compiling techniques
        a.JIT  [just-in-time]
        b.AOT  [Ahead-of-time]
-JIT compiles logic in browser.
-AOT compiles logic at application level.
-JavaScript supports various programming techniques and approaches
        a.functional programming
        b.Structural programming
        c.Imperative programming
        d.Object Oriented Programming etc.
-javascript is not an oop language.it supports only few features  oop.
-javascript is used
    a.client side      HTML
    b.server side      Node.js
    c.database         MongoDB
    d.Animation        ActionScript.Flash,3DS Max etc.
                       CAD
Evolution of JavaScript:
------------------------
-The first browser was "Mosaic"
       -GML,SGML    Presentation
       -ECMA Script Client side
-1990's Tim Berners Lee introduced HTML.
-1994 Netscape Communications-Developed browser
         "Netscape Communicator"
           -HTML
           -ECMA Script
           -Netscape applointed "Brendan Eich" [CERN Labratory]
           -Mocha
           -Live Script
           -Sun Micro Systems - JavaScript
-1998 Microsoft Win-98 -Free Apps
   Internet Explorer Wordstar - Buy
                     Lotus    -
                     FoxPro
                     XP-plug n play

-2004 Netscape stopped its browser,javascript->ECMA
-ECMA script=javascript = ES4-ES20[ECMA script 2020]
              ES6+
                     javascript with HTML
                        [client side]
-javascript is used client side in order to reduce burden on server.
-javascript can reduce burden on server by
        -handling validations client side
        -handling interactions client side
        -DOM manipulations
                a.adding elements
                b.removing elements
                c.rendering new data into elements
```

```
                d.update data in elements etc....
FAQ.what is the role of javascript with html?
A:DOM manipulations
                    Integrating javascripti into page
                    ---------------------------------
1.inline
2.embeded
3.external file

EX: inline
 <body>
    <h1>Click print button to print page...</h1>
   <button onclick="window.print()">Print</button>
</body>

Ex: embeded
   <script type="text/javascript">
        function PrintPage(){
            window.print();
        }
   </script>

   <button onclick="window.PrintPage()">Print</button>

Ex: Exteranal
//printing.js
   function PrintPage(){
         window.print();
 }
//home.html
  <script src="../scripts/printing.js"></script>

    <button onclick="PrintPage()">Print</button>
Note: Always use "minified" script file in "production".
       jsminifier.com

50-----------------------------50
================================
How javascript refers html elements?
1.javascript can refer html element by using DOM hierarchy.

EX:
    <script>

        function bodyload(){
            window.document.images[0].src= "../public/images/laptop.png";
            window.document.forms[0].elements[1].value="Register";
            window.document.forms[1].elements[1].value="Login"
        }
    </script>
</head>
<body onload="bodyload()">
    <img width="100" height="100" border="1">
    <div>
```

```
        <form action="">
            <h2>Register</h2>
            user Name: <input type="text"> <input type="button">
        </form>
    </div>
    <div>
        <h2>Login</h2>
        <form action="">
            Email: <input type="email"> <input type="button" name="" id="">
        </form>
    </div>
</body>
</html>
```

-if you change the position of any element in page,then every time you have to update its position in code.
-it is faster in rendering.

2.javascript can refer element by using "name".

EX:

```
 <script>

        function bodyload(){
           pic.src="../public/images/laptop.png";
           window.document.forms[0].btnRegister.value="Register";[also correct]
           frmRegister.btnRegister.value="Register";
           frmLogin.btnLogin.value="Login";
       }
 </script>
 <body onload="bodyload()">
    <img width="100" name="pic" height="100" border="1">
    <div>
        <form action="" name="frmRegister">
            <h2>Register</h2>
            user Name: <input type="text" name="txtName"> <input type="button"
name="btnRegister">
        </form>
    </div>
    <div>
        <form action="" name="frmLogin">
            <h2>Login</h2>
            Email: <input type="email" name="txtxEmail"> <input type="button"
name="btnLogin" id="">
        </form>
    </div>
</body>
```

-you can access any element directly by using its reference name.
-you can't access directly child elements.
-you have to refer both parent and child hierarchy.

3.you can refer by using "ID".

```
     document.getElementById()
```

EX:

```
    <script>

        function bodyload(){
            document.getElementById("pic").src="../public/images/mobile.png";

            document.getElementById("btnRegister").value="Register";
            document.getElementById("btnLogin").value="Login";
      }
    </script>

<body onload="bodyload()">
    <img width="100" id="pic" name="pic" height="100" border="1">
    <div>
        <form action="" name="frmRegister" >
            <h2>Register</h2>
            user Name: <input type="text" name="txtName"> <input type="button"
id="btnRegister" name="btnRegister">
        </form>
    </div>
    <div>
        <form action="" name="frmLogin">
            <h2>Login</h2>
            Email: <input type="email" name="txtxEmail"> <input type="button"
name="btnLogin" id="btnLogin">
        </form>
    </div>
</body>
```

-every element can be configured with only one ID.
-ID have conflict with CSS reference.
-ID is not required if you are accessing any direct document element.

4.javascript can refer any element by using CSS selectors.
document.querySelector()

EX:

```
    <script>

        function bodyload(){

            document.querySelector("#pic").src="../public/images/mobile.png";
            document.querySelector(".btn-register").value="Register";
            document.querySelector(".btn-login").value="Login";}
    </script>

<body onload="bodyload()">
    <img width="100" id="pic" name="pic" height="100" border="1">
    <div>
        <form action="" name="frmRegister" >
            <h2>Register</h2>
            user Name: <input type="text" name="txtName"> <input type="button"
id="btnRegister" class="btn-register" name="btnRegister">
        </form>
    </div>
```

```
    <div>
        <form action="" name="frmLogin">
          <h2>Login</h2>
            Email: <input type="email" name="txtxEmail"> <input type="button"
name="btnLogin" class="btn-login" id="btnLogin">
        </form>
    </div>
</body>
```

-query selector can apply effects only to the first element.
-However it can handle data for multiple elements.
-it can also handle styles for multiple elements.
syntax:

```
      document.querySelector("h2")          //type selector
      document.querySelector("#pic")        //id selector
      document.querySelector(".pic")        //class selector
```

5.javascript can refer all elements having common name

```
            document.getElementsByName()
```

EX:

```
  <script>
           result= document.getElementsByName("pay");
           alert("total no of payment methods :"+result.length);
        }
    </script>
</head>
<body onload="bodyload()">
    <fieldset>
        <legend>Payment Method</legend>
        <input type="radio" name="pay">Cash
        <input type="radio" name="pay">UPI
        <input type="radio" name="pay">Credit Card
    </fieldset>
</head>
```

6.javascript can referr all elements having common class name

```
       document.getElementsByClassName()
```

EX:

```
    <script>
            result= document.getElementsByClassName("form-check-input");
            alert("Radios with input class :"+result.length);
        }
    </script>
</head>
<body onload="bodyload()">
    <fieldset>
        <legend>Payment Method</legend>
        <input type="radio" class="form-check-input" name="pay">Cash
        <input type="radio" class="form-check-input" name="pay">UPI
        <input type="radio" class="form-check-input" name="pay">Credit Card
    </fieldset>
</body>
```

7.javascript can refer all elements by using tag name

```
      document.getElementsByTagName()
```

```
EX:  <script>
            result= document.getElementsByTagName("h2");
            alert("Total no of headings :"+result.length);
        }
    </script>
</head>
<body onload="bodyload()">
    <h2>Register</h2>
    <fieldset>
        <legend>Payment Method</legend>
        <input type="radio" class="form-check-input" name="pay">Cash
        <input type="radio" class="form-check-input" name="pay">UPI
        <input type="radio" class="form-check-input" name="pay">Credit Card
    </fieldset>
  <h2>Login</h2>
</body>
                      javascript output techniques
-alert()
-confirm()
-console.log(),warn(),success(),error(),debug(),info(),etc.
-innerText
-innerHTML
-outerHTML
-document.write()

51:-----------------------------------:51
----------------------------------------
JavaScript input technique:
---------------------------
-prompt()
-form input elements

prompt()
-it is like alert with input.
syntax:
     prompt("Your Message","Default value");
-prompt returns    null          on cancel
                  ""[empty]      on ok without value
                    value        on ok with value
EX:
    <script>
        function CreateClick(){
            foldername = prompt("Enter Folder Name : ","New_Folder")
            if(foldername == null){
                document.write("you canceld");
            } else if(foldername=="") {
              document.write("Please provide folder name");
            } else {
                document.querySelector("p").innerHTML+="Folder Created :
"+foldername + "<br>";
           }
        }
    </script>
```

```
  </head>
  <body>

    <button onclick="CreateClick()">Create Folder</button>
    <p></p>
  </body>
</html>

-Form Input Elements:
textbox,password,number,email,radio,checkbox,listbox,dropdown etc.

-every form element must have a reference ID.
      <input type="text" id="txtName">
      <select id="lstCities">
      <input type="checkBox" id="optStock">

-you can access element and use its properties
       document.getElementById("txtName").value;
       document.getElementById("lstCities").value;
       document.getElementById("optStock").checked==true/false;

EX:
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet"
href="../node_modules/bootstrap/dist/css/bootstrap.css">
    <link rel="stylesheet" href="">
    <script>
        function SubmitClick(){
             document.getElementById("detailsContainer").style.display="block";
             document.getElementById("registerContainer").style.display="none";

             document.getElementById("lblName").innerHTML =
document.getElementById("txtName").value;
             document.getElementById("lblPrice").innerHTML =
document.getElementById("txtPrice").value;
             document.getElementById("lblCity").innerHTML =
document.getElementById("lstCities").value;


             stock = "";
             stockCheckBox = document.getElementById("optStock");
             if(stockCheckBox.checked){
                stock = "Available"
             }else {
                stock = "Out Of Stock";
             }

             document.getElementById("lblStock").innerHTML = stock;
```

```
        }
    </script>
</head>
<body class="container-fluid">
   <div class="mt-3" id="registerContainer">
    <button class="btn btn-primary" data-bs-target="#registerProduct"
data-bs-toggle="modal">Register New Product</button>
   </div>

    <div class="modal fade" id="registerProduct">
          <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <h2>Register New Product</h2>
                    <button class="btn-close" data-bs-dismiss="modal"></button>
                </div>
                <div class="modal-body">
                    <dl>
                        <dt>Name</dt>
                        <dd><input type="text" class="form-control"
id="txtName"></dd>
                        <dt>Price</dt>
                        <dd><input type="text" class="form-control"
id="txtPrice"></dd>
                        <dt>Shiped to</dt>
                        <dd>
                            <select name="" id="lstCities" class="form-select">
                               <option value="Delhi">Delhi</option>
                               <option value="Hyderabad">Hyd</option>
                            </select>
                        </dd>
                        <dt>Stock</dt>
                        <dd class="form-switch">
                            <input type="checkbox" id="optStock"
class="form-check-input">Available
                        </dd>
                    </dl>
                </div>
                <div class="modal-footer">
                    <button class="btn btn-primary" onclick="SubmitClick()"
data-bs-dismiss="modal">submit</button>
                    <button class="btn btn-danger"
data-bs-dismiss="modal">Cancel</button>
                </div>
            </div>
          </div>
    </div>
    <div id="detailsContainer" style="display: none">
        <h2>Product Details</h2>
        <dl class="row">
            <dt class="col-3">Name</dt>
            <dd class="col-9" id="lblName"></dd>
            <dt class="col-3">Price</dt>
            <dd class="col-9" id="lblPrice"></dd>
```

```
            <dt class="col-3">Shipped to</dt>
            <dd class="col-9" id="lblCity"></dd>
            <dt class="col-3">Stock</dt>
            <dd class="col-9" id="lblstock"></dd>
        </dl>
        <button class="btn btn-info" data-bs-target="#registerProduct"
data-bs-toggle="modal">Edit</button>
    </div>
    <script src="../node_modules/jquery/dist/jquery.js"></script>
    <script
src="../node_modules/bootstrap/dist/js/bootstrap.bundle.js"></script>
</body>
</html>

FAQ:What is strict mode of JavaScript?
A-Strict mode for javascript allows to reduce code in-consistency.
  SO that developers have to follow coding standards.
   <script>
       "use strict";
      x=10;                //invalid-x is not defined
     document.write("x=" + x);
   <script>

            var x;
             x=10;

FAQ:How to write javascript for Legacy Browser?[Old version]
A-By enclosing the code in "HTML comment".
  EX: <script>
      <!--
       "use strict";
      x=10;
     document.write("x=" + x);
      -->
   <script>

FAQ:How to add javascript comments?
A-    //        single line comment
     /*   */    multi line comment
     ///        XML comment
   <!--  -->    HTML comments
52:--------------:52
====================

             JavaScript Language Basics
-Variables
-variables are storage locations in memory where you can also store a value
 and use it as a part of any expression.
-javascript allows to use variables directly if it is not in strict mode.
-In strict mode variable configuration coprises of 3 phases
   a.Declaration
   b.Assignment
   c.Initialization
-Declaring is defining scope and name for variable
```

```
     var username;
-Assigning is rendering a value into variable after declarign.
       username="ksahnihlapa";
-Initialization is rendering a value into variable while declaring.
        var username="ksahnihlapa";
Note:Declaring or initaialization of variable is mandatory if javascript is in
     strict mode.

-Variables in javascript can be declared by using
  a.var
  b.let
  c.const

var:
-it defines function scope variable.
-you can declare in any block of a function and access from any another block in
function.
    function f1(){
     var x;
      x=10;
     if(x==10) {
       var y=20;
     }
   document.write("x= "+x+"<br>"+"y= "+y);
   }
   f1();

-var allows declaring,assigning and initialization.
-var allows shadowing.
FAQ:What is shadowing?
-it is the process of redeclaring same name identifier with in the scope.
    EX:function f1(){
     var x;
      x=10;
     if(x==10) {
       var y=20;
       var y=40;  //shadowing
     }
   }
-var allows hoisting.
FAQ:What is hoisting?
A:It is a compiling technique,where compiler can find declaration of variable
before using it.
  Hence you can use a variable before declaring.
EX:  "use strict";
      function f1(){
      x=10;
     document.write("x= "+x);
      var x;  //hoisting
   }
   f1();

let:
-it is used to define block scope variable.
```

-it is accessible only block where it is declared or to its inner blocks.
-it allows declaring,assigning and initialization.
-it will not allow shadowing.
-it will not allow hoisting.
EX:

```
   function f1(){
     let x;
      x=10;
     if(x==10) {
       let y=20;
       document.write("x= "+x+"<br>"+"y= "+y);
     }
   document.write("x= "+x+"<br>"+"y= "+y);  //scope of y is not there
  }
   f1();
```

const:
-it is used to define block scope variable.
-it will allow only initialization.
-no declaring.
-no assigning.
-no shadowing.
-no hoisting.

FAQ:Why we need a const?
A:const is required to initialize memory.
at the time of loading application or component memory is initailized with some default value.

NOTE.if initialization is missing then by default value will be "undefined"

```
   var x;
  document.write("X="+x);   //x=undefined
  const x;        //invalid
  x=10;         //invalid
  const x=10;  //valid
```

FAQ:Can't const change its value?
A-Dynamically yes.

.Global scope for variable:
-you can declare a variable in module scope.so that it is global and accessible to all functions in module.

EX:

```
"use strict";
   function f1(){
    const x=10; //only allowed
    x=20;  //not allowed
   document.write("x= "+x);
  }
   f1();
```

FAQ>Can we declare a global variable inside function?
A:yes,By using browser "window" object.

```
       function f1(){
```

```
         window.y=20;
       }
       function f2(){
           document.write("y"+y);
      }
53:--------------:53
====================
Variable Naming:
-variable name must start with alphabet or underscore.
   var username;
   var _username;
   var 2021Sales;   //invalid
   var Sales2021;   //valid
   var Sales 2021;  //invalid
   var Sales.2021;  //invalid
   var Sales$2021;  //invalid
   var Sales_2021;  //valid
-The special character "_" is used to indicate that variable not implemented.
 it requires further implementation.
-Variable name can't exceed more than 255 chars.
-Don't use language keywords for variable name.
    var class="UI";  //invalid
    var object="";  //invalid
-Always variable must speek what it is.
   var emp=new Employee();        //not good
   var prod=new Product();       //not good
   var x=1000;                  //not good

   var employee = new Employee();
   var product  = new Product();
   var temporaryEmployee = new TemporaryEmployee();
                     Data Types
                     ----------
-It defines data structure.
-Data type determines the size and type of data.
-Data types are classified into 2 types
     1.primitive data types
     2.non-primitive data types
Primitive data types:
---------------------
-primitive types are immutable types.
-Their structure will not change.
-They hava fixed range for value.
-Value range can't change.
-They use memory stack.
-Stack uses "LIFO".
-JavaScript primitive types are
  a.number
  b.string
  c.boolean
  d.null
  e.undefined
NOTE:w
```

String Type:

-String is a literal with group of chars enclosed in
    a.single quote ''
    b.double quote ""
    c.Back Tick   ``
-Single and DOuble quote are used to configure inner and outer string
combination.
   var link="Home;
-Back tick allows a string with embedded expression "${}".
    ${}   =>Data Binding Expression
EX:

```
      <head>
            <title></title>
            <script>
                 function SubmitClick(){
                      var username=prompt("Enter Your Name");
                      var age=document.getElementById("1stAge").value;
                      var msg1="hello !"+" "+userName+" "+"You will be"+"
"+(parseInt(age)+1)+" "+"next year.<br>";
                      var msg2=`Hello ! ${userName} you will be
${parseInt(age)+1} next year.`;
                      document.write(msg1);
                      document.write(msg1);
                 }
           </script>
      </head>
      <body>
         <dl>
            <dt>Select Age</dt>
            <dd>
                <select id="1stAge">
                   <option value="15">15</option>
                   <option value="15">20</option>
                   <option value="15">5</option>
                </select>
            </dd>
         </dl>
         <button onclick="SubmitClick()">Submit</button>
     </body>
```

Escape Sequence Issues:

-special chars in a string can escape printing.
-you have to print the non-printable chars by using "\".
      \n   new line in console

  new line in HTML
EX:

```
 <script>
   var imagePath="\"D:\\Images\\assets\\mobile.jpg\"";
   document.write(imagePath);
 </script>
```

String Handling in JavaScript:
-JavaScript provides a set of methods and properties to format and manipulate string.

String Formatting Methods:
-bold()
-italic()
-sup()
-sub()
-fontcolor() fontcolor('colorName')
-fontsize() fontsize('4')
-toUpperCase()
-toLowerCase()

String Property:
-length :it returns the total no of chars.[count of chars]

syntax:
```
  var msg="Welcome to javascript";
  msg.bold().italics().fontcolor('green');
  msg.toUpperCase();
```

EX:
```
<!DOCTYPE html>
<html>
  <head>
    <title>54</title>
    <script>
        function VerifyUser(){
            var userName=document.getElementById("txtName").value;
            var msg=document.getElementById("msg");
            if(userName.length>=4 && userName.length<=10){
               msg.innerHTML = "User Name
Verified".bold().fontcolor('green').italics();
            } else {
                msg.innerHTML = "Invalid -  User Name 4 to 10 chars
Only".italics().fontcolor('red');
            }
        }
        function ChangeCase(){
            var userName=document.getElementById("txtName").value;
            document.getElementById('txtName').value=userName.toUpperCase();
        }
    </script>
  </head>
  <body>
    <h2>Register</h2>
    <dl>
      <dt>User Name</dt>
      <dd><input type="text" id="txtName" placeholder="Block Letters"
onkeyup="VerifyUser()" onblur="ChangeCase()"/></dd>
      <dd id="msg"></dd>
    </dl>
```

```
    <button onclick="VerifyUser()">Submit</button>
  </body>
</html>

Events:
-onkeyup           :actions to perform when key is released.
-onblur            :action to perform when control is blured[lost focus]
-onclick           :action to perform when clicked
-onload            :action to perform on page or image load

EX:Formating a String dynamically  using string format functions
->
<!DOCTYPE html>
<html lang="en">
<head>
    <script>
        function ChangeColor(){
            alert(document.querySelector("select").value);
            document.querySelector("p").innerHTML="Welcome to
javascript".fontcolor(document.querySelector("select").value);
        }
    </script>
</head>
<body>
    <fieldset>
        <legend>Choose Effects</legend>
        <dl>
            <dt>Font Color</dt>
            <dd>
                <select onchange="ChangeColor()">
                    <option >Red</option>
                    <option >Green</option>
                    <option >White</option>
                    <option >Yellow</option>
                </select>
            </dd>
        </dl>
    </fieldset>
    <p></p>
</body>
</html>

EX:formating strings by using styles
->
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>String Demo</title>
    <script>
        function ChangeColor(){
              document.querySelector("p").style.color =
```

```
document.querySelector("select").value;
        }

    </script>
</head>
<body>
    <fieldset>
        <legend>Choose Effects</legend>
        <dl>
            <dt>Font Color</dt>
            <dd>
                <select onchange="ChangeColor()">
                    <option >Red</option>
                    <option >Green</option>
                    <option >White</option>
                    <option >Yellow</option>
                </select>
            </dd>
        </dl>
    </fieldset>
    <p>Welcome Home</p>
</body>
</html>

EX:Dynamically applying CSS Class
->
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet"
href="../node_modules/bootstrap/dist/css/bootstrap.css">
    <script>
       function ChangeTheme(){
         var themeCheckBox = document.getElementById("theme");
         var formContainer = document.getElementById("formContainer");
         if(themeCheckBox.checked){
            formContainer.className = "dark-theme";
            document.querySelector("button").className="btn btn-dark w-100";
         }else {
            formContainer.className = "light-theme";
            document.querySelector("button").className="btn btn-primary w-100";
         }
       }

    </script>
    <style>
        .form{
            border: 2px solid black;
            padding: 20px;
            width: 250px;
            margin-top: 20px;
        }
        .dark-theme{
            background-color: black;
```

```
            color: white;
            border: 2px solid black;
            padding: 20px;
            width: 250px;
            margin-top: 20px;
        }
        .light-theme {
            background-color: white;
            color: black;
            border: 2px solid black;
            padding: 20px;
            width: 250px;
            margin-top: 20px;


        }
    </style>
</head>
<body>
    <div class="container-fluid">
        <div class="form" id="formContainer">
            <div class="form-switch">
                <input type="checkbox" onchange="ChangeTheme()"
class="form-check-input" id="theme">Dark Theme
            </div>
            <h2>Register User</h2>
            <dl>
                <dt>User Name</dt>
                <dd><input type="text"></dd>
                <dt>Password</dt>
                <dd><input type="password"></dd>
                <dt>Email</dt>
                <dd><input type="email"></dd>
            </dl>
            <button class="btn w-100">Login</button>
        </div>

    </div>
</body>
</html>

Note: You can define styles and classes dynamically to any element.
  document.querySelector("p").style.color="green";
  document.querySelector("button").className = "btn btn-primary";

                  String Manipulations
                  --------------------
-charAt()         :it returns the character at specified index.
-charCodeAt()     :it returns the character ASCII code present at specified
index.
-slice()          :it can extracts chars between specified index
-substr()         :it can return the specified number of chars from given index.
-substring()      :it can return the chars from specific index in any direction.
```

EX:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <script>
        function ChangeCase(){
            var userName=document.getElementById("txtName").value;
            var firstChar=userName.charAt(0);
            var restChars= userName.substring(1);
            var sentence = firstChar.toUpperCase()+restChars.toLowerCase();
            document.getElementById("txtName").value=sentence;
        }
    </script>
</head>
<body>
    <fieldset>
        <legend>Test</legend>
        <div>
            <input type="text" id="txtName" onblur="ChangeCase()">
        </div>

    </fieldset>
</body>
</html>
```

FAQ:What is charCodeAt()?

->charAt() returns char at specified index

charCodeAt() returns its character code as per UTF standards

A=65,Z=90

EX:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <script>
     function VerifyName(){
        var userName=document.getElementById("txtName").value;
        if(userName.charCodeAt(0)>=65 && userName.charCodeAt(0)<=90) {
            document.querySelector("p").innerHTML  = "";
        }else {
            document.querySelector("p").innerHTML = "Name must start with
uppercase letter ".fontcolor('red');
        }
    }
   </script>
</head>
<body>
    <fieldset>
        <legend>User Name</legend>
        <input type="text" size="40" id="txtName" placeholder="name must start
with uppercase" onkeyup="VerifyName()">
        <p></p>
    </fieldset>
```

```
</body>
</html>

55:--------------------:55
=========================
FAQ:What is difference between slice(),substr() & substring()?
-slice()   :it can read the character between specified start and end index.
            if end is not defined,then it will read upto end of string.
            End index must be greater than start index.

          string.slice(0,7)  //0 to 7
          string.slice(7)    //7 to end
          string.slice(7,0)  //invalid

      Syntax: slice(startIndex,endIndex);
-substr()  :it can read specified number of chars from given index number
    syntax:   substr(startIndex,countOfChars);
              string.substr(7,3);       //from 7 index it will read 3 chars

-substring()   :it can read from specified index to any direction
      Syntax:substring(startIndex,endIndex);
   Note:endIndex can be less than start index.
       string.substring(0,7)       //0 to 7 chars
       string.substring(7)         //7 to end
       string.substring(7,0)       //7 to start[0]

IndexOf() and lastIndexOf()
---------------------------
-These are the functions which can find any character in a string and return its
index position.
-If character not found then it returns -1
-lastIndexOf() will return the last occurance index.
    Welcome  -indexOf("e")       //1
    Welcome  -lastIndexOf("e")   //6

EX:
      <script>
            function VerifyEmail(){
                var email=document.getElementById("txtEmail").value;
                var msg=document.querySelector("h2");
                var atPos = email.indexOf("@");
                if(atPos<2){
                    msg.innerHTML = "Invalid Email";
                }else{
                    msg.innerHTML = "Email Verified";
                }
            }
        </script>
    </head>
    <body>
        <fieldset>
            <legend>Your Email</legend>
            <input type="text" id="txtEmail"><button
onclick="VerifyEmail()">Submit</button>
```

```
        </fieldset>
        <h2></h2>
    </body>

startsWith() and endsWith():
----------------------------
-These are the functions used to verify the starting and ending chars in a
string.
-These functions return boolean true when string is starting and ending with
specified chars.
Syntax:
   string.startsWith(chars);
   string.endsWith(chars);

EX:
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Document</title>
    <link rel="stylesheet"
href="/node_modules/bootstrap/dist/css/bootstrap.css">

    <script>
        function VerifyCard(){
            var cardnumber = document.getElementById("txtCard").value;
            var cardImage = document.getElementById("cardImg");
            if(cardnumber.startsWith("4444")){
                cardImg.src = "/public/images/master.png";
            } else if(cardnumber.startsWith("5555")) {
                cardImg.src = "/public/images/visa.png";
            } else {
                cardImg.src="";
                cardImg.alt="N/A";
            }
        }
        function VerifyEmail(){
            var email = document.getElementById("txtEmail").value;

            if(email.endsWith("gmail.com")){
                document.getElementById("msg").innerHTML = "Email
Verified".fontcolor("green");
            } else {
                document.getElementById("msg").innerHTML = "Invalid
Email".fontcolor("red");
            }
        }
    </script>
</head>
<body class="container-fluid">
    <fieldset>
        <legend>VERIFY DETAILS</legend>
       <div>
        <h3>Card Number</h3>
        <div class="input-group">
```

```
            <input type="text" id="txtCard"
onkeyup="VerifyCard()"class="form-control">
            <img class="input-group-text" class="img-fluid" id="cardImg"
width="100" height="50">
        </div>
        <h3>Your Gmail</h3>
        <div >
           <input type="text" id="txtEmail" class="form-control"
onkeyup="VerifyEmail()">
           <div id="msg">

           </div>
        </div>
       </div>
    </fieldset>
</body>
</html>
```

-match()   :it is used to verify and compare given string with any regular expression.
               :it returns true if string format is as per regular expression.
         In JavaScript regular expression is enclose in "//".

```
syntax:  var regExp = /pattern/;
         var string= "";

      if(string.match(regExp)){

      }
```

EX:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet"
href="/node_modules/bootstrap/dist/css/bootstrap.css">
    <style>
        meter{
            height: 30px;
        }
    </style>
    <script>
        function VerifyPassword(){
            var password = document.getElementById("txtPwd").value;
            var regExp = /(?=.*[A-Z])\w{4,15}/;
            var msg=document.getElementById("msg");
            var grade=document.getElementById("grade");

            function showGrade(min,max,value,low,high){
                    grade.min = min;
                    grade.max=max;
                    grade.value=value;
                    grade.low=low;
                    grade.high=high;
```

```
            }


           if(password==""){
            msg.innerHTML = "password required";
            msg.className = "text-danger";
           } else {
            if(password.match(regExp)) {
                 msg.innerHTML  = "strong Password";
                 msg.className = "text-success";
                 showGrade(1,100,100,0,0);
            } else {
                if(password.length<4) {
                  msg.innerHTML = "Poor Password";
                  msg.className = "text-danger";
                  showGrade(1,100,20,60,80);
                } else {
                  msg.innerHTML  = "Weak Password";
                  msg.className = "text-warning";
                  showGrade(1,100,70,40,80);
               }
           }
           }
        }
        function VerifyMobile(){
            var mobile = document.getElementById("txtMobile").value;
            var mobileExpression = /\+91\d{10}/;
            var mobileError = document.getElementById("mobileError");
            if(mobile.match(mobileExpression)){
                mobileError.innerHTML = "";

            }else{
                mobileError.innerHTML = "Invalid Mobile";
                mobileError.className = "text-danger";
            }
        }
    </script>
</head>
<body class="container-fluid">
    <h2>Verify Password</h2>
    <input type="text" onkeyup="VerifyPassword()" class="form-control"
id="txtPwd">
    <div id="msg"></div>
    <div>
        <meter id="grade" min="1" max="100" class="w-100"></meter>
    </div>
    <h2>Verify Mobile</h2>
    <input type="text" onkeyup="VerifyMobile()" id="txtMobile"
class="form-control">
    <div id="mobileError"></div>
</body>
</html>
```

trim() and split():

```
-------------------
-trim() is used to remove the leading spaces in a string.
-split() is used to split the string at specified delimeter and return an array.

EX:
<!DOCTYPE html>
<html lang="en">
<head>
    <script>
        function VerifyPassword(){
            var pwd = document.getElementById("txtPwd").value;
            if(pwd.trim()=="admin"){
                document.querySelector("h2").innerHTML = "Verified";
            } else {
                document.querySelector("h2").innerHTML = "Invalid password";
            }
        }
    </script>
</head>
<body>
    Your Password :
    <input type="text" id="txtPwd"> <button
onclick="VerifyPassword()">Submit</button>
    <h2></h2>
</body>
</html>

EX:split():
<script>
    var mobileNumbers = "7873719196,6345789445,7894561230";
    var result = mobileNumbers.split(',');
    document.write(result[2]);
</script>

56:====================:56
---------------------------
                                                Boolean Type:
-Boolean type are used in decision making.
-Boolean type in javasscript can handle 2 values.
     a.true
     b.false
-Boolean conditions javascript can use 1 for true and 0 for false.
EX:
<script>
    var name="samsung tv";
    var stock = true;
    var status;
    if(stock==true) {
        status = "Available";
    } else {
        status = "Out of Stock";
    }
    document.write(`Name=${name}<br>Stock=${status}`);
</script>
```

Undefined Type

-It specifies that value is not supplied into a reference.
-Undefined type is verified by using "undefined" keyword.
-It indicates that value is not defined into given reference.
EX:

```
<script>
    var Name="samsung tv";
    var Price;
    if(Price==undefined) {
        document.write(`Name=${Name}`);
    } else {
        document.write(`Name=${Name}<br>Price=${Price}`);
    }

</script>
```

FAQ:What is difference between undefined and not-defined?
Ans: undefined :reference is there but value not defined.
not-defined :reference is not defined.
EX:

```
<script>                                x=undefined
    var x;                              y=not defined
    document.write(`X=${x}<br>Y=${y}`);

</script>
```

Null Type

-It is an exception type.
-Exception occurs at run time.
-Null indicates that value is not supplied during run time.
-You can verify null type by using "null" type.
EX:

```
<script>
   var username = prompt("Enter Name");
   if(username==null){
    document.write("you canceled");
   } else if (username==""){
    document.write("Name cant be empty");
   }
    else {
   document.write(` Hello ! ${username}`);
   }
</script>
```

Summary:
Primitive Types
-number :numeric values
-string :literals
-boolean :true/false
-null :no value at run time
-undefined :no value at compile time

JavaScript Non-Primitime Types

-They are mutable types.
-They can change the structure according to state and situation.
-There is no fixed range for value.
-Value range varies according to memory available.
-They are store in heap memory.
-JavaScript non-primitive types are
  a.array
  b.object
  c.Map

Array Type

-Array means organizing in order and accessing in random order.
-Stack means organizing in order and accessing the last one first.[LIFO]
-Queue means organizing in order and accessing the first one first.[FIFO]

What is the purpose of Array?
-Array is used in computer programming to reduce overhead and complexity.