

# Scripting Languages Lab Manual

## Scripting Languages Lab

1. Develop a HTML document to basic alignments on headers and format the document using suitable tags.
2. Develop a HTML document which displays the arts and science department of your college and the courses offered by the department using list.
3. Develop a HTML document to create table with rows and columns and split them using rows span and column span.
4. Using CSS and HTML, make a webpage that has two columns. Each column should use half of the width of the page. The left half should have a light-gray background and the right half should have a light green background. The left half should have a list of the 5 best-selling books in Amazon's kindle store, and the right should have a list of your five favorite celebrities or athletes.
5. Develop a program to illustrate CSS border style properties.
6. Develop a JavaScript program to compute the sum of an array of integers.
7. Develop a JavaScript program to generate ten random numbers within 1 to 100 and display the numbers in a table.
8. Develop a JavaScript to create an Arithmetic Calculator using user defined Function.
9. Develop a JavaScript for loop that will iterate from 0 to 100. For each iteration, it will check if the current number is odd or even, and display a message to the screen.
10. Develop a JavaScript program to sum of sum digits of a given number.
11. Develop a JavaScript function to demonstrate the mathematical functions.
12. Develop a JavaScript program to demonstrate the various string functions.
13. Develop a Typescript program to demonstrate the control statements.
14. Develop a Typescript program to demonstrate the string functions.

## Scripting Languages Lab Manual

1. Develop a HTML document to basic alignments on headers and format the document using suitable tags.

### Aim:

To develop a HTML document to basic alignment on headers and format a document using suitable tags.

### Procedure:

1. Open a text editor and create a new file named basic\_alignments.html.
2. Add <html>, <head>, and <body> tags with a title in <head>.
3. Use <p align="left"> for left-aligned text and <p align="center"> for center-aligned text.
4. Add <h1> to <h6> tags for varying heading sizes.
5. Use <u> for underline, <i> for italic, <sup> for superscript, and <sub> for subscript.
6. Use <small> to decrease text size and <big> to increase text size.
7. Apply <tt> for teletype font and <del> for strike-through text.
8. Save the file with a .html extension and open it in a web browser to view the output.

### Program:

```
<html><head>
<title> Basic Alignments </title></head>
<body><h1> Basic Alignments </h1>
<p align=left>Hypertext Markup Language a standardized system for tagging text files to achieve
font, colour, graphic and hyperlink effects on worldwide webpages.</p>
<p align=center>Hypertext Markup Language a standardized system for tagging text files to
achieve font, colour, graphic and hyperlink effects on worldwide webpages.</p>
<h1>Different size of headings </h1><br>
<h1>Jamal Mohamed College </h1><br>
<h2>Jamal Mohamed College </h2><br>
<h3>Jamal Mohamed College </h3><br>
<h4>Jamal Mohamed College </h4><br>
<h5>Jamal Mohamed College </h5><br>
<h6>Jamal Mohamed College </h6><br>
<u>Jamal Mohamed College </u><br>
<i>Jamal Mohamed College </i><br>
```

## Scripting Languages Lab Manual

```
x<sup>3</sup>x<sub>3</sub>
<small>Jamal Mohamed College <small><br>
<big>Jamal Mohamed College </big><br>
<tt>Jamal Mohamed College </tt><br>
<del>Jamal Mohamed College </del><br></body></html>
```

### Output:

#### Basic Alignments

Hypertext Markup Language a standardized system for tagging text files to achieve font, colour, graphic and hyperlink effects on worldwide webpages.

Hypertext Markup Language a standardized system for tagging text files to achieve font, colour, graphic and hyperlink effects on worldwide webpages.

#### Different size of headings

## Jamal Mohamed College

### Jamal Mohamed College

#### Jamal Mohamed College

#### Jamal Mohamed College

#### Jamal Mohamed College

#### Jamal Mohamed College

#### Jamal Mohamed College

Jamal Mohamed College  
*Jamal Mohamed College*  
x<sup>3</sup>x<sub>3</sub> Jamal Mohamed College  
Jamal Mohamed College  
Jamal Mohamed College  
Jamal Mohamed College

### Result:

Thus, the HTML document was developed by using basic alignment on headers was executed successfully and the output was verified.

## Scripting Languages Lab Manual

2. Develop a HTML document which displays the arts and science department of your college and the courses offered by the department using list.

### Aim:

To develop a HTML document which display the arts and science department of your college and the courses offered by the department using list.

### Procedure:

1. Open a text editor and create a new file named college.html.
2. Define the basic HTML structure with <html>, <head>, and <body> tags.
3. Define Text Formatting for setting text color
4. Adding Paragraph with Underline:
5. Use <ol> for ordered lists, with the type attribute set to 1 (numbered list) or a (alphabetical list) to display courses offered by various departments.
6. Use <ul> for unordered lists, specifying type="circle", type="disc", and type="square" for different bullet styles for each department's courses.
7. Use the <hr> tag to add a horizontal line between sections for better readability.
8. Save the file as college.html and open it in a web browser to see the formatted text, ordered and unordered lists, and styling.

### Program:

```
<html><head>
<title> college </title>
<font color ="#FFFFFF">
<b> JAMAL MOHAMED COLLEGE <br>
AUTONOMOUS TIRUCHIRAPALLI </font><br>
</head>
<body>
<p align = "left"><u> Jamal Mohamed college is an arts and science college.<br>The courses
were offered by college based on various department </u></p>
<h1 align = "center">Using Ordered list </h1>
<h2><u> Department of Arabic and Tamil</u></h2>
<ol type = "1">
<li> B.A. (Arabic and Tamil)
```

## Scripting Languages Lab Manual

```
<li>M.Phil.
</ol><hr>
<h2><u> Department of English</u></h2>
<ol type = "a">
<li>B.A.(English)
<li>M.A.(English)
<li>M.Phil.
<li>Ph.D.
</ol>
<h2><u> Department of Computer Science</u></h2>
<ol type ="1">
<li>B.Sc(CS & IT)
<li>M.Sc. (CS & IT)
<li>BCA
<li>MCA
<li>M.Phil.
<li>Ph.D.
</ol><hr>
<h1 align="center">Using Unordered list </h1>
<h2><u> Department of Mathematics </u></h2>
<ul type = "circle">
<li>B.Sc
<li>M.Sc
<li>Ph.D
</ul><hr>
<h2><u> Department of Physics & Chemistry</u></h2>
<ul type="disc">
<li>B.Sc(Physics & Chemistry)
<li>M.Sc(Physics & Chemistry)
<li>M.Phil(Physics & Chemistry)
<li>Ph.D
```

## Scripting Languages Lab Manual

```
</ul><hr>
<h2><u>Department of Fashion Technology, N&D</u></h2>
<ul type = "square">
<li>B.Sc(FT & ND)
<li>M.Sc(FT & ND)
</ul><hr>
</body></html>
```

### Output:

**JAMAL MOHAMED COLLEGE**  
**AUTONOMOUS TIRUCHIRAPALLI**

Jamal Mohamed college is an arts and science college.  
The courses were offered by college based on various department

### Using Ordered list

#### Department of Arabic and Tamil

1. B.A. (Arabic and Tamil)
2. M.Phil.

---

#### Department of English

1. B.A.(English)
2. M.A.(English)
3. M.Phil.
4. Ph.D.

---

#### Department of Computer Science

1. B.Sc(CS & IT)
2. M.Sc. (CS & IT)
3. BCA
4. MCA
5. M.Phil.
6. Ph.D.

---

### Using Unordered list

#### Department of Mathematics

- B.Sc
- M.Sc
- Ph.D

---

#### Department of Physics & Chemistry

- B.Sc(Physics & Chemistry)
- M.Sc(Physics & Chemistry)
- M.Phil(Physics & Chemistry)
- Ph.D

### Result:

Thus, the HTML document was developed by using ordered and unordered list was executed successfully and the output was verified.

## Scripting Languages Lab Manual

3. Develop a HTML document to create table with rows and columns and split them using rows span and column span.

### Aim:

To develop a HTML document to create table with rows and columns and split them using row span and column span.

### Procedure:

1. Open a text editor and create a new file named table.html.
2. Add the basic HTML structure with <html>, <head>, and <body> tags.
3. Add the <table> tag with border="10", bgcolor="#80FFFF", and align="center" for styling and use <caption> to add a title to the table.
4. Use colspan and rowspan to make the row span and colspan across table.
5. Align Text and Setting Background Color using align and bgcolor attributes.
6. Save the file as table.html and open it in a web browser to view the table.

### Program:

```
<html><head>
<title> table </title></head>
<body><table border = 10 bgcolor=" #80FFFF" align = "center">
<caption>Major seed producing states in India</caption>
<tr><td colspan = 2 align ="center" bgcolor = "#80FFFF">
<tr><td rowspan = 4 align = center>
<b>TAMILNADU
<td> paddy
<tr><td> cotton
<tr><td> Blackgram
<tr><td> Greengram</b></tr>
<tr><td rowspan = 4 align = center>
<b>GUJARAT
<td> Castro
<tr><td> Greengram
<tr><td> Groundnut
<tr><td> Cotton</b></tr>
```

## Scripting Languages Lab Manual

```
<tr><td rowspan = 4 align = center>  
<b>HARYANA  
<td> Mustard  
<tr><td> wheat<tr><td> paddy  
</table></body></html>
```

### Output:

<b>TAMILNADU</b>	paddy
	cotton
	Blackgram
	Greengram
<b>GUJARAT</b>	Castor
	Greengram
	Groundnut
	Cotton
<b>HARYANA</b>	Mustard
	wheat
	paddy

### Result:

Thus, the HTML document was developed by using rows and columns span was executed in a table successfully and output was verified.



## Scripting Languages Lab Manual

4. Using CSS and HTML, make a webpage that has two columns. Each column should use half of the width of the page. The left half should have a light-gray background and the right half should have a light green background. The left half should have a list of the 5 best-selling books in Amazon's kindle store, and the right should have a list of your five favourite celebrities or athletes.

### Aim:

To develop a web page using css and html.

### Procedure:

1. Open a text editor and create a new file named two\_columns.html.
2. Add the basic HTML structure with <html>, <head>, and <body> tags.
3. Add CSS for Layout.
4. Create the Two Columns using <div> tag.
5. Add Content in Columns.
6. Save the file as two\_columns.html and open it in a web browser to view the two-column layout.

### Program:

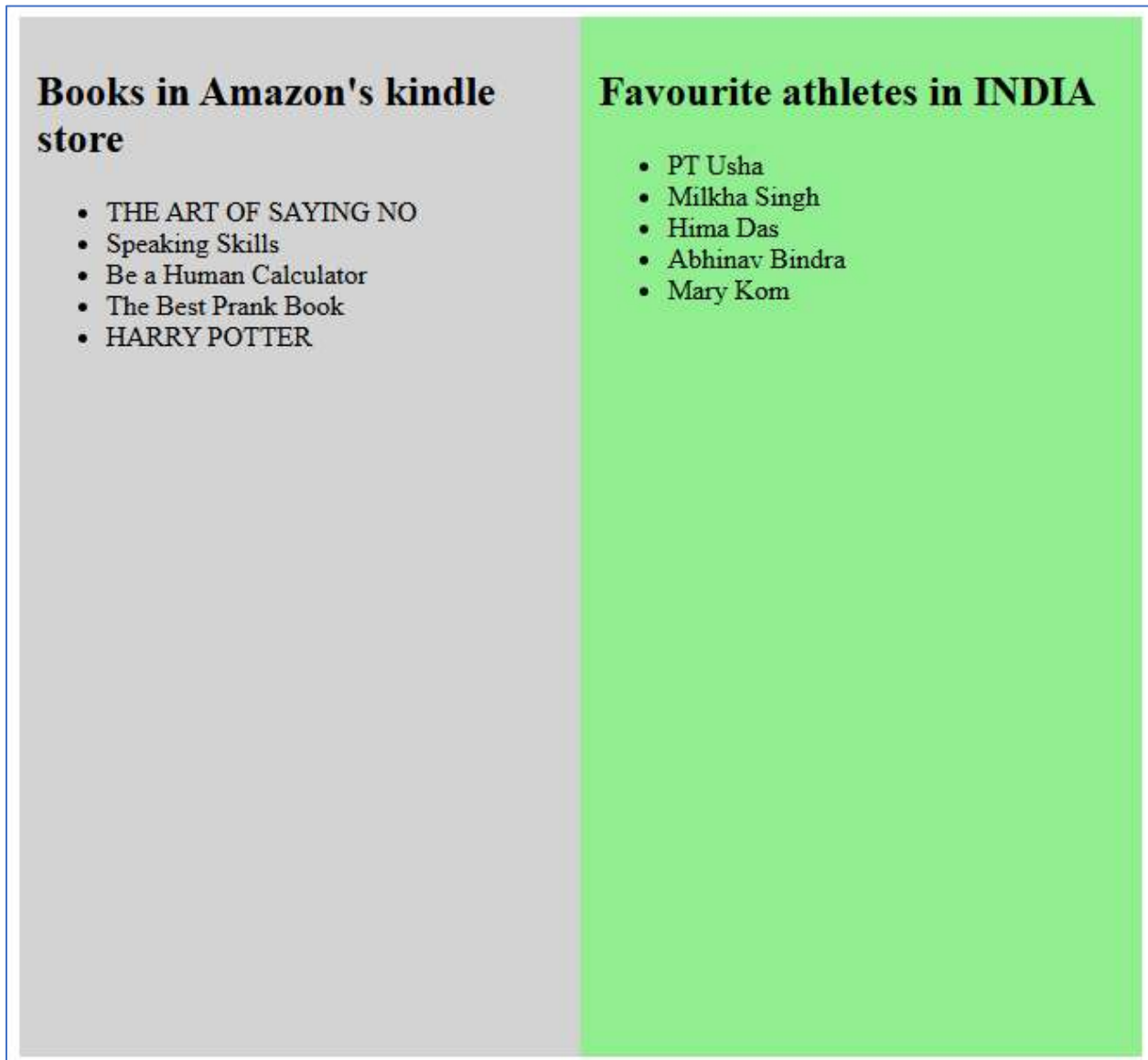
```
<html><head>
<title> WEB PAGE-TWO COLUMNS</title>
<style>
*{
  box-sizing: border-box;
}
.column {
float:left;
  width: 50%;
  padding: 10px;
height: 600px;
}
.row {
  content: "";
  display: table;
  clear: both;
```

## Scripting Languages Lab Manual

```
}
</style>
</head>
<body>
<div class="row">
<div class="column" style="background-color:lightgray;">
<h2>Books in Amazon's kindle store</h2>
<ul>
<li>THE ART OF SAYING NO
<li>Speaking Skills
<li> Be a Human Calculator
<li> The Best Prank Book
<li>HARRY POTTER
</ul>
</div>
<div class="column" style="background-color:lightgreen;">
<h2>Favourite athletes in INDIA</h2>
<ul>
<li>PT Usha
<li>Milkha Singh
<li>Hima Das
<li> Abhinav Bindra
<li>Mary Kom
</ul>
</div></div></body></html>
```

## Scripting Languages Lab Manual

### Output:



### Result:

Thus, the web page was developed successfully using CSS and HTML.

## Scripting Languages Lab Manual

5. Develop a program to illustrate CSS border style properties

### Aim:

To illustrate the use of CSS border style properties in HTML document.

### Procedure:

1. Create the HTML File 'cssborder.html' which includes external CSS file for designing the web page
2. Add Content to the Body using multiple heading tags (<h1>, <h2>, <h3>, <h4>, <h5>, <h6>) with different text describing the type of border applied.
3. Create the CSS File 'borderstyles.css' for setting effects for various tags.
4. Open cssborder.html in a web browser to view the headings with various border styles.

Program:

### cssborder.html

```
<html><head><title>CSS BORDER SYLE PROPERTIES</title>
<link rel="stylesheet" href="borderstyles.css"></link>
</head><body>
<h6> SPECIFIES DASHED BORDER</h6>
<h5> SPECIFIES SOLID BORDER</h5>
<h4> SPECIFIES DOUBLE BORDER</h4>
<h3> SPECIFIES A 3D RIDGED BORDER </h3>
<h2> SPECIFIES A 3D INSET BORDER</h2>
<h1> SPECIFIES A 3D OUTSET BORDER </h1>
</body></html>
```

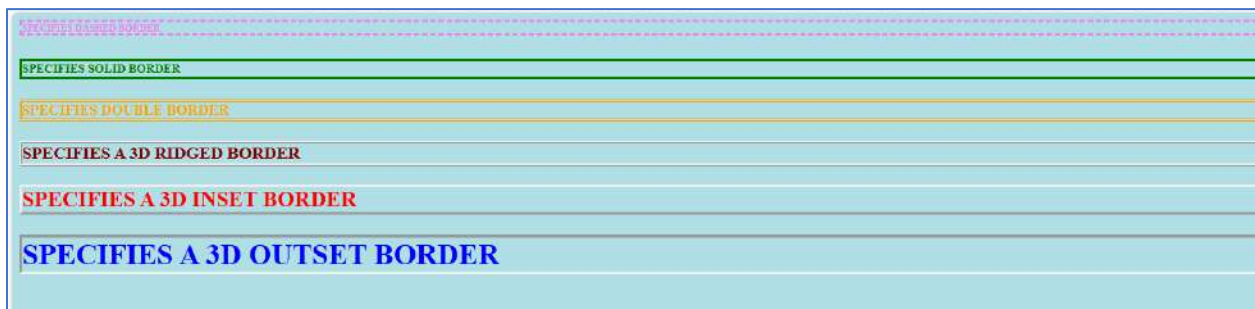
### borderstyles.css

```
body {
background-color:powderblue;
}
h6{
color:violet;
border-style:dashed;
}
h5{
```

## Scripting Languages Lab Manual

```
color:green;
border-style:solid;
}
h4{
color:orange;
border-style:double;
}
h3{
color:maroon;
border-style:ridge;
}
h2{
color:red;
border-style:outset;
}
h1{
color:blue;
border-style:inset;
}
```

### Output:



### Result:

Thus, the HTML document was developed by using CSS border style properties was executed successfully and the output was verified.

## Scripting Languages Lab Manual

6. Develop a JavaScript program to compute the sum of an array of integers.

### Aim:

To write a program to compute the sum of an array of integers using javascript.

### Procedure:

1. Open a text editor and create a new file named sumArray.html.
2. Add the basic HTML structure: <html>, <head>, and <body> tags.
3. Inside the <script> tag, define an array of integers: var array = [1, 2, 3, 4, 5, 6].
4. Initialize a variable s = 0 to store the sum of the array elements.
5. Use a for loop to iterate over the array and add each element to the sum
6. After the loop, use document.write to display the sum of the array on the webpage.
7. Open the HTML file in a web browser to view the result. It will display the sum of the integers in the array.

### Program:

```
<html><head>  
<title>Compute the sum of an array of integers</title>  
<script> var array = [1, 2, 3, 4, 5, 6], s = 0, i;  
for (i = 0; i < array.length; i += 1)  
s += array[i];  
document.write("Sum of array of Integers: "+s);  
</script></head>  
<body></body></html>
```

### Output:

Sum of array of Integers: 21

### Result:

Thus, the calculation of sum of array of integers program was developed and executed successfully.

## Scripting Languages Lab Manual

7. Develop a JavaScript program to generate ten random numbers within 1 to 100 and display the numbers in a table.

### Aim:

To write a program to generate the ten random numbers from 1 to 100 using javascript

### Procedure:

1. Open a text editor and create a new file named generateRandomNumbers.html.
2. Add the basic HTML structure: <html>, <head>, and <body> tags.
3. Inside the <script> tag, use document.write("<table border =1>"); to create a table with a border to display the random numbers.
4. Use a for loop and Math.random() function to generate ten random numbers.
5. Write each random number inside a <td>.
6. Open the HTML file in a web browser to view the result, where ten random numbers between 1 and 100 will be displayed in a table.

### Program:

```
<html>
<head>
<title>Generate random numbers</title>
<script>
document.write("<h2>Ten Random numbers from 1 to 100: </h2>");
document.write("<table border =1>");
for (var i = 0; i < 10; i++)
{
var randno = Math.floor(Math.random() * 100) + 1;
document.write("<td>"+randno);
}
document.write("</td>");
document.write("</table>");
</script> </head>
<body></body></html>
```

## Scripting Languages Lab Manual

**Output:**

**Ten Random numbers from 1 to 100:**

41	44	45	41	25	51	79	21	36	94
----	----	----	----	----	----	----	----	----	----

**Result:**

Thus, the generation of random number program was developed and executed successfully.



## Scripting Languages Lab Manual

8. Develop a JavaScript to create an Arithmetic Calculator using user defined Function

### Aim:

To develop a javascript program to create an Arithmetic Calculator using user defined function.

### Procedure:

1. Open a text editor and create a new file named calculator.html.
2. Add the basic HTML structure, including <html>, <head>, and <body> tags.
3. Inside the <script> tag, define four functions: multiply(), addition(), subtraction(), and division().
4. Each function retrieves the values of two numbers from the input fields, performs the respective arithmetic operation, and displays the result in the total field.
5. The JavaScript functions use Number() to convert the input values into numbers before performing the operations.
6. Create the Form with three text input fields and four buttons to perform the required operations.
7. Open the HTML file in a web browser. Enter numbers in the "Number 1" and "Number 2" fields, then click the buttons to perform the arithmetic operations. The result will be displayed in the "Result" field.

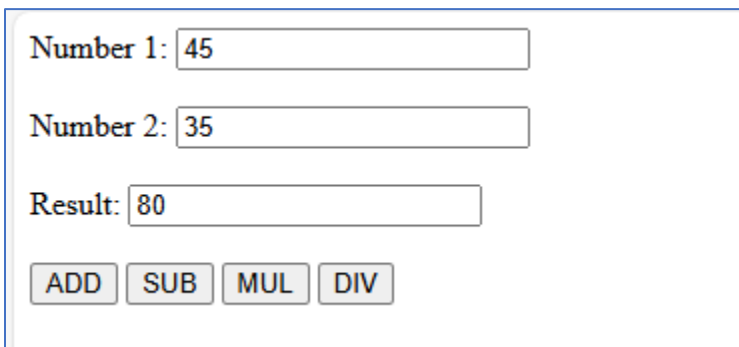
### Program:

```
<html><head><title> Arithmetic Calculator</title>
<script type="text/javascript">
function multiply(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a*b;
document.calculator.total.value=c;}
function addition(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a+b;
document.calculator.total.value=c;}
function subtraction(){
```

## Scripting Languages Lab Manual

```
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a-b;
document.calculator.total.value=c;}
function division(){
a=Number(document.calculator.number1.value);
b=Number(document.calculator.number2.value);
c=a/b;
document.calculator.total.value=c;}
</script></head>
<body><form name="calculator">
Number 1: <input type="text" name="number1"><br><br>
Number 2: <input type="text" name="number2"><br><br>
Result: <input type="text" name="total"> <br><br>
<input type="button" value="ADD" onclick="addition();">
<input type="button" value="SUB" onclick="subtraction();">
<input type="button" value="MUL" onclick="multiply();">
<input type="button" value="DIV" onclick="division();">
</form></body></html>
```

### Output:



Number 1: 45

Number 2: 35

Result: 80

ADD SUB MUL DIV

### Result:

Thus, the program to create arithmetic calculator was developed and executed successfully.

## Scripting Languages Lab Manual

9. Develop a JavaScript for loop that will iterate from 0 to 100. For each iteration, it will check if the current number is odd or even, and display a message to the screen.

### Aim:

To write a javascript program to generate for loop that will iterate from 0 to 100 and display whether it is odd or even.

### Procedure:

1. Open a text editor and create a new file named odd-even.html.
2. Add the basic HTML structure, including <html>, <head>, and <body> tags.
3. Inside the <script> tag, use a for loop to iterate through numbers from 0 to 100.
4. Inside the loop, use an if-else condition to check if a number is even or odd.
5. If the number is 0 or divisible by 2, display it as "even". Otherwise, display it as "odd".
6. The results will be displayed using document.write().
7. Open the HTML file in a web browser. The browser will display numbers from 0 to 100 along with whether they are odd or even.

### Program:

```
<html><head><title>Odd/Even</title>
<script>
for(var num=0;num<=100;num++) {
if(num==0){
document.write(num + " is even");
document.write("<br>");
}
else if (num%2==0) {
document.write(num + " is even");
document.write("<br>");
}
else{
document.write(num + " is odd");
document.write("<br>");
}}
</script></head></html>
```

## Scripting Languages Lab Manual

### Output:

```
0 is even
1 is odd
2 is even
3 is odd
4 is even
5 is odd
6 is even
7 is odd
8 is even
9 is odd
10 is even
11 is odd
12 is even
13 is odd
14 is even
15 is odd
16 is even
17 is odd
18 is even
19 is odd
20 is even
21 is odd
22 is even
23 is odd
24 is even
25 is odd
26 is even
27 is odd
28 is even
29 is odd
30 is even
31 is odd
32 is even
33 is odd
34 is even
35 is odd
36 is even
37 is odd
38 is even
39 is odd
40 is even
41 is odd
42 is even
43 is odd
44 is even
45 is odd
46 is even
47 is odd
48 is even
49 is odd
50 is even
```

### Result:

Thus, the program to print odd or even was developed and executed successfully.

## Scripting Languages Lab Manual

10. Develop a JavaScript program to sum of sum digits of a given number.

### Aim:

To write a javascript program that calculates the sum of the digits of a given number.

### Procedure:

1. Open a text editor and create a new file named sum-of-digits.html.
2. Add the basic HTML structure, including <html>, <head>, and <body> tags.
3. Inside the <script> tag, define a function calculateSumOfDigits() that gets the value from the input field.
4. If the input is invalid (not a number), display an error message in red.
5. If the input is valid, convert it to a string, split the string into individual digits, and sum them.
6. Display the result in green inside the result div.
7. Open the HTML file in a web browser. Enter a number and click the "Calculate" button to see the sum of its digits.

### Program:

```
<html>
<head><title>Sum of Digits</title></head>
<body>
<h1>Find the Sum of Digits</h1>
<label for="numberInput">Enter a number:</label>
<input type="text" id="numberInput" placeholder="Enter a number">
<button onclick="calculateSumOfDigits()">Calculate</button>
<div id="result" class="output"></div>
<script>
function calculateSumOfDigits() {
const input = document.getElementById("numberInput").value;
const resultDiv = document.getElementById("result");
if (isNaN(input) || input.trim() === "") {
resultDiv.textContent = "Please enter a valid number.";
resultDiv.style.color = "red";
return;
}
```

## Scripting Languages Lab Manual

```
const number = Math.abs(parseInt(input, 10));
let sum = 0;
number.toString().split("").forEach(digit => {
sum += parseInt(digit, 10);
});
resultDiv.textContent = `The sum of the digits of ${input} is ${sum}.`;
resultDiv.style.color = "green";
}
</script></body></html>
```

### Output:

## Find the Sum of Digits

Enter a number:

The sum of the digits of 56789 is 35.

### Result:

Thus, the program to sum the digits of a given number was developed and executed successfully.

## Scripting Languages Lab Manual

11. Develop a JavaScript function to demonstrate the mathematical functions.

### Aim:

To write a javascript program that demonstrates the use of various mathematical functions.

### Procedure:

1. Open a text editor and create a new file named math-functions-demo.html.
2. Add a button labeled "Run Math Functions". This button will trigger the JavaScript function when clicked.
3. Add a <div> element with an id of "results" to display the results of the Math functions.
4. Inside the <script> tag, define a function demonstrateMathFunctions() that performs various Math operations such as abs, sqrt, pow, round, random, sin, log, max.
5. For each operation, append the result to the results div.
6. Open the HTML file in a web browser. Click the "Run Math Functions" button to see the results of the various Math operations.

### Program:

```
<html><head><title>Math Functions Demo</title></head>
<body><h1>JavaScript Math Functions</h1>
<button onclick="demonstrateMathFunctions()">Run Math Functions</button>
<div id="results" class="output"></div>
<script>
function demonstrateMathFunctions() {
const resultsDiv = document.getElementById("results");
resultsDiv.innerHTML = "";
const number = -42;
const square = 64;
const base = 3, exponent = 4;
const decimal = 7.8;
const nums = [10, 5, -3, 99, 0];
const angle = Math.PI / 4;
const logNumber = 10;
function appendResult(description, value) {
const paragraph = document.createElement("p");
```

## Scripting Languages Lab Manual

```
paragraph.innerHTML = `${description}:</strong> ${value}`;  
resultsDiv.appendChild(paragraph);  
}  
appendResult(`Absolute value of ${number}`, Math.abs(number));  
appendResult(`Square root of ${square}`, Math.sqrt(square));  
appendResult(`${base} raised to the power of ${exponent}`, Math.pow(base, exponent));  
appendResult(`Rounding ${decimal}`, Math.round(decimal));  
appendResult(`Ceiling of ${decimal}`, Math.ceil(decimal));  
appendResult(`Floor of ${decimal}`, Math.floor(decimal));  
appendResult(`Random number (0-1)`, Math.random().toFixed(4));  
appendResult(`Random integer (1-100)`, Math.floor(Math.random() * 100) + 1);  
appendResult(`Sine of 45 degrees`, Math.sin(angle).toFixed(4));  
appendResult(`Cosine of 45 degrees`, Math.cos(angle).toFixed(4));  
appendResult(`Tangent of 45 degrees`, Math.tan(angle).toFixed(4));  
appendResult(`Natural logarithm of ${logNumber}`, Math.log(logNumber).toFixed(4));  
appendResult(`Logarithm base 10 of ${logNumber}`, Math.log10(logNumber).toFixed(4));  
appendResult(`Maximum in [${nums}]`, Math.max(...nums));  
appendResult(`Minimum in [${nums}]`, Math.min(...nums));  
}  
</script></body></html>
```



Output:

# JavaScript Math Functions

Run Math Functions

**Absolute value of -42:** 42

**Square root of 64:** 8

**3 raised to the power of 4:** 81

**Rounding 7.8:** 8

**Ceiling of 7.8:** 8

**Floor of 7.8:** 7

**Random number (0-1):** 0.9519

**Random integer (1-100):** 73

**Sine of 45 degrees:** 0.7071

**Cosine of 45 degrees:** 0.7071

**Tangent of 45 degrees:** 1.0000

**Natural logarithm of 10:** 2.3026

**Logarithm base 10 of 10:** 1.0000

**Maximum in [10,5,-3,99,0]:** 99

**Minimum in [10,5,-3,99,0]:** -3

**Result:**

Thus, the program to demonstrate the use of various mathematical functions was developed and executed successfully.

## Scripting Languages Lab Manual

12. Develop a JavaScript program to demonstrate the various string functions.

### Aim:

To write a javascript program that demonstrates the use of various string functions.

### Procedure:

1. Open a text editor and create a new file named string-functions-demo.html.
2. Add a button labeled "Run String Functions" that triggers the JavaScript function when clicked.
3. Add a <div> element with the id "results" to display the results of the string functions.
4. Inside the <script> tag, define the function demonstrateStringFunctions() that performs various string operations such as length, toUpperCase, toLowerCase, charAt, indexOf, substring, replace, includes, split, trim, concat and repeat.
5. For each operation, append the result to the results div.
6. Open the HTML file in a web browser. Click the "Run String Functions" button to see the results of the string operations.

### Program:

```
<html><head>
<title>String Functions Demo</title></head>
<body><h1>JavaScript String Functions</h1>
<button onclick="demonstrateStringFunctions()">Run String Functions</button>
<div id="results" class="output"></div>
<script>
function demonstrateStringFunctions() {
const resultsDiv = document.getElementById("results");
resultsDiv.innerHTML = "";
const sampleString = "Hello, World!";
const searchString = "World";
function appendResult(description, value) {
const paragraph = document.createElement("p");
paragraph.innerHTML = `<strong>${description}</strong> ${value}`;
resultsDiv.appendChild(paragraph);
}
appendResult(`Original String`, sampleString);
```

## Scripting Languages Lab Manual

```
appendResult(`Length of String`, sampleString.length);
appendResult(`Uppercase`, sampleString.toUpperCase());
appendResult(`Lowercase`, sampleString.toLowerCase());
appendResult(`Character at index 7`, sampleString.charAt(7));
appendResult(`Index of '${searchString}'`, sampleString.indexOf(searchString));
appendResult(`Substring (0 to 5)`, sampleString.substring(0, 5));
appendResult(`Slice (0 to 5)`, sampleString.slice(0, 5));
appendResult(`Replacing 'World' with 'JavaScript', sampleString.replace("World",
"JavaScript"));
appendResult(`Includes 'World'`, sampleString.includes(searchString));
appendResult(`Starts with 'Hello'`, sampleString.startsWith("Hello"));
appendResult(`Ends with '!'`, sampleString.endsWith("!"));
appendResult(`Split by ','`, sampleString.split(",").join(" | "));
appendResult(`Trimmed String`, " Trim this string! ".trim());
appendResult(`Concatenation`, sampleString.concat(" Let's learn strings!"));
appendResult(`Repeat String (3 times)`, sampleString.repeat(3));
}
</script></body></html>
```

Output:

```
JavaScript String Functions  
  
Run String Functions  
  
Original String: Hello, World!  
Length of String: 13  
Uppercase: HELLO, WORLD!  
Lowercase: hello, world!  
Character at index 7: W  
Index of 'World': 7  
Substring (0 to 5): Hello  
Slice (0 to 5): Hello  
Replacing 'World' with 'JavaScript': Hello, JavaScript!  
Includes 'World': true  
Starts with 'Hello': true  
Ends with '!': true  
Split by ',': Hello | World!  
Trimmed String: Trim this string!  
Concatenation: Hello, World! Let's learn strings!  
Repeat String (3 times): Hello, World!Hello, World!Hello, World!
```

**Result:**

Thus, the program to demonstrate the use of various string functions was developed and executed successfully.

## Scripting Languages Lab Manual

13. Develop a Typescript program to demonstrate the control statements

### Aim:

To write a typescript program that demonstrates the use of various control statements.

### Procedure:

1. Open a text editor and create a new file named typescript-control-statements.html.
2. Include the TypeScript CDN by adding the following `<script>` tag inside the `<head>` section:  
`<script src="https://cdn.jsdelivr.net/npm/typescript/lib/typescript.min.js"></script>`
3. Add a button labeled "Run Control Statements" that will trigger the TypeScript code execution.
4. Add a `<div>` element with the id results to display the output of control statements.
5. Inside the `<script>` tag, define a string `tsCode` containing the TypeScript code to demonstrate the following control structures:

If-Else Statement: Checks if a number is positive.

Switch Statement: Determines the day of the week based on a number (1-7).

For Loop: Iterates through numbers from 1 to 5 and displays them.

6. The function `demonstrateControlStatements()` is used to execute these control structures and display the results in the results div.

7. The TypeScript code is transpiled to JavaScript using the `ts.transpileModule()` method.

8. The transpiled JavaScript is then executed using `eval()`.

9. Open the file in a web browser. Click the "Run Control Statements" button to see the results of the control statements in the results div.

### Program:

```
<html><head>
<title>TypeScript Control Statements</title>
<script src="https://cdn.jsdelivr.net/npm/typescript/lib/typescript.min.js"></script></head>
<body><h1>TypeScript Control Statements</h1>
<button id="runButton">Run Control Statements</button>
<div id="results" class="output"></div>
<script>
const tsCode = `
function demonstrateControlStatements(): void {
const resultsDiv = document.getElementById("results");
```

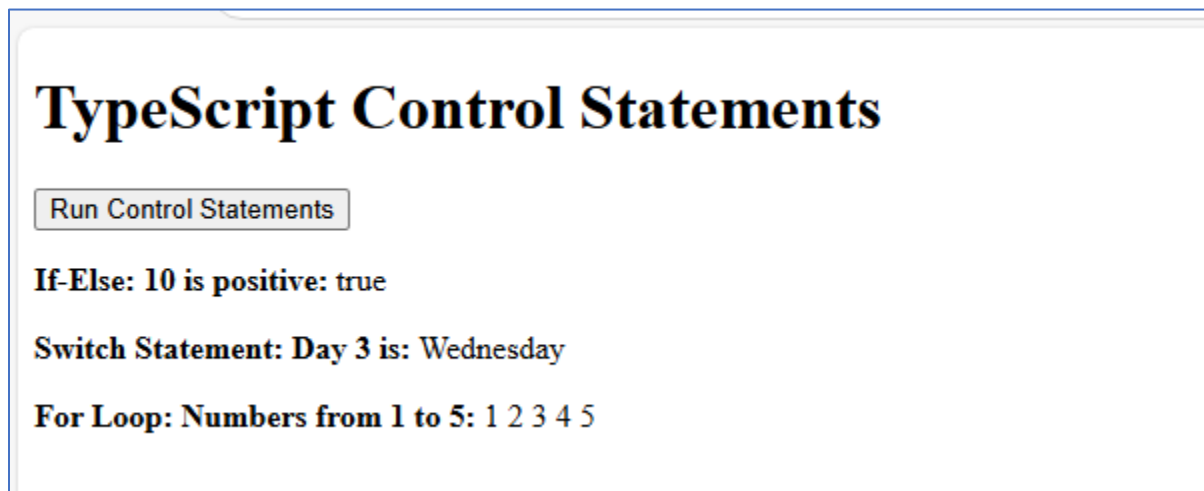
## Scripting Languages Lab Manual

```
if (resultsDiv) resultsDiv.innerHTML = "";
const appendResult = (description: string, value: any) => {
const paragraph = document.createElement("p");
paragraph.innerHTML = `<strong>${description}</strong> ${value}`;
if (resultsDiv) resultsDiv.appendChild(paragraph);
};
const number = 10;
if (number > 0) {
appendResult(`If-Else: ${number} is positive`, true);
} else {
appendResult(`If-Else: ${number} is not positive`, false);
}
const day = 3;
let dayName = "";
switch (day) {
case 1:
dayName = "Monday";
break;
case 2:
dayName = "Tuesday";
break;
case 3:
dayName = "Wednesday";
break;
default:
dayName = "Unknown";
}
appendResult(`Switch Statement: Day ${day} is`, dayName);
let forLoopResult = "";
for (let i = 1; i <= 5; i++) {
forLoopResult += `${i} `;
}
```

## Scripting Languages Lab Manual

```
}
appendResult("For Loop: Numbers from 1 to 5", forLoopResult);
}
`;
document.getElementById("runButton")?.addEventListener("click", () => {
const jsCode = ts.transpileModule(tsCode, { compilerOptions: { module: ts.ModuleKind.ES2015
} }).outputText;
eval(jsCode);
demonstrateControlStatements();
});
</script></body></html>
```

### Output:



### Result:

Thus, the typescript program to demonstrate the use of various control statements was developed and executed successfully.

## Scripting Languages Lab Manual

14. Develop a Typescript program to demonstrate the string functions

### Aim:

To write a typescript program that demonstrates the use of various string functions.

### Procedure:

1. Open a text editor and create a new file named typescript-string-functions.html.
2. Include the TypeScript CDN by adding the following <script> tag inside the <head> section:  
<script src="https://cdn.jsdelivr.net/npm/typescript/lib/typescript.min.js"></script>
3. Add a button labeled "Run String Functions" that will trigger the TypeScript code execution.
4. Add a <div> element with the id results to display the output of string functions.
5. Inside the <script> tag, define a string tsCode containing the TypeScript code that demonstrates various string functions.
6. The demonstrateStringFunctions() function encapsulates the logic to apply each string function and append the results to the results div.
7. Transpile the TypeScript code to JavaScript using ts.transpileModule() method.
8. Use eval() to run the transpiled code.
9. Open the file in a web browser. Click the "Run String Functions" button to see the output of the TypeScript string functions in the results div.

### Program:

```
<html><head><title>TypeScript String Functions</title>
<script src="https://cdn.jsdelivr.net/npm/typescript/lib/typescript.min.js"></script></head>
<body><h1>TypeScript String Functions</h1>
<button id="runButton">Run String Functions</button>
<div id="results" class="output"></div>
<script>
const tsCode = `
function demonstrateStringFunctions(): void {
const resultsDiv = document.getElementById("results");
if (resultsDiv) resultsDiv.innerHTML = "";
const appendResult = (description: string, value: any) => {
const paragraph = document.createElement("p");
paragraph.innerHTML = `<strong>\${description}</strong> \${value}`;

```



## Scripting Languages Lab Manual

```
if (resultsDiv) resultsDiv.appendChild(paragraph);
};
const sampleString = "Hello, TypeScript!";
const searchString = "TypeScript";
appendResult(`Original String`, sampleString);
appendResult(`Length of String`, sampleString.length);
appendResult(`Uppercase`, sampleString.toUpperCase());
appendResult(`Lowercase`, sampleString.toLowerCase());
appendResult(`Character at Index 7`, sampleString.charAt(7));
appendResult(`Index of "${searchString}`, sampleString.indexOf(searchString));
appendResult(`Substring (0 to 5)`, sampleString.substring(0, 5));
appendResult(`Slice (0 to 5)`, sampleString.slice(0, 5));
appendResult(`Replacing 'TypeScript' with 'World`, sampleString.replace("TypeScript",
"World"));
appendResult(`Includes 'TypeScript`, sampleString.includes(searchString));
appendResult(`Starts with 'Hello`, sampleString.startsWith("Hello"));
appendResult(`Ends with '!`, sampleString.endsWith("!"));
appendResult(`Split by ',`, sampleString.split(",").join(" | "));
appendResult(`Trimmed String`, " Trim this string! ".trim());
appendResult(`Concatenation`, sampleString.concat(" Let's learn strings!"));
appendResult(`Repeat String (3 times)`, sampleString.repeat(3));
}
`;
document.getElementById("runButton")?.addEventListener("click", () => {
const jsCode = ts.transpileModule(tsCode, { compilerOptions: { module: ts.ModuleKind.ES2015
} }).outputText;
eval(jsCode);
demonstrateStringFunctions();
});
</script></body></html>
```

### Output:

```
TypeScript String Functions  
  
Run String Functions  
  
Original String: Hello, TypeScript!  
Length of String: 18  
Uppercase: HELLO, TYPESCRIPT!  
Lowercase: hello, typescript!  
Character at Index 7: T  
Index of 'TypeScript': 7  
Substring (0 to 5): Hello  
Slice (0 to 5): Hello  
Replacing 'TypeScript' with 'World': Hello, World!  
Includes 'TypeScript': true  
Starts with 'Hello': true  
Ends with '!': true  
Split by ',': Hello | TypeScript!  
Trimmed String: Trim this string!  
Concatenation: Hello, TypeScript! Let's learn strings!  
Repeat String (3 times): Hello, TypeScript!Hello, TypeScript!Hello, TypeScript!
```

### Result:

Thus, the typescript program to demonstrate the use of various string functions was developed and executed successfully.