



Channabasaveshwara Institute of Technology

(Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi)
(NAAC Accredited & ISO 9001:2015 Certified Institution)
NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka.

QMP 7.1 D/F



Department of Computer Science & Engineering

LAB MANUAL

(2018-19)

15CSL77

WEB TECHNOLOGY LABORATORY WITH MINI PROJECT

VII Semester CSE

Name : _____

U S N : _____

Batch : _____ **Section :** _____

'Instructions to the Candidates'

1. Students should come with thorough preparation for the experiment to be conducted.
2. Students will not be permitted to attend the laboratory unless they bring the practical record fully completed in all respects pertaining to the experiment conducted in the previous class.
3. Practical record should be neatly maintained.
4. They should obtain the signature of the staff-in-charge in the observation book after completing each experiment.
5. Theory regarding each experiment should be written in the practical record before procedure in your own words.
6. Ask lab technician for assistance if you have any problem.
7. Save your class work, assignments in system .
8. Do not download or install software without the assistance of the laboratory technician.
9. Do not alter the configuration of the system.
10. Turnoff the systems after use.

SYLLABUS

1. Write a JavaScript to design a simple calculator to perform the following operations: sum, product, difference and quotient.
2. Write a JavaScript 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.
3. 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.
4. Develop and demonstrate a HTML5 file that includes JavaScript script that uses functions for the following problems:
 - a. Parameter: A string
 - b. Output: The position in the string of the left-most vowel
 - c. Parameter: A number
 - d. Output: The number with its digits in the reverse order
5. Design an XML document to store information about a student in an engineering college affiliated to VTU. The information must include USN, Name, and Name of the College, Branch, Year of Joining, and email id. Make up sample data for 3 students. Create a CSS style sheet and use it to display the document.
6. Write a PHP program to keep track of the number of visitors visiting the web page and to display this count of visitors, with proper headings.
7. Write a PHP program to display a digital clock which displays the current time of the server.
8. Write the PHP programs to do the following:
 - a. Implement simple calculator operations.
 - b. Find the transpose of a matrix.
 - c. Multiplication of two matrices
9. Write a PHP program named states.py that declares a variable states with value "Mississippi Alabama Texas Massachusetts Kansas". write a PHP 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 caseinsensitive 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.

10. Write a PHP program to sort the student records which are stored in the database using selection sort.

Study Experiment / Project:

Develop a web application project using the languages and concepts learnt in the theory and exercises listed in part A with a good look and feel effects. You can use any web technologies and frameworks and databases.

Note:

1. In the examination each student picks one question from part A.
2. A team of two or three students must develop the mini project. However during the examination, each student must demonstrate the project individually.
3. The team must submit a brief project report (15-20 pages) that must include the following
 - a. Introduction
 - b. Requirement Analysis
 - c. Software Requirement Specification
 - d. Analysis and Design
 - e. Implementation
 - f. Testing

XHTML TUTORIAL

What is XHTML

XHTML stands for EXtensible HyperText Markup Language

XHTML is almost identical to HTML 4.01

XHTML is a stricter and cleaner version of HTML

XHTML is HTML defined as an XML application

XHTML is a W3C Recommendation

XHTML 1.0 became a W3C Recommendation January 26, 2000.

XHTML is compatible with HTML 4.01. All browsers support XHTML.

Why XHTML?

Many pages on the internet contain "bad" HTML.

The following HTML code will work just fine if you view it in a browser (even if it does NOT follow the HTML rules):

```
<html>
<head>
<title>This is bad HTML</title>
<body>
<h1>Bad HTML
</body>
```

XML is a markup language where everything must be marked up correctly, which results in "well-formed" documents.

XML is designed to describe data, and HTML is designed to display data.

Today's market consists of different browser technologies, some browsers run on computers, and some browsers run on mobile phones or other small devices. The last-mentioned do not have the resources or power to interpret a "bad" markup language.

Therefore - by combining HTML and XML, and their strengths, we got a markup language that is useful now and in the future - XHTML.

The Most Important Differences:

- XHTML elements must be properly nested
- XHTML elements must always be closed
- XHTML elements must be in lowercase
- XHTML documents must have one root element

XHTML Elements Must Be Properly Nested

In HTML, some elements can be improperly nested within each other, like this:

```
<b><i>This text is bold and italic</b></i>
```

In XHTML, all elements must be properly nested within each other, like this:

```
<b><i>This text is bold and italic</i></b>
```

Note: A common mistake with nested lists, is to forget that the inside list must be within `` and `` tags.

This is wrong:

```
<ul>
  <li>Coffee</li>
  <li>Tea
    <ul>
      <li>Black tea</li>
      <li>Green tea</li>
    </ul>
  <li>Milk</li>
</ul>
```

This is correct:

```
<ul>
  <li>Coffee</li>
  <li>Tea
    <ul>
      <li>Black tea</li>
      <li>Green tea</li>
    </ul>
  </li>
  <li>Milk</li>
</ul>
```

Notice that we have inserted a `` tag after the `` tag in the "correct" code example.

XHTML Elements Must Always Be Closed

Non-empty elements must have a closing tag.

This is wrong:

```
<p>This is a paragraph
<p>This is another paragraph
```

This is correct:

```
<p>This is a paragraph</p>
<p>This is another paragraph</p>
```

Empty Elements Must Also Be Closed

Empty elements must also be closed.

This is wrong:

```
A break: <br>
A horizontal rule: <hr>
An image: 
```

This is correct:

```
A break: <br />
A horizontal rule: <hr />
An image: 
```

XHTML Elements Must Be In Lower Case

Tag names and attributes must be in lower case.

This is wrong:

```
<BODY>
<P>This is a paragraph</P>
</BODY>
```

This is correct:

```
<body>
<p>This is a paragraph</p>
</body>
```

XHTML Documents Must Have One Root Element

All XHTML elements must be nested within the <html> root element. Child elements must be in pairs and correctly nested within their parent element. The basic document structure is:

```
<html>
<head> ... </head>
<body> ... </body>
```

```
</html>
```

Some More XHTML Syntax Rules

- Attribute names must be in **lower case**
- Attribute values must be **quoted**
- Attribute minimization is **forbidden**
- The id attribute **replaces** the name attribute
- The XHTML DTD defines **mandatory** elements

The id Attribute Replaces The name Attribute

HTML 4.01 defines a name attribute for the elements applet, frame, iframe, and img. In XHTML the name attribute is deprecated. Use id instead.

This is wrong:

```

```

This is correct:

```

```

Note: To interoperate with older browsers for a while, you should use both name and id, with identical attribute values, like this:

```

```

IMPORTANT Compatibility Note:

To make your XHTML compatible with today's browsers, you should add an extra space before the "/" symbol.

The Lang Attribute

The lang attribute applies to almost every XHTML element. It specifies the language of the content within an element.

If you use the lang attribute in an element, you must also add the xml:lang attribute, like this:

```
<div lang="no" xml:lang="no">Heia Norge!</div>
```

Mandatory XHTML Elements

All XHTML documents must have a DOCTYPE declaration. The html, head, title, and body elements must be present.

This is an XHTML document with a minimum of required tags:

```
<!DOCTYPE Doctype goes here>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Title goes here</title>
</head>

<body>
</body>

</html>
```

Note: The DOCTYPE declaration is not a part of the XHTML document itself. It is not an XHTML element. You will learn more about the XHTML DOCTYPE in the next chapter.

Note: The xmlns attribute in <html>, specifies the xml namespace for a document, and is required in XHTML documents. However, the HTML validator at w3.org does not complain when the xmlns attribute is missing. This is because the namespace "xmlns=http://www.w3.org/1999/xhtml" is default, and will be added to the <html> tag even if you do not include it.

XHTML DTD (Document Type Definitions)

An XHTML document consists of three main parts:

- the DOCTYPE declaration
- the <head> section
- the <body> section

The basic document structure is:

```
<!DOCTYPE ...>
<html>
<head>
<title>... </title>
</head>
<body> ... </body>
</html>
```

Note: The DOCTYPE declaration is always the first line in an XHTML document!

An XHTML Example

This is a simple (minimal) XHTML document:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html>
<head>
<title>simple document</title>
</head>
```

```
<body>
<p>a simple paragraph</p>
</body>
</html>
```

The DOCTYPE declaration above defines the document type. The rest of the document looks like HTML.

Document Type Definitions (DTD)

- A DTD specifies the syntax of a web page in SGML
- DTDs are used by SGML applications, such as HTML, to specify rules for documents of a particular type, including a set of elements and entity declarations
- An XHTML DTD describes in precise, computer-readable language, the allowed syntax of XHTML markup

There are three XHTML DTDs:

- STRICT
- TRANSITIONAL
- FRAMESET

XHTML 1.0 Strict

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Use the strict DOCTYPE when you want really clean markup, free of presentational clutter. Use it together with CSS.

XHTML 1.0 Transitional

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Use the transitional DOCTYPE when you want to still use HTML's presentational features.

XHTML 1.0 Frameset

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Use the frameset DOCTYPE when you want to use HTML frames.

Empty Tags: <hr> ,
 and

Empty tags are not allowed in XHTML. The <hr> and
 tags should be replaced with <hr /> and
.

A general "find-and-replace" function was executed to swap the tags.

We decided not to close the tags with , but with /> at the end of the tag. This was done manually.

HTML and XHTML Quick Reference:

Tag name	Attributes	Description	Example
Anchor <a>..	href="URI" name="text"	used as a hypertext link or a named fragment within the document.	
Bold ..	Only core tag attributes	Renders the enclosed text in a bold font.	Turn left onto Blackstone Blvd..
Blockquote <blockquote> . .. </blockquote>	Only core tag attributes	Indicates a long quotation. Its content is some number of blocklevel elements, such as paragraphs.	<blockquote cite="http://www.example.com"> <h1>Fascinating Evidence</h1> <p>This is the beginning of a lengthy quoted passage (text continues . . .) </p> </blockquote>
Body <body> . . . </body>	Background="URI" bgcolor="#rrggbb" or "color name"	The body of a document contains the document's content.	
Br 	Only core tag attributes	Inserts a line break in the content	Channabasaveshwara Institute of Technology Gubbi
(Comments) <!-- . . . -->	Not applicable	Inserts notes or scripts into the document that are not displayed by the browser.	<!-- start secondary navigation here --> . . . (markup continues)
Div <div> . . . </div>	Only core tag attributes	Denotes a generic "division" within the flow of the document.	<div id="summary"> <h1>In Closing</h1> <p>We can summarize as follows...</p> </div>
dl <dl> . . . </dl>	Only core tag attributes	Indicates a definition list. Each item in a definition list consists of two parts: a term (dt) and description (dd), which can represent terms and definitions or other name-value pairs.	<dl> <dt><code>em</code></dt> <dd>Indicates emphasized text.</dd> </dl>
Em . . . 		Indicates emphasized text.	This is exactly what you've been looking for.
Font . . . 	color="#RRGGBB/color name" face="typeface" size="value"	An outdated method for affecting the style (color, typeface, and size) of the enclosed text	Obsolete.
Form <form> . . . </form>	action="URL" Required. method="get post" name="text"	Indicates an interactive form that contains controls for collecting user input and other page content.	<form action="/cgi-bin/guestbook.pl" method="get"> <input type="submit" /> <input type="reset"> </p> </form>
h1, h2, h3, h4, h5, h6 <hn > . . . </hn >	Only core tag attributes	Specifies a heading that briefly describes the section it introduces. There are six levels of headings, from h1 (most important)	<h1>Story Title</h1> <p>In the beginning . . . </p>

		to h6 (least important).	
head <head> . . . </head>	id="text" profile="URLs"	Defines the head portion of the document that contains information about the document	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/HTML4.01/strict.dtd"> <html> <head> <title>Document Title</title> <style type="text/css">h1 {color: #333;}</style> </head> </html>
hr HTML: <hr>; XHTML: <hr/> or <hr />	Only core tag attributes	Adds a horizontal rule to the page that can be used as a divider between sections of content. It is a block-level element.	<p>These are notes from Thursday.</p> <hr> <p>These are notes from Friday.</p>
html <html> . . . </html>	id="text" manifest="URL" version="-//W3C//DTD HTML 4.01//EN" xmlns="http://www.w3.org/1999/xhtml"	This is the root element of HTML and XHTML documents, meaning all other elements are contained within it.	<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en"> <head>... </head> <body>... </body> </html>
i <i> . . . </i>	Only core tag attributes	Enclosed text is displayed in italic.	The Western Black Widow Spider, <i>Latrodectus hesperus</i>, is commonly found
img HTML: ; XHTML: or 	align="bottom left middle right top" alt="text" src="URL" height="number" width="number"	Places an image on the page.	<p>Your ideal pet: </p>
input HTML: <input>; XHTML: <input/> or <input />	accept="MIME type" align="bottom left middle right top" name="text" src="URL" size="number" type="text password checkbox radio submit reset file hidden image button"	The input element is used to create a variety of form input controls.	<input type="button" value="Push Me!">
li . . . 	type="format" start="number" value="number"	Defines an item in a list. It is used within the ol , ul , menu , and dir list elements	 About News
meta HTML: <meta>; XHTML: <meta/> or <meta />	content="text" name="text"	Provides additional information about the document. It should be placed within the head of the document	<meta name="copyright" content="CIT gubbi">
ol	type="1 A a I i"	Defines an ordered	

<code> ... </code>		(numbered) list that consists of one or more list items (li).	<code>Get out of bed Take a shower </code>
<code>p</code> <code><p> ... </p></code>	<code>align="center left right"</code>	Denotes a paragraph	<code><p> Paragraphs are the most rudimentary elements of a text document.</p></code>
<code>pre</code> <code><pre> ... </pre></code>	<code>width="number"</code>	Delimits "preformatted" text, meaning that lines are displayed exactly as they are typed in, honoring whitespace such as multiple character spaces and line breaks.	<code><pre> This is an example of text with a lot of Curious whitespace. </pre></code>
<code>script</code> <code><script> ... </script></code>	<code>id="text"</code> <code>src="URL"</code> <code>type="content-type"</code>	Places a script in the document (usually JavaScript for web documents).	<code><script type="text/javascript"> // <![CDATA[... JavaScript code goes here ... //]]> </script></code>
Table <code><table> ... </table></code>	<code>align="left right center"</code> <code>cellpadding="number"</code> <code>cellspacing="number"</code> <code>border="number"</code> <code>bgcolor="#rrggbb" or "color name"</code> <code>width="number" or "percentage"</code> <code>height="number" or "percentage"</code>	Indicates a table used for displaying rows and columns of data or information.	<code><table width="70%" cellpadding="10"> <tr> <td>cell 1</td><td>cell 2</td> </tr> <tr> <td>cell 3</td><td>cell 4</td> </tr> </table></code>
<code>td <td> ... </td></code>	<code>align="left right center justify char"</code> <code>bgcolor="#rrggbb" or "color name"</code> <code>colspan="number"</code> <code>height="pixels" or "percentage"</code> <code>rowspan="number"</code> <code>width="pixels" or "percentage"</code>	Defines a table data cell.	<code><table> <tr> <td colspan="2">Cell 1</td> </tr> <tr> <td>Cell 3</td><td>Cell 4</td> </tr> </table></code>
<code>tr</code> <code><tr> ... </tr></code>	<code>align="left right center justify char"</code> <code>bgcolor="#rrggbb" or "color name"</code> <code>valign="top middle bottom baseline"</code>	Defines a row of cells within a table	<code><table> <tr> <td>cell 1</td><td>cell 2</td> </tr> <tr> <td>cell 3</td><td>cell 4</td> </tr> </table></code>

CSS TUTORIAL

What is CSS?

- **CSS** stands for **Cascading Style Sheets**
- Styles define **how to display** HTML elements
- Styles are normally stored in **Style Sheets**
- Styles were added to HTML 4.0 **to solve a problem**
- **External Style Sheets** can save a lot of work
- External Style Sheets are stored in **CSS files**
- Multiple style definitions will **cascade** into one

Styles solved a big problem

The original HTML was never intended to contain tags for formatting a document. HTML tags were intended to define the content of a document, like:

```
<p>This is a paragraph.</p>
```

```
<h1>This is a heading</h1>
```

When tags like `` and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large web sites where fonts and color information had to be added to every single Web page, became a long, expensive and unduly painful process.

To solve this problem, the World Wide Web Consortium (W3C) - responsible for standardizing HTML - created CSS in addition to HTML 4.0.

With HTML 4.0, all formatting can be removed from the HTML document and stored in a separate CSS file.

All browsers support CSS today.

Styles save a lot of work

Styles sheets define HOW HTML elements are to be displayed.

Styles are normally saved in external .css files. External style sheets enable you to change the appearance and layout of all the pages in a Web site, just by editing one single CSS document!

Multiple styles will cascade into one

Style sheets allow style information to be specified in many ways.

Styles can be specified:

- inside an HTML element
- inside the head section of an HTML page
- in an external CSS file

Tip: Even multiple external style sheets can be referenced inside a single HTML document.

Cascading order - What style will be used when there is more than one style specified for an HTML element?

Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number four has the highest priority:

1. Browser default
2. External style sheet
3. Internal style sheet (in the head section)
4. Inline style (inside an HTML element)

So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or in a browser (a default value).

If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!

Syntax

The CSS syntax is made up of three parts: a selector, a property and a value:

```
selector {property:value}
```

The selector is normally the HTML element/tag you wish to define, the property is the attribute you wish to change, and each property can take a value. The property and value are separated by a colon, and surrounded by curly braces:

```
body {color:black}
```

Note: If the value is multiple words, put quotes around the value:

```
p {font-family:"sans serif"}
```

Note: If you want to specify more than one property, you must separate each property with a semicolon. The example below shows how to define a center aligned paragraph, with a red text color:

```
p {text-align:center;color:red}
```

To make the style definitions more readable, you can describe one property on each line, like this:

```
P {  
text-align:center;  
color:black;  
font-family:arial  
}
```

Grouping

You can group selectors. Separate each selector with a comma. In the example below we have grouped all the header elements. All header elements will be displayed in green text color:

```
h1,h2,h3,h4,h5,h6
{
color:green
}
```

The class Selector

With the class selector you can define different styles for the same type of HTML element.

Say that you would like to have two types of paragraphs in your document: one right-aligned paragraph, and one center-aligned paragraph. Here is how you can do it with styles:

```
p.right {text-align:right}
p.center {text-align:center}
```

You have to use the class attribute in your HTML document:

```
<p class="right">This paragraph will be right-aligned.</p>
<p class="center">This paragraph will be center-aligned.</p>
```

Note: To apply more than one class per given element, the syntax is:

```
<p class="center bold">This is a paragraph.</p>
```

The paragraph above will be styled by the class "center" AND the class "bold".

You can also omit the tag name in the selector to define a style that will be used by all HTML elements that have a certain class. In the example below, all HTML elements with class="center" will be center-aligned:

```
.center {text-align:center}
```

In the code below both the h1 element and the p element have class="center". This means that both elements will follow the rules in the ".center" selector:

```
<h1 class="center">This heading will be center-aligned</h1>
<p class="center">This paragraph will also be center-aligned.</p>
```

Do **NOT** start a class name with a number! It will not work in Mozilla/Firefox.

Add Styles to Elements with Particular Attributes

You can also apply styles to HTML elements with particular attributes.

The style rule below will match all input elements that have a type attribute with a value of "text":

```
input[type="text"] {background-color:blue}
```

The id Selector

You can also define styles for HTML elements with the id selector. The id selector is defined as a #.

The style rule below will match the element that has an id attribute with a value of "green":

```
#green {color:green}
```

The style rule below will match the p element that has an id with a value of "para1":

```
p#para1
{
text-align:center;
color:red
}
```

Do **NOT** start an ID name with a number! It will not work in Mozilla/Firefox.

CSS Comments

Comments are used to explain your code, and may help you when you edit the source code at a later date. A comment will be ignored by browsers. A CSS comment begins with "/*", and ends with "*/", like this:

```
/*This is a comment*/
p
{
text-align:center;
/*This is another comment*/
color:black;
font-family:arial
}
```

How to Insert a Style Sheet

When a browser reads a style sheet, it will format the document according to it. There are three ways of inserting a style sheet:

External Style Sheet

An external style sheet is ideal when the style is applied to many pages. With an external style sheet, you can change the look of an entire Web site by changing one file. Each page must link to the style sheet using the <link> tag. The <link> tag goes inside the head section:

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

The browser will read the style definitions from the file mystyle.css, and format the document according to it.

An external style sheet can be written in any text editor. The file should not contain any html tags. Your style sheet should be saved with a .css extension. An example of a style sheet file is shown below:

```
hr {color:sienna}
p {margin-left:20px}
body {background-image:url("images/back40.gif")}
```

Do not leave spaces between the property value and the units! "margin-left:20 px" (instead of "margin-left:20px") will only work in IE6, but it will not work in Firefox or Opera.

Internal Style Sheet

An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag, like this:

```
<head>
<style type="text/css">
hr {color:sienna}
p {margin-left:20px}
body {background-image:url("images/back40.gif")}
</style>
</head>
```

The browser will now read the style definitions, and format the document according to it.

Note: A browser normally ignores unknown tags. This means that an old browser that does not support styles, will ignore the <style> tag, but the content of the <style> tag will be displayed on the page. It is possible to prevent an old browser from displaying the content by hiding it in the HTML comment element:

```
<head>
```

```
<style type="text/css">
<!--
hr {color:sienna}
p {margin-left:20px}
body {background-image:url("images/back40.gif")}
-->
</style>
</head>
```

Inline Styles

An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly, such as when a style is to be applied to a single occurrence of an element.

To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property. The example shows how to change the color and the left margin of a paragraph:

```
<p style="color:sienna;margin-left:20px">This is a paragraph.</p>
```

Multiple Style Sheets

If some properties have been set for the same selector in different style sheets, the values will be inherited from the more specific style sheet.

For example, an external style sheet has these properties for the h3 selector:

```
h3 {
color:red;
text-align:left;
font-size:8pt
}
```

And an internal style sheet has these properties for the h3 selector:

```
h3 {
text-align:right;
font-size:20pt
}
```

If the page with the internal style sheet also links to the external style sheet the properties for h3 will be:

```
color:red;
text-align:right;
font-size:20pt
```

JAVASCRIPT TUTORIAL

What is JavaScript?

- JavaScript was designed to add interactivity to HTML pages
- JavaScript is a scripting language
- A scripting language is a lightweight programming language
- JavaScript is usually embedded directly into HTML pages
- JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
- Everyone can use JavaScript without purchasing a license

Are Java and JavaScript the same?

NO!

Java and JavaScript are two completely different languages in both concept and design!

Java (developed by Sun Microsystems) is a powerful and much more complex programming language - in the same category as C and C++.

What can a JavaScript do?

- **JavaScript gives HTML designers a programming tool** - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages
- **JavaScript can put dynamic text into an HTML page** - A JavaScript statement like this: `document.write("<h1>" + name + "</h1>")` can write a variable text into an HTML page
- **JavaScript can react to events** - A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element
- **JavaScript can read and write HTML elements** - A JavaScript can read and change the content of an HTML element
- **JavaScript can be used to validate data** - A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing
- **JavaScript can be used to detect the visitor's browser** - A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser
- **JavaScript can be used to create cookies** - A JavaScript can be used to store and retrieve information on the visitor's computer

The Real Name is ECMAScript

JavaScript's official name is ECMAScript.

ECMAScript is developed and maintained by the [ECMA organization](#).

ECMA-262 is the official JavaScript standard.

The language was invented by Brendan Eich at Netscape (with Navigator 2.0), and has appeared in all Netscape and Microsoft browsers since 1996.

The development of ECMA-262 started in 1996, and the first edition of was adopted by the ECMA General Assembly in June 1997.

The standard was approved as an international ISO (ISO/IEC 16262) standard in 1998.

The development of the standard is still in progress.

The HTML <script> tag is used to insert a JavaScript into an HTML page.

Put a JavaScript into an HTML page

The example below shows how to use JavaScript to write text on a web page:

```
<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>
```

The example below shows how to add HTML tags to the JavaScript:

```
<html>
<body>
<script type="text/javascript">
document.write("<h1>Hello World!</h1>");
</script>
</body>
</html>
```

Example Explained

To insert a JavaScript into an HTML page, we use the <script> tag. Inside the <script> tag we use the type attribute to define the scripting language.

So, the <script type="text/javascript"> and </script> tells where the JavaScript starts and ends:

```
<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>
```

The **document.write** command is a standard JavaScript command for writing output to a page.

By entering the document.write command between the <script> and </script> tags, the browser will recognize it as a JavaScript command and execute the code line. In this case the browser will write Hello World! to the page:

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

```
document.write("Hello World!");
</script>
</body>
</html>
```

Note: If we had not entered the `<script>` tag, the browser would have treated the `document.write("Hello World!")` command as pure text, and just write the entire line on the page.

How to Handle Simple Browsers

Browsers that do not support JavaScript, will display JavaScript as page content.

To prevent them from doing this, and as a part of the JavaScript standard, the HTML comment tag should be used to "hide" the JavaScript.

Just add an HTML comment tag `<!--` before the first JavaScript statement, and a `-->` (end of comment) after the last JavaScript statement, like this:

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

The two forward slashes at the end of comment line (`//`) is the JavaScript comment symbol. This prevents JavaScript from executing the `-->` tag.

JavaScripts in the body section will be executed WHILE the page loads.

JavaScripts in the head section will be executed when CALLED.

Where to Put the JavaScript

JavaScripts in a page will be executed immediately while the page loads into the browser. This is not always what we want. Sometimes we want to execute a script when a page loads, other times when a user triggers an event

Scripts in `<head>`

Scripts to be executed when they are called, or when an event is triggered, go in the head section.

If you place a script in the head section, you will ensure that the script is loaded before anyone uses it.

```
<html>
<head>
<script type="text/javascript">
function message()
{
alert("This alert box was called with the onload event");
}
</script>
```

```
</script>
</head>

<body onload="message()" >
</body>
</html>
```

Scripts in <body>

Scripts to be executed when the page loads go in the body section.

If you place a script in the body section, it generates the content of a page.

```
<html>
<head>
</head>

<body>
<script type="text/javascript">
document.write("This message is written by JavaScript");
</script>
</body>

</html>
```

Scripts in <head> and <body>

You can place an unlimited number of scripts in your document, so you can have scripts in both the body and the head section.

```
<html>
<head>
<script type="text/javascript">
....
</script>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Using an External JavaScript

If you want to run the same JavaScript on several pages, without having to write the same script on every page, you can write a JavaScript in an external file.

Save the external JavaScript file with a .js file extension.

Note: The external script cannot contain the <script> tag!

To use the external script, point to the .js file in the "src" attribute of the <script> tag:

```
<html>
<head>
<script type="text/javascript" src="xxx.js"></script>
```

```
</head>  
<body>  
</body>  
</html>
```

Note: Remember to place the script exactly where you normally would write the script!

JavaScript is a sequence of statements to be executed by the browser.

JavaScript is Case Sensitive

Unlike HTML, JavaScript is case sensitive - therefore watch your capitalization closely when you write JavaScript statements, create or call variables, objects and functions.

JavaScript Statements

A JavaScript statement is a command to a browser. The purpose of the command is to tell the browser what to do.

This JavaScript statement tells the browser to write "Hello Dolly" to the web page:

```
document.write("Hello Dolly");
```

It is normal to add a semicolon at the end of each executable statement. Most people think this is a good programming practice, and most often you will see this in JavaScript examples on the web.

The semicolon is optional (according to the JavaScript standard), and the browser is supposed to interpret the end of the line as the end of the statement. Because of this you will often see examples without the semicolon at the end.

Note: Using semicolons makes it possible to write multiple statements on one line.

JavaScript Code

JavaScript code (or just JavaScript) is a sequence of JavaScript statements.

Each statement is executed by the browser in the sequence they are written.

This example will write a heading and two paragraphs to a web page:

```
<script type="text/javascript">  
document.write("<h1>This is a heading</h1>");  
document.write("<p>This is a paragraph.</p>");  
document.write("<p>This is another paragraph.</p>");  
</script>
```

JavaScript Blocks

JavaScript statements can be grouped together in blocks.

Blocks start with a left curly bracket {, and ends with a right curly bracket }.

The purpose of a block is to make the sequence of statements execute together.

This example will write a heading and two paragraphs to a web page:

```
<script type="text/javascript">
{
document.write("<h1>This is a heading</h1>");
document.write("<p>This is a paragraph.</p>");
document.write("<p>This is another paragraph.</p>");
}
</script>
```

The example above is not very useful. It just demonstrates the use of a block. Normally a block is used to group statements together in a function or in a condition (where a group of statements should be executed if a condition is met).

You will learn more about functions and conditions in later chapters.

INTRODUCTION TO PHP

What is PHP?

PHP (recursive acronym for *PHP: Hypertext Preprocessor*) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

Nice, but what does that mean? An example:

Example #1 An introductory example

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Example</title>
  </head>
  <body>

    <?php
      echo "Hi, I'm a PHP script!";
    ?>

  </body>
</html>
```

Instead of lots of commands to output HTML (as seen in C or Perl), PHP pages contain HTML with embedded code that does "something" (in this case, output "Hi, I'm a PHP script!"). The PHP code is enclosed in special start and end processing instructions `<?php` and `?>` that allow you to jump into and out of "PHP mode."

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

What can PHP do?

Anything. PHP is mainly focused on server-side scripting, so you can do anything any other CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. But PHP can do much more.

There are three main areas where PHP scripts are used.

- Server-side scripting. This is the most traditional and main target field for PHP. You need three things to make this work. The PHP parser (CGI or server module), a web server and a web browser. You need to run the web server, with a connected PHP installation. You can access the PHP program output with a web browser, viewing the PHP page through the server. All these can run on your home machine if you are just experimenting with PHP programming. See the installation instructions section for more information.
- Command line scripting. You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed using cron (on *nix or Linux) or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks. See the section about Command line usage of PHP for more information.
- Writing desktop applications. PHP is probably not the very best language to create a desktop application with a graphical user interface, but if you know PHP very well, and would like to use some advanced PHP features in your client-side applications you can also use PHP-GTK to write such programs. You also have the ability to write cross-platform applications this way.

PHP can be used on all major operating systems, including Linux, many Unix variants (including HP-UX, Solaris and OpenBSD), Microsoft Windows, Mac OS X, RISC OS, and probably others. PHP has also support for most of the web servers today. This includes Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet servers, O'Reilly Website Pro server, Caudium, Xitami, OmniHTTPd, and many others. For the majority of the servers, PHP has a module, for the others supporting the CGI standard, PHP can work as a CGI processor.

Lab Experiments

Experiment No: 1

Date:

Aim:

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

```
<html>
<head>
<title>My calculator</title>
<script type="text/javascript">
function call(click_id)
{
var v1=parseFloat(document.getElementById("ip1").value);
var v2=parseFloat(document.getElementById("ip2").value);

if(isNaN(v1) || isNaN(v2))
alert("enter a valid number");

else if(click_id=="add")
document.getElementById("output").value=v1+v2;

else if(click_id=="sub")
document.getElementById("output").value=v1-v2;

else if(click_id=="mul")
document.getElementById("output").value=v1*v2;

else if(click_id=="div")
document.getElementById("output").value=v1/v2;
}
</script>
</head>
<body>
<center>
<h1> A SIMPLE CALCULATOR PROGRAM</h1>
<table style="background-color:yellow" align=="center">
<tr>
<td>
<form method="get" action="">
<div width=50% align="center">
<label>OP1<input type="text" id="ip1"/></label>
<label>op2<input type="text" id="ip2"/></label>
<label>total<input type="text" id="output"/></label>
</div>
<br>
<div width=50% align="center">
<input type="button" value="+" id="add" onclick="call(this.id)"/>
<input type="button" value="-" id="sub" onclick="call(this.id)"/>
<input type="button" value="*" id="mul" onclick="call(this.id)"/>
<input type="button" value="/" id="div" onclick="call(this.id)"/>
<input type="reset" value="clear"/>
</div>
</form>
</td>
</tr>
```

```
</table>
</center>
</body>
</html>
```

OUTPUT:-

A SIMPLE CALCULATOR PROGRAM

OP1	<input type="text"/>	op2	<input type="text"/>	total	<input type="text"/>
<input type="button" value="+"/> <input type="button" value="-"/> <input type="button" value="*"/> <input type="button" value="/"/> <input type="button" value="clear"/>					

A SIMPLE CALCULATOR PROGRAM

OP1	<input type="text" value="3"/>	op2	<input type="text" value="4"/>	total	<input type="text" value="7"/>
<input type="button" value="+"/> <input type="button" value="-"/> <input type="button" value="*"/> <input type="button" value="/"/> <input type="button" value="clear"/>					

Experiment No: 2**Date:****Aim:**

Write a JavaScript 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.

```
<html>
<head>
</head>
<body>
<table align="center" border=1>
<tr><td>number</td><td>square</td><td>cube</td></tr>
<script type="text/javascript">
for(var n=0; n<=10; n++)
{
document.write( "<tr><td>" + n + "</td><td>" + n*n + "</td><td>" + n*n*n
+ "</td></tr>" );
}
</script>
</table>
</body>
</html>
```

OUTPUT:-

The Sqaure & the Cube from 0 to 10

number	square	cube
0	0	0
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Experiment No: 3**Date:****Aim:**

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.

```
<html>
<body>
<p id="demo"></p>
<script>
var var1 = setInterval(inTimer, 1000);
var size = 5;
var ids = document.getElementById("demo");
function inTimer() {
ids.innerHTML = 'TEXT GROWING';
ids.setAttribute('style', "font-size: " + size + "px; color: red");
size += 5;
if(size >= 50 ){
clearInterval(var1);
var2 = setInterval(deTimer, 1000);
}
}
function deTimer() {
size -= 5;
ids.innerHTML = 'TEXT SHRINKING';
ids.setAttribute('style', "font-size: " + size + "px; color: blue");
if(size == 5 ){
clearInterval(var2);
}
}
</script>
</body>
</html>
Output:-
```

TEXT GROWING

TEXT SHRINKING

Experiment No: 4**Date:****Aim:**

Develop and demonstrate a HTML5 file that includes JavaScript script that uses functions for the following problems:

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

```
<html>
<body>
<script type="text/javascript">
var str = prompt("Enter the Input","");
if(isNaN(str))
{
str = str.toUpperCase();
for(var i = 0; i < str.length; i++) {
var chr = str.charAt(i);
if(chr == 'A' || chr == 'E' || chr == 'I' || chr == 'O' || chr == 'U')break;
}
if( i < str.length )
alert("The position of the left most vowel is "+(i+1));
else
alert("No vowel found in the entered string");
}
else
{
var num,rev=0,remainder;
num = parseInt(str);
while(num!=0) {
remainder = num%10;
num = parseInt(num/10);
rev = rev * 10 + remainder;
}
alert("Reverse of "+str+" is "+rev);
}
</script>
</body>
</html>
```

Output:-

localhost says

Enter the Input

localhost says

The position of the left most vowel is 1

localhost says

Enter the Input

localhost says

Reverse of 121 is 121

Experiment No: 5**Date:****Aim:**

Design an XML document to store information about a student in an engineering college affiliated to VTU. The information must include USN, Name, and Name of the College, Branch, Year of Joining, and email id. Make up sample data for 3 students. Create a CSS style sheet and use it to display the document.

```
<?xml-stylesheet type="text/css" href="6.css" ?>
<!DOCTYPE HTML>
<html>
<head>
<h1> STUDENTS DESCRIPTION </h1>
</head>
<students>
<student>
<USN>USN : 1CG15CS001</USN>
<name>NAME : SANTHOS</name>
<college>COLLEGE : CIT</college>
<branch>BRANCH : Computer Science and Engineering</branch>
<year>YEAR : 2015</year>
<e-mail>E-Mail : santosh@gmail.com</e-mail>
</student>
<student>
<USN>USN : 1CG15IS002</USN>
<name>NAME : MANORANJAN</name>
<college>COLLEGE : CITIT</college>
<branch>BRANCH : Information Science and Engineering</branch>
<year>YEAR : 2015</year>
<e-mail>E-Mail : manoranjana@gmail.com</e-mail>
</student>
<student>
<USN>USN : 1CG15EC101</USN>
<name>NAME : CHETHAN</name>
<college>COLLEGE : CITIT</college>
<branch>BRANCH : Electronics and Communication Engineering
</branch>
<year>YEAR : 2015</year>
<e-mail>E-Mail : chethan@gmail.com</e-mail>
</student>
</students>
</html>
6.CSS
```

```
student{
display:block; margin-top:10px; color:Navy;
}
USN{
display:block; margin-left:10px;font-size:14pt; color:Red;
}
name{
```

```
display:block; margin-left:20px;font-size:14pt; color:Blue;
}
college{
display:block; margin-left:20px;font-size:12pt; color:Maroon;
}
branch{
display:block; margin-left:20px;font-size:12pt; color:Purple;
}
year{
display:block; margin-left:20px;font-size:14pt; color:Green;
}
e-mail{
display:block; margin-left:20px;font-size:12pt; color:Blue;
}
```

OUTPUT:-

STUDENTS DESCRIPTION

USN : 1CG15CS001

NAME : SANTHOS

COLLEGE : CIT

BRANCH : Computer Science and Engineering

YEAR : 2015

E-Mail : santosh@gmail.com

USN : 1CG15IS002

NAME : MANORANJAN

COLLEGE : CITIT

BRANCH : Information Science and Engineering

YEAR : 2015

E-Mail : manoranjan@gmail.com

USN : 1CG15EC101

NAME : CHETHAN

COLLEGE : CITIT

BRANCH : Electronics and Communication Engineering

YEAR : 2015

E-Mail : chethan@gmail.com

Experiment No: 6**Date:****Aim:**

Write a PHP program to keep track of the number of visitors visiting the web page and to display this count of visitors, with proper headings.

```
<?php
print "<h3> REFRESH PAGE </h3>";
$name="counter.txt";
$file = fopen($name,"r");
$hits= fscanf($file,"%d");
fclose($file);
$hits[0]++;
$file = fopen($name,"w");
fprintf($file,"%d",$hits[0]);
fclose($file);
print "Total number of views: ".$hits[0];
?>
```

OUTPUT:-


REFRESH PAGE

Total number of views: 6

Experiment No: 7**Date:****Aim:**

Write a PHP program to display a digital clock which displays the current time of the server.

```
<html>
<head>
<meta http-equiv="refresh" content="1"/>
<style>
p {
color:white;
font-size:90px;
position: absolute;
top: 50%;
left: 50%;
transform: translate(-50%, -50%);
}
body{background-color:black;}
</style>
<p> <?php echo date(" h: i : s A");?> </p>
</head>
</html>
Output :-
```



10: 44 : 08 AM

Experiment No: 8**Date:****Aim:****Write the PHP 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.**

```

<html>
<head>
<style>
table, td, th
{
border: 1px solid black;
width: 35%;
text-align: center;
background-color: DarkGray;
}
table { margin: auto; }
input,p { text-align:right; }
</style>
</head>
<body>
<form method="post">
<table>
<caption><h2> SIMPLE CALCULATOR </h2></caption>>
<tr><td>First Number:</td><td><input type="text" name="num1" /></td>
<td rowspan="2"><input type="submit" name="submit"
value="calculate"></td></tr>
<tr><td>Second Number:</td><td><input type="text"
name="num2"/></td></tr>
</form>
<?php
if(isset($_POST['submit']))
{
$num1 = $_POST['num1'];
$num2 = $_POST['num2'];
if(is_numeric($num1) andis_numeric($num1) )
{
echo "<tr><td> Addition :</td><td><p>".($num1+$num2)."</p></td>";
echo "<tr><td> Subtraction :</td><td><p> ".($num1-$num2)."</p></td>";
echo "<tr><td> Multiplication :</td><td><p>".($num1*$num2)."</p></td>";
echo "<tr><td>Division :</td><td><p> ".($num1/$num2)."</p></td>";
echo "</table>";
}
else
{
echo"<script type='text/javascript' > alert(' ENTER VALID
NUMBER');</script>";
}
}

```

```
?>
</body>
</html>
```

OUTPUT:-**SIMPLE CALCULATOR**

First Number:	50	calculate
Second Number:	25	
Addition :	75	
Subtraction :	25	
Multiplication :	1250	
Division :	2	

8b.php

```
<?php
$a = array(array(1,2,3),array(4,5,6),array(7,8,9));
$b = array(array(7,8,9),array(4,5,6),array(1,2,3));
$m=count($a);
$n=count($a[2]);
$p=count($b);
$q=count($b[2]);
echo "the first matrix :".<br/>";
for ($row = 0; $row < $m; $row++) {
for ($col = 0; $col < $n; $col++)
echo " ".$a[$row][$col];
echo "<br/>";
}
echo "the second matrix :".<br/>";
for ($row = 0; $row < $p; $row++) {
for ($col = 0; $col < $q; $col++)
echo " ".$b[$row][$col];
echo "<br/>";
}
echo "the transpose for the first matrix is:".<br/>";
```

```
for ($row = 0; $row < $m; $row++) {
for ($col = 0; $col < $n; $col++)
echo " ".$a[$col][$row];
echo "<br/>";
}
if(($m===$p) and ($n===$q)) {
echo "the addition of matrices is:".<br/>";
for ($row = 0; $row < 3; $row++) {
for ($col = 0; $col < 3; $col++)
echo " ".$a[$row][$col]+$b[$row][$col]." ";
echo "<br/>";
}
}
if($n===$p){
echo " The multiplication of matrices: <br/>";
$result=array();
for ($i=0; $i < $m; $i++) {
for($j=0; $j < $q; $j++){
$result[$i][$j] = 0;
for($k=0; $k < $n; $k++)
$result[$i][$j] += $a[$i][$k] * $b[$k][$j];
}
}
for ($row = 0; $row < $m; $row++) {
for ($col = 0; $col < $q; $col++)
echo " ".$result[$row][$col];
echo "<br/>";
}
}
```


?>

Output:-

the first matrix :

1 2 3

4 5 6

7 8 9

the second matrix :

7 8 9

4 5 6

1 2 3

the transpose for the first matrix is:

1 4 7

2 5 8

3 6 9

the addition of matrices is:

8 10 12

8 10 12

8 10 12

The multiplication of matrices:

18 24 30

54 69 84

90 114 138

Experiment No: 9**Date:****Aim:**

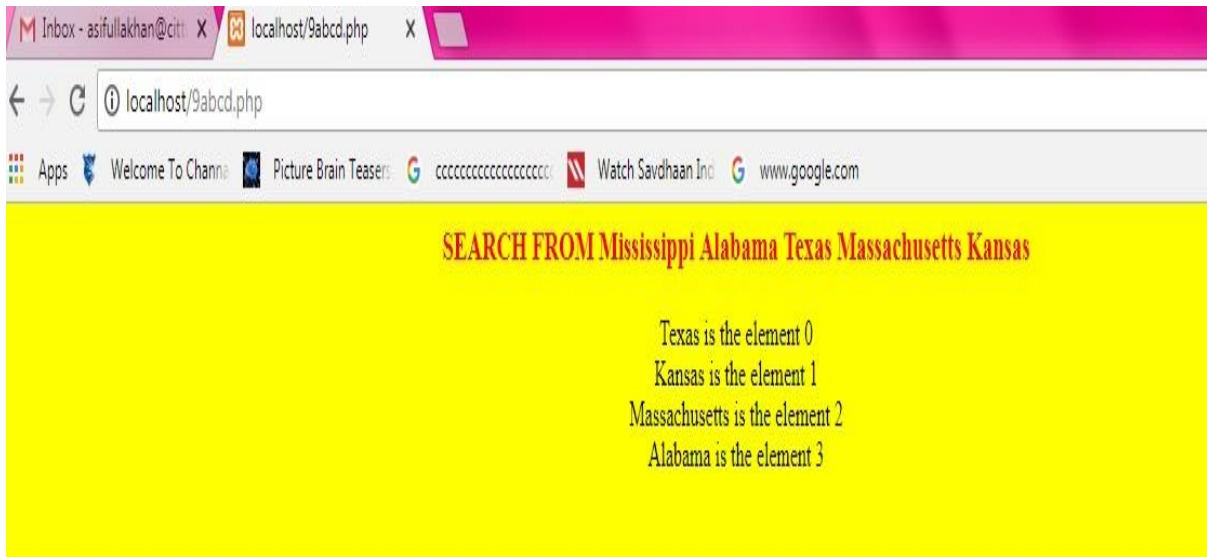
9. Write a PHP program named states.py that declares a variable states with value "Mississippi Alabama Texas Massachusetts Kansas". write a PHP program that does the following:

- 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.

```
<?php
    $states = "Mississippi Alabama Texas Massachusetts Kansas";
    $statesArray = [];
    $states1 = explode(' ', $states)
    foreach($states1 as $state) {
        if(preg_match( '/xas$/', ($state)))
            $statesArray[0] = ($state);
    }
    foreach($states1 as $state) {
        if(preg_match( '/Kansas$/', ($state)))
            $statesArray[1] = ($state);
    }
    foreach($states1 as $state) {
        if(preg_match( '/Massachusetts$/', ($state)))
            $statesArray[2] = ($state);
    }
```

```
foreach($states1 as $state) {  
    if(preg_match( '/Alabama$/', ($state)))  
        $statesArray[3] = ($state);  
}  
  
foreach ( $statesArray as $element => $value ){  
    print( $value." is the element ". $element."<br/>");  
}  
?>
```

OUTPUT:-



Experiment No: 10**Date:****Aim:**

Write a PHP program to sort the student records which are stored in the database using selection sort.

```
Goto Mysql and then type
create database weblab;
use weblab;
create table student(usn varchar(10),name varchar(20),address varchar(20));
<html>
<body>
<style>
table, td, th
{
border: 1px solid black;
width: 33%;
text-align: center;
border-collapse: collapse;
background-color: lightblue;
}
table { margin: auto; }
</style>
<?php
$servername = "localhost";
$username = "root";
$password = "root";
$dbname = "weblab";
$a=[];
// Create connection
// Opens a new connection to the MySQL server
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection and return an error description from the last
connection error, if any
if ($conn->connect_error)
die("Connection failed: " . $conn->connect_error);
$sql = "SELECT * FROM student";
// performs a query against the database
$result = $conn->query($sql);
echo "<br>";
echo "<center> BEFORE SORTING </center>";
echo "<table border='2'>";
echo "<tr>";
echo "<th>USN</th><th>NAME</th><th>Address</th></tr>";
if ($result->num_rows > 0)
{
// output data of each row and fetches a result row as an
associative array
while($row = $result->fetch_assoc()){
echo "<tr>";
echo "<td>". $row["usn"]."</td>";
```

```

echo "<td>". $row["name"]."</td>";
echo "<td>". $row["addr"]."</td></tr>";
array_push($a,$row["usn"]);
}
}
else
echo "Table is Empty";
echo "</table>";
$n=count($a);
$b=$a;
for ( $i = 0 ; $i< ($n - 1) ; $i++ )
{
$pos= $i;
for ( $j = $i + 1 ; $j < $n ; $j++ ) {
if ( $a[$pos] > $a[$j] )
$pos= $j;
}
if ( $pos!= $i ) {
$stemp=$a[$i];
$a[$i] = $a[$pos];
$a[$pos] = $stemp;
}
}
$c=[];
$d=[];
$result = $conn->query($sql);
if ($result->num_rows> 0)// output data of each row
{
while($row = $result->fetch_assoc()) {
for($i=0;$i<$n;$i++) {
if($row["usn"]== $a[$i]) {
$c[$i]=$row["name"];
$d[$i]=$row["addr"];
}
}
}
}
echo "<br>";
echo "<center> AFTER SORTING </center>";
echo "<table border='2'>";
echo "<tr>";
echo "<th>USN</th><th>NAME</th><th>Address</th></tr>";
for($i=0;$i<$n;$i++) {
echo "<tr>";
echo "<td>". $a[$i]."</td>";
echo "<td>". $c[$i]."</td>";
FDP-Web Technology Lab With Mini Project – 15CSL77 2018-2019
Dept. of ISE & CSE, RNSIT Page 30
echo "<td>". $d[$i]."</td></tr>";
}
echo "</table>";

```

```
$conn->close();
```

```
?>
```

```
</body>
```

```
</html>
```

OUTPUT:-

BEFORE SORTING

USN	NAME	Address
1rn14	chandan	bengaluru
1rn07	arun	mysore
1rn01	abhi	tumkur
1rn38	Manoranjan	Mandya

AFTER SORTING

USN	NAME	Address
1rn01	abhi	tumkur
1rn07	arun	mysore
1rn14	chandan	bengaluru
1rn38	Manoranjan	Mandya