

USN

--	--	--	--	--	--	--	--	--	--

Internal Assessment Test 1 – Nov. 2018

Sub:	Web Technologies				Sub Code:	18MCA13	Branch:	MCA
Date:	24.11.2018	Duration:	90 min's	Max Marks:	50	Sem:	1	

PART I

1 Explain request phase of HTTP? [10]

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

Accept: text/plain

Accept: text/html

Can be written as

Accept: text/*

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 HTTP1.1.

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 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 http://www.vtucs.com
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.

OR

2 a) What is web server? Explain its characteristics? [05]

Web servers are programs that provide documents to requesting browsers. Example: Apache.

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

b) Explain DNS with neat diagram [05]

The IP addresses are numbers. Hence, it would be difficult for the users to remember IP address. To solve this problem, text based names were introduced. These are technically known as **domain name system (DNS)**.

These names begin with the names of the host machine, followed by progressively larger enclosing collection of machines, called **domains**. There may be two, three or more domain names. DNS is of the form **hostname.domainName.domainName** .

The steps for conversion from DNS to IP:

- The DNS has to be converted to IP address before destination is reached.
- This conversion is needed because computer understands only numbers.
- The conversion is done with the help of *name server*.
- As soon as domain name is provided, it will be sent across the internet to contact name servers.
- This name server is responsible for converting domain name to IP
- If one of the *name servers* is not able to convert DNS to IP, it contacts other name server.
- This process continues until IP address is generated.
- Once the IP address is generated, the host can be accessed.
- The hostname and all domain names form what is known as FULLY QUALIFIED DOMAIN NAME.

This is as shown below:

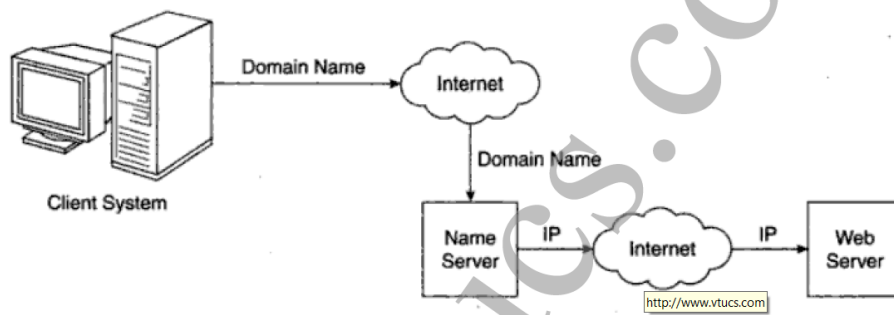


Figure 1.1 Domain name conversion

PART II

3 a) Describe syntactic difference between HTML and XHTML [04]

There are some significant differences between the syntactic rules of HTML (or lack thereof) and those of XHTML. This section describes these differences.

Case sensitivity. In HTML, tag and attribute names are case insensitive, meaning that `<FORM>`, `<form>`, and `<Form>` are equivalent. In XHTML, all tag and attribute names must be in lowercase.

Closing tags. In HTML, closing tags may be omitted if the processing agent (usually a browser) can infer their presence. For example, in HTML, paragraph elements often do not have closing tags. The appearance of another opening paragraph tag is used to infer the closing tag on the previous paragraph. For example:

```

<p>
During Spring, flowers are born. ...
<p>
During Fall, flowers die. ...
  
```

In XHTML, all elements must have closing tags. For elements that do not include content, in which the closing tag appears to serve no purpose, a slash can be included at the end of the opening tag as an abbreviation of the closing tag. For example, the following two lines are equivalent:

```

<input type = "text" name = "address" > </input>
  
```

and

```

<input type = "text" name = "address" />
  
```

Recall that some browsers can be confused if the slash at the end is not preceded by a space.

Quoted attribute values. In HTML, attribute values must be quoted only if there are embedded special characters or whitespace characters. Numeric attribute values are rarely quoted in HTML. In XHTML, all attribute values must be double quoted, regardless of what characters are included in the value.

Explicit attribute values. In HTML, some attribute values are implicit; that is, they need not be explicitly stated. For example, if the `border` attribute appears in a `<table>` tag without a value, it specifies a default width border on the table. For example:

```

<table border>
  
```

This is illegal in XHTML, in which such an attribute is assigned a string of the name of the attribute. For example:

```

<table border = "border">
  
```

Other such attributes are `checked`, `multiple`, and `selected`.

id and name attributes. HTML markup often uses the name attribute for elements. This attribute was deprecated for some elements in HTML 4.0. The id attribute was added to nearly all elements with this same version of HTML. In XHTML, the use of id is encouraged, and the use of name is discouraged. In fact, the name attribute was removed for the anchor and map elements in XHTML 1.1. However, form elements must still use the name attribute because it is used in processing form data.


Element nesting. Although HTML has rules against improper nesting of elements, they are not enforced. Examples of nesting rules are: 1) an anchor element cannot contain another anchor element, and a form element cannot contain another form element; 2) if an element appears inside another element, the closing tag of the inner element must appear before the closing tag of the outer element; 3) block elements cannot be nested in inline elements; 4) text cannot be directly nested in body or form elements; and 5) list elements cannot be directly nested in list elements. In XHTML, these nesting rules are strictly enforced.

All of the XHTML syntactic rules are checked by the W3C validation software.

b) Explain the following tags with example [05]

i)Image ii)Link iii)List

i)Image

- Image can be displayed on the web page using tag.
- When the tag is used, it should also be mentioned which image needs to be displayed. This is done using src attribute.
- Attribute means extra information given to the browser
- Whenever tag is used, alt attribute is also used.
- Alt stands for alert.
- Some very old browsers would not be having the capacity to display the images.
- In this case, whatever is the message given to alt attribute, that would be displayed.
- Another use of alt is  when image display option has been disabled by user. The option is normally disabled when the size of the image is huge and takes time for downloading.

```
<html>
  <head> <title>display image</title> </head>
  <body>
    
  </body>
</html>
```

ii) Link

Hyperlinks are the mechanism which allows the navigation from one page to another.

The term “hyper” means beyond and “link” means connection

Whichever text helps in navigation is called hypertext

Hyperlinks can be created using <a> (anchor tag)

The attribute that should be used for <a> is href

Program: hyper.html

```
<html>
```

```

<head>
  <title> hyperlink </title>
</head>
<a href = "link.html"> CLICK HERE </a>
</html>

```

iii)List

Unordered Lists:

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.

```

<html>
  <head> <title> Unordered List </title> </head>
  <body>
    <h1>Engine Aircraft </h1>
    <ul>
      <li>Cessna Skyhawk</li>
      <li>Beechcraft Bonanza</li>
    </ul>
  </body>
</html>

```

Ordered Lists:

Ordered lists are lists in which the order of items is important. This orderedness 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.

```

<html>
  <head> <title> Unordered List </title> </head>
  <body>
    <h1>Engine Aircraft </h1>
    <ol>
      <li>Cessna Skyhawk</li>
      <li>Beechcraft Bonanza</li>
    </ol>
  </body>
</html>

```

Definition Lists:

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. <html>

```
<head> <title> Definition List </title> </head>
<body>
  <h1> Cessna Skyhawk </h1>
  <dl>
    <dt>152 </dt>
    <dd>Two-place trainer</dd>
    <dt> 172 </dt>
    <dd>Smaller four-place airplane</dd>
    <dt> 120 </dt>
    <dd>Six-place airplane- high performance</dd>
  </dl>
</body>
</html>
```

OR

4 a) Describe standard XHTML document structure with an example [05]

Standard XHTML Document Structure

- Every XHTML document must begin with an xml declaration element that simply identifies the document as being one based on XML. This element includes an attribute that specifies the version number 1.0.
- The xml declaration usually includes a second attribute, encoding, which specifies the encoding used for the document [utf-8].
- Following is the xml declaration element, which should be the first line of every XHTML document:

```
<?xml version = "1.0" encoding = "utf-8"?>
```

- Note that this declaration must begin in the first character position of the document file.
- The xml declaration element is followed immediately by an SGML DOCTYPE command, which specifies the particular SGML document-type definition (DTD) with which the document complies, among other things.
- The following command states that the document in which it is included complies with the XHTML 1.0 Strict standard:

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

- An XHTML document must include the four tags <html>, <head>, <title>, and <body>.
- The <html> tag identifies the root element of the document. So, XHTML documents always have an <html> tag immediately following the DOCTYPE command, and they always end with the closing html tag, </html>.

- The html element includes an attribute, xmlns, that specifies the XHTML namespace, as shown in the following element:

```
<html xmlns = "http://www.w3.org/1999/xhtml">
```
- Although the xmlns attribute's value looks like a URL, it does not specify a document. It is just a name that happens to have the form of a URL.
- An XHTML document consists of two parts, named the head and the body.
- The <head> element contains the head part of the document, which provides information about the document and does not provide the content of the document.
- The body of a document provides the content of the document.
- The content of the title element is displayed by the browser at the top of its display window, usually in the browser window's title bar.

b) Create a web page using XHTML containing two vertical frames, with the right frame three times the width of the left frame. The right frame is further divided horizontally with top frame twice that of the bottom frame. [05]

```
<?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">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Time table</title>
  </head>
  <frameset cols="*,3*">
    <frame src="file1.html" />
    <frameset rows="2*,*">
      <frame src="file2.html" />
      <frame src="file3.html" />
    </frameset>
  </frameset>
</html>
```

PART III

5) Explain the basic structure of HTML5 document with example. [10]

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title of the document</title>
  </head>
```



```
<body>
```

The content of the document.....

```
</body>
```

```
</html>
```

The <! DOCTYPE> declaration must be the very first thing in your HTML document, before the <html> tag.

The <! DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

In HTML 4.01, the <! DOCTYPE> declaration refers to a DTD, because HTML 4.01 was based on SGML. The DTD specifies the rules for the markup language, so that the browsers render the content correctly.

HTML5 is not based on SGML, and therefore does not require a reference to a DTD.

OR

6 Explain all new features of HTML5. [10]

HTML 5 adds a lot of new features to the HTML specification, and it is easy to implement. You use the HTML 5 doctype, which is simple and streamlined:

```
<!doctype html>
```

Yes, that's it. Just two words "doctype" and "html." It can be this simple because HTML 5 is no longer [part of SGML](#) but is instead a [markup language](#) all on its own.

The character set for HTML 5 is streamlined as well. It uses UTF-8, and you define it with just one meta tag:

```
<meta charset="UTF-8">
```

HTML 5 New Structure

HTML 5 recognizes that webpages have a structure, just like books and [other XML documents](#) have a structure. In general, webpages have navigation, body content, sidebar content, headers, footers, and other features. HTML 5 has tags to support those elements of the page. They are:

- **<section>** defines sections of pages.

- **<header>** defines the header of a page.
- **<footer>** defines the footer of a page.
- **<nav>** defines the navigation on a page.
- **<article>** defines the article or primary content on a page.
- **<aside>** defines extra content like a sidebar on a page.
- **<figure>** defines images that annotate an article.

HTML 5 New Inline Elements

The new inline elements define some basic concepts and keep them semantically marked up:

- **<mark>** indicates content that is marked in some fashion.
- **<time>** indicates content that is a time or date.
- **<meter>** indicates content that is a fraction of a known range such as disk usage.
- **<progress>** indicates the progress of a task towards completion.

HTML 5 New Dynamic Pages Support

HTML 5 was developed to help web application developers, so there are a lot of new features that make it easy to create dynamic HTML pages:

- **Context menus** – HTML 5 supports the creation and use of context menus within webpages and applications.
- **href** is not required on a tag. This allows you to use a tag with scripts and in web applications without needing a place to send that anchor.
- **async attribute** – This is added to the script tag to tell the browser that the script should be loaded asynchronously so that it doesn't slow down the load and display of the rest of the page.
- **<details>** – This provides details about an element. This would be like tooltips in non-web applications.
- **<datagrid>** creates a table that is built from a database or other dynamic source.
- **<menu>** is an old tag brought back and given new life allowing you to create a menu system on your webpages.
- **<command>** defines actions that should happen when a dynamic element is activated.

HTML 5 New Form Types

HTML 5 supports all the standard form input types, but it adds a few more:

- datetime
- datetime-local
- date

- month
- week
- time
- number
- range
- email
- url

HTML 5 New Elements

There are a few exciting new elements in HTML 5:

- **<canvas>** – This element gives you a drawing space in JavaScript on your webpages. It can add images or graphs to tooltips or create dynamic graphs on your webpages, built on the fly.
- **<video>** – Add video to your webpages with this simple tag.
- **<audio>** – Add sound to your [webpages](#) with this simple tag.

HTML 5 Removes Some Elements

Some elements in HTML 4 are no longer be supported by HTML 5. Most are already deprecated and shouldn't be surprising. They are:

- acronym
- applet
- basefont
- big
- center
- dir
- font
- frame
- frameset
- isindex
- noframes
- noscript
- s
- strike
- tt
- u

PART IV

7) Create a XHTML document that describes the form for Registration. [10]

Registration form

First Name

Last Name

Nick Name

e-mail

Password

Date of Birth

Gender Male Female Others

Mobile

Address

```

<?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">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
  </head>
  <body>
    <table>
      <caption> Regsitration Form </caption>
      <form>
        <tr>
          <td><label> First Name :</label></td>
          <td><input type="text" name="fname"/> </td>
        </tr>
        <tr>
          <td><label> Last Name :</label></td>
          <td><input type="text" name="lname"/></td>
        </tr>
        <tr>
          <td><label> Nick Name :</label></td>
          <td><input type="text" name="nname"/></td>
        </tr>
      </form>
    </table>
  </body>
</html>

```

```

        <td><label> Email :</label></td>
        <td><input type="text" name="email"/></td>
    </tr>
    <tr>
        <td><label> Password :</label></td>
        <td><input type="password" name="pass"/></td>
    </tr>
    <tr>
        <td><label> Date of Birth :</label></td>
        <td><input type="text" name="dob"/></td>
    </tr>
    <tr>
        <td><label> Gender: </label></td>
        <td><input type="radio" name="Male" value="0"/> Male
        <input type="radio" name="Female" value="1"/> Feamle
        <input type="radio" name="Female" value="2"/> Others </td>
    </tr>
    <tr>
        <td><label> Mobile :</label></td>
        <td><input type="text" name="mobile"/></td>
    </tr>
    <tr>
        <td><label> Address :</label></td>
        <td><textarea rows="3" cols="10"></textarea></td>
    </tr>
    <tr>
        <td><input type="submit" value="submit"/></td>
        <td><input type="reset" value="reset"/></td>
    </tr>
</form>
</table>
</body>
</html>

```

OR

8 Write a XHTML program to create following table [10].

Time Table					
	Mon	Tue	Wed	Thu	Fri
Hours	Science	Maths	Science	Maths	Arts
	Social	History	English	Social	Sports
	Lunch				
	Science	Maths	Science	Maths	Project
	Social	History	English	Social	

```
<?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">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
```

```
<title>Time table</title>
```

```
</head>
```

```
<body>
```

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

```
<tr align="center">
```

```
<th colspan="6">Time Table </th>
```

```
</tr>
```

```
<tr>
```

```
<th rowspan="6">Hours</th>
```

```
<th>Mon</th>
```

```
<th>Tue</th>
```

```
<th>Wed</th>
```

```
<th>Thu</th>
```

```
<th>Fri</th>
```

```
</tr>
```

```
<tr>
```

```
<td>Science</td>
```

```
<td>Maths</td>
```

```
<td>Science</td>
```

```
<td>Maths</td>
```

```
<td>Arts</td>
```

```
</tr>
```

```
<tr>
```

```
<td>Social</td>
```

```
<td>History</td>
```

```
<td>English</td>
```

```
<td>Social</td>
```

```
<td>Sports</td>
```

```
</tr>
```

```
<tr align="center">
```

```
<td colspan="5">Lunch</td>
```

```
</tr>
```

```
<tr>
```

```
<td>Science</td>
```

```
<td>Maths</td>
```

```
<td>Science</td>
```

```
<td>Maths</td>
```

```
        <td rowspan="2">Project</td>
    </tr>
    <tr>
        <td>Social</td>
        <td>History</td>
        <td>English</td>
        <td>Social</td>
    </tr>
</table>
</body>
</html>
```

PART V

9) Explain HTML 5 Media Elements (<audio> & <video>) and form element attributes (search, range, number, email) with appropriate example [10]

audio

```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
  Your browser does not support the audio tag.
</audio>
```

The <audio> tag defines sound, such as music or other audio streams.

Currently, there are 3 supported file formats for the <audio> element: MP3, WAV, and OGG

Video

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

The <video> tag specifies video, such as a movie clip or other video streams.

Currently, there are 3 supported video formats for the <video> element: MP4, WebM, and Ogg

- MP4 = MPEG 4 files with H264 video codec and AAC audio codec
- WebM = WebM files with VP8 video codec and Vorbis audio codec
- Ogg = Ogg files with Theora video codec and Vorbis audio codec

OR

10) Explain HTML 5 semantic elements (<nav>, <section>, <article>,<aside>, <header>, <footer>) with example. [10]

1) Nav

The <nav> tag is a new element in HTML5. It is used to define a block of navigation links, either within the current document or to other documents. Examples of navigation blocks are menus, tables of contents, and indexes.

One HTML document may contain several <nav> tags, for example, one for site navigation and one for intra-page navigation.

Note that not all links in the HTML document can be placed inside the <nav> element; it can only include major navigation blocks. For example, the <nav> tag is not placed in the <footer> tag for defining links in the footer of the website.

Example:

```
<nav>
  <a href="/learn-html.html">HTML</a> | <a href="/learn-css.html">CSS</a> | <a
href="/learn-javascript.html">JavaScript</a> | <a href="/learn-php.html">PHP</a> |
</nav>
```

2) Section

The HTML <section> tag specifies a section in a document.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <title>HTML Section Tag</title>
```

```
</head>
```

```
<body>
```

```
  <section>
```

```
    <h1>Java</h1>
```



```
<h3>Inheritance</h3>
<p>Inheritance defines the relationship between superclass and subclass.</p>
</section>
</body>
```

```
</html>
```

3) Article

The `<article>` tag specifies independent, self-contained content.

An article should make sense on its own and it should be possible to distribute it independently from the rest of the site.

Potential sources for the `<article>` element:

- Forum post
- Blog post
- News story
- Comment

```
<article>
  <h1>Google Chrome</h1>
  <p>Google Chrome is a free, open-source web browser developed by Google,
  released in 2008.</p>
</article>
```

4) Aside

The `<aside>` tag defines some content aside from the content it is placed in.

The aside content should be related to the surrounding content.

```
<p>My family and I visited The Epcot center this summer.</p>
```

```
<aside>
  <h4>Epcot Center</h4>
  <p>The Epcot Center is a theme park in Disney World, Florida.</p>
</aside>
```

5) Header

The `<header>` element represents a container for introductory content or a set of navigational links.

A <header> element typically contains:

one or more heading elements (<h1> - <h6>)

logo or icon

authorship information

You can have several <header> elements in one document.

Note: A <header> tag cannot be placed within a <footer>, <address> or another <header> element.

```
<article>
```

```
  <header>
```

```
    <h1>Most important heading here</h1>
```

```
    <h3>Less important heading here</h3>
```

```
    <p>Some additional information here</p>
```

```
  </header>
```

```
  <p>Lorem Ipsum dolor set amet....</p>
```

```
</article>
```

6) Footer

The <footer> tag defines a footer for a document or section.

A <footer> element should contain information about its containing element.

A <footer> element typically contains:

authorship information

copyright information

contact information

sitemap

back to top links

related documents

You can have several <footer> elements in one document.