CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | |

Internal Assessment Test 2 – November 2016

Sub:	**Programming The WEB**					Code:	10CS73
Date:	03/ 11/ 16	Duration:	90 Mins	Max Marks:	50	Sem:	VII A,B
						Branch:	CSE/ISE

Note: **Answer to the point. Sketch figures wherever necessary.**

Answer any 5 full questions.

		Marks	CO	RBT
1	**a)** Declare a XML document containing data of 3 Employee's (emp_id, emp_name, emp_designation, emp_age, emp_phone, emp_address) display this XML using CSS , with following rules :emp_id in font size 28pt and color red, emp_name and emp_designation in font size 18 pt and color blue ,emp_age , emp_phone and emp_address in font size 15pt and color black	**6M**	CO3	L3
	b) Explain the following tags with example: i) <div> ii) <span>	**4M**	CO1	L3
2	**a)** Write an XHTML document that has six short paragraphs of text. Define three different paragraph styles p1, p2, and p3. The p1 style must use left and right margins of 20 pixels, a background color of yellow and a fore ground color of blue. The p2 style must use font size of 18 points, font name 'Arial' and font style in italic form. The p3 style must use a text indent of 1 cm, a background color of green, and foreground color of white. The 1st and 4th paragraph must use p1,the second and 5th must use p2, and 3rd and 6th must use p3?	**5M**	CO1	L3
	b) What is XML Schema? Explain defining schema with example?	**5M**	CO3	L2
3	Explain the different selector forms provided in CSS. Illustrate the use of each with suitable example	**10M**	CO1	L2
4	**a)**What is a query string? How is it transmitted to the server with the GET and POST methods?	**5M**	CO4	L2
	b) Explain the string functions in Perl with example	**5M**	CO4	L1
5	What is DTD? What is the difference between External and Internal DTDs? Write the syntax and example for declaring elements , Attributes and Entities ,CDATA,PCDATA?	**10M**	CO3	L2
6	**a)**Illustrate with the help of a diagram the XSLT processing**?**	**4M**	CO3	L1
	b) Describe with example various types of variables in Perl.	**6M**	CO4	L2
7	How the files are handled in Perl? Write a Perl program to copy contents of one file to another file.	**10M**	CO4	L3

Course Outcomes		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1:	Design Multimedia and web pages that include the CSS and standard textual components needed on web pages	2	2	2	1	3	1	1	0	1	2	1	1
CO2:	Create interactive websites using javascript and DHTML	2	1	2	1	3	1	0	0	0	2	1	1
CO3:	Write well-formed XML document for a given schema using DTD, XSLT	1	1	1	1	3	0	0	0	0	0	0	0
CO4:	Write simple CGI programs using PERL	2	2	1	1	2	0	1	0	0	1	0	0
CO5:	Implement a simple web application using PHP language	1	2	2	1	3	1	1	0	0	1	2	2
CO6:	Implement a simple web application using Ruby On Rails.	2	2	1	1	2	1	2	0	1	1	1	1

Cognitive level	KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PO1 - *Engineering knowledge*; PO2 - *Problem analysis*; PO3 - *Design/development of solutions*; PO4 - *Conduct investigations of complex problems*; PO5 - *Modern tool usage*; PO6 - *The Engineer and society*; PO7- *Environment and sustainability*; PO8 – *Ethics*; PO9 - *Individual and team work*; PO10 - *Communication*; PO11 - *Project management and finance*; PO12 - *Life-long learning*

Scheme and Solution- IAT2-Nov 2016
Programming the Web (10CS73)

Q.1a) Declare a XML document containing data of 3 Employee's (emp_id, emp_name, emp_designation, emp_age, emp_phone, emp_address) display this XML using CSS , with following rules :emp_id in font size 28pt and color red, emp_name and emp_designation in font size 18 pt and color blue ,emp_age , emp_phone and emp_address in font size 15pt and color black**. (6M)**

Each employee 1M*3= 3M

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css href=employee.css"?>
<employee>
        <emp_id>11    </emp_id>
        <emp_name>  ABC  </emp_name>
       <emp_designation> Manager   </emp_designation>
        <emp_age>   50  </emp_age>
        <emp_phone> 123456 </emp_phone>
        <emp_address> Bangalore  </emp_address>
    </employee>
    <employee>

    </employee>
    <employee>

    </employee>
```

<u>**employee.css**</u>

```
employee{display:block;margin-top:15px;}
emp_id {display:block;     color:red;font-size:28pt} -1M
emp_name,emp_designation { display:block;     color:blue;font-size:18pt;}-1M
emp_age , emp_phone  and emp_address { display:block;     color:black;font-size:15pt;}-1M
```

Q.1b)Explain the following tags with example: i) <div>   ii) <span>   **(4M)**

Each explanation(1M) + Example(1M) =2*2=4M

i)The <div> tag defines a division or a section in an HTML document. The <div> tag is used to group block-elements to format them with CSS.

```
<div style="color:#0000FF"> -

  <h3>This is a heading in a div element</h3>

  <p>This is some text in a div element.</p>
```

```
</div>
```

ii) <span>

The HTML <span> tag is used for grouping and applying styles to inline elements.There is a difference between the span tag and the div tag. The span tag is used with inline elements whilst the div tag is used with block-level content.

```
<!DOCTYPE html>

<html>

<head>

<title>HTML span Tag</title>

</head>

<body>

<p>This is a paragraph <span style="color:#FF0000;">

This is a paragraph</span>

This is a paragraph</p>

<p><span style="color:#8866ff;">

This is another paragraph</span></p>

</body>

</html>
```

Q.2a) Write an XHTML document that has six short paragraphs of text. Define three different paragraph styles p1, p2, and p3. The p1 style must use left and right margins of 20 pixels, a background color of yellow and a fore ground color of blue. The p2 style must use font size of 18 points, font name 'Arial' and font style in italic form. The p3 style must use a text indent of 1 cm, a background color of green, and foreground color of white. The 1st and 4th paragraph must use p1,the second and 5th must use p2, and 3rd and 6th must use p3? **(5M)**

Each paragraph style 1M*3=3M
Paragraph writing 2M

```
<?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">
```

```
<!-- fonts.html
     An example to illustrate font properties
     -->
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head> <title> Font properties </title>
    <style type = "text/css">
      p.p1 {
              margin-left:20px;
             margin-right:20px;
             background-color:yellow;
              color:blue;
            }
      p.p2 {
          font-size: 18pt;
           font-family:'Arial';
          font-style:italic;
}
p.p3 {
         text-indent:1cm;
         background-color:green;
          color:white;
}
  </style>
  </head>
  <body>
```

```
   <p class = “p1">      If a job is worth doing, it's worth doing right.    </p>

   <p class = "p2"> Two wrongs don't make a right, but they certainly  can get you in a lot of trouble.
</p>

 <p class = "p3">   Hello    </p>

 <p class = "p1">  Hello World     </p>

<p class = "p2"> Hello hi    </p>

<p class = "p3">  Web Programming    </p>

 </body> </html>
```

Q.2 b)What is XML Schema? Explain defining schema with example? **(5M)**

Definition -1M

An *XML Schema* describes the structure of an *XML* document, just like a DTD. An *XML* document with correct syntax is called "Well Formed". An *XML* document validated against an *XML Schema* is both "Well Formed" and "Valid"

Schema Definition-2M

```
<?xml version="1.0"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

Any Example -2M

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Q.3 Explain the different selector forms provided in CSS. Illustrate the use of each with suitable example **(10M)**

Selector can have variety of forms like:

1.Simple selector form
2.Class selector
3.Generic selector
4.Id selector
5.Universal selector
6.Pseudo classes

Simple selector form

Simple selector form is a list of style rules, as in the content of a <style> tag for document-level style sheets. The selector is a tag name or a list of tag names, separated by commas. Consider the following examples, in which the property is font-size and the property value is a number of points :

```
h1, h3 { font-size: 24pt ;}
   h2 { font-size: 20pt ;}
```

Class Selectors

Used to allow different occurrences of the same tag to use different style specifications. A style class has a name, which is attached to the tag's name with a period.

p.narrow {*property-value list*} p.wide {*property-value list*}

The class you want on a particular occurrence of a tag is specified with the class attribute of the tag.

For example,

<p class = "narrow">

Once upon a time there lived a king in the place called Ayodhya.
</p>

...

<p class = "wide">

Once upon a time there lived a king in the place called Ayodhya.
</p>

Generic Selectors

A generic class can be defined if you want a style to apply to more than one kind of tag. A generic class must be named, and the name must begin with a period without a tag name in its name. For Example:

```
.really-big { … }
```

Use it as if it were a normal style class

```
<h1 class = "really-big"> This Tuesday is a holiday </h1>...
<p class = "really-big"> … </p>
```

```
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
<title> Absolute positioning </title>
<style type = "text/css">
.regtext {font-family: Times; font-size: 14pt; width: 600px}
.abstext {position: absolute; top: 25px; left: 50px; font-family: Times; font-size: 24pt; font-
style: italic; letter-spacing: 1em; color: rgb(102,102,102); width: 500px}
</style>
</head>
 <body>
<p class = "regtext">
```

Id Selectors
An id selector allow the application of a style to one specific element. The general form of an id selector is as follows :
#specific-id {property-value list}

Example:

#section14 {font-size: 20} specifies a font size of 20 points to the element

<h2 id =“section14”> Alice in wonderland</h2>

Universal selector
The universal selector, denoted by an asterisk(*), which applies style to all elements in the document. For example:

  □ **{color: red;}**

makes all elements in the document red.

Pseudo Classes
Pseudo classes are styles that apply when something happens, rather than because the target element simply exists. Names of pseudo classes begin with colons hover classes apply when the mouse cursor is over the element focus classes apply when an element has focus i.e. the mouse

cursor is over the element and the left mouse button is clicked. These two pseudo classes are supported by FX2 but IE7 supports only hover.

<html xmlns = "http://www.w3.org/1999/xhtml"> <head> <title> Checkboxes </title>

```
  <style type = "text/css">
    input:hover {color: red;}
    input:focus {color: green;}
  </style> </head> <body>
  <form action = ""> <p>
      Your name:
      <input type = "text" />
    </p> </form>
 </body>
</html>
```

Q.4a)What is a query string? How is it transmitted to the server with the GET and POST methods? **(5M)**

Definition query string 1M+ Explanation for GET method 2M and POST method 2M

In World Wide Web, a **query string** is the part of a Uniform Resource Locator (URL) that contains data to be passed to web applications such as CGI programs.

The Mozilla URL location bar showing an URL with the query string title=Main_page&action=raw

When a web page is requested via the Hypertext Transfer Protocol, the server locates a file in its file system based on the requested URL. This file may be a regular file or a program. In the second case, the server may (depending on its configuration) run the program, sending its output as the required page. The query string is a part of the URL which is passed to the program. Its use permits data to be passed from the HTTP client (often a web browser) to the program which generates the web page

Structure

A typical URL containing a query string is as follows:

http://server/path/program?query_string

When a server receives a request for such a page, it runs a program (if configured to do so), passing the query_string unchanged to the program. The question mark is used as a separator and is not part of the query string.

A link in a web page may have a URL that contains a query string. However, the main use of query strings is to contain the content of an HTML form, also known as web form. In particular, when a form containing the fields field$_1$, field$_2$, field$_3$ is submitted, the content of the fields is encoded as a query string as follows:

field$_1$=value$_1$&field$_2$=value$_2$&field$_3$=value$_3$...

- The query string is composed of a series of field-value pairs.
- The field-value pairs are each separated by an equal sign.
- The series of pairs is separated by the ampersand, '&' or semicolon, ';'.

For each field of the form, the query string contains a pair field=value. Web forms may include fields that are not visible to the user; these fields are included in the query string when the form is submitted

This convention is a W3C recommendation. W3C recommends that all web servers support semicolon separators in the place of ampersand separators.

Technically, the form content is only encoded as a query string when the form submission method is GET. The same encoding is used by default when the submission method is POST, but the result is not sent as a query string, that is, is not added to the action URL of the form. Rather, the string is sent as the body of the request.

Q.4 a) Explain the string functions in Perl with example **(5M)**

Any five functions with example-1M*5=5M

String Functions

Functions and operators are closely related in Perl

chomp( ) – removes any terminating newline characters from input string

```
$string1 = "This is test\n";
$retval  = chomp( $string1 );

print " Choped String is : $string1\n";
print " Number of characters removed : $retval\n";
```

```
out put: Choped String is : This is test
Number of characters removed : 1
```

lc( ) –returns input string letters converted to lowercase

```
#!/usr/bin/perl

$orig_string = "This is Test and CAPITAL";
$changed_string = lc( $orig_string );

print "Changed String is : $changed_string\n";
```

out put- `Changed String is : this is test and capital`

uc( ) returns input string letters converted to uppercase

```
$string = 'the cat sat on the mat.';
$u_string = uc($string);

print "First String |$string|\n";
print "Second String |$u_string|\n";
```

***output**: First String |the cat sat on the mat.|*
Second String |THE CAT SAT ON THE MAT.|

join( ) –returns a string constructed by catenating the string of the second and subsequent strings together ,with the parameter character inserted between them.

```
$string = join( "-", "one", "two", "three" );
print"Joined String is $string\n";

$string = join( "", "one", "two", "three" );
print"Joined String is $string\n";
```

output: Joined String is one-two-three
Joined String is onetwothree

length( ) returns the number of characters in its parameter string

```
$orig_string = "This is Test and CAPITAL";
$string_len =  length( $orig_string );

print "Length of  String is : $string_len\n";
```

output: Length of String is : 24

Q. 5 What is DTD? What is the difference between External and Internal DTDs? Write the syntax and example for declaring elements , Attributes and Entities ,CDATA,PCDATA?

Document Type Definitions: 1M
A DTD is a set of structural rules called declarations. These rules specify a set of elements, along with how and where they can appear in a document

Purpose: provide a standard form for a collection of XML documents an define a markup language for them.

Internal and External DTD's: 3M
• Internal DTDs
```
<!DOCTYPE planes [
<!– The DTD for planes -->
]>
```

• External DTDs

```
<!DOCTYPE XML_doc_root_name SYSTEM
"DTD_file_name">
```

For example

```
<!DOCTYPE planes_for_sale SYSTEM
"planes.dtd">
<?xml version = "1.0" encoding = "utf-8"?>
<!-- planes.xml - A document that lists ads for used airplanes -->
<!DOCTYPE planes_for_sale SYSTEM "planes.dtd">
<planes_for_sale>
<ad>
<year> 1977 </year>
<make> &c; </make>
<model> Skyhawk </model>
<color> Light blue and white </color>
<description> New paint, nearly new interior,
685 hours SMOH, full IFR King avionics </description>
<price> 23,495 </price>
<seller phone = "555-222-3333"> Skyway Aircraft </seller>
<location>
<city> Rapid City, </city>
<state> South Dakota </state>
</location>
</ad>
</planes_for_sale>
```

1M * 5(Element,attributes,entity,CDATA,PCDATA)=5M
1.Declaring Elements:
An element declaration specifies the name of an element and its structure
• If the element is a leaf node of the document tree, its structure is in terms of characters

Child elements can have modifiers,
+ => One or more occurrences
* => Zero or more occurrences
? => Zero or one occurrences
<!ELEMENT person (parent+, age, spouse?, sibling*)>

2.PCDATA
Leaf nodes specify data types of content of their parent nodes which are elements
PCDATA (parsable character data)
Example of a leaf declaration:
<!ELEMENT name (#PCDATA)>
<!ELEMENT age(#PCDATA)>

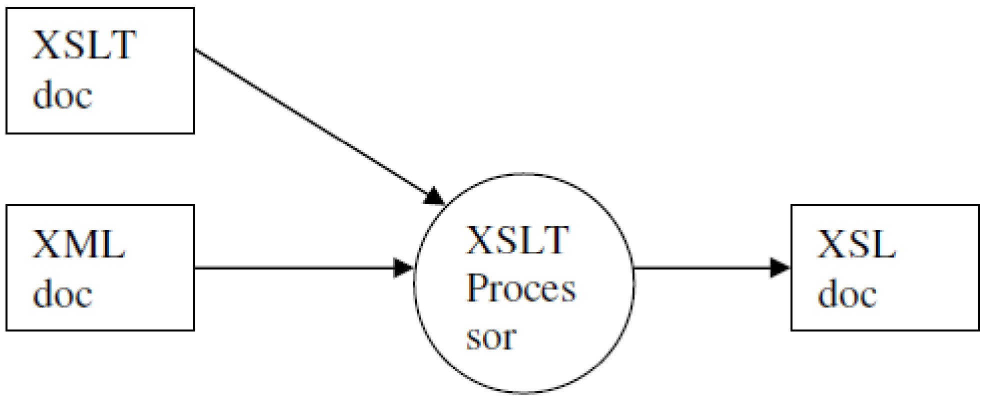
3.Declating Attributes-
Attributes are declared separately from the element declarations
<!ATTLIST element_name attribute_name attribute_type [default _value]>

Value - value ,which is used if none is specified
#Fixed value - value ,which every element have and can't be changed
# Required - no default value is given ,every instance must specify a value

#Implied - no default value is given ,the value may or may not be specified
Example :
<!ATTLIST car doors CDATA "4">
<!ATTLIST car engine_type CDATA #REQUIRED>
<!ATTLIST car make CDATA #FIXED "Ford">
<!ATTLIST car price CDATA #IMPLIED>
<car doors = "2" engine_type = "V8">
...
</car>
4. **Declaring Entities :** A general entity can be referenced anywhere in the content of an XML document
<!ENTITY [%] entity_name "entity_value">
% when present it specifies declaration parameter entity
Example :
<!ENTITY jfk "John Fitzgerald Kennedy">
A reference above declared entity: &jfk;

5.CDATA
The term **CDATA**, meaning character data, is used for distinct, but related, purposes in the markup languages SGML and **XML**. The term indicates that a certain portion of the document is general character data, rather than non-character data or character data with a more specific, limited structure.
<!ATTLIST seller phone CDATA #REQUIRED>
<!ATTLIST seller email CDATA #IMPLIED

Q.5a) Illustrate with the help of a diagram the XSLT processing**?**
Diagram 2M + explanation 3M
XSLT is a functional-style programming language.
XSLT includes functions, parameters, names to which values can be bound , selection constructs and conditional expressions for multiple selection.
The syntactic structure of XSLT is XML, so each stmt is specified with an element. XSLT document is a program to be executed and XML doc is input data to that program.
Parts of XML document are selected possibly modified and merged with other parts of XSLT document to form new doc called XSL document which could be again input to an XSLT processor. o/p can be stored for future use by applications like browsers.
XSLT documents consists of templates which use XPath to describe element/attribute patterns in the input XML document
Each template has XSLT code which is executed when a match to the template is found in the XML document. So each template describes a function, which is executed whenever the XSLT processor finds a match to the templates pattern.
XSLT processor sequentially examines the i/p XML documents searching for parts that match one of the templates in XSLT doc.
XML doc consists of nodes where nodes are elements, attributes, comments, text and processing instructions.
If the template matches an element, the element is not processed until the closing tag is found.
When a template matches an element, the child elements of that element may or may not be processed.
XSLT has two models of processing XML data
1. *Template –Driven model* _ data consists of multiple instances of regular data collections files of records.
2. *Data – Driven model* _ irregular and recursive data



Q.5b) Describe with example various types of variables in Perl**. (5M)**

PERL has three categories of variables 1.Scalar, arrays and Hashes
Scalar type is specified by preceding the name with a $.
1.Scalars: 1M
Scalars are variables that can store either numbers, strings, or references
Numbers are stored in double format; integers are rarely used
Numeric literals have the same form as in other common languages

Perl has two kinds of string literals, those delimited by double quotes and those delimited by single quotes

Example: $age=10;
Sname=”ABC”;
$number=12.9;

2.Arrays: 2M

An array stores an ordered list of values. While a scalar variable can only store one value, an array can store many. Perl array names are prefixed with an @-sign. Here is an example:

```
my @colors = ("red","green","blue");

print "$colors[0]\n";          # prints "red"

print "$colors[1]\n";          # prints "green"

print "$colors[2]\n";          # prints "blue"
```

A much easier way to do this is to use a foreach loop:

```
my @colors = ("red","green","blue");

foreach my $i (@colors) {

   print "$i\n";

}
```

3.Hashes: 2M

A hash is a special kind of array — an associative array, or paired list of elements. Each pair consists of a string key and a data value.

Perl hash names are prefixed with a percent sign (%). Here's how they're defined:

```
Hash Name       key  value

my %colors = ( "red", "#ff0000",
               "green", "#00ff00",
               "blue", "#0000ff",
               "black", "#000000",
               "white", "#ffffff" );
```

This particular example creates a hash named %colors which stores the RGB HEX values for the named colors. The color names are the hash keys; the hex codes are the hash values.

Q.7 How the files are handled in Perl ? Write a perl program to copy contents of one file to another file **(10M)**

Maybe you have a guestbook program and want to keep a log of the names and email addresses of visitors, or a page counter that must update a counter file, or a program that scans a flat-file database and draws info from it to generate a page. You can do this by reading and writing data files (often called file I/O).

File permissions can be changed with following command -1M

```
chmod 666 myfile.dat
```

Opening Files (File Modes) – 2M

Reading and writing files is done by opening a file and associating it with a filehandle. This is done with the statement:

```
open(filehandle,filename);
open(INF,"<out.txt")  opens out.txt for reading
open(INF,">out.txt")  opens out.txt for writing(overwriting)
open(INF,">>out.txt")  opens out.txt for appending
open(INF,"+<out.txt")  opens out.txt for reading and writing
```

The INF is the file handle.

To read data from the file 1M

Here DATA is the file handle which will be used to read the file. Here is the example which will open a file and will print its content over the screen

```
#!/usr/bin/perl

open(DATA, "<file.txt") or die "Couldn't open file file.txt, $!";

while(<DATA>){
  print "$_";
}
```

Write to file 1M

The <FILEHANDLE> operator in a list context, it returns a list of lines from the specified filehandle. For example, to import all the lines from a file into an array –

```
#!/usr/bin/perl

open(DATA, "<file.txt") or die "Couldn't open file file.txt, $!";

while(<DATA>){
  print "$_";
}
```

Closing the file -1M

To close a filehandle, and therefore disassociate the filehandle from the corresponding file, you use the **close** function. This flushes the filehandle's buffers and closes the system's file descriptor.

```
close(DATA) || die "Couldn't close file properly";
```

Program to copy contents of one file to another file-4M

```
#!/usr/bin/perl

# Open file to read
open(DATA1, "<file1.txt");

# Open new file to write
open(DATA2, ">file2.txt");

# Copy data from one file to another.
while(<DATA1>)
{
  print DATA2  $_;
}
close( DATA1 );
close( DATA2 );
```