Internet Tool and Technology Lab (ITT 3266) of VI sem B. Tech. IT Degree at MIT, Manipal
CONTENTS

Course Objectives

- To design and develop static and dynamic web pages.
- To familiarize with Client-Side Programming, Server-Side Programming, Active Server Pages.
- To learn Database Connectivity to web applications

Course Outcomes

At the end of this course, students will be able to

- Design and develop dynamic web pages with good aesthetic sense of designing and latest technical know-how's.
- Have a good understanding of Web Application Terminologies, Internet Tools other web services.
- Learn how to link and publish web sites.

Evaluation plan

Split up of 60 marks for Regular Lab Evaluation
Six regular evaluations will be carried out in alternate weeks. Each evaluation is for 10 marks and following is the split up: Record : 4 Marks Evaluation: 4 Marks Execution: 2 Marks Total = 10 Marks Total Internal Marks: 6 * 10 =60 Marks
End Semester Lab evaluation: 20 marks (Duration 2 hrs)
Program write up: 08Marks Program execution: 12 Marks Total: 8 + 12 = 20 Marks
Project evaluation: 20 marks

INSTRUCTIONS TO THE STUDENTS
Pre- Lab Session Instructions

1. Students should carry the Lab Manual Book and the required stationery to every lab session
2. Be in time and follow the institution dress code
3. Must sign in the log register provided
4. Make sure to occupy the allotted seat and answer the attendance
5. Adhere to the rules and maintain the decorum

In- Lab Session Instructions

- Follow the instructions on the allotted exercises
- Show the program and results to the instructors on completion of experiments
- On receiving approval from the instructor, copy the program and results in the lab record
- Prescribed textbooks and class notes can be kept ready for reference if required

General Instructions for the exercises in Lab

- Implement the given exercise individually and not in a group.
- The programs should meet the following criteria:
  - Programs should be interactive with appropriate prompt messages, error messages if any, and descriptive messages for outputs.
  - Comments should be used to give the statement of the problem.
  - Statements within the program should be properly indented.
- Plagiarism (copying from others) is strictly prohibited and would invite severe penalty in evaluation.
- In case a student misses a lab, he/ she must ensure that the experiment is completed before the next evaluation with the permission of the faculty concerned.
- Students missing out lab on genuine reasons like conference, sports or activities assigned by the Department or Institute will have to take **prior permission** from the HOD to attend **additional lab**(with other batch) and complete it **before** the student goes on leave. The student could be awarded marks for the write up for that day provided he submits it during the **immediate** next lab.

- Students who fall sick should get permission from the HOD for evaluating the lab records. However attendance will not be given for that lab.
- Students will be evaluated only by the faculty with whom they are registered even though they carry out additional experiments in other batch.
- Presence of the student during the lab end semester exams is mandatory even if the student assumes he has scored enough to pass the examination
- Minimum attendance of 75% is mandatory to write the final exam.
- If the student loses his book, he/she will have to rewrite all the lab details in the lab record.
- Questions for lab tests and examination are not necessarily limited to the questions in the manual, but may involve some variations and / or combinations of the questions.

THE STUDENTS SHOULD NOT
- Bring mobile phones or any other electronic gadgets to the lab.
- Go out of the lab without permission.

LAB NO: 1   **Date:**

HTML - I

Objectives

- To familiarize with various HTML tags.
- To gain knowledge about how to develop simple web pages using various tags.

HTML

HTML stands for Hypertext Markup Language, is the predominant markup language for web pages. HTML provides it means to create a structured document such as headings, paragraphs, lists, links, quotes, and other so many items.

Basic html Program

```
<html>
<head>
<title>My First Webpage</title>
</head>
<body>
This is my first homepage.
</body>
</html>
```

Explanation

The text between <html> and </html> describes an HTML document.
The text between <head> and </head> provides information about the document
The text between <title> and </title> provides a title for the document.
The text between <body> and </body> describes the visible page content.

HTML Tags

- HTML tags are keywords (tag names) surrounded by angle brackets.
- The end tag is written like the start tag, but with a **slash** before the tag name.
- The start tag is often called the opening tag. The end tag is often called the closing tag.

HTML Elements

- HTML elements are written with a **start** tag, with an **end** tag, with the **content** in between.
- Syntax : *<tagname>content</tagname>*

HTML Attributes

- HTML elements can have attributes
- Attributes provide additional information about an element
- Attributes are always specified in the start tag
- Attributes come in name/value pairs like: name="value"
- Syntax : *<tagname attributename = "value">content</tagname>*

HTML Colors

There are following three different methods to set colors in your web page:

- Color names - You can specify color names directly like green, blue or red.
- Hex codes - A six-digit code representing the amount of red, green, and blue that makes up the color. The first two digits (RR) represent a red value, the next two are a green value(GG), and the last are the blue value(BB).
  E.g. Black = #000000.
- Color decimal or percentage values - This value is specified using the rgb( ) property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.
  E.g. Black = rgb(0,0,0).

Commonly used tags and elements.

The commonly used tags and elements in html are listed below with suitable examples.

Table 1.1 HTML Tags

Tag	Name	Example	Explanation
<!-- -->	Comment	<!-- Write your comments here -->	Comment tags <!-- and --> are used to insert comments in HTML.
<p>	Paragraph	<body> This is first Paragraphs </body>	Introduces paragraphs in the web documents.
<h1> to <h6>	Heading	<body> Heading Tag Heading Tag Heading Tag Heading Tag </body>	Produces headings of different font sizes.
 	Line Break	<body> Hello You delivered your assignment ontime. Thanks Mahnaz	Anything following the tag starts in next line.

<pre>	Preformatte-d text	<pre>
		  My Bonnie lies over the ocean.
		
		  My Bonnie lies over the sea.
		
		  My Bonnie lies over the ocean.
		
		  Oh, bring back my Bonnie to me.
		</pre>
			The text inside a <pre> element is displayed in a fixed-width font (usually Courier), and it preserves both spaces and line breaks

style	Style attribute	<body style="background-color:lightgrey;">	Setting the style of an HTML element, can be done with the style

			attribute. The HTML style attribute has the following syntax: *style="property: value;"* where the property is a CSS property and the value is a CSS value.
<center>	Center	<center> This text is in the center. </center>	Used to center the contents.
<a>	Anchor tag	This is a link	HTML links are defined with the <a> tag. The link's destination is specified in the href attribute.
<img>	Image	<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">	HTML images are defined with the <img> tag.The source file (src), alternative text (alt), and size (width and height) are provided as attributes
 	Bold Strong	The following word uses a bold typeface.	Anything that appears within ... or … element, is displayed in bold typeface.
<i>	Italic	The following word uses a <i>italicized</i> typeface.	Anything that appears within <i>...</i> element is displayed in italicized format.
<u>	Underline	The following word uses a <u>underlined</u> typeface.	Anything that appears within <u>...</u> element, is displayed with underline.
<mark>	Marked formatting	HTML <mark>Marked</mark> Formatting	The HTML <mark> element defines marked or highlighted text.
<sub>	Subscript Formatting	This is _{subscripted} text.	The HTML <sub> element defines subscripted text.
<sup>	Superscript Formatting	This is ^{superscripted} text.	The HTML <sup> element defines superscripted text.

`<q>` or `<blockquote>`	Quotations	`<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>`	Browsers usually insert quotation marks around the `<q>` element. For long quotations `<blockquote>` element can be used.
`<cite>`	Text Citations	`<p>This HTML tutorial is derived from <cite>W3 Standard for HTML</cite>.</p>`	`<cite>` element defines the title of a work.

`<ul>`	Unordered List	`<ul>`
		`<li>Coffee</li>`
		`<li>Tea</li>`
		`<li>Milk</li>`
		`</ul>`
			An unordered list starts with the `<ul>` tag. Each list item starts with the `<li>` tag.

`<ol>`	Ordered List	`<ol>`
		` <li>Coffee</li>`
		` <li>Tea</li>`
		` <li>Milk</li>`
		`</ol>`
			An ordered list starts with the `<ol>` tag. Each list item starts with the `<li>` tag. To specify the type of numbering you require type attribute can be affixed with the `<ol>` tag.
			
			• *<ol type="1"> - Default-Case Numerals.*
			• *<ol type="I"> - Upper-Case Numerals.*
			• *<ol type="i"> - Lower-Case Numerals.*
			• *<ol type="a"> - Lower-Case Letters.*
			• *<ol type="A"> - Upper-Case Letters.*

`<dl>`	Definition List	`<body>`
		`<dl>`
		`<dt><b>HTML</b></dt>`
		`<dd>This stands for Hyper Text Markup Language</dd>`
			Definition List makes use of following three tags.
			
			• `<dl>` - Defines the start of the list
			• `<dt>` - A term

		<dt>HTTP</dt> <dd>This stands for Hyper Text Transfer Protocol</dd> </dl> </body>	• <dd> - Term definition • </dl> - Defines the end of the list
<table>	Table	<body> <table border="1"> <tr> <th>Name</th> <th>Salary</th> </tr> <tr> <td>Ramesh Raman</td> <td>5000</td> </tr> </table> </body>	The HTML tables are created using the <table> tag in which • <tr> tag is used to create table rows • <td> tag is used to create data cells • <th> tag creates table headings • border is an attribute used to create table borders
<form>	Form	<form action="action_page.php" method="get"> First name:
  <input type="text" name="firstname">
  Last name:
  <input type="text" name="lastname"> <input type="radio" name="gender" value="male" checked> Male
  <input type="radio" name="gender" value="female"> Female
 <input type="submit" value="Submit"> </form>	i. HTML forms are used to collect user input. ii. The <input> element has many variations, depending on the type attribute. Type attributes are defined as follows. • text: Defines normal text input • radio: Defines radio button input (for selecting one of many choices) • submit: Defines a submit button (for submitting the form)

			iii. The action attribute defines the action to be performed when the form is submitted. iv. The method attribute specifies the HTTP method (GET or POST) to be used when submitting the forms.
			i.

Lab exercises

Write an HTML Code for the following programs.

1. Create a web page containing following elements:
   a) A link to file image.html located in the text subdirectory.
   b) A link to file background.html located in the same directory.
   c) A link to the email address.

2. A local university has asked you to create an HTML document that allows potential students to provide feedback about their campus visit. Your HTML document should contain a form with text boxes for a name, address and e-mail. Provide check boxes that allow prospective students to indicate what they liked most about the campus. These check boxes should include: students, location, campus, atmosphere, dorm rooms and sports. Also, provide radio buttons that ask the prospective student how they became interested in the university. Options should include: friends, television, Internet and other. In addition, provide a text area for additional comments, a submit button and a reset button.

3. Design the student registration form using necessary fields.

4. Develop static pages (using only HTML) of an online Book store. The website should consist the following pages. Home page, Registration and user Login, User profile page, Books catalog, Shopping cart, Payment By credit card, order confirmation

Additional Exercises

1. Create an XHTML document that marks up your resume.

2. Create an XHTML document that displays a tic-tac-toe table with player X winning. Use **<h2>** to markup both Xs and Os. Center the letters in each cell horizontally. Title the game using **<h1>** tag. This title should span all three columns. Set the table border to one.

3. A University has asked you to create an XHTML document that allows potential students to provide feedback about their campus visit. The XHTML document should contain a form with text boxes for a name, address and e-mail. Provide check boxes that allow prospective students to indicate what they liked most about the campus. These check boxes should include: students, location, campus, atmosphere, dorm rooms and sports. Also, provide radio buttons that ask the prospective student how they became interested in the university. Options should include: friends, television, Internet and other. In addition, provide a text area for additional comments, a submit button and a reset button.

LAB NO: 2   **Date:**

<div align="center">

HTML - II

</div>

Objectives

- To understand frames and frame sets
- To familiarize with image maps

HTML Frames and Frame sets

The <frameset> tag defines a frameset. The <frameset> element holds one or more <u><frame></u> elements. Each <frame> element can hold a separate document. The <frameset> element specifies how many columns or rows there will be in the frameset, and how much percentage/pixels of space will occupy each of them.

- To use frames on a page we use <frameset> tag instead of <body> tag.
- The <frameset> tag defines how to divide the window into frames.
- The **rows** attribute of <frameset> tag defines horizontal frames and **cols** attribute defines vertical frames.
- Each frame is indicated by <frame> tag and it defines which HTML document shall open into the frame.
- Frame tag can have various attributes like src (gives filename that should be loaded in the frame. Its value can be any URL.) name (gives a name to a frame) and so on.

Example:

```
<frameset cols="25%,50%,25%">
 <frame name="left" src="/html/top_frame.htm" />
 <frame name="center" src="/html/main_frame.htm" />
  <frame name="right" src="/html/bottom_frame.htm" />
   </frameset>
```

Iframe
An iframe is used to display a web page within a web page. It is also called as hidden frame.

Example

```
<body>
<iframe src="http://www.w3schools.com/html/html_iframe.asp" width="200"
height="200"></iframe>
</body>
```

Image Maps

An image-map is an image with clickable areas. The <map> tag is used to define a client-side image-map.The required name attribute of the <map> element is associated with the <img>'s usemap attribute and creates a relationship between the image and the map.

The <map> element contains a number of <area> elements that defines the clickable areas in the image map.

Example

```
<html>
<body>
<p>Click on the sun or on one of the planets to watch it closer:</p>
<img src="planets.gif" width="145" height="126" alt="Planets" usemap="#planetmap">
<map name="planetmap">
  <area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm">
  <area shape="circle" coords="90,58,3" alt="Mercury" href="mercur.htm">
  <area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm">
</map>
</body>
</html>
```

Lab exercises

1. Create an image map using a photograph of your institute. Include an event handler that writes your department name to the status bar using the default Status property. Create an individual webpage that opens when you click on the respective department. Also include a link back to main page.

2.   Display the following content on the screen.
   a) Create a web page that holds a bulleted list of the names of your friends. Make sure that the bullets are in plain circle.
   b) Create a web page to display the maximum and minimum temperature of 5 cities using table.
   c) Create a Frame which would hold both the web page that was created earlier. The frame should be split row-wise into equal halves.
   d) Create a frameset shown below

File: pgm1.html; Name: Contact Height: 150 pixels	
File: menu.html; Name: menu Height: balance Width: 175 pixels	File: content.html; Name: content Height: balance Width: 175 pixels
File: pgm2.html; Name: frmBottom Height: 100 pixels	

Create another file testTarget.html, in this file write information about IT department. Create 8 links in file menu.html to testTarget.html but use 8 different targets, the targets should be contact, content, frmBottom, blank, top, self, parent and do not use the target attribute.

3.   Develop and demonstrate a XHTML document that illustrates the use external style sheet and inline style sheet, ordered list, table, borders, padding, color, and the <span> tag.

Additional Exercises

1.   Write an XHTML document that shows the results of a color survey. The document should contain a form with radio buttons that allows users to vote for their favorite color. One of the colors should be selected as a default. The document should also contain a table showing various colors and the corresponding percentage of votes for each color. (Each row should be

displayed in the color to which it is referring.) Use attributes to format width, border and cell spacing for the table.

2. Create the frame set shown in Figure 2.1

Figure 2.1

LAB NO: 3   **Date:**

JAVA SCRIPT - I

Objective

- To apply the concepts of CSS to enrich the web documents
- To introduce basic java script programming.

Introduction to CSS

The major points of CSS are given below:

- CSS stands for Cascading Style Sheet.
- CSS is used to design HTML tags.
- CSS is a widely used language on the web.
- HTML, CSS and JavaScript are used for web designing. It helps the web designers to apply style on HTML tags.

CSS Syntax

A CSS rule set contains a selector and a declaration block.

Syntax

Selector{Property1: value1; Property2: value2; ....;}

H1{color: yellow; font-size: 11px;}

Selector: Selector indicates the HTML element you want to style. It could be any tag like <h1>, <title> etc.

Declaration Block: The declaration block can contain one or more declarations separated by a semicolon. For the above example, there are two declarations:

1. color: yellow;

2. font-size: 11 px;

Each declaration contains a property name and value, separated by a colon.

Property: A Property is a type of attribute of HTML element. It could be color, border etc.

Value: Values are assigned to CSS properties. In the above example, value "yellow" is assigned to color property.

Types of CSS

1. Inline CSS

We can apply CSS in a single element by inline CSS technique.

Syntax:

<htmltag style="cssproperty1:value; cssproperty2:value;"> </htmltag>

Snippet:

```
<html>
<body>
<h1 style="color:red;margin-left:40px;">Inline CSS is applied on
this heading.</h1>
<p>This paragraph is not affected.</p>
</body>
</html>
```

2. Internal CSS

The internal style sheet is used to add a unique style for a single document. It is defined in <head> section of the HTML page inside the <style> tag.

Snippet:

```
<html>
<head>
<style>
body {
background-color: linen;
}
h1 {
color: red;
margin-left: 80px;   }
```

```
</style>
</head>
<body>
<h1>The internal style sheet is applied on this
heading.</h1>
<p>This paragraph will not be affected.</p>
</body>
</html>
```

3. External CSS

- The external style sheet is generally used when you want to make changes on multiple pages.
- It is ideal for this condition because it facilitates you to change the look of the entire web site by changing just one file.
- It uses the <link> tag on every pages and the <link> tag should be put inside the head section.

Snippet:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

The external style sheet may be written in any text editor but must be saved with a .css extension. This file should not contain HTML elements.

```
File: mystyle.css
body {
    background-color: lightblue;
}
h1 {
    color: navy;
    margin-left: 20px;
}
```

4. Imported CSS

@import is used to import an external stylesheet in a manner similar to the <link> element.

Syntax:

```
<head>
  <@import "URL";
</head>
Snippet:
<head>
  @import "mystyle.css";
</head>
```

Introduction to Java Script

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.

Syntax:

```
<script language="javascript" type="text/javascript">

   JavaScript code

</script>
```

The script tag takes two important attributes −

1. Language − This attribute specifies what scripting language you are using.

2. Type − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".

Simple JavaScript Program

```
<html>
  <body>
```

```
<script language="javascript" type="text/javascript">
  <!--
    document.write("Hello World!")
  //-->
</script>
</body>
</html>
```

Javascript Data Types

JavaScript allows you to work with three primitive data types −

- **Numbers,** eg. 123, 120.50 etc.
- **Strings** of text e.g. "This text string" etc.
- **Boolean** e.g. true or false.

Syntax:

```
<script type="text/javascript">
    var name = "Ali";
    var money;
    money = 2000.50;
</script>
```

JavaScript Statements

This statement tells the browser to write "Hello Dolly." inside an HTML element with id="demo":

Example:

document.getElementById("demo").innerHTML = "Hello Dolly.";

Java Script Functions

Syntax:

```
<script type="text/javascript">
  <!--
    function functionname(parameter-list)
```

```
           {
             statements
           }
         //-->
       </script>
```

Java Script Strings

Table 3.1 Java Script Strings

Method	Example	Description
charAt()	var str = "HELLO WORLD"; str.charAt(0); // returns H	Returns the character at the specified index.
Length	<script> var txt = "NISHA"; document.getElementById("demo").inner HTML = txt.length; </script>	The length property returns the length of a string
charCodeAt()	var str = "HELLO WORLD"; str.charCodeAt(0); // returns 72	Returns a number indicating the Unicode value of the character at the given index.
concat()	var text1 = "Hello"; var text2 = "World"; text3 = text1.concat(" ",text2);	Combines the text of two strings and returns a new string.
indexOf()	var str = "Please locate where 'locate' occurs!"; var pos = str.indexOf("locate");	Returns the index within the calling String object of the first occurrence of the specified value, or -1 if not found.

lastIndexOf()	var str = "Please locate where 'locate' occurs!"; var pos = str.lastIndexOf("locate");	Returns the index within the calling String object of the last occurrence of the specified value, or -1 if not found.
match()	var str = "The rain in SPAIN stays mainly in the plain"; var res = str.match(/ain/g);	Used to match a regular expression against a string.
replace()	str = "Please visit Microsoft!"; var n = str.replace("Microsoft","W3Schools");	Used to find a match between a regular expression and a string, and to replace the matched substring with a new substring.
search()	var str = "Please locate where 'locate' occurs!"; var pos = str.search("locate");	Executes the search for a match between a regular expression and a specified string.
slice()	<script> var str = "Information Technology"; document.getElementById("demo").inner HTML = str.slice(2,5); </script>	Extracts a section of a string and returns a new string.
split()	var str = "How are you doing today?"; var res = str.split(" ");	Splits a String object into an array of strings by separating the string into substrings.
substr()	var str = "Hello world!"; var res = str.substr(1, 4);	Returns the characters in a string beginning at the specified location through the

		specified number of characters.
substring()	<script> var str = "Harry Potter"; document.getElementById("demo").inner HTML = str.substring(6,12); </script>	Returns the characters in a string between two indexes into the string.

toLowerCase()	var text1 = "Hello World!"; // String
	var text2 = text1.toLowerCase();
		Returns the calling string value converted to lower case.

toString()	var str = "Hello World!"; var res = str.toString();	Returns a string representing the specified object.

toUpperCase()	var text1 = "Hello World!"; // String
	var text2 = text1.toUpperCase();
		Returns the calling string value converted to uppercase.

valueOf()	var str = "Hello World!"; var res = str.valueOf();	Returns the primitive value of the specified object.

Java Script Numbers

Table 3.2 Java Script Numbers

Method	Example	Description
toExponential()	var num = 5.56789; var n = num.toExponential();	Forces a number to display in exponential notation, even if the number is in the range in which JavaScript normally uses standard notation.
toFixed()	var num = 5.56789; var n = num.toFixed(2);	Formats a number with a specific number of digits to the right of the decimal.

toPrecision()	var num = 13.3714; var n = num.toPrecision(2);	Defines how many total digits (including digits to the left and right of the decimal) to display of a number.

toString()	var num = 15;
	var a = num.toString();
	var b = num.toString(2);
	var c = num.toString(8);
	var d = num.toString(16);
		Returns the string representation of the number's value. The parameter given to the function indicates various bases.

valueOf()	var num = 15; var n = num.valueOf();	Returns the number's value.

Java Script Looping Constructs

Table 3.3 Java Script Looping Constructs

Name	Syntax/ Example	Explanation

if …else	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
	}
		If executes the block of code within it if the specified condition is true If the same condition is false then else block is executed.

Switch	switch(expression) {
	case n:
	code block
	break;
	case n:
	code block
	break;
	default:
	default code block
	}
		- 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.

for	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) has been executed.

While	while (expression){
	Statement(s) to be executed if expression is true
	}
		Executes a statement or code block repeatedly as long as an **expression** is true.

Break	for (i = 0; i < 10; i++) {
	   if (i === 3)
	{ break; }
	   text += "The number is " + i + "
	";
	}
		The break statement can also be used to jump out of a loop.

Continue	for (i = 0; i < 10; i++) {
	   if (i === 3) { continue; }
	   text += "The number is " + i + "
	";
	}
		The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop

for…in	for (variablename in object){
	statement or block to execute
	}
		In each iteration, one property from object is assigned to variablename and this loop continues till all the properties of the object are exhausted.

Array Methods

Table 3.4 JavaScript Array Methods

Method	Example	Description

concat()	var hege = ["Cecilie", "Lone"];
	var stale = ["Emil", "Tobias", "Linus"];
	var children = hege.concat(stale);
		Returns a new array comprised of this array joined with other array(s) and/or value(s).

every()	var ages = [32, 33, 16, 40];
	
	function checkAdult(age) {
	   return age >= 18;
	}
	
	function myFunction() {
	   document.getElementById("demo").innerHTML = ages.every(checkAdult);
	}
		Returns true if every element in this array satisfies the provided testing function.

filter()	var ages = [32, 33, 16, 40];
	
	function checkAdult(age) {
	   return age >= 18;
	}
	function myFunction() {
	   document.getElementById("demo").innerHTML = ages.filter(checkAdult);
	}
		Creates a new array with all of the elements of this array for which the provided filtering function returns true.

forEach()	Multiply with: <input type="number" id="multiplyWith" value="10">
	<button onclick="numbers.forEach(myFunction)">Try it</button>
	
	 Updated array:
	
	<script>
	var numbers = [65, 44, 12, 4];
	
	function myFunction(item,index,arr) {
		Calls a function for each element in the array.

	arr[index] = item * document.getElementById("multiplyWith").value;
	   demo.innerHTML=numbers;
	}
	</script>	

indexOf()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; var a = fruits.indexOf("Apple");	Returns the first (least) index of an element within the array equal to the specified value, or -1 if none is found.
join()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; var energy = fruits.join(" and ");	Joins all elements of an array into a string.
lastIndexOf()	var fruits = ["Banana", "Orange", "Apple", "Mango", "Orange"]; var a = fruits.lastIndexOf("Orange");	Returns the last (greatest) index of an element within the array equal to the specified value, or -1 if none is found.

map()	var numbers = [4, 9, 16, 25];
	
	function myFunction() {
	   x = document.getElementById("demo")
	   x.innerHTML = numbers.map(Math.sqrt);
	}
		Creates a new array with the results of calling a provided function on every element in this array.

pop()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.pop();	Removes the last element from an array and returns that element.
push()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.push("Kiwi");	Adds one or more elements to the end of an array and returns the new length of the array.

reduce()	var numbers = [65, 44, 12, 4];
	
	function getSum(total, num) {
	   return total + num;
	}
	function myFunction(item) {
		Apply a function simultaneously against two values of the array (from left-to-right) as to reduce it to a single value.

	document.getElementById("demo").innerHTML = numbers.reduce(getSum); }	
reverse()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.reverse();	Reverses the order of the elements of an array -- the first becomes the last, and the last becomes the first.
shift()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.shift();	Removes the first element from an array and returns that element.
slice()	var fruits = ["Banana", "Orange", "Lemon", "Apple" , "Mango"]; var citrus = fruits.slice(1, 3);	Extracts a section of an array and returns a new array.

some()	var ages = [3, 10, 18, 20];
	
	function checkAdult(age) {
	  return age >= 18;
	}
	
	function myFunction() {
	  document.getElementById("demo").innerHTML = ages.some(checkAdult);
	}
		Returns true if at least one element in this array satisfies the provided testing function.

toSource()	<script>
	
	function employee(name,jobtitle,born)
	{
	this.name=name;
	this.jobtitle=jobtitle;
	this.born=born;
	}
	
	var fred=new employee("Fred
	Flintstone","Caveman",1970);
	
	document.write(fred.toSource());
	
	</script>
		Represents the source code of an object

sort()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.sort();	Represents the source code of an object
splice()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.splice(2, 1, "Lemon", "Kiwi");	Adds and/or removes elements from an array.
unshift()	var fruits = ["Banana", "Orange", "Apple", "Mango"]; fruits.unshift("Lemon","Pineapple");	Adds one or more elements to the front of an array and returns the new length of the array.

Lab exercises

1.   Write a JavaScript code that displays text "TEXT-GROWING" with increasing font size in the interval of 100ms in RED COLOR, when the font size reaches 50pt it displays "TEXT-SHRINKING" in BLUE color. Then the font size decreases to 5pt.

2.   Write a JavaScript that encodes English language phrases into pig Latin. Pig Latin is a form of coded language often used for amusement. Many variations exist in the methods used to form pig Latin phrases. For simplicity, use the following algorithm: To form a pig Latin phrase from an English language phrase, tokenize the phrase into an array of words using String method split. To translate each English word into a pig Latin word, place the first letter of the English word at the end of the word and add the letters "ay." Thus the word "jump" becomes "umpjay," the word "the" becomes "hetay" and the word "computer" becomes "omputercay." Blanks between words remain as blanks. Assume the following: The English phrase consists of words separated by blanks, there are no punctuation marks and all words have two or more letters. Function printLatinWord should display each word. Each token (i.e., word in the sentence) is passed to method printLatinWord to print the pig Latin word. Enable the user

to input the sentence through an XHTML form. Keep a running display of all the converted sentences in an XHTML text area.

3. Write a JavaScript that inputs a telephone number as a string in the form (555) 555-5555. The script should use String method split to extract the area code as a token, the first three digits of the phone number as a token and the last four digits of the phone number as a token. Display the area code in one text field and the seven-digit phone number in another text field.

4. Write a JavaScript that reads a five-letter word from the user and produces all possible three letter words that can be derived from the letters of the five-letter word. For example, the three-letter words produced from the word "bathe" include the commonly used words "ate," "bat," "bet," "tab,", "hat," "the" and "tea." Output the results in an HTML text area.

5. Write a Java Script that calculates the squares and cubes of the numbers from 0 to 10 and outputs HTML text that displays the resulting values in an HTML table format.

Additional Exercises

1. A parking garage, charges Rs. 2.00 minimum fee to park for, up to three hours. The garage charges additional Rs. 0.50 per hour in excess of three hours. Write a JavaScript that calculates and displays the parking charges for each customer who parked a car in the garage. The user should enter the hours parked by each customer. The program should display the charges for each customer and list of customers who have parked in the previous day. The program should use the function calculate charges() to determine the charge of each customer.

2. Write a script to do the following. Choose and store a random number between 1 and 15. Allow the user to choose the number until getting it right. Tell the user in each turn whether the guess was too high or too low.

3. Write a JavaScript to solve the following problem. A company pays its salespeople on commission basis. The salespeople receives $200 per week plus 9% of their gross sales for that week. For example, a salesperson whose gross crosses $5000 in sales in a week receives $200 plus 9% of $5000 so the total is

$650. Determine how many salespeople earned salaries in each of the following ranges as shown in the following table (assume that each salesperson's salary is truncated to an integer amount).

1	$300-$399
2	$400-$499
3	$500-$599
4	$600-$699
5	$700-$799
6	$800-$899
7	$900-$999
8	$1000 and over

LAB NO: 4   **Date:**

JAVASCRIPT - II

Objectives

- To acquire knowledge on the application of JavaScript operators, variables, arrays, strings and control structures.
- To learn the usage of JavaScript events and functions.
- To learn the application of regular expression in order to validate the web pages.

Events

The <body> and <frameset> Level Events

There are only two attributes which can be used to trigger any JavaScript or VBScript code, when any event occurs at document level.

Table 4.1 <body> and <frameset> Level Events

Attribute	Value	Description

Onload	`<body onload="myFunction()">`
	`<h1>Hello World!</h1>`
	`<script>`
	`function myFunction() {`
	`alert("Page is loaded");`
	`}`
	`</script>`
	`</body>`
		Script runs when a XHTML document loads.

onunload	`<body onunload="myFunction()">`
	`<script>`
	`function myFunction() {`
	`alert("Thank you for visiting W3Schools!");`
	`}`
	`</script>`
	`</body>`
		Script runs when a XHTML document unloads.

The <form> Level Events

There are following six attributes which can be used to trigger any JavaScript or VBScript code when any event occurs at form level.

<div align="center">

Table 4.2 <form> Level Events

</div>

Attribute	Example	Description

onchange	<head>
	<title>example of onchange event handler</title>
	<script>
	function valid(input) {
	alert("you have changed the value from 10 to " + input);
	}
	</script>
	</head>
	<body>
	<form>
	<input type="text" value="10" onchange="valid(this.value)">
	</form>
	</body>
		Script executes when the element changes.

onsubmit	<body> <form name="myform" onsubmit="alert('thank you ' + myform.data.value +'!')"> <input type="text" name="data"> <input type="submit" value="submit this form"> </form> </body>	Script executes when the form is submitted.
onreset	<body> <form onreset="alert('this will reset the form!')"> <input type="text">	Script executes when the form is reset.

	<input type="reset" value="reset form" > </form> </body>	
Onselect	<body> <form> <input type="text" value="select this" onselect="alert('this is an example of onselect!!')"> </form> </body>	Script executes when the element is selected.

Onblur	<body>
	Enter your name: <input type="text"
	id="fname" onblur="myFunction()">
	<p>When you leave the input field, a
	function is triggered which transforms the
	input text to upper case.</p>
	<script>
	function myFunction() {
	var x = document.getElementById("fname");
	x.value = x.value.toUpperCase();
	}
	</script>
	</body>
		Script executes when the element loses focus.

Onfocus	<body> <h3>example of onfocus event handler</h3> click your mouse in the text box:
 <form> <input type="text" onfocus='alert("you focused in the textbox!!")'> </form> </body>	Script runs when the element gets focus.

Keyboard Events

The following three events are generated by keyboard.

<div align="center">

Table 4.3 Keyboard Events

</div>

Attribute	Value	Description

Onkeydown	```
	<body>
	<input type="text"
	onkeydown="myFunction()">
	<script>
	function myFunction() {
	alert("You pressed a key inside the input field");
	}
	</script>
	</body>
	``
		Script executes on key press.

onkeypress	```
	<body>
	<input type="text"
	onkeypress="myFunction()">
	<script>
	function myFunction() {
	alert("You pressed a key inside the input field");
	}
	</script>
	</body>
	``
		Script executes on key press and release.

onkeyup	```
	<body>
	Enter your name: <input type="text"
	id="fname" onkeyup="myFunction()">
	<script>
	function myFunction() {
	 var x = document.getElementById("fname");
	x.value = x.value.toUpperCase();
	}
	</script>
	</body>
	``
		Script executes key release.

Mouse Events

The following seven events are generated by mouse when it comes in contact with any HTML tag.

Table 4.4 Mouse Events

Attribute	Value	Description
onclick	```<body><button onclick="myFunction()">Click me</button><p id="demo"></p><script>function myFunction() {document.getElementById("demo").innerHTML = "Hello World";}</script></body>```	Script executes on a mouse click.
ondblclick	```<body><p ondblclick="myFunction()">Double-click this paragraph to trigger a function.</p><script>function myFunction() { document.write("Hello World");}</script></body>```	Script executes on a mouse double-click.
onmousemove	```<head><style>div { width: 200px; height: 100px; border: 1px solid black; }```	Script executes when mouse pointer moves.

	`</style>`
	`</head>`
	`<body>`
	`<div onmousemove="myFunction(event)"`
	`onmouseout="clearCoor()"></div>`
	`<p>Mouse over the rectangle above, and`
	`get the coordinates of your mouse`
	`pointer.</p>`
	`<p id="demo"></p>`
	`<script>`
	`function myFunction(e) {`
	`var x = e.clientX;`
	`var y = e.clientY;`
	`var coor = "Coordinates: (" + x + "," + y +`
	`")";`
	`document.getElementById("demo").inner`
	`HTML = coor;`
	`}`
	`function clearCoor() {`
	`document.getElementById("demo").inner`
	`HTML = "";`
	`}`
	`</script>`
	`</body>`	

onmouseout	`<body>`
	`<img onmouseover="bigImg(this)"`
	`onmouseout="normalImg(this)"`
		Script executes when mouse pointer moves out of an element.

Onmouseover	`border="0" src="smiley.gif" alt="Smiley"`
	`width="32" height="32">`
	`<p>The function bigImg() is triggered`
	`when the user moves the mouse pointer`
	`over the image.</p>`
	`<p>The function normalImg() is triggered`
	`when the mouse pointer is moved out of`
	`the image.</p>`
		Script executes when mouse pointer moves over an element.

	```<script>```
	```function bigImg(x) {```
	```x.style.height = "64px";```
	``` x.style.width = "64px";```
	```}```
	```function normalImg(x) {```
	``` x.style.height = "32px";```
	```x.style.width = "32px";```
	```}```
	```</script>```
	```</body>```	

onmouseup	```<body>``` ```<p id="myP"```	Script executes when mouse button is released.

Onmousedow n	```onmousedown="mouseDown()"```
	```onmouseup="mouseUp()">```
	Click the text! The mouseDown() function is triggered when the mouse button is pressed down over this paragraph, and sets the color of the text to red. The mouseUp() function is triggered when the mouse button is released, and sets the color of the text to green.
	```</p>```
	```<script>```
	```function mouseDown() {```
	```document.getElementById("myP").style.c```
	```olor = "red";```
	```}```
	```function mouseUp() {```
	```document.getElementById("myP").style.c```
	```olor = "green";```
	```}```
	```</script>```
	```</body>```
		Script executes when mouse button is pressed.

Lab exercises

1. Develop and demonstrate a XHTML file that includes JavaScript script for the following problems:

   - Input: A number n obtained using prompt
     Output: The first n Fibonacci numbers
   - Input: A number n obtained using prompt
     Output: A table of numbers from 1 to n and their squares using alert

2. Develop and demonstrate a XHTML file that includes JavaScript script that uses functions for the following problems:

   - Parameter: A string
     Output: The position in the string of the left-most vowel
   - Parameter: A number
     Output: The number with its digits in the reverse order

3. Develop and demonstrate, using JavaScript script, a XHTML document that contains three short paragraphs of text, stacked on top of each other, with only enough of each showing so that the mouse cursor can be placed over some part of them. When the cursor is placed over the exposed part of any paragraph, it should rise to the top to become completely visible. Also modify the above document so that when a paragraph is moved from the top stacking position, it returns to its original rather than to the bottom.

4. Develop and demonstrate, using JavaScript script, a XHTML document that collects the USN (the valid format is: A digit from 1 to 4 followed by two upper-case characters followed by two digits followed by two upper-case characters followed by three digits; no embedded spaces allowed) of the user. Event handler must be included for the form element that collects this information to validate the input. Messages in the alert windows must be produced when errors are detected. Also modify the above program to get the current semester also.

5. Write a JavaScript to design a simple calculator to perform the following operations: sum, product, difference and quotient.

6. Write a JavaScript that has list of color names. As the user moves the cursor over the text the background color is changed.

Additional Exercises

1. Write a JavaScript to change the font-size when the user moves mouse over it.

2. Write a function that responds to a click anywhere on the page by displaying an **alert** dialog. Display the event name if the user held *Shift* during the mouse click. Display the element name that triggered the event if the user held *Ctrl* during the mouse click.

3. Write a JavaScript to do the following:

    a) Add two elements to that users can click.
    b) Use the **MIT.gif** image file as the first element. When the user clicks the image, display an **alert** dialog box with the text "you clicked the image."
    c) For the second element, create a one-row table containing a text string. Set the table border to one.
    d) When the user clicks the table element, display an **alert** dialog box containing "you clicked the table." In the two accompanying functions, set each event object to **true**.

LAB NO.: 5   **Date:**

PYTHON PROGRAMMING - I

Objectives

- To demonstrate significant experience with the python program development environment.
- To write suitable python programs using operators, conditionals, loops etc..
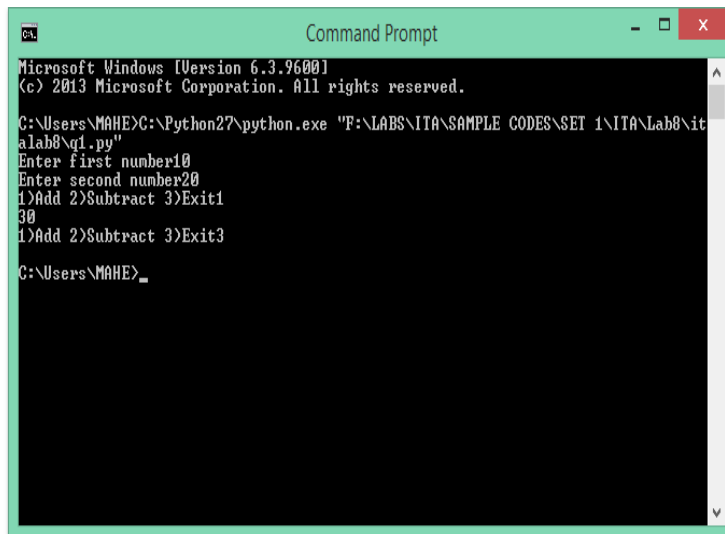
Introduction to python

Python is a high-level, interpreted, interactive and object-oriented scripting language.

Simple Python Script

```
#!/usr/bin/python3
print ("Hello, Python!")
```

Execution Steps

Via Command Prompt

1. Open command prompt
2. Type the location of python .exe file and the python file (use .py extension) which you want to run.

Via Idle GUI

1. Run IDLE.
2. Click File, New Window.
3. Enter your script in the "Untitled" window.
4. In the "Untitled" window, select Run, Run Module (or press F5) to run your script.
5. A dialog "Source Must Be Saved.
6. Give a name with .py extension and then run the program.

Comments in Python

A hash sign (#) that is not inside a string literal begins a comment.

Assigning Values to Variables

```
#!/usr/bin/python
counter = 100        # An integer assignment
miles   = 1000.0     # A floating point
name    = "John"     # A string
a = b = c = 1
print counter
print miles
print name
print a
```

Operators

Python Arithmetic Operators

```
#!/usr/bin/python
a = 21
b = 10
c = 0
c = a + b # Addition
print "Line 1 - Value of c is ", c
c = a – b # substraction
print "Line 2 - Value of c is ", c
```

```
c = a * b # multiplication
print "Line 3 - Value of c is ", c
c = a / b # division
print "Line 4 - Value of c is ", c
c = a % b # modulus
print "Line 5 - Value of c is ", c
a = 2
b = 3
c = a**b #exponent
print "Line 6 - Value of c is ", c
a = 10
b = 5
c = a//b  # Floor Division
print "Line 7 - Value of c is ", c
```

Python Comparison Operators

```
#!/usr/bin/python
a = 21
b = 10
c = 0
if ( a == b ):
   print "Line 1 - a is equal to b"
else:
   print "Line 1 - a is not equal to b"
if ( a != b ):
   print "Line 2 - a is not equal to b"
else:
   print "Line 2 - a is equal to b"
if ( a <> b ):
   print "Line 3 - a is not equal to b"
else:
   print "Line 3 - a is equal to b"
if ( a < b ):
   print "Line 4 - a is less than b"
else:
```

```
   print "Line 4 - a is not less than b"
if ( a > b ):
   print "Line 5 - a is greater than b"
else:
   print "Line 5 - a is not greater than b"
a = 5;
b = 20;
if ( a <= b ):
   print "Line 6 - a is either less than or equal to  b"
else:
   print "Line 6 - a is neither less than nor equal to  b"

if ( b >= a ):
   print "Line 7 - b is either greater than  or equal to b"
else:
   print "Line 7 - b is neither greater than  nor equal to b"
```

Python Assignment Operators

```
#!/usr/bin/python
a = 21
b = 10
c = 0
c = a + b
print "Line 1 - Value of c is ", c
c += a
print "Line 2 - Value of c is ", c
c *= a
print "Line 3 - Value of c is ", c
c /= a
print "Line 4 - Value of c is ", c
c  = 2
c %= a
print "Line 5 - Value of c is ", c
c **= a
print "Line 6 - Value of c is ", c
```

```
c //= a
print "Line 7 - Value of c is ", c
```

Python Bitwise Operators

```
#!/usr/bin/python

a = 60          # 60 = 0011 1100
b = 13          # 13 = 0000 1101
c = 0
c = a & b;       # 12 = 0000 1100
print "Line 1 - Value of c is ", c

c = a | b;       # 61 = 0011 1101
print "Line 2 - Value of c is ", c

c = a ^ b;       # 49 = 0011 0001
print "Line 3 - Value of c is ", c

c = ~a;          # -61 = 1100 0011
print "Line 4 - Value of c is ", c

c = a << 2;      # 240 = 1111 0000
print "Line 5 - Value of c is ", c

c = a >> 2;      # 15 = 0000 1111
print "Line 6 - Value of c is ", c
```

Python Operators Precedence

```
#!/usr/bin/python

a = 20
b = 10
c = 15
d = 5
```

e = 0

e = (a + b) * c / d   #(30 * 15) / 5
print "Value of (a + b) * c / d is ", e

e = ((a + b) * c) / d   # (30 * 15) / 5
print "Value of ((a + b) * c) / d is ", e

e = (a + b) * (c / d);   # (30) * (15/5)
print "Value of (a + b) * (c / d) is ", e

e = a + (b * c) / d;   # 20 + (150/5)
print "Value of a + (b * c) / d is ", e

Python Strings

Python treats single quotes the same as double quotes. Creating strings is as simple as assigning a value to a variable. For example −
var1 = 'Hello World!'
var2 = "Python Programming"

Python Looping Constructs

Syntax of the loops resemble JavaScript code.

String Special Operators

Assume string variable a holds 'Hello' and variable b holds 'Python', then –

Table 5.1 Python Strings

Operator	Description	Example
+	Concatenation - Adds values on either side of the operator	a + b will give HelloPython
*	Repetition - Creates new strings, concatenating multiple copies of the same string	a*2 will give - HelloHello

[]	Slice - Gives the character from the given index	a[1] will give e
[:]	Range Slice - Gives the characters from the given range	a[1:4] will give ell
in	Membership - Returns true if a character exists in the given string	H in a will give 1
not in	Membership - Returns true if a character does not exist in the given string	M not in a will give 1

Mathematical Functions

Python includes following functions that perform mathematical calculations.

Table 5.2 Mathematical Functions

Function	Returns (description)
abs(x)	The absolute value of x: the (positive) distance between x and zero.
ceil(x)	The ceiling of x: the smallest integer not less than x
cmp(x, y)	-1 if x < y, 0 if x == y, or 1 if x > y
exp(x)	The exponential of x: e^x
fabs(x)	The absolute value of x.
floor(x)	The floor of x: the largest integer not greater than x
log(x)	The natural logarithm of x, for x> 0
log10(x)	The base-10 logarithm of x for x> 0 .
max(x1, x2,...)	The largest of its arguments: the value closest to positive infinity
min(x1, x2,...)	The smallest of its arguments: the value closest to negative infinity
modf(x)	The fractional and integer parts of x in a two-item tuple. Both parts have the same sign as x. The integer part is returned as a float.
pow(x, y)	The value of x**y.
round(x [,n])	x rounded to n digits from the decimal point. Python rounds away from zero as a tie-breaker: round(0.5) is 1.0 and round(-0.5) is -1.0.
sqrt(x)	The square root of x for x > 0

Python Files I/O

Examples for file read and file write operations are stated below

File Write

```
#!/usr/bin/python

# Open a file
fo = open("foo.txt", "wb")
fo.write( "Python is a great language.\nYeah its great!!\n");

# Close opend file
fo.close()
```

File Read

```
#!/usr/bin/python

# Open a file
fo = open("foo.txt", "r+")
#10 is the count of no. of bytes read from the opened file
str = fo.read(10);
print "Read String is : ", str
# Close opend file
fo.close()
```

Python Lists

Simple Example:

```
#!/usr/bin/python

list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5, 6, 7 ];

print "list1[0]: ", list1[0]
print "list2[1:5]: ", list2[1:5]
```

Lab Exercises

1. Write the Python programs to do the following:
   a) Implement simple calculator operations.
   b) Find the transpose of a matrix.
   c) Multiplication of two matrices.
   d) Addition of two matrices.
   e) Sum of natural numbers using recursion.
   f) Fibonacci sequence using recursion.
   g) To convert decimal to binary, octal and hexadecimal.
   h) To illustrate different set operations.
   i) Write a program to find whether a list is palindrome or not.
   j) Write a Python program to create a Caesar encryption without using maketrans and translate functions. In cryptography, a Caesar cipher, is one of the simplest and most widely known encryption techniques. In this method each alphabet in the input is replaced by another alphabet with some fixed no of positions down the alphabet.
      Example: ZOO is replaced with CRR

Additiona1 Exercises

1. Write a Python program to generate the prime numbers between m and n.
2. Write a Python program to sort the contents of a list without using in built in sort method.
3. Write a Python program to find the sum of the elements of the list using recursion
4. Write a program to remove the duplicate elements of a list
5. Write a Python program to reverse the list without using any of the methods/slicing

LAB NO. 6   **Date:**

PYTHON PROGRAMMING - II

Objectives

- To apply and manipulate several core data structures like Lists, Strings etc..
- To learn and understand various python libraries

Python includes following list methods

Table 6.1 Python Lists

SN	Function with Description
1	cmp(list1, list2) Compares elements of both lists.
2	len(list) Gives the total length of the list.
3	max(list) Returns item from the list with max value.
4	min(list) Returns item from the list with min value.
5	list(seq) Converts a tuple into list.

Table 6.2 Python List Methods

Sr.No	Methods with Description
1	list.append(obj) Appends object obj to list
2	list.count(obj)

	Returns count of how many times obj occurs in list
3	list.extend(seq) Appends the contents of seq to list
4	list.index(obj) Returns the lowest index in list that obj appears
5	list.insert(index, obj) Inserts object obj into list at offset index
6	list.pop(obj=list[-1]) Removes and returns last object or obj from list
7	list.remove(obj) Removes object obj from list
8	list.reverse() Reverses objects of list in place
9	list.sort([func]) Sorts objects of list, use compare func if given

Lab Exercises

1. Write the Python programs to do the following:
a) To check whether the string is palindrome or not.
b) To remove punctuations from the string.
c) Sort words in alphabetic order.
d) To merge mails.
e) To find and replace a string using regular expressions.
f) To display environment variables.

Additional Exercises

1. Write a Python program named states.py that declares a variable states with value "Mississippi Alabama Texas Massachusetts Kansas". write a Python program that does the following:

   a) Search for a word in variable states that ends in xas. Store this word in element 0 of a list named statesList.

   b) Search for a word in states that begins with k and ends in s. Perform a case-insensitive comparison. [*Note:* Passing re.I as a second parameter to method compile performs a case-insensitive comparison.] Store this word in element 1 of statesList.

   c) Search for a word in states that begins with M and ends in s. Store this word in element 2 of the list.

   d) Search for a word in states that ends in a. Store this word in element 3 of the list.

   e)   Search for a word that begins with M in states at the beginning of the string. Store this word at element 4 of the list.

   f) Output the array states List to the screen.

LAB NO. 7   **Date:**

PYTHON - III

Objectives

- To create and connect to a database.
- To apply the concepts of database programming

Introduction Python to SQLite connection

Connecting To Database

Following Python code shows how to connect to an existing database. If database does not exist, then it will be created and finally a database object will be returned.

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
```

Create a Table

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute('''CREATE TABLE COMPANY
    (ID INT PRIMARY KEY     NOT NULL,
    NAME        TEXT    NOT NULL,
    AGE         INT    NOT NULL,
    ADDRESS       CHAR(50),
    SALARY        REAL);''')
print "Table created successfully";
conn.close()
```

INSERT Operation

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
    VALUES (1, 'Paul', 32, 'California', 20000.00 )");
conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
    VALUES (2, 'Allen', 25, 'Texas', 15000.00 )");
conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
    VALUES (3, 'Teddy', 23, 'Norway', 20000.00 )");
conn.execute("INSERT INTO COMPANY (ID,NAME,AGE,ADDRESS,SALARY) \
    VALUES (4, 'Mark', 25, 'Rich-Mond ', 65000.00 )");
conn.commit()
print "Records created successfully";
conn.close()
```

SELECT Operation

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
cursor = conn.execute("SELECT id, name, address, salary  from COMPANY")
for row in cursor:
  print "ID = ", row[0]
  print "NAME = ", row[1]
  print "ADDRESS = ", row[2]
  print "SALARY = ", row[3], "\n"
print "Operation done successfully";
```

conn.close()

UPDATE Operation

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("UPDATE COMPANY set SALARY = 25000.00 where ID=1")
conn.commit
print "Total number of rows updated :", conn.total_changes
cursor = conn.execute("SELECT id, name, address, salary  from COMPANY")
for row in cursor:
   print "ID = ", row[0]
   print "NAME = ", row[1]
   print "ADDRESS = ", row[2]
   print "SALARY = ", row[3], "\n"
print "Operation done successfully";
conn.close()
```

DELETE Operation

```
#!/usr/bin/python
import sqlite3
conn = sqlite3.connect('test.db')
print "Opened database successfully";
conn.execute("DELETE from COMPANY where ID=2;")
conn.commit
print "Total number of rows deleted :", conn.total_changes
cursor = conn.execute("SELECT id, name, address, salary  from COMPANY")
for row in cursor:
   print "ID = ", row[0]
   print "NAME = ", row[1]
```

```
  print "ADDRESS = ", row[2]
  print "SALARY = ", row[3], "\n"
print "Operation done successfully";
conn.close()
```

Lab Exercises

1. Write a Python program to sort and display the student records which are stored in the database.(Student ID, Reg No, Branch, Semester, Section, CGPA, email)

2. Develop a Python program using data base connectivity to perform registration with required validation.

3. Write a Python program to search a specific student details from the database based on regno.

4. Write a python program to update CGPA based on student ID.

Additional Exercises

1. Write a Python program to validate the phone number and the email id stored in the customer data base

2. Write a Python program to compute the total marks of five subjects based on the data stored in the student database. Also grade the student and display the display results based on the total marks.

3. Write a Python script to create and populate the customer database. The data base consists of three fields Account Number, Name and Balance.
   a) Search and display the Account details based on the Account Number
   b) Display all the customers having the Balance > Rs. 50000/-

LAB NO. 8   **Date:**

JSON AND JQUERY

Objectives
- To implement a mini project using the concepts of JSON and JQUERY

JSON

JSON stands for **J**ava**S**cript **O**bject **N**otation.
- JSON is a syntax for storing and exchanging data.
- JSON is text, written with JavaScript object notation.
- JSON is a lightweight data-interchange format
- JSON is "self-describing" and easy to understand
- JSON is language independent * (JSON uses JavaScript syntax, but the JSON format is text only)
- Text can be read and used as a data format by any programming language.

The JSON format is text only, it can easily be sent to and from a server, and used as a data format by any programming language.

JavaScript has a built in function to convert a string, written in JSON format, into native JavaScript objects:

JSON.parse()

When exchanging data between a browser and a server, the data can only be text.

JSON is text, and we can convert any JavaScript object into JSON, and send JSON to the server.

We can also convert any JSON received from the server into JavaScript objects.

Sending Data

```
var myObj= { "name":"John", "age":31, "city":"NewYork" };
var myJSON= JSON.stringify(myObj);
window.location = "demo_json.php?x=" + myJSON;
```

Receiving Data

```
var myJSON='{       "name":"John",       "age":31,       "city":"New       York"       }';
var myObj=JSON.parse(myJSON);
document.getElementById("demo").innerHTML = myObj.name;
```

Storing and Retrieving Data

Following code is an example of storing and retrieving data

```
//Storing data:
myObj = { "name":"John", "age":31, "city":"New York" };
myJSON = JSON.stringify(myObj);
localStorage.setItem("testJSON", myJSON);

//Retrieving data:
text=localStorage.getItem("testJSON");
obj= JSON.parse(text);
document.getElementById("demo").innerHTML = obj.name;
```

JQUERY

JQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

Table 8.1

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"

$("p:first")	Selects the first <p> element
$("ul li:first")	Selects the first element of the first
$("ul li:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

Example:

```
$(document).ready(function(){
    $("button").click(function(){
        $("p").hide();
    });
});
```

JQuery Events

Table 8.2

Mouse Events	Keyboard Events	Form Events	Document/Window Events
Click	keypress	submit	load
Dblclick	keydown	change	Resize

Mouseenter	keyup	focus	Scroll
Mouseleave		blur	unload

Example:

```
$("#p1").mouseenter(function(){
    alert("You entered p1!");
});
```

Lab Exercises

Design and implement an interactive front end for medical inventory management system (using the following concept JSON, jQuery).

LAB NO. 9   **Date:**

APPLICATION DEVELOPMENT USING DJANGO

Objective

- To develop web application using Django framework

What is Django ?

Django is a Python open-source web development framework used for web application development. A web framework is a software that supports the development of dynamic websites, applications, and services. It provides a set of tools and functionalities that solves many common problems associated with web development, such as security features, database access, sessions, template processing, URL routing, internationalization, localization, and much more.
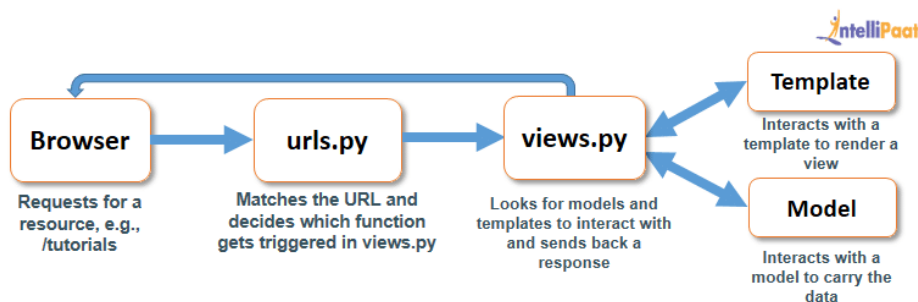

Figure 9.1 : Working of Django Framework

When a user requests for a resource, Django considers the request as a URL and matches it with the existing URL paths in the urls.py file. Once the URL is mapped, Django jumps to the views.py folder and calls a view. The triggered view looks for models and templates to interact with, and then it returns the response back to the user. Here, the model deals with the data associated with the user request. The template deals with the HTML and the static files, such as CSS files, JS files, images, etc., that are required to complete the view.

Installing Django Framework

Step 1: Open the command prompt

Step 2: Run the pip command

```
C:\Users\Mahe>pip install  django
```

Developing Greeting Application

Step 1: Open the command prompt

Step 2: create a new project using below command

```
D:\itt>django-admin startproject Greeting
```

This will create a folder Greeting inside itt folder. Greeting folder contains a package named Greeting and manage.py file. Greeting package contains 4 python files namely __init__, settings, urls and wsgi.

Step 3:   Testing the web server

 Step 3.1: Goto Greeting folder and run the below command.

```
D:\itt\Greeting>python manage.py runserver
```
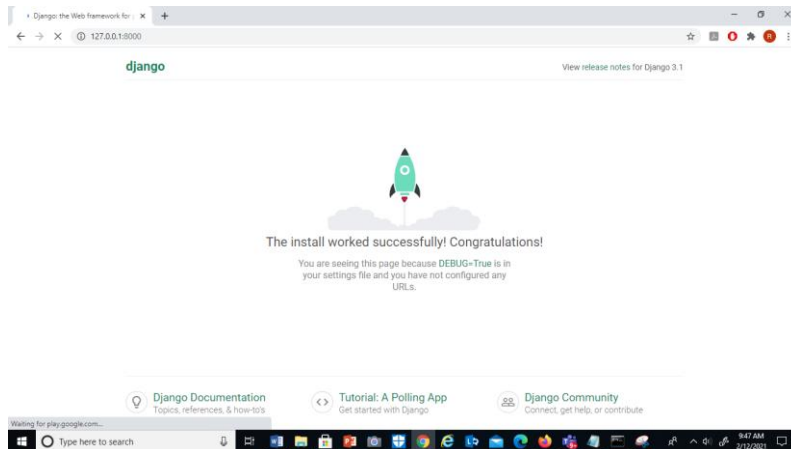
You will get the below message containing the url http://127.0.0.1:8000/

```
D:\itt\Greeting>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until y
ou apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
February 12, 2021 - 09:41:53
Django version 3.1.6, using settings 'Greeting.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Step 3.2: Open the browser and type the above URL. (ie http://127.0.0.1:8000/). Below screen indicates that web server is up and running.

Step 4: create a folder named 'mytemplate' inside Greeting folder


```
D:\itt\Greeting>mkdir mytemplate
```

Step 5: in setting.py, under TEMPLATES set the DIRS as mytemplate as shown below.

'DIRS': ['mytemplate']

Step 6:   create a first.html file inside mytempate

```html
<html>
<body>
   <h2> Enter your name </h2>
   <form action="{% url 'disp' %}">
      <input  type="text" name="a">
      <input  type="submit" value="Greet">
   </form>
</body>
</html>
```

Step 7: create a disp.html file inside mytemplate

```html
<html>
  <body>
    <h1> Welcome {{name}}</h1>
  </body>
</html>
```

Step 8: create a new python file (views.py) and add the below content

```python
from django.shortcuts import render

def readUser(request):
    return render(request,'first.html')

def greetUser(request):
    nm = request.GET['a']
    return render(request,'disp.html',{'name':nm})
```
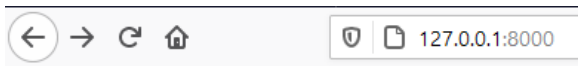
Step 9: add the below content in urls.py

```python
from django.contrib import admin
from django.urls import path
from . import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('',views.readUser,name="first"),
    path('disp/',views.greetUser,name="disp")
]
```

Step 10: open the url http://127.0.0.1:8000/

Additional Details

form tag

The <form> tag is used to create an HTML form for user input. The <input> tag specifies an input field where the user can enter data.

```
<form action="hello.py" method="get">
First Name: <input type="text" name="first_name">  <br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
```

In the above example, we create a form with 2 inputfield and a submit button. Here action attribute of the form specifies the action to be performed when the submit button is clicked. In the above example, when the submit button is clicked, hello.py is called.

GET and POST Methods

The "GET" method indicates that form data is to be encoded (by a browser) into a URL and the "POST" method means that the form data is to appear within a message body. As a simplification, we might say that "GET" is basically for just getting (retrieving) data whereas "POST "may involve anything, like storing or updating data, or ordering a product, or sending E-mail.

Lab Exercise

1 Design and implement a web application on medical inventory management system (use database).

Additional Exercises

1. Write a simple spell checker. Read all words from the 'words' file into a hash. Then read lines from standard input and check each word is spelt correctly. Make sure it is not case sensitive

2. Write a Python program to do the following.
   a) Create a form having field names Name, phone number and email address.
   b) Validate Name (First letter is uppercase letter and length is 10)
   c) Validate Phone number(+91-1234567812)
   d) Validate email address (name@string.com/edu)
   e) Greet the user with some message and display the collected information.

LAB NO. 10   **Date:**

MINI PROJECT - I

Objectives

- To design and develop a mini project employing all the concepts learnt.

Lab exercises

1. Design and develop a web application.

LAB NO. 11 **Date:**

MINI PROJECT - II

Objectives
- To implement a mini project employing all the concepts learnt.

Lab exercises

1. Implementation of the web application.

LAB NO: 12   **Date:**

TESTING AND VALIDATION OF THE MINI PROJECT

Objectives
- To test and validate the project suitably.

Lab Exercises

1. Give the testing and validation details of the mini project.

REFERENCES

1. Deitel, Deitel, Goldberg, "Internet & World Wide Web How To Program", Fourth Edition, Pearson Education, 2008.

2. Chris Bates, "Web Programming: Building Internet Application", 3rd Edition, Wiley India, 2006.

3. Jeffrey C.Jackson, "Web Technologies: A Computer Science Perspective", Pearson Education, 2007.

4. Adrian Holovaty, Jacob K. Moss, "The Definitive Guide to Django: Web Development Done Right",2nd edition, APRESS, 2009