

Internal Assessment Test 1 – Aug. 2024							
Web Technologies				Sub Code:	22MCA24	Branch:	MCA
13/8/2024	Duration:	90 min's	Max Marks:	50	Sem	II	OBE

Q1) Briefly explain the following i) Web browsers ii) URL iii) MIME

i) Web browsers

Web server operations:

- All the communications between a web client and a web server use the HTTP
- When a web server begins execution, it informs the OS under which it is running & it runs as a background process
- A web client or browser, opens a network connection to a web server, sends information requests and possibly data to the server, receives information from the server and closes the connection.
- The primary task of web server is to monitor a communication port on host machine, accept HTTP commands through that port and perform the operations specified by the commands.
- When the URL is received, it is translated into either a filename or a program name

General characteristics of web server:

- The file structure of a web server has two separate directories
- The root of one of these is called document root which stores web documents
- The root of the other directory is called the server root which stores server and its support software's
- The files stored directly in the document root are those available to clients through top level URLs
- The secondary areas from which documents can be served are called virtual document trees.
- Many servers can support more than one site on a computer, potentially reducing the cost of each site and making their maintenance more convenient. Such secondary hosts are called virtual hosts.
- Some servers can serve documents that are in the document root of other machines on the web; in this case they are called as proxy servers

ii) URL

- Uniform Resource Locators (URLs) are used to identify different kinds of resources on Internet.
- If the web browser wants some document from web server, just giving domain name is not sufficient because domain name can only be used for locating the server.
- It does not have information about which document client needs. Therefore, URL should be provided.
- The general format of URL is: scheme: object-address

Example: <http://www.vtu.ac.in/results.php>

- The scheme indicates protocols being used. (http, ftp, telnet...)
- In case of http, the full form of the object address of a URL is as follows:
//fully-qualified-domain-name/path-to-document
- URLs can never have embedded spaces
- It cannot use special characters like semicolons, ampersands and colons
- The path to the document for http protocol is a sequence of directory names and a filename, all separated by whatever special character the OS uses. (Forward or backward slashes)
- The path in a URL can differ from a path to a file because a URL need not include all directories on the path
- A path that includes all directories along the way is called a complete path.
Example: <http://www.gumboco.com/files/f99/storefront.html>
- In most cases, the path to the document is relative to some base path that is specified in the configuration files of the server. Such paths are called partial paths.
- Example: <http://www.gumboco.com/storefront.htm>

iii) MIME

- MIME stands for Multipurpose Internet Mail Extension.
- The server system apart from sending the requested document, it will also send MIME information.
- The MIME information is used by web browser for rendering the document properly.
- The format of MIME is: type/subtype
- Example: text/html , text/doc , image/jpeg , video/mpeg
- When the type is either text or image, the browser renders the document without any problem
- However, if the type is video or audio, it cannot render the document
- It has to take the help of other software like media player, win amp etc.,
- These softwares are called as helper applications or plugins
- These non-textual information are known as HYPER MEDIA
- Experimental document types are used when user wants to create a customized information & make it available in the internet
- The format of experimental document type is: type/x-subtype
Example: database/x-xbase , video/x-msvideo
- Along with creating customized information, the user should also create helper applications.
- This helper application will be used for rendering the document by browser.
- The list of MIME specifications is stored in configuration file of web server.

Q2) Explain request phase and response phase of HTTP.

Request Phase:

The general form of an HTTP request is as follows:

1. HTTP method Domain part of the URL HTTP version

2. Header fields
3. Blank line
4. Message body

The following is an example of the first line of an HTTP request:

```
GET /storefront.html HTTP/1.1
```

Table 1.1 HTTP request methods

Method	Description
GET	Returns the contents of the specified document
HEAD	Returns the header information for the specified document
POST	Executes the specified document, using the enclosed data
PUT	Replaces the specified document with the enclosed data
DELETE	Deletes the specified document

The format of a header field is the field name followed by a colon and the value of the field.

There are four categories of header fields:

1. General: For general information, such as the date
2. Request: Included in request headers
3. Response: For response headers
4. Entity: Used in both request and response headers

A wildcard character, the asterisk (*), can be used to specify that part of a MIME type can be anything.

The Host: host name request field gives the name of the host. The Host field is required for HTTP 1.1. The If-Modified-Since: date request field specifies that the requested file should be sent only if it has been modified since the given date. If the request has a body, the length of that body must be given with a Content-length field. The header of a request must be followed by a blank line, which is used to separate the header from the body of the request.

The Response Phase:

The general form of an HTTP response is as follows:

1. Status line
2. Response header fields
3. Blank line
4. Response body

The status line includes the HTTP version used, a three-digit status code for the response, and a short textual explanation of the status code.

For example, most responses begin with the following:

HTTP/1.1 200 OK

The status codes begin with 1, 2, 3, 4, or 5. The general meanings of the five categories specified by these first digits are shown in Table 1.2.

Table 1.2 First digits of HTTP status codes

First Digit	Category
1	Informational
2	Success
3	Redirection
4	Client error
5	Server error

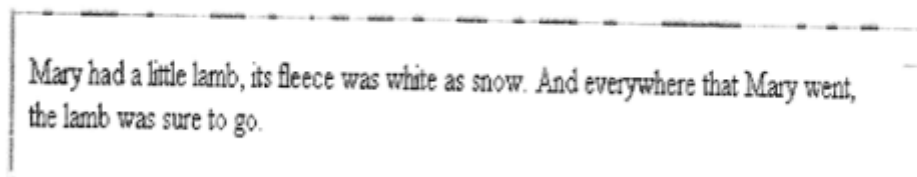
One of the more common status codes is one user never want to see: 404 Not Found, which means the requested file could not be found.

Q3) Explain any five text markup tags

1) Paragraphs

- XHTML does not allow text to be directly placed into the document. Text is normally organised into paragraphs
- It begins with and ends with Multiple paragraphs may appear in a single document

```
<p>
  Mary had
a
  little lamb, its fleece was white as snow. And
everywhere that
  Mary went, the lamb
was sure to go.
</p>
```

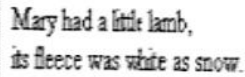


Mary had a little lamb, its fleece was white as snow. And everywhere that Mary went, the lamb was sure to go.

2) Line Breaks

- Sometime text requires a line break without the preceding blank line.
- This exactly what the break tag does. Break tag does not have any content therefore it is self-closing tag
- The break tag is specified as `
` The slash indicates that the tag is both an opening and closing tag.

```
<p>
Mary had a little lamb, <br />
  its fleece was white as snow.
</p>
```

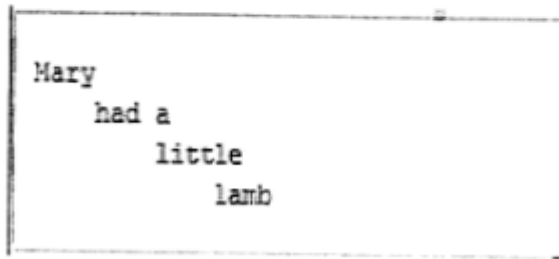


Mary had a little lamb,
its fleece was white as snow.

3) Preserving Whitespace

Sometimes it is desirable to preserve the white space in text—that is, to prevent the browser from eliminating multiple spaces and ignoring embedded line breaks. This can be specified with the `<pre>` tag.

```
<p><pre>
Mary
  had a
    little
      lamb
</pre>
```



Mary
had a
little
lamb

4) Headings

In XHTML, there are six levels of headings, specified by the tags `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, and `<h6>`, where `<h1>` specifies the highest-level heading. Headings are usually displayed in a boldface font whose default size depends on the number in the heading tag. On most browsers, `<h1>`, `<h2>`, and `<h3>` use font sizes that are larger than that of the default size of text, `<h4>` uses the default size, and `<h5>` and `<h6>` use smaller sizes. The heading tags always break the current line, so their content always appears on a new line. Browsers usually insert some vertical space before and after all headings

```
<html>
<head><title> Headings </title></head>
<body>
<h1> Heading 1 </h1>
<h2> Heading 2 </h2>
<h3> Heading 3 </h3>
<h4> Heading 4 </h4>
<h5> Heading 5 </h5>
<h6> Heading 6 </h6>
</body>
```

</html>

5) Block Quotations

The <blockquote> tag is used to make the contents look different from the surrounding text.

<p>Here is a quote from WWF's website:</p>

<blockquote cite="http://www.worldwildlife.org/who/index.html">

For 50 years, WWF has been protecting the future of nature. The world's leading conservation organization, WWF works in 100 countries and is supported by 1.2 million members in the United States and close to 5 million globally.

</blockquote>

Q4) Explain the following tags with example i)Image ii)Link iii)List

1. Image

The Tag

The image tag, , which is an inline tag, specifies an image that is to appear in a document. In its simplest form, the image tag includes two attributes: **src**, which specifies the file containing the image; and **alt**, which specifies text to be displayed when it is not possible to display the image. If the file is in the same directory as the XHTML file of the document, the value of **src** is just the image's filename. In many cases, image files are stored in a subdirectory of the directory where the XHTML files are stored. For example, the image files might be stored in a subdirectory named **images**. If the image file's name is **stars.jpg** and it is stored in the **images** subdirectory, the value of **src** would be as follows:

```
"images/stars.jpg"
```

Two optional attributes of **img**, **width** and **height**, can be included to specify (in pixels) the size of the rectangle for the image. These can be used to scale the size of the image (that is, to make it larger or smaller). Care must be taken to ensure that the image is not distorted in the resizing. For example, if the image is square, the **width** and **height** attribute values must be equal.

The following is an example of an image element:

```
<img src = "c210.jpg" alt = "Picture of a Cessna 210" />
```

2. Link

Links are specified in an attribute of an anchor tag (<a>), which is an inline tag. The anchor tag that specifies a link is called the *source* of that link. The document whose address is specified in a link is called the *target* of that link.

As is the case with many tags, the anchor tag can include many different attributes. However, for creating links only one is required, href (an acronym for hypertext reference). The value assigned to href specifies the target of the link. If the target is in another document in the same directory, the target is just the document's filename. If the target document is in some other directory, the UNIX pathname conventions are used. So, an XHTML file named c210data.html in a subdirectory of the directory in which the source XHTML file—say, named airplanes—is specified in the href attribute as airplanes/c210data.html. This is the relative method of document addressing. Absolute file addresses could be used in which the entire pathname for the file is given. However, relative links are easier to maintain, especially if a hierarchy of XHTML files must be moved. If the document is on some other machine (not on the server providing the document that includes the link), the complete URL obviously must be used.

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- link.html
  An example to illustrate a link
-->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> A link </title>
  </head>
  <body>
    <h1> Aidan's Airplanes </h1>
    <h2> The best in used airplanes </h2>
    <h3> "We've got them by the hangarful" </h3>
    <h2> Special of the month </h2>
    <p>
      1960 Cessna 210 <br />
      <a href = "C210data.html"> Information on the Cessna 210 </a>
    </p>
  </body>
</html>
```

Links can include images in their content, in which case the browser displays the image with the link:

```
<a href = "c210data.html" >
  <img src = "small-airplane.jpg"
    alt = "An image of a small airplane" />
  Information on the Cessna 210
</a>
```

Targets within Documents

If the target of a link is not at the beginning of a document, it must be some element within the document, in which case there must be some means of specifying it. The target element can include an id attribute, which can then be used to identify it in an href attribute. Consider following example

```
<h2 id="avionics"> Avionics</h2>
```

The target is specified in the href attribute value by preceding the id value with a pound sign (#), as in the following example

```
<a href="#avionics">What about avionics? </a>
```

When target is a part or fragment of another document, the name of the part is specified at the end of URL, separated by pound sign(#)

```
<a href="AIDAN1.html#avionics">Avionics</a>
```

3. List

1) Unordered List

The tag, which is a block tag, creates an unordered list. Each item in a list is specified with an tag (li is an acronym for list item). Any tags can appear in a list item, including nested lists. When displayed, each list item is implicitly preceded by a bullet

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- unordered.html
  An example to illustrate an unordered list
-->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Unordered list </title>
  </head>
  <body>
    <h3> Some Common Single-Engine Aircraft </h3>
    <ul>
      <li> Cessna Skyhawk </li>
      <li> Beechcraft Bonanza </li>
      <li> Piper Cherokee </li>
    </ul>
  </body>
</html>
```

2) Ordered List

Ordered lists are lists in which the order of items is important. This ordered-ness of a list is shown in the display of the list by the implicit attachment of a sequential value to the beginning of each item. The default sequential values are Arabic numerals, beginning with 1.

An ordered list is created within the block tag . The items are specified and displayed just as are those in unordered lists, except that the items in an ordered list are preceded by sequential values instead of bullets.


```

<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- ordered.html
An example to illustrate an ordered list
-->
<html xmlns = "http://www.w3.org/1999/xhtml">
<head> <title> Ordered list </title>
</head>
<body>
<h3> Cessna 210 Engine Starting Instructions </h3>
<ol>
<li> Set mixture to rich </li>
<li> Set propeller to high RPM </li>
<li> Set ignition switch to "BOTH" </li>
<li> Set auxiliary fuel pump switch to "LOW PRIME" </li>
<li> When fuel pressure reaches 2 to 2.5 PSI, push
starter button
</li>
</ol>
</body>
</html>

```

3) Definition List

As the name implies, definition lists are used to specify lists of terms and their definitions, as in glossaries. A definition list is given as the content of a <dl> tag, which is a block tag. Each term to be defined in the definition list is given as the content of a <dt>tag. The definitions themselves are specified as the content of <dd> tags. The defined terms of a definition list are usually displayed in the left margin; the definitions are usually shown indented on the line or lines following the term.

```

<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<!-- definition.html
An example to illustrate definition lists
-->
<html xmlns = "http://www.w3.org/1999/xhtml">
<head> <title> Definition lists </title>
</head>
<body>
<h3> Single-Engine Cessna Airplanes </h3>
<dl>
<dt> 152 </dt>
<dd> Two-place trainer </dd>
<dt> 172 </dt>
<dd> Smaller four-place airplane </dd>
<dt> 182 </dt>
<dd> Larger four-place airplane </dd>
<dt> 210 </dt>
<dd> Six-place airplane - high performance </dd>
</dl>
</body>
</html>

```

Q5) Explain Frameset and frame tags with examples

The browser window can be used to display more than one document at a time. The window can be divided into rectangular areas, each of which is a frame. Each frame is capable of displaying its own document.

Framesets:

- The number of frames and their layout in the browser window are specified with the <frameset> tag.
- A frameset element takes the place of the body element in a document. A document

has either a body or a frameset but cannot have both.

- The <frameset> tag must have either a rows or a cols attribute. (or both)
- To create horizontal frames, rows attribute is used.
- To create vertical frames, cols attribute is used.
- The values for these attributes can be numbers, percentages and asterisks. Two or more values are separated by commas & given in quoted string.
 - ☑ Number- specify height of one row in pixels.
<frameset rows="200, 300, 400">
 - ☑ Percentages – Given as number followed immediately by percentage sign.
Percentage value specify the percentage of total browser window height that a row should occupy.
<frameset rows="22%, 33%, 45%">
 - ☑ Asterisks – it means the remainder of window height
<frameset rows="22%, 33%, *">
- The cols attribute is very much like the rows attribute, except it specifies the number of columns of frames.
- <frameset rows = " 23%, 33%, 33% " cols= " 25%, *" >

Frames

- The content of frame is specified with <frame> tag, which can appear only in the content of a frameset element.
- The frames in the frameset appear by rows.
<frame src="apples.html">
- The content of a frame is specified as the value of the src attribute in the <frame> tag. If <frame> tag has no src attribute browser displays empty frame.
- If the content of frame does not fit in given frameset scroll bars are implicitly included.
- If you want frame to have scroll bar, regardless of the size of its content, the <frame> attribute scrolling can be set to yes

Frameset can also be nested

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

<!-- nested_frames.html
  An example to illustrate nested frames
-->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Nested frames </title>
  </head>
  <frameset cols = "40%, *">
    <frameset rows = "50%, *">
      <frame src = "frame1.html" />
      <frame src = "frame2.html" />
    </frameset>
    <frameset rows = "20%, 35%, *">
      <frame src = "frame3.html" />
      <frame src = "frame4.html" />
      <frame src = "frame5.html" />
    </frameset>
  </frameset>
</html>
```

Q6) Write a XHTML program to create following table

Students Information	Name	Address	
		City	House No
	A	HND	1
	B	LHR	2
	C	SWL	3
	D	BWP	4

```
<!DOCTYPE HTML>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>MCA Department</title>
  </head>
  <body>

    <table border="1" cellspacing="0" cellpadding="8">

      <tr valign="center">
        <td rowspan="6">Student Information</td>
        <td rowspan="2">Name</td>
        <td colspan="2">Address</td>
      </tr>
      <tr valign="top">
        <td>City</td>
        <td>House No.</td>
      </tr>
      <tr >
        <td align="center">A</td>
        <td align="left">HND</td>
        <td align="right">1</td>
      </tr>
      <tr >
        <td align="center">B</td>
        <td align="left">LHR</td>
        <td align="center">2</td>
      </tr>
      <tr >
        <td align="center">C</td>
        <td align="left">SWL</td>
        <td align="left">3</td>
      </tr>
      <tr >
        <td align="center">D</td>
        <td align="left">BWP</td>
        <td align="right">4</td>
      </tr>
    </table>
  </body>
</html>
```

Q7) Explain HTML 5 Media Elements (<audio> & <video>) with appropriate examples

Audio Tag

One of the most important multimedia tags in HTML5 is the <audio>tag. This tag is used to embed audio content into web pages. It supports several audio formats, such as MP3, OGG, and WAV. The tag has several attributes, including src to specify the location of the audio file and controls to display the default audio player controls. Here's an example:

```
<audio src="audiofile.mp3" controls>
  Your browser does not support the audio element.
</audio>
```

In this example, the src attribute specifies the location of the audio file audiofile.mp3, while the controls attribute displays the default audio player controls. If the browser does not support the <audio>tag, it will display the text between the opening and closing tags.

Attributes of Audio tag:

src: specifies the URL of the audio file

controls: displays the default audio player controls

autoplay: automatically starts playing the audio when the page loads

loop: continuously repeats the audio file

preload: specifies whether the audio should be preloaded or not

Video Tag

The <video>tag is used to embed video files in a web page. It supports several video formats, such as MP4, WebM, and OGG. The tag has several attributes, including src to specify the location of the video file and controls to display the default video player controls. Here's an example:

```
<video src="videofile.mp4" controls>
  Your browser does not support the video element.
</video>
```

In this example, the src attribute specifies the location of the video file videofile.mp4, while the Controls attribute displays the default video player controls. If the browser does not support the <video>tag, it will display the text between the opening and closing tags.

Attributes of Video tag:

src: specifies the URL of the video file

controls: displays the default video player controls

autoplay: automatically starts playing the video when the page loads

loop: continuously repeats the video file

preload: specifies whether the video should be preloaded or not

poster: specifies an image to be displayed before the video starts playing

width and height: specifies the dimensions of the video display area in pixels

Source Tag

The <source>tag is used to specify alternative versions of a multimedia file, such as different formats or bitrates, that the browser can choose from based on the user's device and internet connection. The tag has several attributes, including src to specify the location of the multimedia file and type to specify the MIME type of the file. Here's an example:

```
<video controls>
  <source src="videofile.mp4" type="video/mp4">
  <source src="videofile.webm" type="video/webm">
  Your browser does not support the video element.
</video>
```

In this example, the <video>tag includes two <source>tags with different video formats (mp4 and webm). The browser will choose the appropriate format based on the user's device and internet connection. If the browser does not support the <video>tag, it will display the text between the opening and closing tags.

Attributes of Source tag:

src: specifies the URL of the multimedia file

type: specifies the MIME type of the multimedia file

Q8) Explain form input element type attributes (search, range, number, email, color, file, date, time, week, month) with appropriate example

The `<input type="color">` is used for input fields that should contain a color. Depending on browser support, a color picker can show up in the input field.

```
<form>
  <label for="favcolor">Select your favorite color:</label>
  <input type="color" id="favcolor" name="favcolor">
</form>
```

The `<input type="date">` is used for input fields that should contain a date. Depending on browser support, a date picker can show up in the input field.

```
<form>
  <label for="birthday">Birthday:</label>
  <input type="date" id="birthday" name="birthday">
</form>
```

The `<input type="email">` is used for input fields that should contain an e-mail address. Depending on browser support, the e-mail address can be automatically validated when submitted. Some smartphones recognize the email type, and add ".com" to the keyboard to match email input.

```
<form>
  <label for="email">Enter your email:</label>
  <input type="email" id="email" name="email">
</form>
```

The `<input type="file">` defines a file-select field and a "Browse" button for file uploads.

```
<form>
  <label for="myfile">Select a file:</label>
  <input type="file" id="myfile" name="myfile">
</form>
```

The `<input type="month">` allows the user to select a month and year. Depending on browser support, a date picker can show up in the input field.

```
<form>
  <label for="bdaymonth">Birthday (month and year):</label>
  <input type="month" id="bdaymonth" name="bdaymonth">
</form>
```

The `<input type="number">` defines a numeric input field. You can also set restrictions on what numbers are accepted. The following example displays a numeric input field, where you can enter a value from 1 to 5:

```
<form>
  <label for="quantity">Quantity (between 1 and 5):</label>
  <input type="number" id="quantity" name="quantity" min="1" max="5">
</form>
```

The `<input type="range">` defines a control for entering a number whose exact value is not important (like a slider control). Default range is 0 to 100. However, you can set restrictions on what numbers are accepted with the `min`, `max`, and `step` attributes:

```
<form>
  <label for="vol">Volume (between 0 and 50):</label>
  <input type="range" id="vol" name="vol" min="0" max="50">
</form>
```

The `<input type="search">` is used for search fields (a search field behaves like a regular text field).

```
<form>
  <label for="gsearch">Search Google:</label>
  <input type="search" id="gsearch" name="gsearch">
</form>
```

The `<input type="time">` allows the user to select a time (no time zone). Depending on browser support, a time picker can show up in the input field.

```
<form>
  <label for="appt">Select a time:</label>
  <input type="time" id="appt" name="appt">
</form>
```

The `<input type="week">` allows the user to select a week and year. Depending on browser support, a date picker can show up in the input field.

```
<form>
  <label for="week">Select a week:</label>
  <input type="week" id="week" name="week">
</form>
```

Q9) Explain the various levels of style sheet with example for each.

The format of style specification depends on the level of stylesheet.

Inline Style Specification: appears as values of the style attribute of a tag, the general form is as follows:

Style = "Property1 : Value1; Property2 : Value2; Property3 : Value3; Property_n:Value_n;"

It is recommended that last property/value pair be followed by a semicolon.

Eg:

```
<h1 style="font-family: 'Lucida Handwriting'; font-size: 50pt; color: Red;">Web Technology</h1>
```

Document Style Specification: appears as the content of a style element within the

header of a document, general form of the content of a style element is as follows:

```
<style type="text/css">
```

Rule list

```
</style>
```

Each style rule in a rule list has two parts: a selector, which indicates the tag or tags affected

by the rule, and a list of property–value pairs. The list has the same form as the quoted list for inline style sheets, except that it is delimited by braces rather than double quotes. So, the form of a style rule is as follows:

```
Selector { Property1 : Value1; Property2 : Value2; Property3 : Value3; .....  
Property_n:Value_n; }
```

Eg:

```
<style type = "text/css">  
h1 {  
font-family: 'Lucida Handwriting';  
font-size: 50pt;  
color: Red;  
}  
</style>
```

External Style Sheet: have a form similar to that of document style sheets. The external file consists of a list of style rules.

Eg

```
<head>  
<link rel="stylesheet" type="text/css" href="cssfile.css">  
</head>  
Cssfile.css  
P{  
Background-color:blue;  
Color:red;  
}
```

Q10) Explain the various selector forms with examples.

1) Simple Selector Forms:

In case of simple selector, a tag is used. If the properties of the tag are changed, then it reflects at all the places when used in the program. The selector can be any tag. If the new properties for a tag are not mentioned within the rule list, then the browser uses default behaviour of a tag.

Eg:

```
h1 { font-size : 24pt; }
```

```
h2, h3 { font-size : 20pt; }
```

```
body b em { font-size : 14pt; }
```

Only applies to the content of 'em' elements that are descendent of bold element in the body of the document. This is a contextual selector

2) Class Selectors:

Class selectors are used to allow different occurrences of the same tag to use different style specifications.

Eg

```
<head>
```

```
<style type = "text/css">
```

```
p.one { font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }
```

```
p.two { font-family: 'Monotype Corsiva'; font-size: 50pt; color: green; }
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p class = "one">Web Technology</p>
```

```
<p class = "two">Web Technology</p>
```

```
</body>
```

3) Generic Selectors:

Sometimes it is convenient to have a class of Style specification that applies to the content of more than one kind of tag. This is done by using a generic class, which is defined without a tag name in its name. In place of the tag name, you use the name of the generic class, which must begin with a period.

Eg

```
<head>
```



```
<style type = "text/css">
.sale{ font-family: 'Monotype Corsiva'; color: green; }
</style>
</head>
<body>
<p class = "sale">Weekend Sale</p>
<h1 class = "sale">Weekend Sale</h1>
<h6 class = "sale"> Weekend Sale</h6>
</body>
```

4) id Selectors:

An id selector allows the application of a style to one specific element.

Eg:

```
<head>
<style type = "text/css">
#one { font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }
#two { font-family: 'Monotype Corsiva'; font-size: 50pt; color: green; }
</style>
</head>
<body>
<p id = "one">Web Technology</p>
<p id = "two">Web Technology</p>
</body>
```

5) Universal Selectors:

The universal selector, denoted by an asterisk (*), applies its style to all elements in a document.

```
<head>
<style type = "text/css">
*{ font-family: 'Lucida Handwriting'; font-size: 25pt; color: Red; }
```

</style>

</head>

<body>

<p>Web Technology</p>

<p>Web Technology</p>

</body>