V.E.S. Institute of Technology, Collector Colony,

Chembur, Mumbai

Department of M.C.A

**INDEX**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr.** | **Contents** |  | **Date Of** | **Date Of** | **Marks** | **Sign** |
| **No** |  |  | **Performance** | **Submission** |  |  |
|  |  | |  |  |  |  |
| 1 | Program to design a calculator along with data | |  |  |  |  |
|  | validation in JavaScript using variable, | |  |  |  |  |
|  | functions and control structures. | |  |  |  |  |
|  |  | |  |  |  |  |
| 2 | Program in JavaScript on manipulation of | |  |  |  |  |
|  | strings. (Palindrome, length calculation , | |  |  |  |  |
|  | concatenation, comparison , | find position of |  |  |  |  |
|  | substring, index of character , | uppercase(using |  |  |  |  |
|  | unfocus), lowercase etc.) |  |  |  |  |  |
|  |  | |  |  |  |  |
| 3 | A simple stopwatch script, with Start, Stop, and | |  |  |  |  |
|  | Reset options. Displays seconds, and minutes, | |  |  |  |  |
|  | and can be set to stop and execute a function | |  |  |  |  |
|  | when the time of the stopwatch reaches to a | |  |  |  |  |
|  | certain value. |  |  |  |  |  |
|  |  | |  |  |  |  |
| 4 | a) Write down a menu driven program for | |  |  |  |  |
|  | sorting, integers and strings. User should be | |  |  |  |  |
|  | prompted to enter number of strings/integers to | |  |  |  |  |
|  | be sorted.( Javascript sort function should not | |  |  |  |  |
|  | be used) |  |  |  |  |  |
|  | b) Write a JavaScript code to create an | |  |  |  |  |
|  | Employee object with attributes (Name, | |  |  |  |  |
|  | Region, Sales, Comm). All values should be | |  |  |  |  |
|  | accepted from user through text box except | |  |  |  |  |
|  | Comm. Calculate Sales Commissions as per the | |  |  |  |  |
|  | total sales value is entered. Accepted total sales | |  |  |  |  |
|  |  |  |  |  |  |  |

in Rs. In a box, after clicking on a “Find

Commission” button all values should be

displayed in “Alert Box”.

Criteria for calculating Commission Sales >50

and sales then comm. = θ

Sales >500 then sales <=500 then Comm = 10

* of sales Sales > 500 then comm = 50 + 8% of sales. Commission value should also be stored in Employee object.

1. Program to design a registration form with fields like first name, last name, address, e-mail id, password and confirm password, write appropriate validation checks for each field. If the textbox has been left empty, popup an Alert indicating which textbox has been left empty.

When alert’s OK button is clicked on, set focus to that specific textbox.

1. a) Program to design a drop down list with categories of items, on selecting any category its item list should be displayed in another list, in sorted order. Show an appropriate message after selecting both. (e.g for Category Grocery it should display items like sugar, oil, salt , pulses etc.)
   1. Design the following TextArea counter.
2. a) Program using Ajax for making a server request.
   1. Write a program of AJAX with PHP for displaying the multiplication table of the number entered.

1. Program using Ajax for loading html scriptlets from server using Ajax events.
2. Program for making an Ajax style file upload.
3. Design a web page in Bootstrap using different fonts, content drop down supported heading styles, different column layout Image styles, alignment of dropdowns to left, right and centre.
4. Pricing table is a way for representing pricing schemes, features and categories of products in a website. (Mostly pricing table is used for showing different kinds of plans and packages by companies. Users can easily compare

Prices between different packages from lists of pricing columns and choose best plan that suit for their business.) Design a web page to show pricing table with 4 colour theme.

1. Design a Joomla template that can be used for ecommerce websites which is easy to customize and adapt to the most crucial requirements.
2. Design one page Joomla template with features like On Screen Responsive Content Slider, Top Fixed Menu, Custom 404 error page, maximum module positions and Google Map Integration.
3. Mini Project

**Practical: 1**

**JavaScript**

Javascript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

* JavaScript is a lightweight, interpreted programming language.
* Designed for creating network-centric applications.
* Complementary to and integrated with Java.
* Complementary to and integrated with HTML.
* Open and cross-platform

**JavaScript Variables**

Like many other programming languages, JavaScript has variables. Variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container. Before using a variable in a JavaScript program, you must declare it.

Variables are declared with the **var** keyword as follows.

Storing a value in a variable is called variable initialization. We can do variable initialization at the time of variable creation or at a later point in time when you need that variable.

JavaScript is untyped language. This means that a JavaScript variable can hold a value of any data type. Unlike many other languages, we don't have to tell JavaScript during variable declaration what type of value the variable will hold. The value type of a variable can change during the execution of a program and JavaScript takes care of it automatically.

**JavaScript Variable Names**

JavaScript variable names should not start with a numeral (0-9). They must begin with a letter or an underscore character. For example, 123test is

an invalid variable name but \_123test is a valid one.

JavaScript variable names are case-sensitive.

example, Name and name are two different variables.

For

**JavaScript – Functions**

A function is a group of reusable code which can be called anywhere in your program. This eliminates the need of writing the same code again and again. It helps programmers in writing modular codes. Functions allow a programmer to divide a big program into a number of small and manageable functions.

Like any other advanced programming language, JavaScript also supports all the features necessary to write modular code using functions.

**Function Definition**

Before we use a function, we need to define it. The most common way to define a function in JavaScript is by using the function keyword, followed by a unique function name, a list of parameters (that might be empty), and a statement block surrounded by curly braces.

**Syntax**

function *name*(*parameter1, parameter2, parameter3*)

{

*code to be executed*

}

**Control Structures**

1. **Conditional statements The if Statement**

Use the **if** statement to specify a block of JavaScript code to be executed if a condition is true.

**Syntax:**

if (*condition*) {

*block of code to be executed if the condition is true*

}

**The else Statement**

Use the else statement to specify a block of code to be executed if the condition is false.

**Syntax:**

if (*condition*) {

*block of code to be executed if the condition is true* } else {

*block of code to be executed if the condition is false*

}

**The else if Statement**

Use the **else if** statement to specify a new condition if the first condition is false.

**Syntax:**

if (*condition1*) {

*block of code to be executed if condition1 is true*

} else if (*condition2*) {

*block of code to be executed if the condition1 is false and condition2 is true*

} else {

*block of code to be executed if the condition1 is false and condition2 is false*

}

**The JavaScript Switch Statement**

Use the switch statement to select one of many blocks of code to be executed.

**Syntax**

switch(*expression*) {

case *n*:

*code block*

break;

case *n*:

*code block*

break;

default:

*code block*

}

This is how it works:

The switch expression is evaluated once.

The value of the expression is compared with the values of each case.

If there is a match, the associated block of code is executed.

1. **Looping statements The For Loop**

The for loop is often the tool you will use when you want to create a

loop.

The for loop has the following syntax:

for (*statement 1*; *statement 2*; *statement 3*) { *code block to be executed*

}

**Statement 1** is executed before the loop (the code block) starts. **Statement 2** defines the condition for running the loop (the code

block).

**Statement 3** is executed each time after the loop (the code block) hasbeen executed.

**The While Loop**

The while loop loops through a block of code as long as a specified condition is true.

**Syntax**

while (*condition*) {

*code block to be executed*

}

**The Do/While Loop**

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

**Syntax**

do {

*code block to be executed*

}

while (*condition*);

**The Break Statement**

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch() statement.

The break statement can also be used to jump out of a loop.

The break statement breaks the loop and continues executing the code after the loop (if any)

**The Continue Statement**

The **continue statement** breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

**JavaScript Labels**

To label JavaScript statements you precede the statements with a label name and a colon:

label:statements

The break and the continue statements are the only JavaScript statements that can "jump out of" a code block.

**Syntax:**

break *labelname*;

continue *labelname*;

The continue statement (with or without a label reference) can only be used to skip one loop iteration.

The break statement, without a label reference, can only be used to jump out of a loop or a switch.

With a label reference, the break statement can be used to jump out of any code block:

**Write a program to design a calculator along with data validations on JavaScript using variables, functions and control structures.**

**Code:**

<html>

<head>

<title>Java Script</title>

<script>

function addition()

{

var a=Number(document.form.t1.value);

var b=Number(document.form.t2.value);

if(!Number.isInteger(a))

{ alert ("Enter Number Only"); } if(!Number.isInteger(b))

{ alert ("Enter Number Only"); } var c=a+b; document.form.t3.value=c;

}

function subtraction()

{

var a=Number(document.form.t1.value); var b=Number(document.form.t2.value); if(!Number.isInteger(a))

{ alert ("Enter Number Only"); } if(!Number.isInteger(b))

{ alert ("Enter Number Only"); }

var c=a-b;

document.form.t3.value=c;

}

function multiplication()

{

var a=Number(document.form.t1.value);

var b=Number(document.form.t2.value);

if(!Number.isInteger(a))

{ alert ("Enter Number Only"); } if(!Number.isInteger(b))

{ alert ("Enter Number Only"); } var c=a\*b; document.form.t3.value=c;

}

function division()

{

var a=Number(document.form.t1.value); var b=Number(document.form.t2.value); if(!Number.isInteger(a))

{ alert ("Enter Number Only"); } if(!Number.isInteger(b))

{ alert ("Enter Number Only"); }

var c=a/b;

document.form.t3.value=c;

}

function modulus()

{

var a=Number(document.form.t1.value);

var b=Number(document.form.t2.value);

if(!Number.isInteger(a))

{ alert ("Enter Number Only"); } if(!Number.isInteger(b))

{ alert ("Enter Number Only"); }

var c=a%b;

document.form.t3.value=c;

}

</script>

<head>

<body>

<form name="form" method="POST">

Enter first number:<input type="text" name="t1"/><br/><br/>

Enter second number:<input type="text" name="t2"/><br/><br/>

Answer:<input type="text" name="t3"/><br/><br/>

<br>

<input type="button" onclick="addition()" name="b1" value="Add"/>

<input type="button" onclick="subtraction()" name="b2" value="Subtract"/> <input type="button" onclick="multiplication()" name="b3" value="Multiply"/>

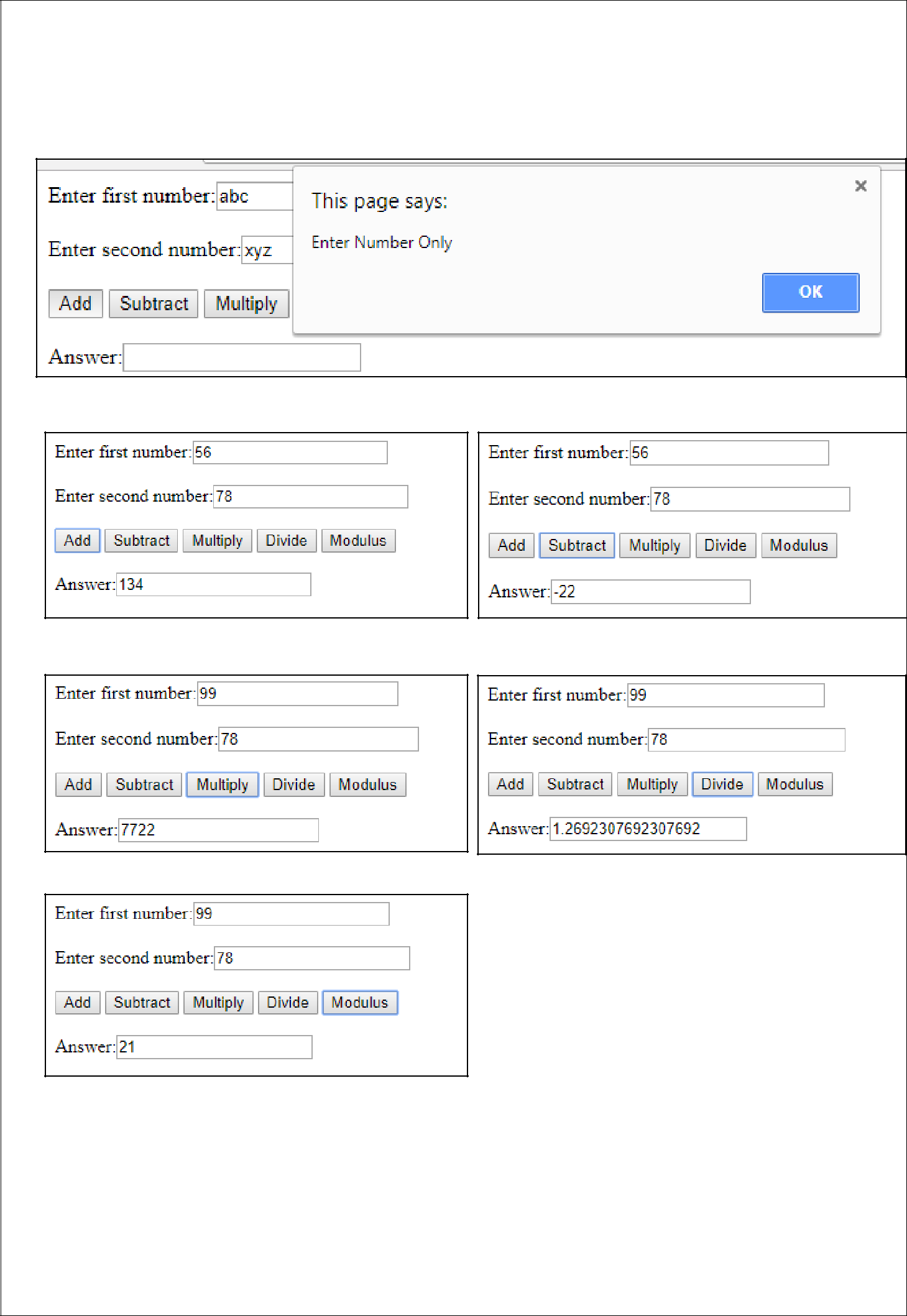
<input type="button" onclick="division()" name="b4" value="Divide"/>

<input type="button" onclick="modulus()" name="b5" value="Modulus"/>

</form>

</body>

</html>

**Output:**

**Practical: 2**

**JavaScript Strings**

A string can be any text inside double or single quotes. String indexes are zero-based: The first character is in position 0, the second in 1, and so on.

**Properties and methods:**

Strings have their own built-in variables and functions, also known as properties and methods.

**length**

A string’s **length** property keeps track of how many characters it has.

**toLowerCase**

A string’s **toLowerCase** method returns a copy of the string with its letters converted to *lowercase*. Numbers, symbols, and other characters are not affected.

**toUpperCase**

A string’s **toUpperCase** method returns a copy of the string with its letters converted to *capitals*. Numbers, symbols, and other characters are not affected.

|  |  |  |  |
| --- | --- | --- | --- |
| **indexOf()** |  |  |  |
| The | **indexOf()** method returns the index of (the position | | of) |
| the **first** | occurrence of a specified text in a string | |  |
| **lastIndexOf()** | |  |  |
| The | **lastIndexOf()** method returns the index of the **last** occurrence | | of |
| a specified text in a string | | |  |

**search()**

The **search()** method searches a string for a specified value and returns the position of the match

**replace()**

The **replace()** method replaces a specified value with another value in a string:

**concat()**

**concat()** joins two or more strings. The **concat()** method can be usedinstead of the plus operator.

**charAt()**

The **charAt()** method returns the character at a specified index (position) in a string

**split()**

A string can be converted to an array with the **split()** method

**reverse()**

The **reverse()** method reverses the order of the elements in an array.

**join()**

The **join()** method joins the elements of an array into a string, and returns the string. The elements will be separated by a specified separator. The default separator is comma (,).

**substr()**

Extracts the characters from a string, beginning at a specified start position, and through the specified number of character

**substring()**

Extracts the characters from a string, between two specified indices

**trim()**

Removes whitespace from both ends of a string

**Nan()**

Not a number value

**isInteger()**

This method determines whether the passed value is an integer.

1. **WAP in JavaScript on manipulation of strings (palindrome, length calculation, concatenation, comparison, find position of substring, index of character, uppercase, lower case etc).**

**Code:**

<html>

<head>

<title>Javascript</title>

<script>

function concatenation()

{

var a=document.form.txt1.value;

var b=document.form.txt2.value;

var res=a.concat(b);

document.form.txt3.value=res;

}

function length1()

{

var a=document.form.txt1.value;

var res=a.length;

document.form.txt3.value=res;

}

function comparison()

{

var a=document.form.txt1.value;

var b=document.form.txt2.value;

if(a==b)

document.form.txt3.value="Strings are equal";

else

document.form.txt3.value="Strings are not equal";

}

function lcase()

{

var a=document.form.txt1.value;

var res=a.toLowerCase();

document.form.txt3.value=res;

}

function ucase()

{

var a=document.form.txt1.value;

var res=a.toUpperCase();

document.form.txt3.value=res;

}

function palindromee()

{

var a=document.form.txt1.value;

var res=a.split("");

var t=res.join();

var rev=res.reverse();

var fin=rev.join();

if(t==fin)

document.form.txt3.value="It is a palindrome";

else

document.form.txt3.value="It is not a palindrome";

}

</script>

</head>

<body>

<form name="form" id="fm"><br>

Enter first string:<input type="text" name="txt1"><br><br>

Enter second string:<input type="text" name="txt2"><br><br> String manipulation:<input type="text" name="txt3"><br><br> <input type="button" name="btn1" value="Palindrome" onclick="palindromee()">

<input type="button" name="btn2" value="Length" onclick="length1()"> <input type="button" name="btn3" value="Concatenation" onclick="concatenation()">

<input type="button" name="btn4" value="Comparison" onclick="comparison()">

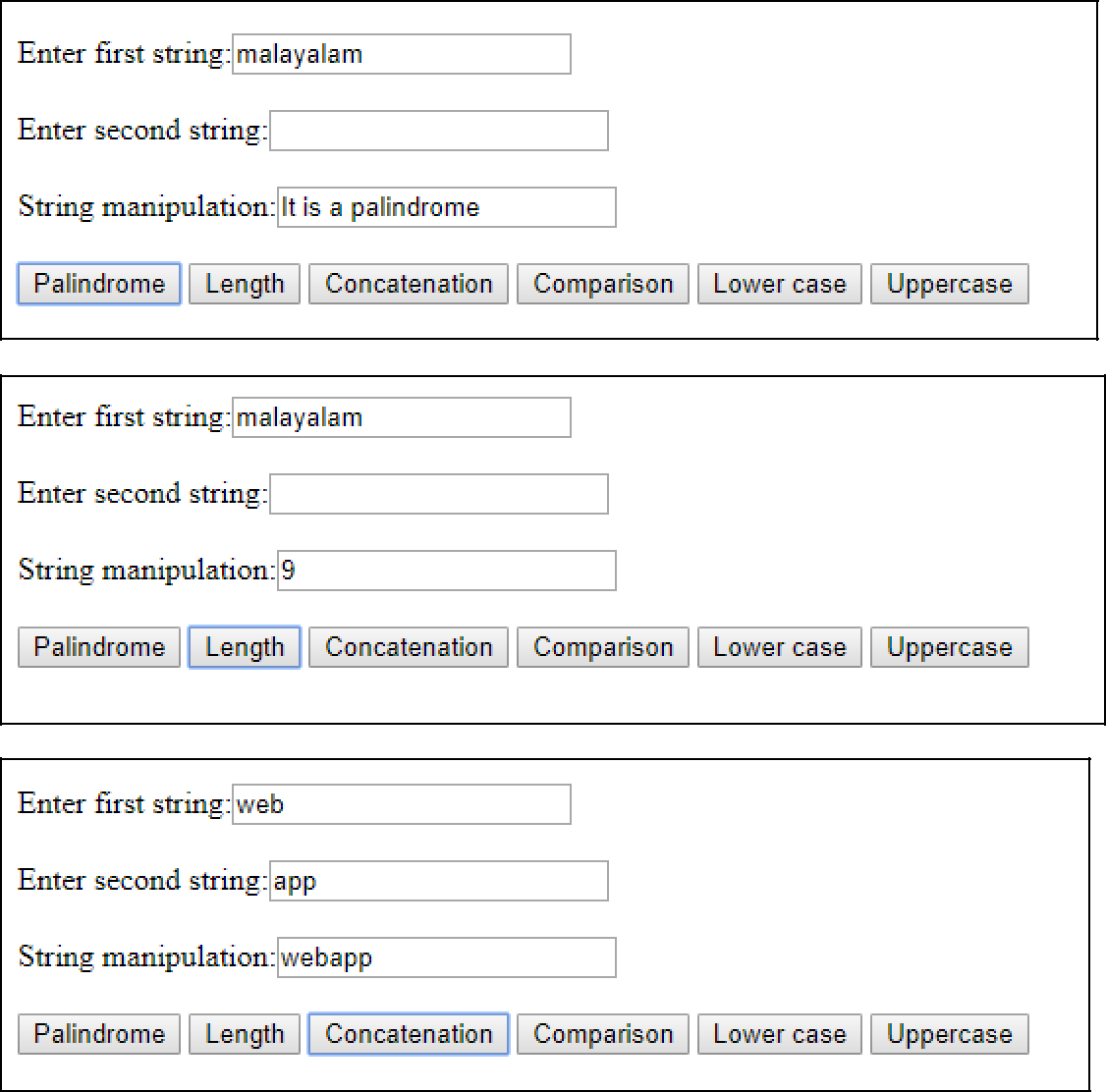
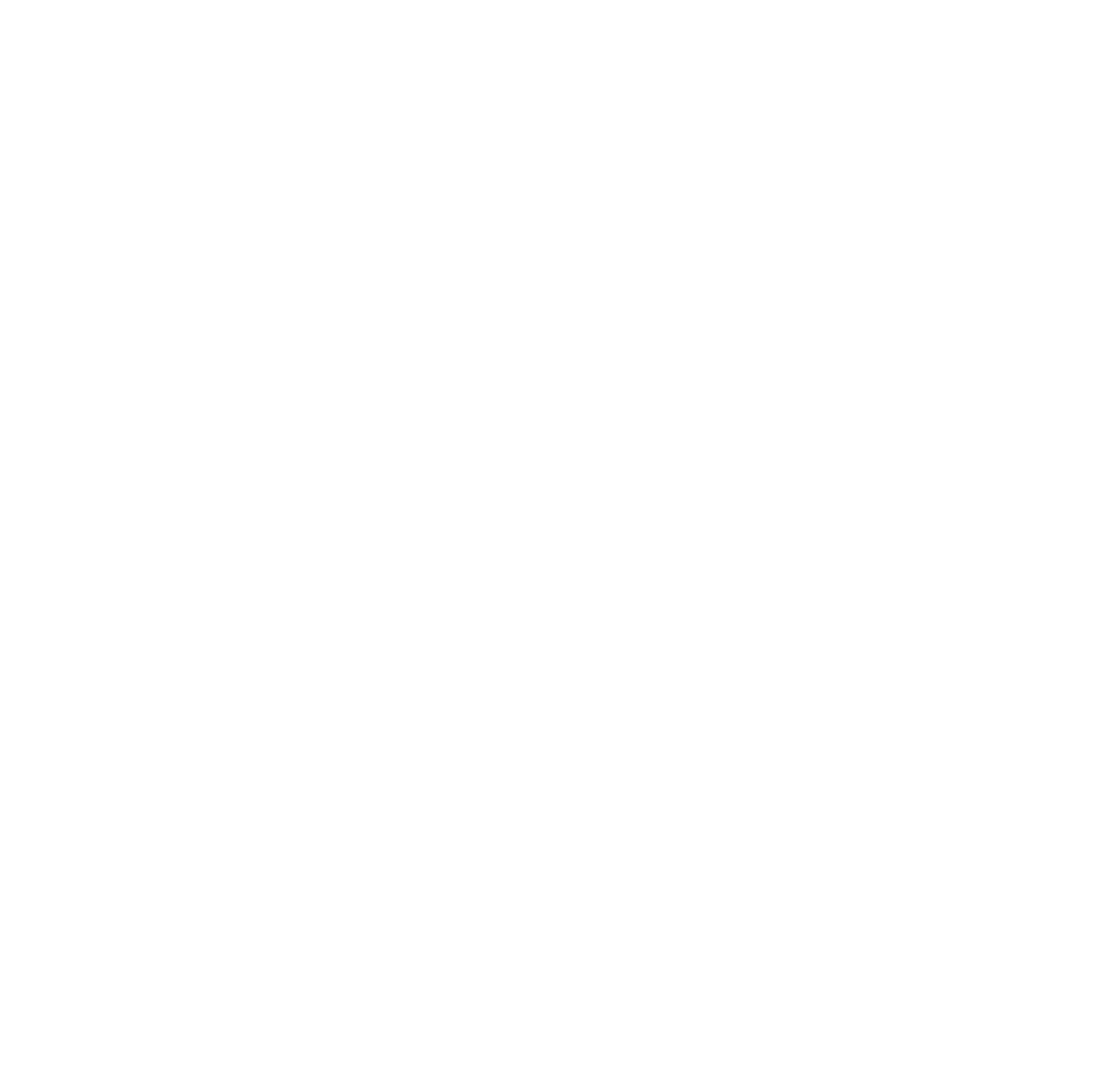
<input type="button" name="btn5" value="Lower case" onclick="lcase()">

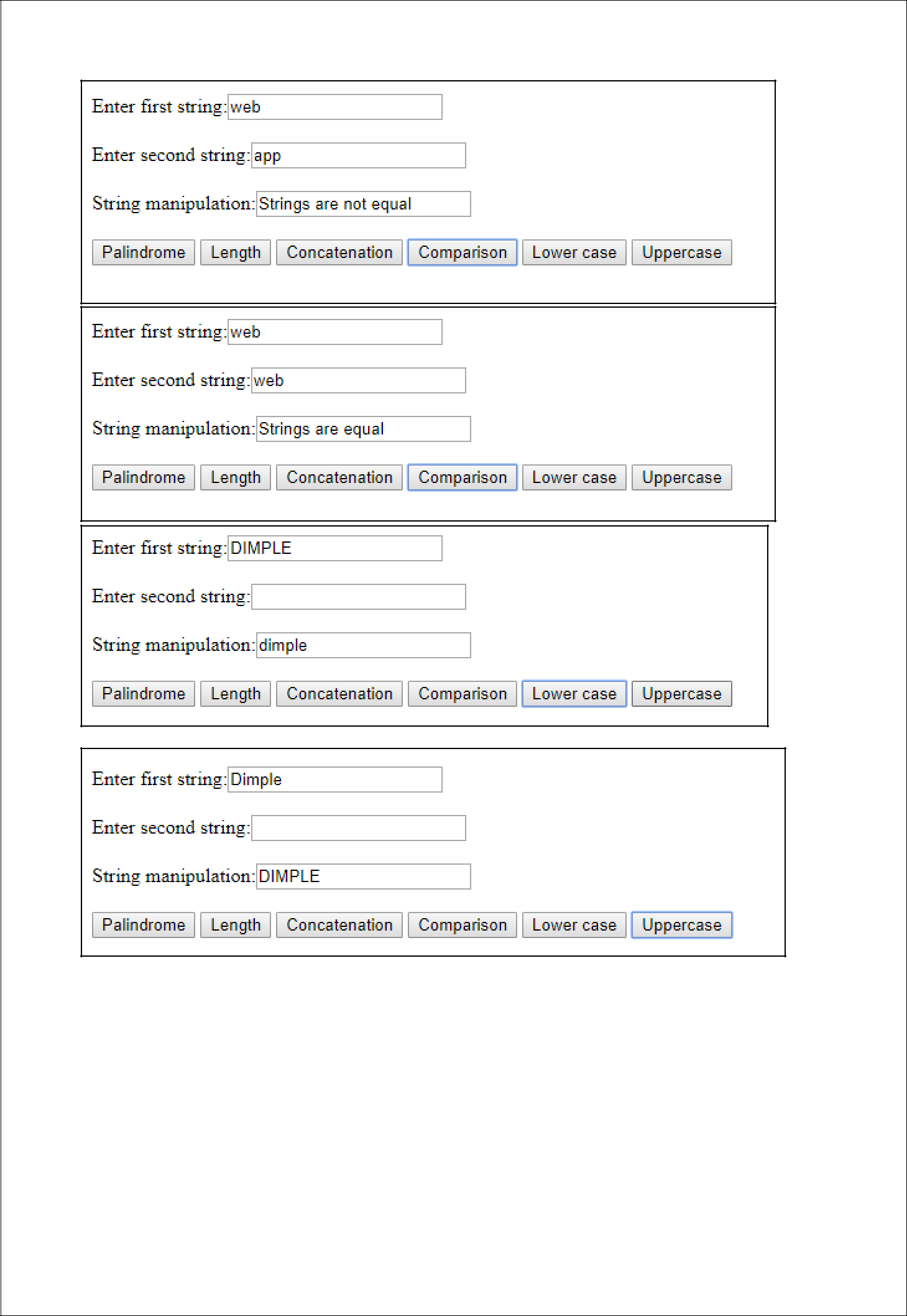
<input type="button" name="btn6" value="Uppercase" onclick="ucase()">

</body>

</html>

**Output:**





**Practical: 3**

**The setTimeout() Method:**

The setTimeout() method calls a function or evaluates an expression after a specified number of milliseconds.

**Syntax:** window.setTimeout(function, milliseconds);

* 1000 ms = 1 second.
* The function is only executed once. If you need to repeat execution, use the setInterval() method.
* The window.setTimeout() method can be written without the window prefix.
* The first parameter is a function to be executed.
* The second parameter indicates the number of milliseconds before execution.

**Parameter Values:**

**Parameter** *function milliseconds param1,param2, ...*

**Description**

Required. The function that will be executed

Optional. The number of milliseconds to wait before executing

the code. If omitted, the value 0 is used

Optional. Additional parameters to pass to the *function*

**Example**

Click a button. Wait 3 seconds, and the page will alert "Hello":

<button onclick="setTimeout(myFunction, 3000)">Try it</button> <script>

function myFunction() {

alert('Hello');

}

</script>

**The clearTimeout() Method:**

* The clearTimeout() method clears a timer set with the setTimeout() method.
* The ID value returned by setTimeout() is used as the parameter for the clearTimeout() method.
* To be able to use the clearTimeout() method, you must use a global variable when creating the timeout method:
* myVar = setTimeout("javascript function", milliseconds);
* Then, if the function has not already been executed, you will be able to stop the execution by calling the clearTimeout() method.

**Syntax:** clearTimeout(id\_of\_settimeout)

|  |  |  |
| --- | --- | --- |
| **Parameter Values** |  |  |
| **Parameter** | | **Description** |
| id\_of\_settimeout | | Required. The ID value of the timer returned by the |
|  |  | setTimeout() method |

**Write a program to design a stopwatch using JavaScript with start, stop & reset buttons. Display Hours, Minutes and Seconds.**

**Code:**

<html>

<head>

<title>Stop Watch</title>

</head>

<body>

<div id="t1">00:00:00</div>

<br>

<button onclick="startTimer()">Start</button> <button onclick="stopTimer()">Stop</button> <button onclick="resetTimer()">Reset</button> <script>

var seconds=0, minutes=0, hours=0, t;

function add() {

seconds++;

if (seconds >= 60) {

seconds = 0;

minutes++;

if (minutes >= 60) {

minutes = 0;

hours++;

}

}

t1.textContent = (hours ? (hours > 9 ? hours : "0" + hours) : "00") +

":" + (minutes ? (minutes > 9 ? minutes : "0" + minutes) : "00") + ":" +

(seconds > 9 ? seconds : "0" + seconds);

startTimer();

}

function startTimer() {

t = setTimeout(add, 1000);

}

function stopTimer() {

clearTimeout(t);

}

function resetTimer() {

t1.textContent = "00:00:00";

stopTimer();

seconds = 0; minutes = 0; hours = 0;

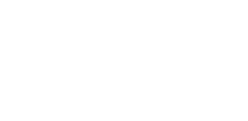
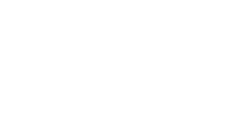
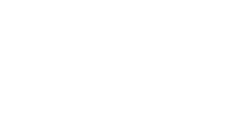
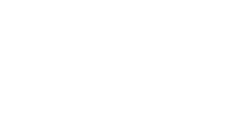
}

</script>

</body>

</html>

**Output:**



**Practical: 4**

**Array:**

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

JavaScript arrays are used to store multiple values in a single variable.Using an array literal is the easiest way to create a JavaScript Array.

**Syntax**:

var *array\_name* = [*item1*, *item2*, ...];

**Example:**

var cars = ["Saab", "Volvo", "BMW"];

Spaces and line breaks are not important. A declaration can span multiple lines:

var cars = [

"Saab",

"Volvo",

"BMW"

];

**Array Length Property:**

The length property sets or returns the number of elements in an array.

**Syntax:**

Return the length of array:

*array*.length

Set the length of array:

*array*.length = *number*

**Example:**

var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.length;

**String localeCompare() Method**

The localeCompare() method compares two strings in the current locale.

The locale is based on the language settings of the browser.

The localeCompare() method returns a number indicating whether the *string* comes before, after or is equal as

the *compareString* in **sort order**.

**Syntax:**

*string.localeCompare(compareString)*

**Example:**

var str1 = "ab";

var str2 = "cd";

var n = str1.localeCompare(str2);

1. **Design a menu driven program for sorting integers and strings.**

**Code:**

**sorting.html**

<html>

<head><title>Select Sorting</title></head> <body>

<form name="myform">

<h3>Select the type to sort..</h3>

<b>

<input type="radio" name="sort" id="num">Sort Numbers<br> <input type="radio" name="sort" id="str">Sort Strings<br> </b><br>

<input type="button" value="Submit" onclick="sub()">

</form>

<script>

function sub()

{

var inputs=document.myform.sort.value; if(document.getElementById("num").checked)

window.open("sortnum.html");

else if(document.getElementById("str").checked)

window.open("sortstr.html");

}

</script>

</body>

</html>

**sortnum.html**

<html>

<head>

<title>Sorting Numbers</title>

</head>

<body>

<form name="form1">

Enter 1<sup>st</sup> number:<input type="number" name="txt1"

id="t1"><br><br>

Enter 2<sup>nd</sup> number:<input type="number" name="txt2"

id="t2"><br><br>

Enter 3<sup>rd</sup> number:<input type="number" name="txt3"

id="t3"><br><br>

Enter 4<sup>th</sup> number:<input type="number" name="txt4"

id="t4"><br><br>

Enter 5<sup>th</sup> number:<input type="number" name="txt5"

id="t5"><br><br>

<input type="button" name="submit" onclick="sortt()" value="Sort">

<p id="demo"></p>

<script>

function sortt()

{

var a=parseInt(document.form1.t1.value);

var b=parseInt(document.form1.t2.value);

var c=parseInt(document.form1.t3.value);

var d=parseInt(document.form1.t4.value);

var e=parseInt(document.form1.t5.value);

var arr=[a,b,c,d,e];

var i,temp;

for (i=0;i<arr.length-1;i++)

{

for(j=0;j<=arr.length-i-1;j++)

{

if(arr[j]>arr[j+1])

{

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

document.getElementById("demo").innerHTML=arr;

}

</script>

</form>

</html>

**sortstr.html**

<html>

<head>

<title>Sorting Strings</title>

</head>

<body>

<form name="form1">

Enter 1<sup>st</sup> string:<input type="text" name="txt1" id="t1"><br><br> Enter 2<sup>nd</sup> string:<input type="text" name="txt2" id="t2"><br><br> Enter 3<sup>rd</sup> string:<input type="text" name="txt3" id="t3"><br><br> Enter 4<sup>th</sup> string:<input type="text" name="txt4" id="t4"><br><br> Enter 5<sup>th</sup> string:<input type="text" name="txt5" id="t5"><br><br> <input type="button" name="submit" onclick="sortt()" value="Sort"> <p id="demo"></p>

<script>

function sortt()

{

var tx1=document.form1.t1.value;

var tx2=document.form1.t2.value;

var tx3=document.form1.t3.value;

var tx4=document.form1.t4.value;

var tx5=document.form1.t5.value;

var arr=[tx1,tx2,tx3,tx4,tx5];

var i,temp;

for (i=0;i<arr.length-1;i++)

{

for(j=0;j<=arr.length-i-1;j++)

{

if(arr[j].localeCompare(arr[j+1])==1)

{

temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

document.getElementById("demo").innerHTML=arr;

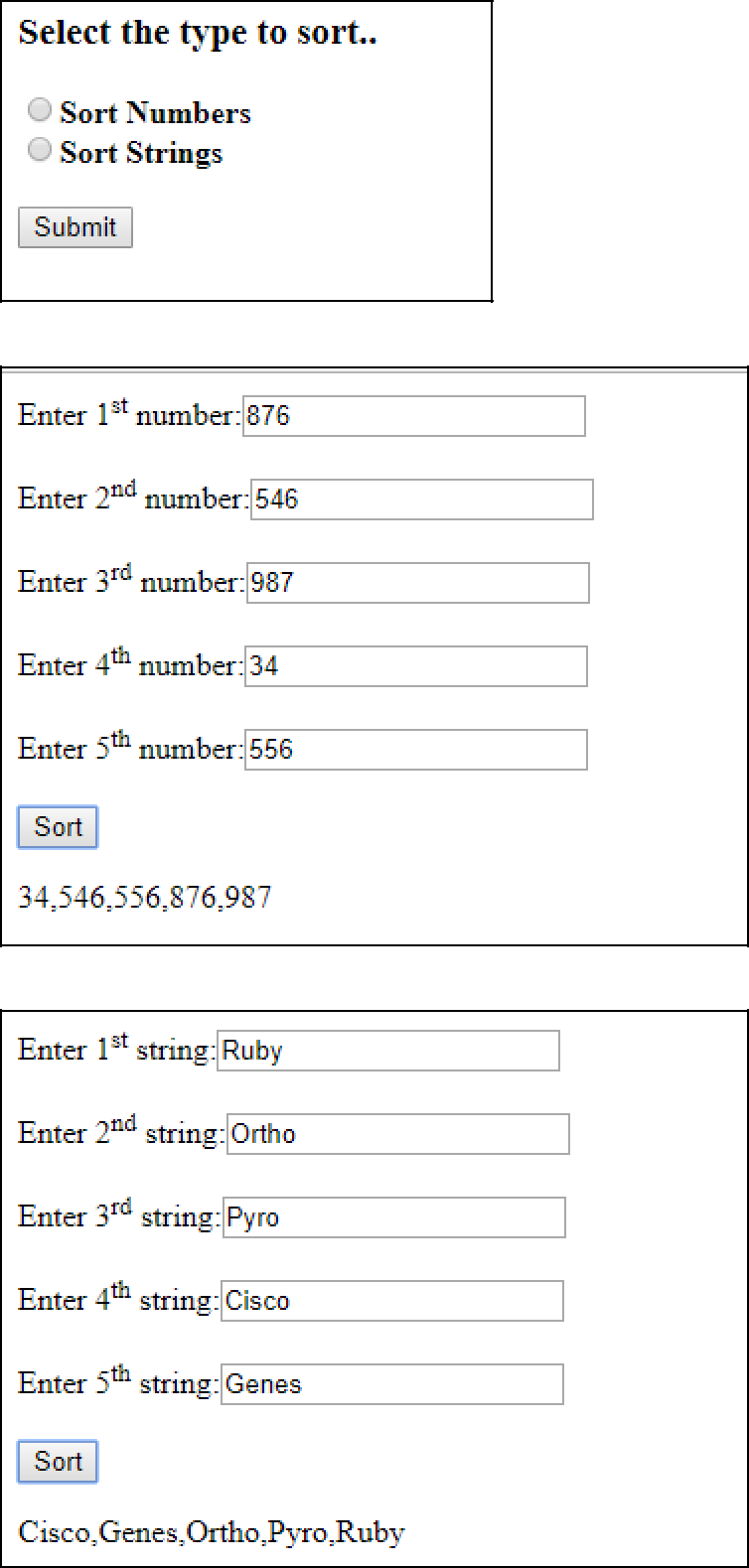
}

</script>

</form>

</html>

**Output:**



**JavaScript Control Structure:**

Control structure actually controls the flow of execution of a program.

Following are the several control structure supported by javascript.

* if … else
* switch case
* do while loop
* while loop
* for loop

**If … else:**

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

**Syntax**

if (expression)

{

Statement(s) to be executed if expression is true

}

**Example**

<script type="text/javascript">

<!--

var age = 20;

if( age > 18 ){

document.write("<b>Qualifies for driving</b>");

}

//-->

</script>

1. Write a JavaScript code to create an Employee object with attributes (Name, Region, Sales, Comm). All values should be accepted from user through text box except Comm. Calculate Sales Commissions as per the total sales value is entered. Accepted total sales in Rs. In a box, after clicking on a “Find Commission” button all values should be displayed in “Alert Box”. Criteria for calculating Commission Sales &gt;50 and sales then comm. = θ Sales &gt;500 then sales &lt;=500 then Comm = 10 % of sales

Sales &gt; 500 then comm = 50 + 8% of sales. Commission value should also be stored in Employee object.

**Program:**

**sales.html**

<html>

<head>

<title>Sales Commission</title>

</head>

<form method="GET" name="frm">

<h3>JavaScript Program to find Sales Commission</h3>

<table border="1">

<tr>

<td align = "right">Enter Name: </td>

<td><input type="text" name="name" placeholder="Name"> </td> </tr> <tr>

<td align = "right">Enter Region: </td>

<td><input type="text" name="region" placeholder="Region"> </td> </tr> <tr>

<td align = "right">Enter Sales: </td>

<td><input type="number" name="sales" placeholder="Sales"> </td>

</tr>

<tr>

<td></td>

<td><input type="button" name="submit" value="Calculate"

onclick="calculate()"></td>

</tr>

</table>

</form>

<script>

function calculate(){

var b=document.frm.name.value;

var d=document.frm.region.value;

var a=document.frm.sales.value;

var cal;

if(a<500)

{

alert("Name: " + b + "\n" + "Region: "

* d + "\n" + "Sales: " + a + "\n" + "No Commision allowed on Rs. " + a);

}

else if(a>=500 && a<=5000)

{

cal = parseInt(a\*10)/100;

alert("Name: " + b + "\n" + "Region: "

* d + "\n" + "Sales: " + a + "\n" + "10% Commision on sales is Rs. " + a + " is " + cal + " Rs.");

}

else if(a>5000)

{

cal = parseInt(a\*8)/100+50;

alert("Name: " + b + "\n" + "Region: "

* d + "\n" + "Sales: " + a + "\n" + "50Rs. + 8% Commision on sales is Rs. "
* a + " is " + cal + " Rs.");

} else

{

alert("Enter proper details");

}

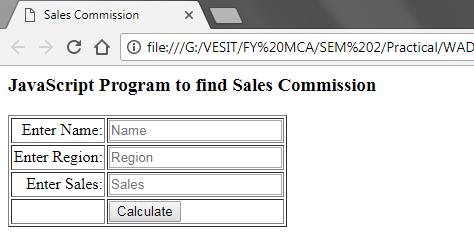
}

</script>

</html>

OUTPUT:

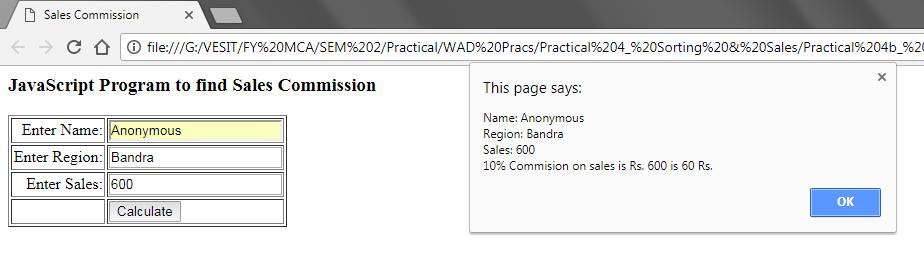
Home Page:



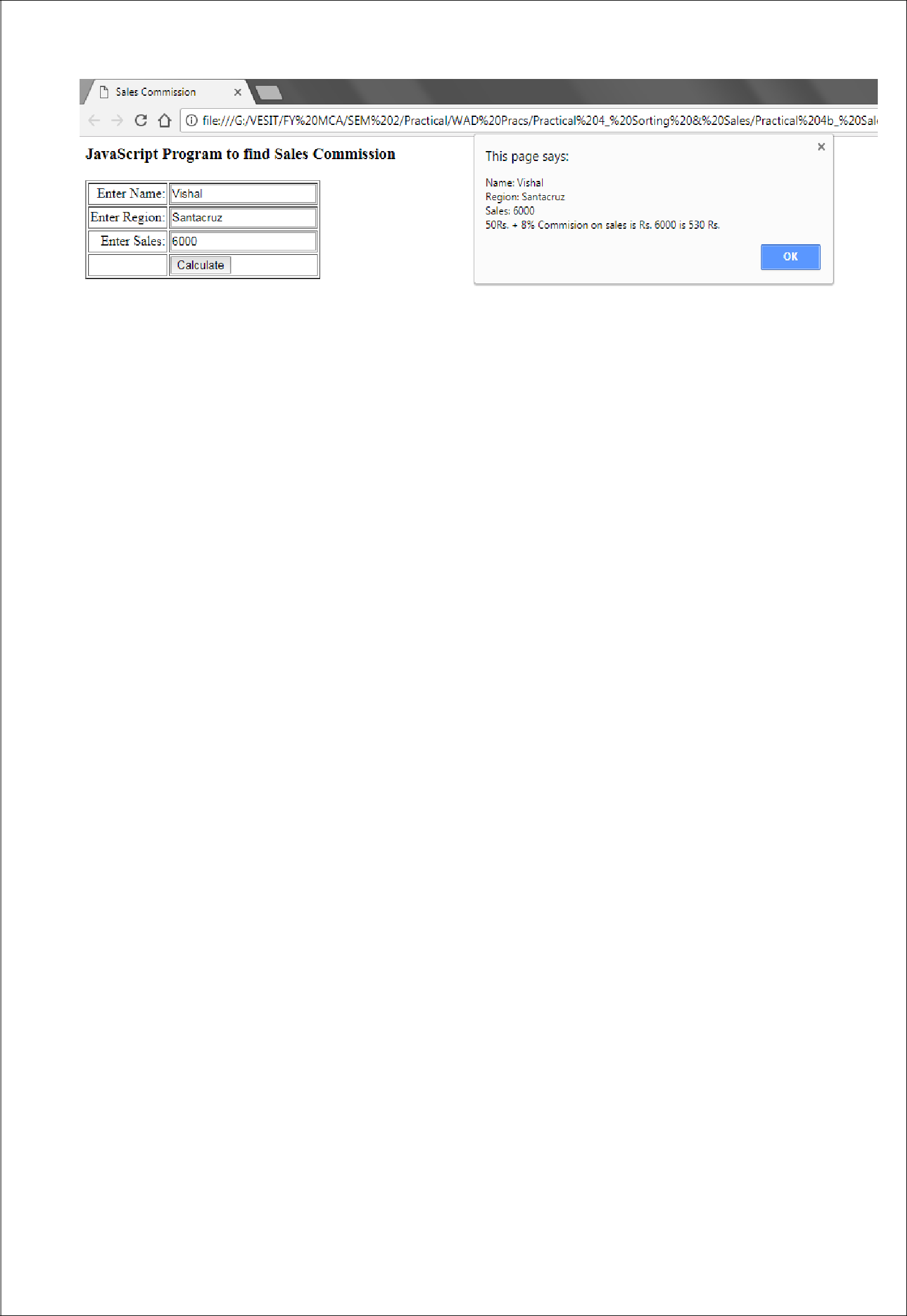
When the sales is less than 500:



When the sales is greater than 500 and less than 5000:



When the sales is greater than 5000:



**Practical: 5**

**JAVASCRIPT:**

**getElementById() Method**

The getElementById() method returns the element that has the ID attribute with the specified value. This method is one of the most common methods in the HTML DOM, and is used almost every time you want to manipulate, or get info from, an element on your document. Returns *null* if no elements with the specified ID exists.

An ID should be unique within a page. However, if more than one element with the specified ID exists, the getElementById() method returns the first element in the source code.

**Syntax**

document.getElementById(*elementID*)

**Window alert() Method**

The alert() method displays an alert box with a specified message and an OK button. An alert box is often used if you want to make sure information comes through to the user.

**Note:** The alert box takes the focus away from the current window, andforces the browser to read the message. Do not overuse this method, as it prevents the user from accessing other parts of the page until the box is closed.

**Syntax**

alert(*message*)

**JavaScript Functions**

A JavaScript function is a block of code designed to perform a particular task. A JavaScript function is executed when "something" invokes it (calls it).

**JavaScript Function Syntax**

A JavaScript function is defined with the **function** keyword, followed by a **name**, followed by parentheses **()**.

Function names can contain letters, digits, underscores, and dollar signs (same rules as variables). The parentheses may include parameter names separated by commas: **(*parameter1, parameter2, ...*)**

The code to be executed, by the function, is placed inside curly brackets: **{}**

function *name* (*parameter1, parameter2,* *parameter3*) {

*code to be executed*

}

Function **parameters** are listed inside the parentheses () in the function definition. Function **arguments** are the **values** received by the function when it is invoked. Inside the function, the arguments (the parameters) behave as local variables

**HTML Form:**

The HTML **<form>** element defines a form that is used to collect user

input:

<form>

.

*form elements*

.

</form>

An HTML form contains **form elements**. Form elements are different types of input elements, like text fields, checkboxes, radio buttons, submit buttons, and more.

The **<input>** element is the most important form element. The <input> element can be displayed in several ways, depending on the **type** attribute.

**Text Input**

**<input type="text">** defines a one-line input field for **text input**:

**Radio Button Input**

**<input type="radio">** defines a **radio button**. Radio buttons let a userselect ONE of a limited number of choices:

**The Submit Button**

**<input type="submit">** defines a button for **submitting** the form data to a **form-handler**. The form-handler is typically a server page with a script forprocessing input data.

1. **WAP to design a registration form with fields like first name, last name, address, e-mail id, password and confirm password, write appropriate validation on messages for each field.**

**Code:**

<html>

<head>

<script>

function val()

{

var name=document.getElementById("fname").value;

var lname=document.getElementById("lname").value;

var address=document.getElementById("address").value;

var email=document.getElementById("email").value;

var password=document.getElementById("password").value;

var cpass=document.getElementById("cpassword").value;

if(name=="")

{

window.alert("name cannot be empty");

}

if(isNaN(name)==false)

{

window.alert("name cannot be a number");

}

if(lname=="")

{

window.alert("Last name cannot be empty");

}

if(isNaN(lname)==false)

{

window.alert("Last name cannot be a number");

}

if(address=="")

{

window.alert("address cannot be empty");

}

if(email=="")

{

window.alert("email cannot be empty");

}

if(email.indexOf("@")<1 || email.lastIndexOf(".")<email.indexOf("@")+2 || email.lastIndexOf(".")+2>=email.length) {

window.alert("not a valid email");

}

if(password=="")

{

window.alert("Password cannot be empty");

}

if(cpass=="")

{

window.alert("Last name cannot be empty");

}

if(password != cpass)

{

window.alert("confirm password should be same as that of password");

}

}

</script>

</head>

<body>

<form>

Name:

<input type="text" id="fname"></br></br>

Last Name:

<input type="text" id="lname"></br></br>

Address:

<input type="textarea" id="address" rows="20" cols="5"></br></br>

Email:

<input type="text" id="email"></br></br>

Password:

<input type="password" id="password"></br></br>

Confirm Password:

<input type="password" id="cpassword"></br></br>

<input type="submit" name="submit" onclick="val()"

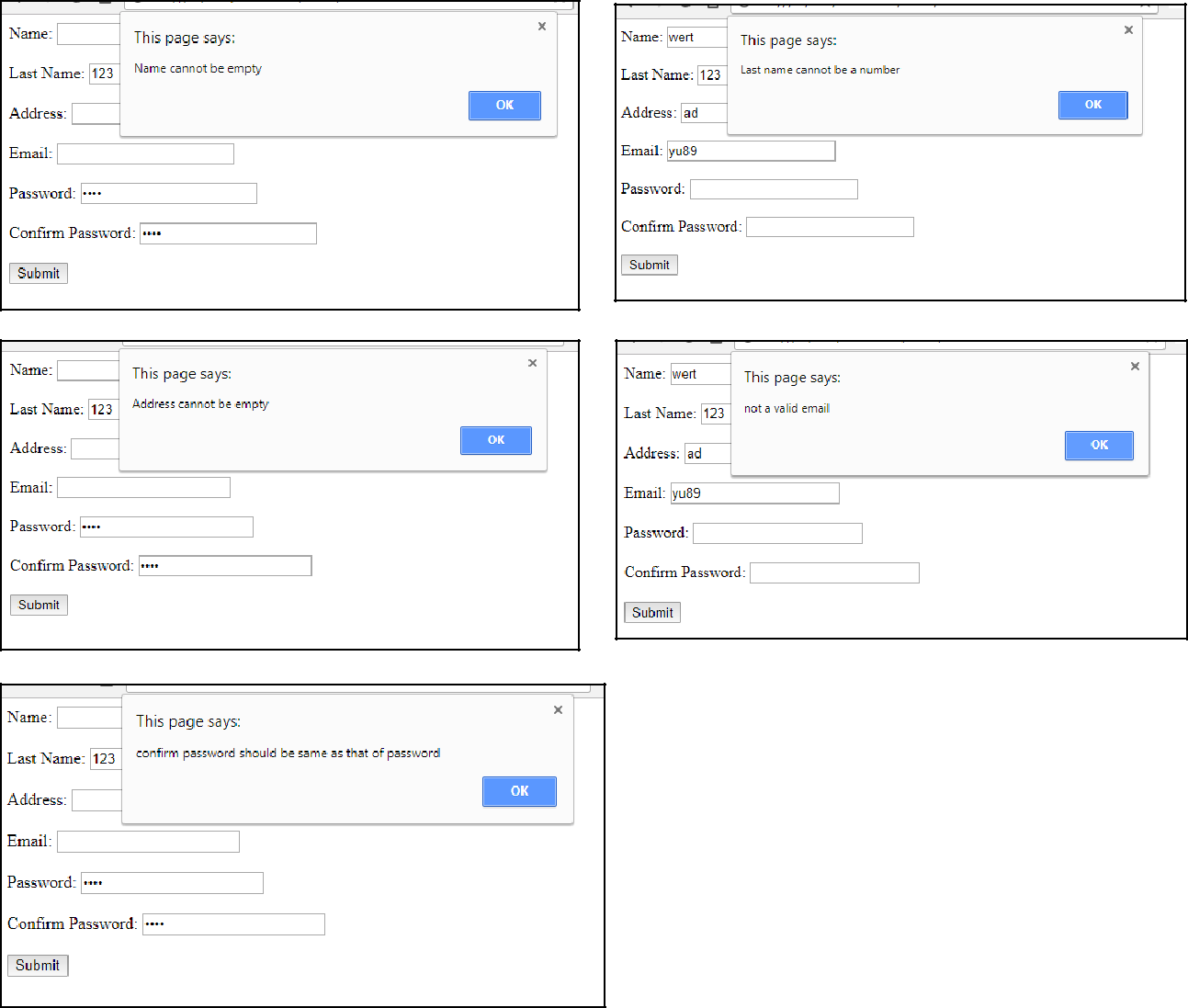
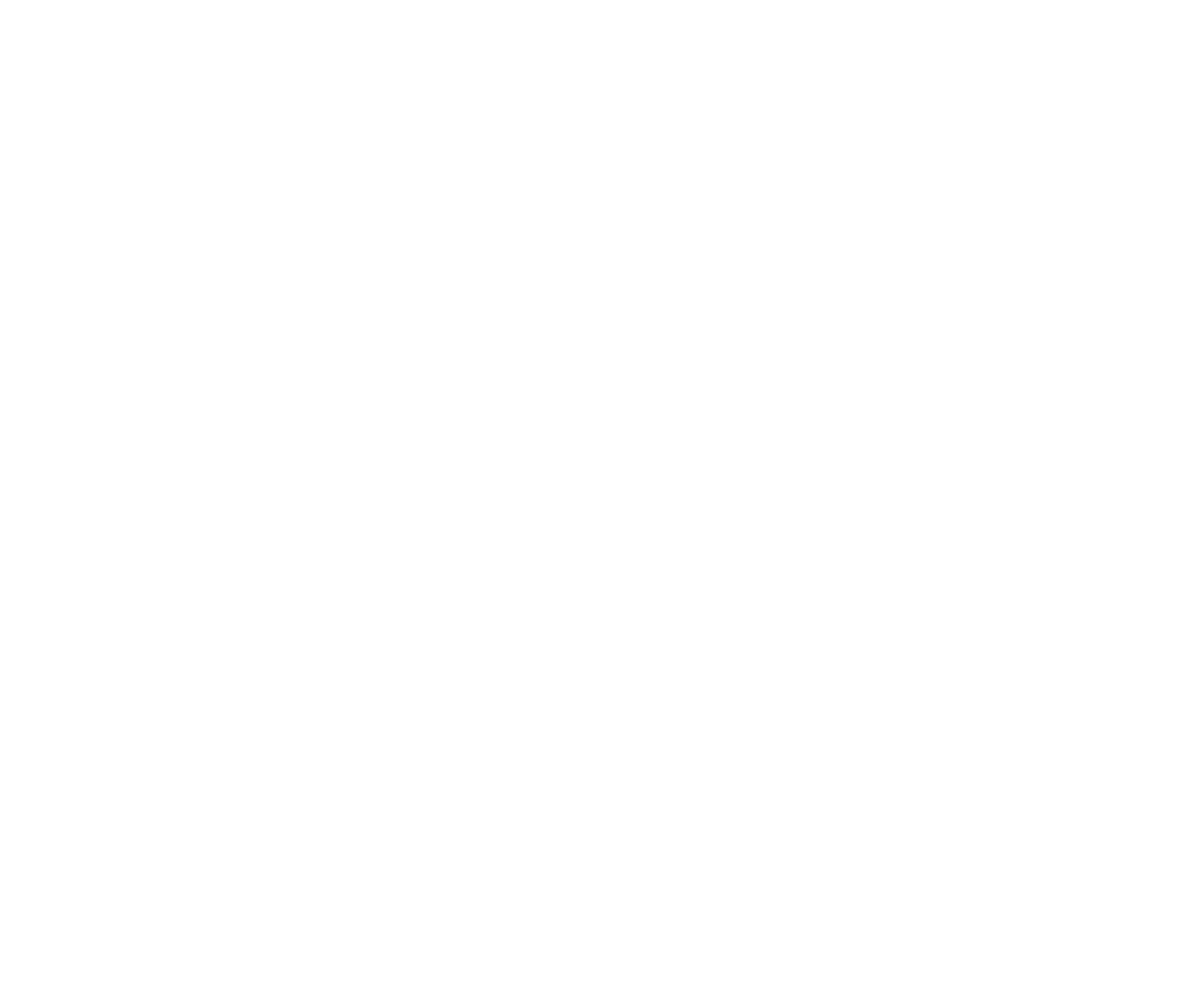
value="Submit"></br></br>

</form>

</body>

</html>

**Output:**



**Practical: 6**

**Array:**

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

JavaScript arrays are used to store multiple values in a single variable.

Using an array literal is the easiest way to create a JavaScript Array.

**Syntax:**

var array\_name = [item1, item2, ...];

**Example:**

var cars = ["Saab", "Volvo", "BMW"];

Spaces and line breaks are not important. A declaration can span multiple lines:

var cars = [

"Saab",

"Volvo",

"BMW"

];

**Select:**

* The <select> element is used to create a drop-down list.
* The option tags inside the <select> element define the available options in the list.
* The <select> element is a form control and can be used in a form to collect user input.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | **Description** |  |
|  |  |  |  |
| **autofocus** | autofocus | Specifies that the drop-down list should |  |
| automatically get focus when the page loads |  |
|  |  |  |
|  |  |  |  |
| **disabled** | disabled | Specifies that a drop-down list should be disabled |  |
|  |  |  |  |
| **form** | form\_id | Defines one or more forms the select field belongs |  |
| to |  |
|  |  |  |
|  |  |  |  |
| **multiple** | multiple | Specifies that multiple options can be selected at |  |
| once |  |
|  |  |  |
|  |  |  |  |
| **name** | name | Defines a name for the drop-down list |  |
|  |  |  |  |
| **required** | required | Specifies that the user is required to select a |  |
| value before submitting the form |  |
|  |  |  |
|  |  |  |  |
| **size** | number | Defines the number of visible options in a drop- |  |
| down list |  |
|  |  |  |
|  |  |  |  |

1. **Program to design a drop down list with categories of items, on selecting any category its item list should be displayed in another list, in sorted order. Show an appropriate message after selecting both. (e.g for Category Grocery it should display items like sugar, oil, salt , pulses etc.)**

**Program:**

**categories.html**

<html>

<head>

<title>Dynamic Select Statements</title> <script type="text/javascript"> var stateLists = new Array(4)

stateLists["empty"] = ["Select"];

stateLists["Grocery"] = ["Oil", "Pulses","Salt", "Sugar"];

stateLists["Fruits"] = ["Apple", "Banana", "Grapes", "Mango"];

stateLists["Vegetables"] = ["Cabbage", "Coli-flower", "Spinach", "Tomato"];

stateLists["Frozen"]= ["Butter", "Cheese", "ICe-cream"];

var i,temp;

for (i=0;i<stateLists.length-1;i++)

{

for(j=0;j<=stateLists.length-i-1;j++)

{

if(stateLists[j].localeCompare(stateLists[j+1])==1)

{

temp=stateLists[j];

stateLists[j]=stateLists[j+1];

stateLists[j+1]=temp;

}

}

}

document.getElementById("city").innerHTML=stateLists; function stateChange(selectObj) {

var idx = selectObj.selectedIndex;

var which = selectObj.options[idx].value; cList = stateLists[which];

var cSelect = document.getElementById("city"); var len=cSelect.options.length; while (cSelect.options.length > 0)

{

cSelect.remove(0);

}

var newOption;

// create new options

for (var i=0; i<cList.length; i++)

{

newOption = document.createElement("option"); newOption.value = cList[i]; newOption.text=cList[i]; try

{

cSelect.add(newOption);

}

catch (e)

{

cSelect.appendChild(newOption);

}

}

alert(" Options are selected and sorted.");

}

</script>

</head>

<body>

<h2>Select a States:</h2>

<label for="state">Select</label>

<select id="state" onchange="stateChange(this);"> <option value="empty">Select an options:</option> <option value="Grocery">Grocery</option> <option value="Fruits">Fruits</option>

<option value="Vegetables">Vegetables</option> <option value="Frozen">Frozen</option>

</select>

<br/>

<label for="city">Items are:</label>

<select id="city">

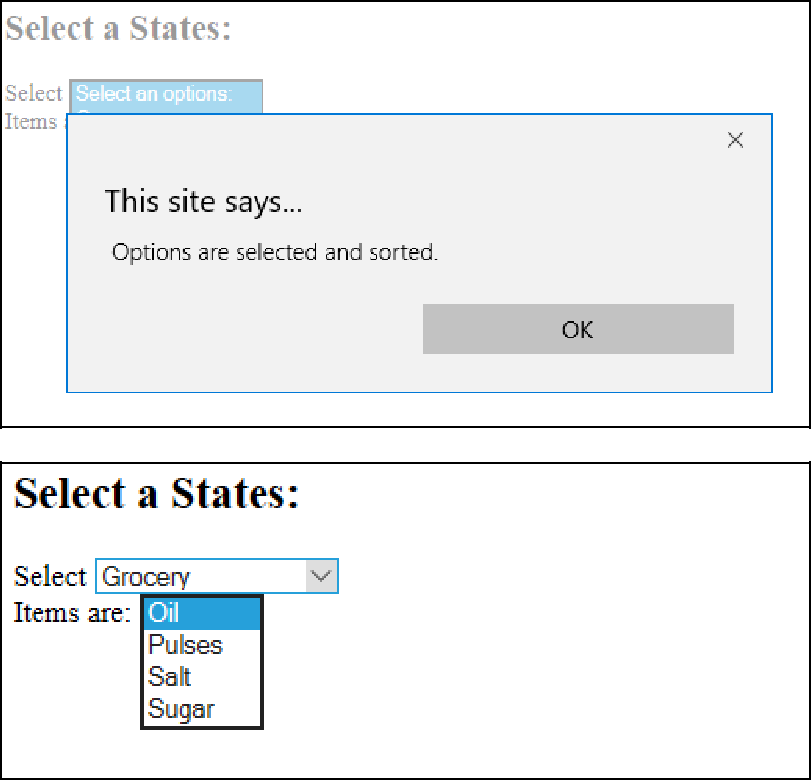
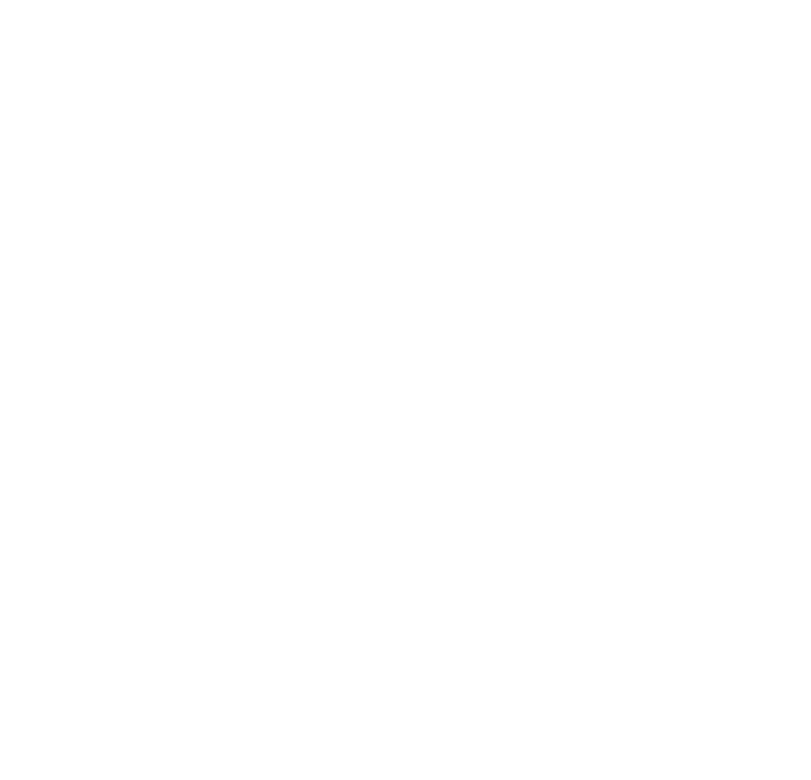
<option value="0">Options</option>

</select>

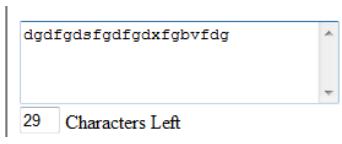
</body>

</html>

**Output:**



**Aim**: Design the following TextArea counter.



**Program:**

**textarea.html**

<html>

<head>

<title>Text Area Calculation</title>

</head>

<script>

function textcount()

{

var c;

var text\_max=50;

c=document.f1.message.value.length;

document.f1.keepcount.value=text\_max-c;

}

</script>

<form name="f1">

<h3> Type your Text here!!!!</h3>

<textarea name="message" rows="8" cols="30" maxlength="50"

onkeydown="textcount();" onkeyup="textcount();"></textarea>

<br><br>

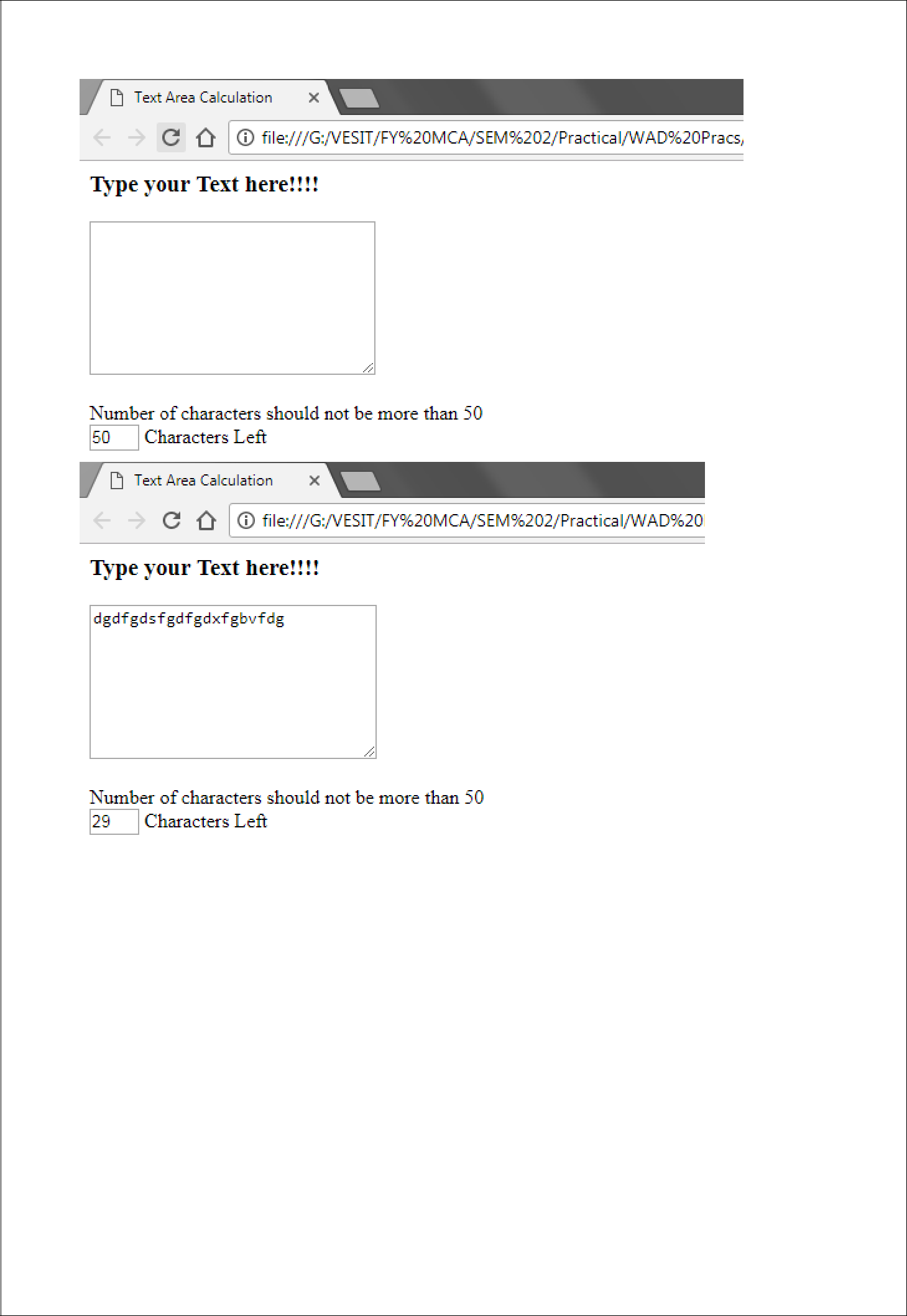
<p1>Number of characters should not be more than 50</p1><br>

<input type="text" name="keepcount" value="50" size="1"> Characters Left

</form>

</html>

OUTPUT:



**Practical: 7**

**AJAX**

AJAX = **A**synchronous **J**avaScript **A**nd **X**ML.

AJAX is not a programming language.

AJAX just uses a combination of:

-A browser built-in XMLHttpRequest object (to request data from a web server)

-JavaScript and HTML DOM (to display or use the data)

AJAX is a misleading name. AJAX applications might use XML to transport data, but it is equally common to transport data as plain text or JSON text.

AJAX allows web pages to be updated asynchronously by exchanging data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

1. An event occurs in a web page (the page is loaded, a button is clicked)
2. An XMLHttpRequest object is created by JavaScript
3. The XMLHttpRequest object sends a request to a web server
4. The server processes the request
5. The server sends a response back to the web page
6. The response is read by JavaScript
7. Proper action (like page update) is performed by JavaScript

**XMLHttpRequest Object Methods**

-**abort()**

Cancels the current request.

-**getAllResponseHeaders()**

Returns the complete set of HTTP headers as a string.

-**getResponseHeader( headerName )**

Returns the value of the specified HTTP header.

-**open( method, URL )**

**open( method, URL, async )**

**open( method, URL, async, userName )**

**open( method, URL, async, userName, password )**

Specifies the method, URL, and other optional attributes of a request.

The method parameter can have a value of "GET", "POST", or "HEAD". Other

HTTP methods such as "PUT" and "DELETE" (primarily used in REST

applications) may be possible.

-**send( content )**

Sends the request.

-**setRequestHeader( label, value )**

Adds a label/value pair to the HTTP header to be sent.

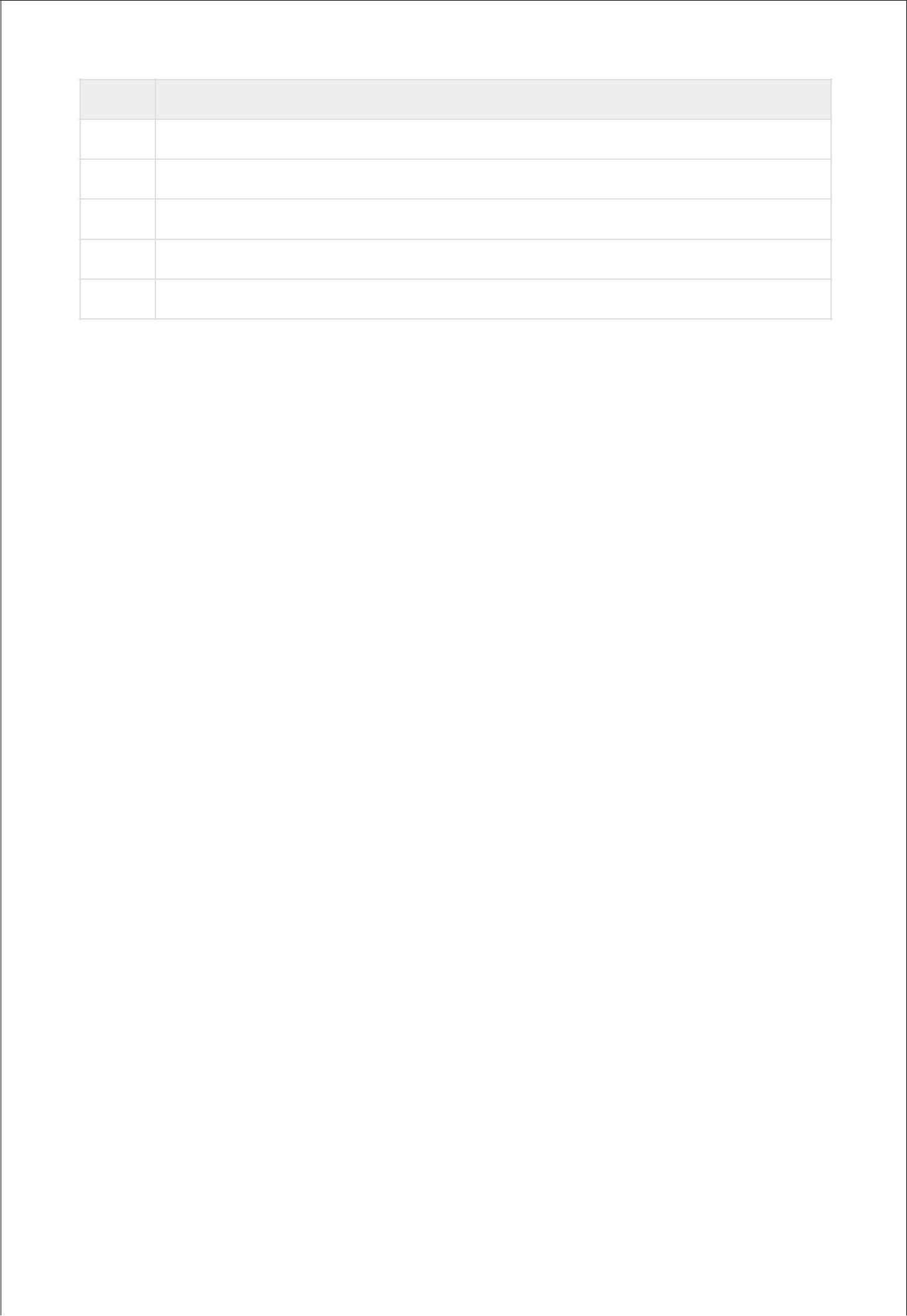
**XMLHttpRequest Properties**

-**onreadystatechange**

An event handler for an event that fires at every state change. -**readyState**

The readyState property defines the current state of the XMLHttpRequest object.

The following table provides a list of the possible values for the readyState property –



State Description

1. The request is not initialized.
2. The request has been set up.
3. The request has been sent.
4. The request is in process.
5. The request is completed.

readyState = 0 After you have created the XMLHttpRequest object, but before you have called the open() method.

readyState = 1 After you have called the open() method, but before you have called send().

readyState = 2 After you have called send().

readyState = 3 After the browser has established a communication with the server, but before the server has completed the response.

readyState = 4 After the request has been completed, and the response data has been completely received from the server. -**responseText**

Returns the response as a string.

-**responseXML**

Returns the response as XML. This property returns an XML document object,

which can be examined and parsed using the W3C DOM node tree methods and

properties.

-**status**

Returns the status as a number (e.g., 404 for "Not Found" and 200 for "OK").

-**statusText**

Returns the status as a string (e.g., "Not Found" or "OK").

**a)Program using Ajax for making a server request.**

**Code:**

**Request.html**

<html>

<body>

<h1>The XMLHttpRequest Object</h1>

<div id="demo">

<button type="button" onclick="loadDoc()">Change Content</button>

</div>

<script>

function loadDoc()

{

var xhttp = new XMLHttpRequest();

xhttp.onreadystatechange = function()

{

if(this.readyState == 4 && this.status == 200)

{

document.getElementById("demo").innerHTML =this.responseText;

}

};

xhttp.open("GET", "data.txt", true);

xhttp.send();

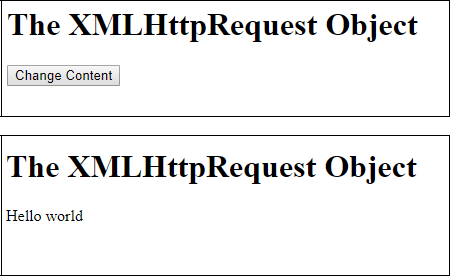
}

</script>

</body>

</html>

**Output:**



1. **Write a program of AJAX with PHP for displaying the multiplication table of the number entered.**

**Code:**

**Multiplication.html**

<html>

<body>

<input type="text" id="num">

<div id="demo">

<button type="button" onclick="loadDoc()">Change Content</button>

</div>

<script>

function loadDoc()

{

var num=document.getElementById("num").value; var xhttp = new XMLHttpRequest(); xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200)

{

document.getElementById("demo").innerHTML =this.responseText;

}

};

xhttp.open("GET", "mul.php?q="+num, true);

xhttp.send();

}

</script>

</body>

</html>

**Mul.php**

<?php

$num=$\_GET["q"];

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

{

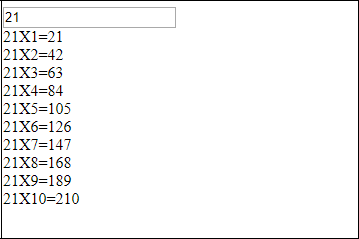
$res=$num\*$i;

echo $num."X".$i."=".$res."</br>";

}

?>

**Output:**



**Practical: 8**

**The XMLHttpRequest Object**

All modern browsers support the XMLHttpRequest object.

The XMLHttpRequest object can be used to exchange data with a web server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

**Creating an XMLHttpRequest Object**

All modern browsers (Chrome, Firefox, IE7+, Edge, Safari, Opera) have a built-in XMLHttpRequest object.

Syntax for creating an XMLHttpRequest object:

*variable* = new XMLHttpRequest();

**Example:**

var xhttp = new XMLHttpRequest();

**XMLHttpRequest Properties**

-**onreadystatechange**

An event handler for an event that fires at every state change. -**readyState**

The readyState property defines the current state of the XMLHttpRequest object.

The following table provides a list of the possible values for the readyState property –



State Description

1. The request is not initialized.
2. The request has been set up.
3. The request has been sent.
4. The request is in process.
5. The request is completed.

readyState = 0 After you have created the XMLHttpRequest object, but before you have called the open() method.

readyState = 1 After you have called the open() method, but before you have called send().

readyState = 2 After you have called send().

readyState = 3 After the browser has established a communication with the server, but before the server has completed the response.

readyState = 4 After the request has been completed, and the response data has been completely received from the server. -**responseText**

Returns the response as a string.

-**status**

Returns the status as a number (e.g., 404 for "Not Found" and 200 for "OK").

-**statusText:**Returns the status as a string (e.g., "Not Found" or "OK").

**Program using Ajax for loading html scriptlets from server using Ajax events.**

**Code:**

<html>

<body>

<div id="demo">

<h1>The XMLHttpRequest Object</h1>

<button type="button" onclick="loadDoc()">Change Content</button>

</div>

<script>

function loadDoc() {

var xhttp = new XMLHttpRequest(); xhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) { document.getElementById("demo").innerHTML =this.responseText; }

};

xhttp.open("GET", "scriplet.html", true); xhttp.send();

}

</script>

</body>

</html>

**scriplet.html**

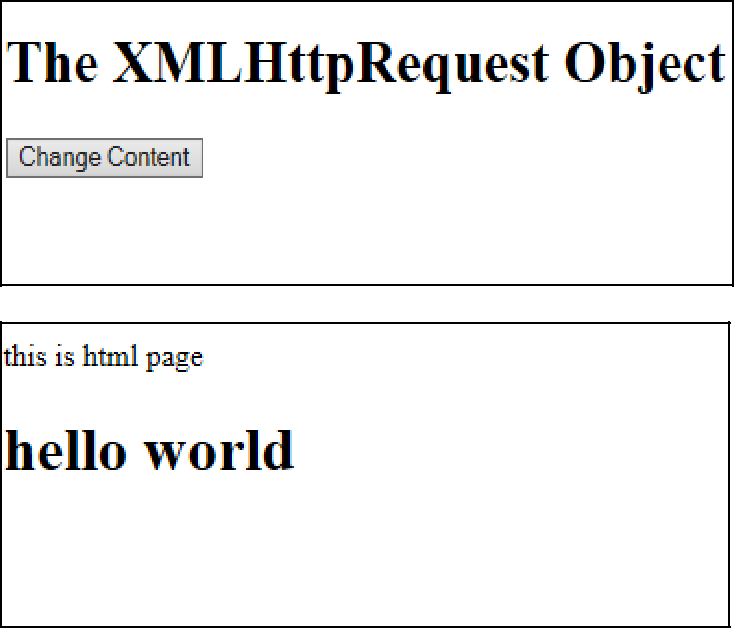
<html>

<p>this is html page</p>

<h1>hello world</h1>

</html>

**Output:**



**Practical: 9**

**Ajax Style File Upload**

Some rules to follow for the HTML form above:

* Form uses method="post"
* The form also needs the following attribute: enctype="multipart/form-data". It specifies which content-type to use when submitting the

form

Without the requirements above, the file upload will not work.

Other important things:

* The type="file" attribute of the <input> tag shows the input field as a file-select control, with a "Browse" button next to the input control
* The move\_uploaded\_file() function moves an uploaded file to a new location.

This function returns TRUE on success, or FALSE on failure.

**Syntax:**

move\_uploaded\_file(file,newloc)

file: Specifies the file to be moved.

newloc: Specifies the new location of the file.

**Progress Bar:**

The <progress> element represents the completion progress of a task.

<progress id="progress" value="0"></progress>

HTML5 progress element supports following attributes:

* **max** - specifies how much work the task requires in total. Must be afloating-point number > 0
* **value** - specifies how much of the task has been completed. Must be afloating-point number >= 0 and <= max

**Program for making an Ajax style file upload.**

**Code:**

**Fileupload.html**

<html>

<head>

<title>Ajax file upload with percentage progress bar</title> <meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<!-- include jQuery -->

<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.1/jquery.min.js"></sc

ript>

<!-- include bootstrap files -->

<!-- Latest compiled and minified CSS --> <link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css

* integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Optional theme --> <link rel="stylesheet"

href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript --> <script

src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>

<script>

$(function () {

$('#btn').click(function () {

$('.myprogress').css('width', '0'); $('.msg').text('');

var filename = $('#filename').val(); var myfile = $('#myfile').val();

if (filename == '' || myfile == '') { alert('Please enter file name and select file'); return;

}

var formData = new FormData(); formData.append('myfile', $('#myfile')[0].files[0]); formData.append('filename', filename); $('#btn').attr('disabled', 'disabled'); $('.msg').text('Uploading in progress...'); $.ajax({

url: 'uploadscript.php',

data: formData,

processData: false,

contentType: false,

type: 'POST',

// this part is progress bar xhr: function () {

var xhr = new window.XMLHttpRequest(); xhr.upload.addEventListener("progress", function (evt) {

if (evt.lengthComputable) {

var percentComplete = evt.loaded / evt.total;

percentComplete = parseInt(percentComplete \* 100);

$('.myprogress').text(percentComplete + '%');

$('.myprogress').css('width', percentComplete + '%');

}

}, false);

return xhr;

},

success: function (data) {

$('.msg').text(data);

$('#btn').removeAttr('disabled');

}

});

});

});

</script>

</head>

<body>

<div class="container">

<div class="row">

<h3>Ajax file upload with progress bar</h3> <form id="myform" method="post"> <div class="form-group">

<label>Enter the file name: </label>

<input class="form-control" type="text" id="filename" /> </div>

<div class="form-group">

<label>Select file: </label>

<input class="form-control" type="file" id="myfile" /> </div>

<div class="form-group">

<div class="progress">

<div class="progress-bar progress-bar-success myprogress"

role="progressbar" style="width:0%">0%</div>

</div>

<div class="msg"></div>

</div>

<input type="button" id="btn" class="btn-success" value="Upload" />

</form>

</div>

</div>

</body>

</html>

**uploadscript.php**

<?php

error\_reporting(0);

if (isset($\_POST) and $\_SERVER['REQUEST\_METHOD'] == "POST") { $path = "uploads/"; //set your folder path //set the valid file extensions

$valid\_formats = array("jpg", "png", "gif", "bmp", "jpeg", "GIF", "JPG", "PNG", "doc", "txt", "docx", "pdf", "xls", "xlsx"); //add the formats you want to upload

$name = $\_FILES['myfile']['name']; //get the name of the file $size = $\_FILES['myfile']['size']; //get the size of the file

if (strlen($name)) { //check if the file is selected or cancelled after pressing the browse button.

list($txt, $ext) = explode(".", $name); //extract the name and extension of the file

if (in\_array($ext, $valid\_formats)) { //if the file is valid go on. if ($size < 2098888) { // check if the file size is more than 2 mb $file\_name = $\_POST['filename']; //get the file name $tmp = $\_FILES['myfile']['tmp\_name'];

if (move\_uploaded\_file($tmp, $path . $file\_name.'.'.$ext)) { //check if it the file move successfully.

echo "File uploaded successfully!!";

} else {

echo "failed";

}

} else {

echo "File size max 2 MB";

}

} else {

echo "Invalid file format..";

}

} else {

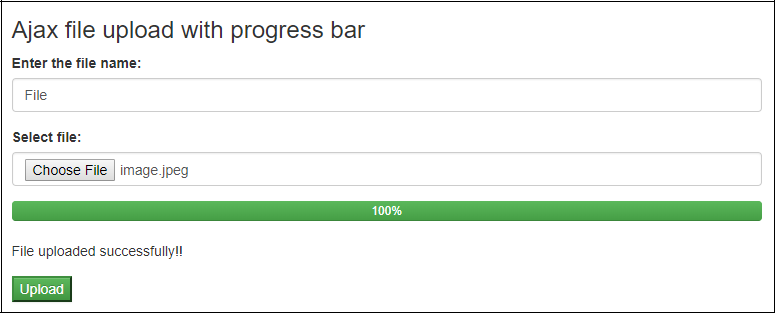
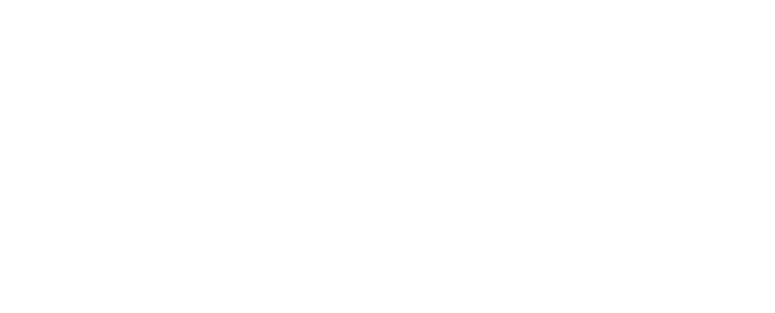
echo "Please select a file..!";

}

exit;

}

**Output:**



**Practical: 10**

**Bootstrap** is a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source_software) front-end [web framework](https://en.wikipedia.org/wiki/Web_framework) for designing [websites](https://en.wikipedia.org/wiki/Website) and [web applications.](https://en.wikipedia.org/wiki/Web_application) It contains [HTML-](https://en.wikipedia.org/wiki/HTML) and [CSS-](https://en.wikipedia.org/wiki/CSS)based design templates for [typography,](https://en.wikipedia.org/wiki/Typography) forms, buttons, navigation and other interface components, as well as optional [JavaScript](https://en.wikipedia.org/wiki/JavaScript) extensions. Unlike many web frameworks, it concerns itself with [front-end development](https://en.wikipedia.org/wiki/Front-end_web_development) only.

Bootstrap is the second most-starred project on [GitHub,](https://en.wikipedia.org/wiki/GitHub) with more than 107,000 stars and 48,000 forks.

Drop down:

Example

<div class="dropdown">

<button class="btn btn-primary dropdown-

toggle" type="button" data-toggle="dropdown">Dropdown Example <span class="caret"></span></button>

<ul class="dropdown-menu">

<li><a href="#">HTML</a></li>

<li><a href="#">CSS</a></li>

<li><a href="#">JavaScript</a></li>

</ul>

</div>

Example Explained

The .dropdown class indicates a dropdown menu. To open the dropdown menu, use a button or a link with a class of .dropdown-toggle and the datatoggle="dropdown" attribute. The .caret class creates a caret arrow icon (), which indicates that the button is a dropdown.

Add the .dropdown-menu class to a <ul> element to actually build the dropdown menu

**Aim: Design a web page in Bootstrap using different fonts, content drop down supported heading styles, different column layout Image styles, alignment of dropdowns to left, right and centre**

**Code:**

<html>

<head>

<title>Bootstrap</title>

<link href="css/bootstrap.min.css" rel="stylesheet"> <link href="css/mdb.min.css" rel="stylesheet"> <body>

<div class="container">

<h2><strong>Bootstrap <small class="text-muted">Responsive

Data</small></strong></h2>

<table class="table">

<thead class="blue-grey lighten-4"><th>Name</th> <th>Stream</th></thead>

<tbody>

<tr><td>Dimple</td><td>MCA</td></tr>

<tr><td>Aru</td><td>EXTC</td></tr>

<tr><td>Neha</td><td>CS</td></tr>

</tbody></table>

<h2>Dropdown Right</h2> <div class="dropdown">

<button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown">Dropdown Example

<span class="caret"></span></button>

<ul class="dropdown-menu dropdown-menu-right"> <li><a href="#">HTML</a></li>

<li><a href="#">CSS</a></li>

<li><a href="#">JavaScript</a></li>

<li class="divider"></li>

<li><a href="#">About Us</a></li>

</ul>

</div>

<h2>Dropdown Left</h2> <div class="dropdown">

<button class="btn btn-default dropdown-toggle" type="button" data-toggle="dropdown">Dropdown Example

<span class="caret"></span></button>

<ul class="dropdown-menu dropdown-menu-left"> <li><a href="#">HTML</a></li>

<li><a href="#">CSS</a></li>

<li><a href="#">JavaScript</a></li>

<li class="divider"></li>

<li><a href="#">About Us</a></li>

</ul>

</div>

<h2>Bootstrap Responsive Image</h2>

<img src="scene.jpg" class="img-fluid z-depth-2 rounded-circle"

width="500px" height="500px">

</div>

<script type="text/javascript" src="js/jquery-3.2.1.min.js"></script>

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

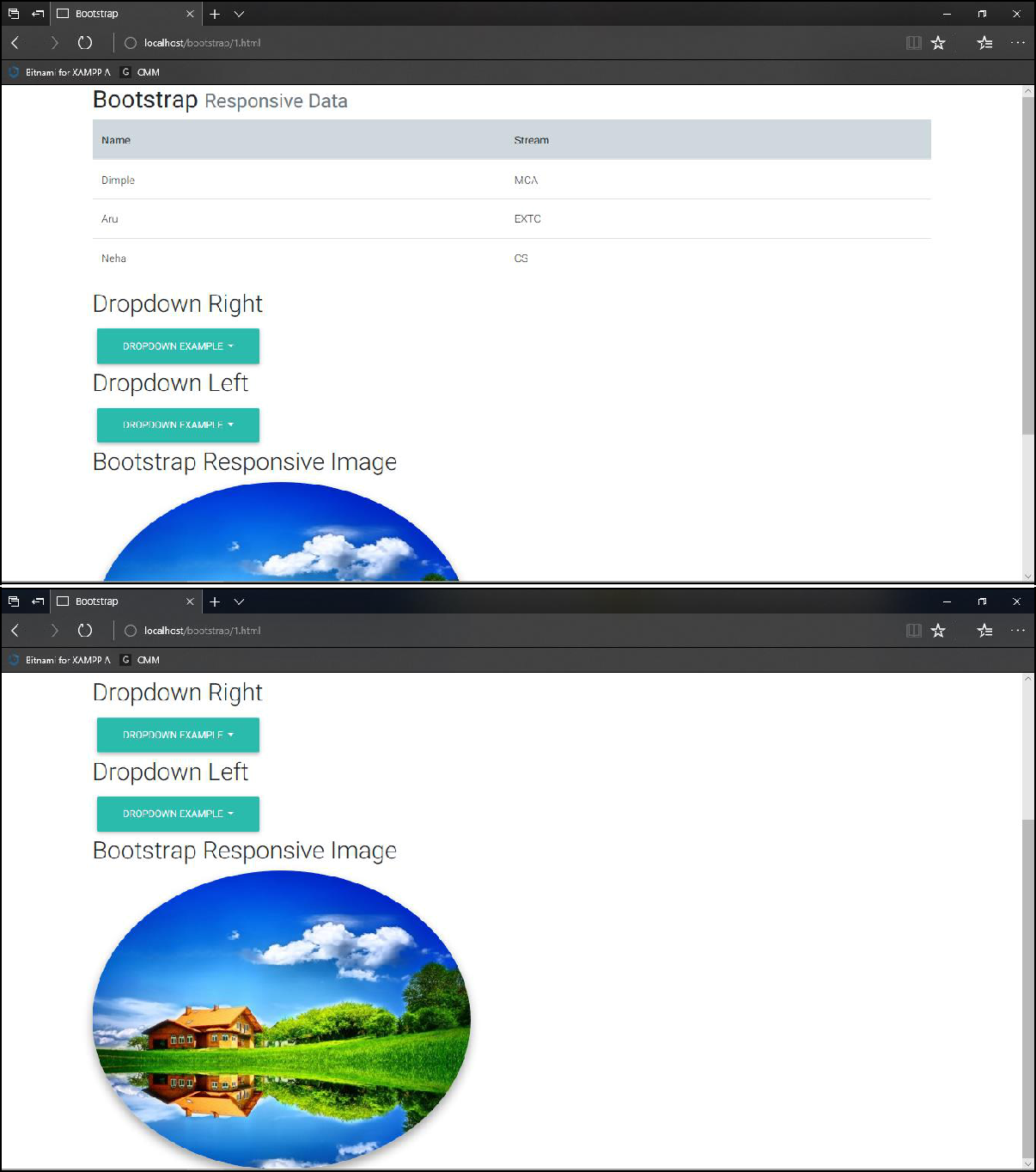
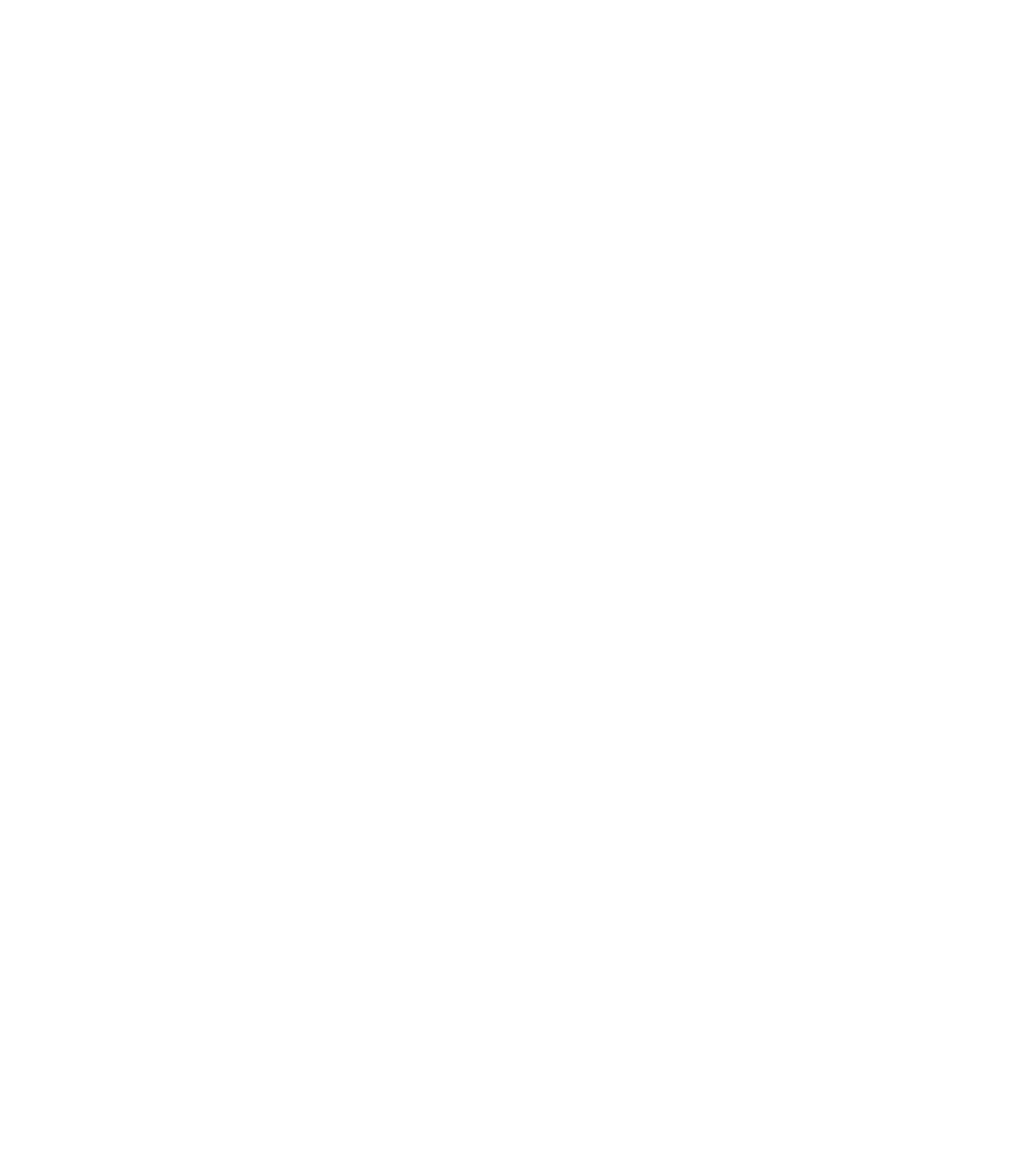
<script type="text/javascript" src="js/bootstrap.min.js"></script>

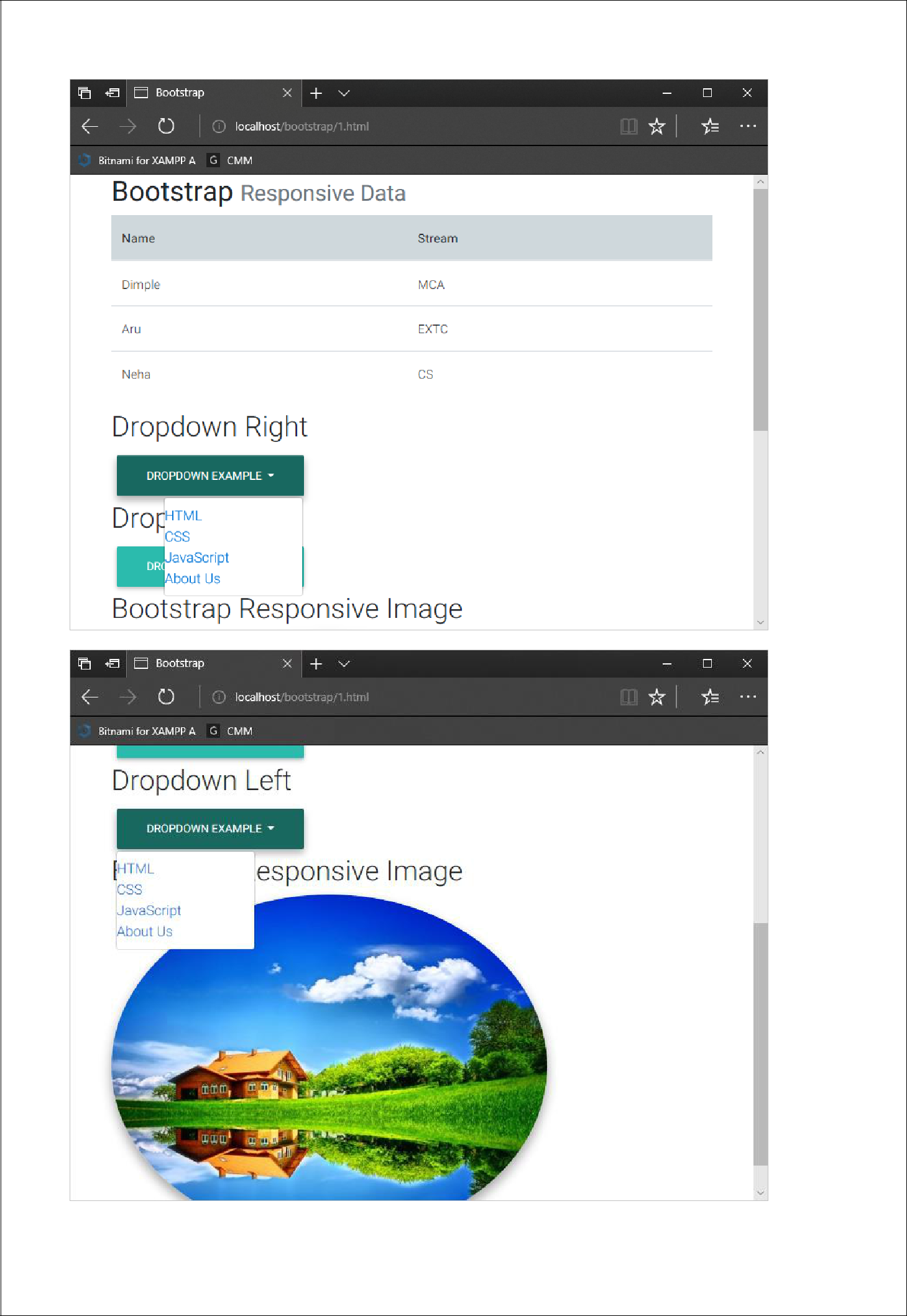
<script type="text/javascript" src="js/mdb.min.js"></script>

</body>

</html>

**Output:**





**Practical: 11**

**AIM:** Pricing table is a way for representing pricing schemes, features andcategories of products in a website. (Mostly pricing table is used for showing different kinds of hosting plans and packages by web hosting companies. Users can easily compare prices between different packages from lists of pricing columns and choose best plan that suit for their business.) Design a web page to show pricing table with 4 colour theme.

**Introduction:**

A basic Bootstrap table has a light padding and only horizontal dividers.

The .table class adds basic styling to a table:

Example

|  |  |  |
| --- | --- | --- |
| **Firstname** | **Lastname** | **Email** |
|  |  |  |
| John | Doe | john@example.com |
|  |  |  |
| Mary | Moe | mary@example.com |
|  |  |  |
| July | Dooley | july@example.co |

**Bordered Table**

The .table-bordered class adds borders on all sides of the table and cells:

Example

|  |  |  |
| --- | --- | --- |
| **Firstname** | **Lastname** | **Email** |
|  |  |  |
| John | Doe | john@example.com |
|  |  |  |
| Mary | Moe | mary@example.com |
|  |  |  |
| July | Dooley | july@example.com |
|  |  |  |

**Code:**

<html>

<head>

<title>Pricing</title>

<link href="http://netdna.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css"

rel="stylesheet" id="bootstrap-css">

<link href="style.css" rel="stylesheet">

<style>

@import url(http://fonts.googleapis.com/css?family=Open+Sans:300);

.price\_table\_container{

text-align:center;

color:#666;

margin-top:35px;

font-family: 'Open Sans', sans-serif;

}

.price\_table\_heading{

font-size:36px;

padding:10px;

background:#EEE;

}

.price\_table\_row {

padding:15px;

background: #FFF;

}

.cost{

padding:30px;

font-size:30px;

}

.price\_table\_row:nth-of-type(even) {

background: #F8F8F8;

}

.btn{

border-radius:0px;

}

.recommended{

/\*USER DEFINED COLOUR\*/

background:#FF3A3A;

color:#FFF;

/\*USER DEFINED COLOUR\*/

padding:3px 0 3px 0;

margin-top:10px;

text-align:center;

margin-bottom:-35px;

font-family: 'Open Sans', sans-serif;

}

.primary-bg{background:#337AB7;color:#FFF;}

.success-bg{background:#5CB85C;color:#FFF;}

.info-bg{background:#5BC0DE;color:#FFF;}

.warning-bg{background:#F0AD4E;color:#FFF;}

</style>

<body>

<div class="container">

<h2><center>Hosting Plans By Different Companies</center></h2> <div class="row">

<div class="col-md-3 col-sm-6 col-xs-12 float-shadow"> <div class="price\_table\_container">

<div class="price\_table\_heading">Company A</div> <div class="price\_table\_body">

<div class="price\_table\_row cost warning-bg"><strong>Rs 199</strong><span>/Month</span></div>

<div class="price\_table\_row">10 Websites</div>

<div class="price\_table\_row">100 GB Storage</div> <div class="price\_table\_row">10 GB Bandwidth</div> <div class="price\_table\_row">10 Email Addresses</div>

<div class="price\_table\_row">Free Backup</div> <div class="price\_table\_row">Full Time Support</div>

</div>

<a href="#" class="btn btn-warning btn-lg btn-block">Sign Up</a> </div>

</div>

<div class="col-md-3 col-sm-6 col-xs-12 float-shadow">

<div class="recommended"><strong><span class="glyphicon glyphicon-heart" aria-hidden="true"></span> RECOMMENDED</strong></div>

<div class="price\_table\_container">

<div class="price\_table\_heading">Company B</div> <div class="price\_table\_body">

<div class="price\_table\_row cost primary-bg"><strong>Rs 299</strong><span>/Month</span></div>

<div class="price\_table\_row">10 Websites</div> <div class="price\_table\_row">100 GB Storage</div>

<div class="price\_table\_row">100 GB Bandwidth</div> <div class="price\_table\_row">50 Email Addresses</div> <div class="price\_table\_row">Free Backup</div>

<div class="price\_table\_row">Full Time Support</div> </div>

<a href="#" class="btn btn-primary btn-lg btn-block">Sign Up</a> </div>

</div>

<div class="col-md-3 col-sm-6 col-xs-12 float-shadow"> <div class="price\_table\_container">

<div class="price\_table\_heading">Company C</div> <div class="price\_table\_body">

<div class="price\_table\_row cost success-bg"><strong>Rs 449</strong><span>/Month</span></div>

<div class="price\_table\_row">10 Websites</div> <div class="price\_table\_row">10 GB Storage</div> <div class="price\_table\_row">100 GB Bandwidth</div> <div class="price\_table\_row">100 Email Addresses</div>

<div class="price\_table\_row">Free Backup</div>

<div class="price\_table\_row">Full Time Support</div> </div>

<a href="#" class="btn btn-success btn-lg btn-block">Sign Up</a> </div>

</div>

<div class="col-md-3 col-sm-6 col-xs-12 float-shadow"> <div class="price\_table\_container">

<div class="price\_table\_heading">Company D</div> <div class="price\_table\_body">

<div class="price\_table\_row cost info-bg"><strong>Rs 349</strong><span>/Month</span></div>

<div class="price\_table\_row">10 Websites</div> <div class="price\_table\_row">10 GB Storage</div> <div class="price\_table\_row">100 GB Bandwidth</div> <div class="price\_table\_row">1000 Email Addresses</div> <div class="price\_table\_row">Free Backup</div>

<div class="price\_table\_row">Full Time Support</div> </div>

<a href="#" class="btn btn-info btn-lg btn-block">Sign Up</a> </div>

</div>

</div>

</div>

<script src="http://code.jquery.com/jquery-1.10.2.min.js"></script>

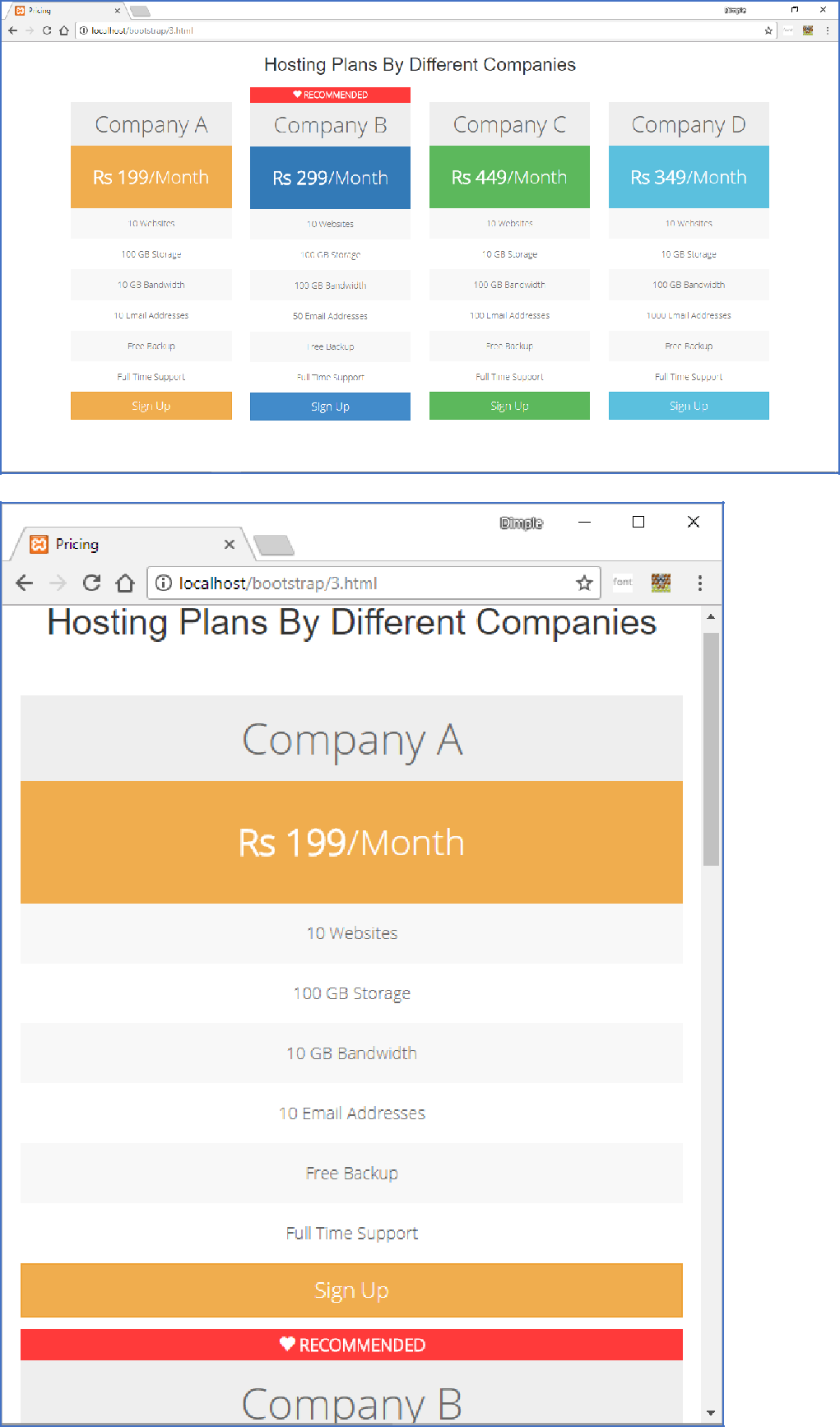
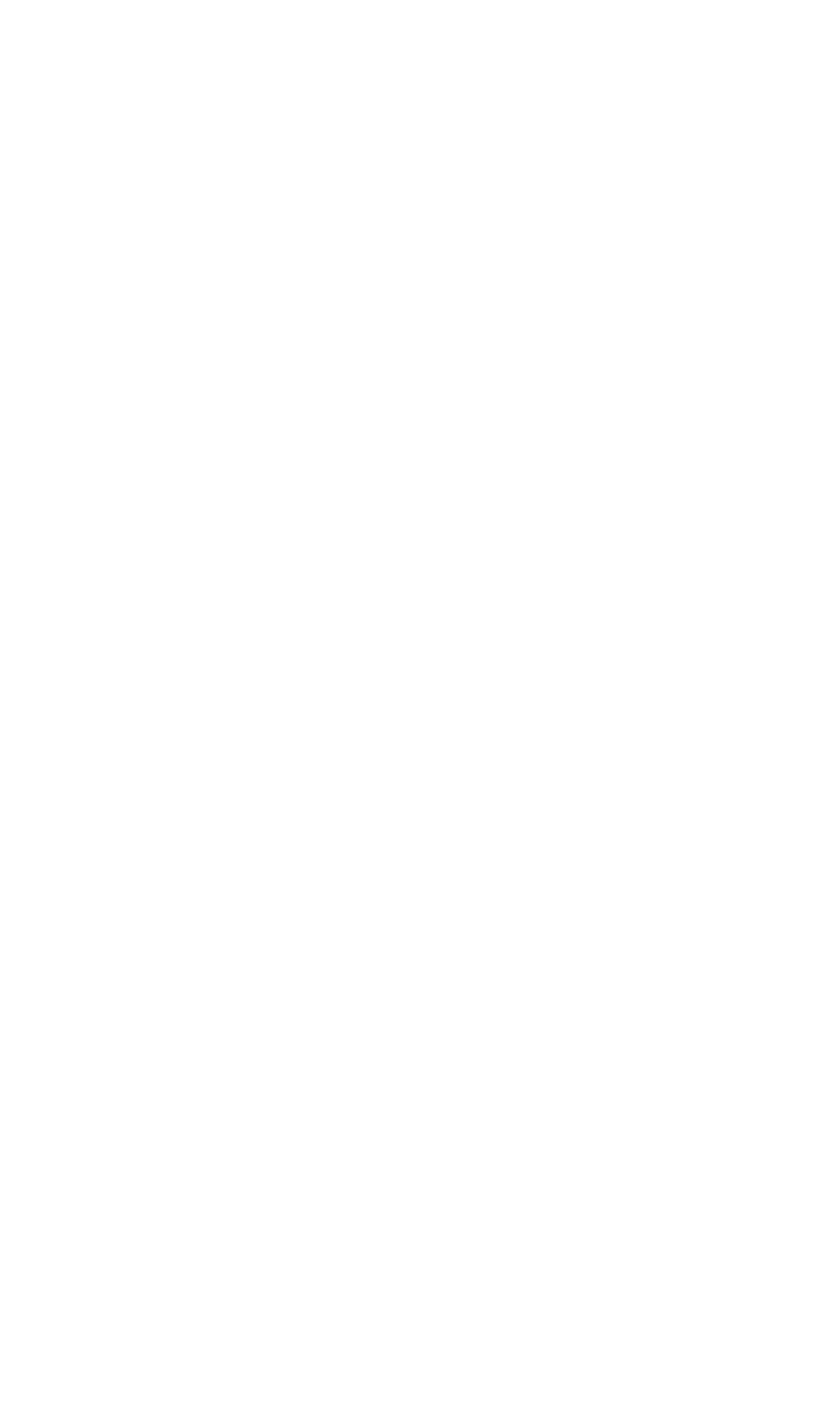
<script

src="http://netdna.bootstrapcdn.com/bootstrap/3.2.0/js/bootstrap.min.js"></script>

</body>

</html>

**Output:**



**Practical: 12**

**Joomla**

Joomla! is an content management system (CMS), which enables you to build Web sites and powerful online applications. Many aspects, including its ease-of-use and extensibility, have made Joomla! the most popular Web site software available. Best of all, Joomla! is an open source solution that is freely available to everyone.

Joomla! is designed to be easy to install and set up even if you're not an advanced user. Many Web hosting services offer a single-click install, getting your new site up and running in just a few minutes. Since Joomla! is so easy to use, as a Web designer or developer, you can quickly build sites for your clients. Then, with a minimal amount of instruction, you can empower your clients to easily manage their own sites themselves.

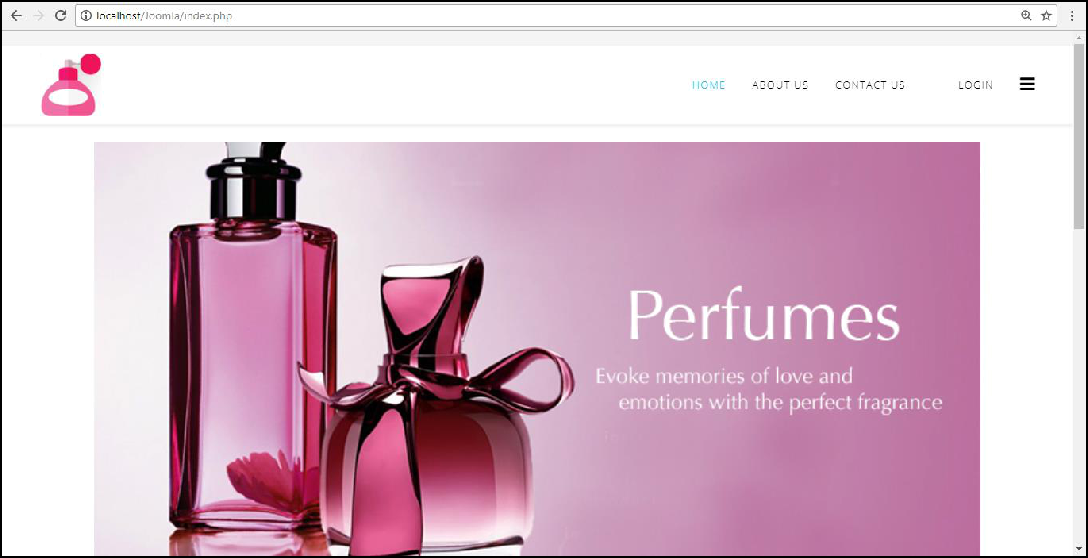
JA Content Slider module is used to slide your articles from Joomla categories with cool effects. The module's rich back-end configuration covers all built-in functionality like: layout, animation control, auto thumbnail creation, images size, number of articles, sorting and much more.

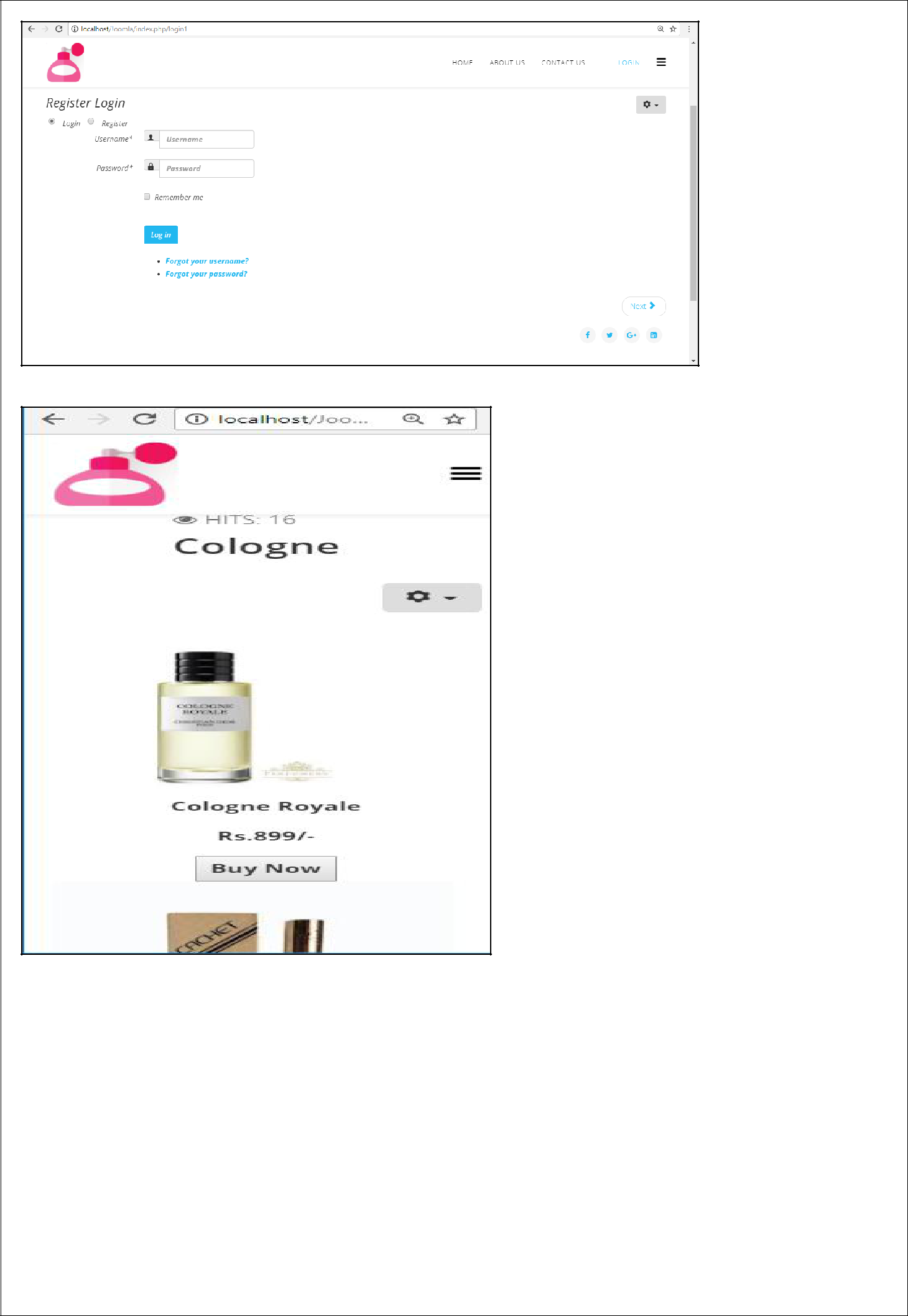
* Easy and elegant display of article list
* Support Joomla content and K2 content • Support 2 slide modes: Vertical and

Horizontal

* Item size is configurable
* Show or hide article image • Include number of settings for animation Deployment Many web hosts have control panels for automatic installation of Joomla. On Windows, Joomla can be installed using the Microsoft Web Platform Installer, which automatically detects and installs dependencies, such as PHP or MySQL

**Design a Joomla template that can be used for e-commerce websites which is easy to customize and adapt to the most crucial requirements.**





**Practical: 13**

**Joomla**

Joomla! is an content management system (CMS), which enables you to build Web sites and powerful online applications. Many aspects, including its ease-of-use and extensibility, have made Joomla! the most popular Web site software available. Best of all, Joomla! is an open source solution that is freely available to everyone.

Joomla! is designed to be easy to install and set up ev en if y ou're not an adv anced user. Many Web hosting services offer a single -click install, getting y our new site up and running in just a few minutes.

Since Joomla! is so easy to use, as a Web designer or dev eloper, y ou can quickly build sites for y our clients. Then, with a minimal amount of instruction, y ou can empower your clients to easily manage their own sites themselves.

JA Content Slider module is used to slide your articles from Joomla categories with cool effects.

The module's rich back-end configuration covers all built-in functionality like: layout, animation control, auto thumbnail creation, images size, number of articles, sorting and much more.

* Easy and elegant display of article list
* Support Joomla content and K2 content
* Support 2 slide modes: Vertical and Horizontal
* Item size is configurable
* Show or hide article image
* Include number of settings for animation

Deployment

Many web hosts have control panels for automatic installation of Joomla. On Windows,

Joomla can be installed using the Microsoft Web Platform Installer, which automatically detects and installs dependencies, such as PHP or MySQL

Google Map Integration :

Adding a map with your address to your Joomla.com website is done easily thanks to the special inbuilt Google Maps module. You can practically place the module in any position of your website, but it is designed to look great as a header on your Contacts page or as a side element in your articles. Y ou can also add an unlimited number of maps throughout your website. In this tutorial, we will show you how.

* Locate the module and add your address
* Adding map to your Contact page
* Adding map to a side column
* Adding multiple maps on your website

**Design one page Joomla template with features like On Screen Responsive Content Slider, Top Fixed Menu, Custom 404 error page, maximum module positions and Google Map Integration.**

