USN										

Internal Assessment Test 3 – Nov. 2017

Sub:	Programming the Web					Sub Code:	10CS73	Branch:	CSE	
Date:	20-11-2017	Duration:	90 min's	Max Marks:	50	Sem / Sec:	7 – A, B , C		OBE	

Answer any 5 questions

		Marks	CO	RBT
1.a)	What is hypertext? Explain HTTP phases. Mention various methods and status codes of HTTP	7	CO1	L2
1.b)	Write an XHTML document to describe an ordered list of four states. Each element of the list must have an unordered list of atleast two cities in the state	3	CO1	L3
2.a)	Explain syntactic differences between HTML and XHTML.	5	CO1	L3
2. b)	Write an XHTML document to display an image and three buttons. The buttons should be labeled 1,2 and 3. When pressed, each button should change the content of the image to that of a different image	5	CO2	L3
3.a)	Explain all selector forms with example for each.	5	CO3	L2
3.b)	Create XHTML document that has 2 frames. The left frame displays content.html and right frame displays cars.html where the second frame is a target of link from the first frame. Note : contents.html is a list of links to the cars description.	5	CO3	L3
4.a)	Explain the purposes of XML processors	5	CO5	L2
4.b)	Create a XHTML document with PHP to display the number, square root, square, cube and quadruple using sqrt and pow functions.	5	CO3	L3

USN										

Internal Assessment Test 3 – Nov. 2017

Sub:	Programming the Web					Sub Code:	10CS73	Branch:	CSE	
Date:	20-11-2017	Duration:	90 min's	Max Marks:	50	Sem / Sec:	7 – A, B , C		OBE	

Answer any 5 questions

		Marks	CO	RBT
1.a)	What is hypertext? Explain HTTP phases. Mention various methods and status codes of HTTP	7	CO1	L2
b)	Write an XHTML document to describe an ordered list of four states. Each element of the list must have an unordered list of atleast two cities in the state	3	CO1	L3
2.a)	Explain syntactic differences between HTML and XHTML.	5	CO1	L3
2. b)	Write an XHTML document to display an image and three buttons. The buttons should be labeled 1,2 and 3. When pressed, each button should change the content of the image to that of a different image	5	CO2	L3
3.a)	Explain all selector forms with example for each.	5	CO3	L2
3.b)	Create XHTML document that has 2 frames. The left frame displays content.html and right frame displays cars.html where the second frame is a target of link from the first frame. Note : contents.html is a list of links to the cars description.	5	CO3	L3
4.a)	Explain the purposes of XML processors	5	CO5	L2
4.b)	Create a XHTML document with PHP to display the number, square root, square, cube and quadruple using sqrt and pow functions.	5	CO3	L3

5.a)	Describe the logical internal structure of an array in PHP	5	CO2	L2
5.b)	Create a DTD for a catalog of cars, where each car has the child elements make, model, year, color, engine, number-of-doors, transmission type and accessories. The engine element has the child elements, number of cylinders and fuel system. The accessories element has the attributes radio, air-conditioning, power-windows and power-brakes, each of which is required and has the possible values yes and no. Entities must be declared for the names of popular car make.	5	CO5	L3
6.a)	What is Document Type Definition? Describe the approach to declare elements, entities and attributes.	5	CO5	L2
6.b)	Explain any 5 string functions in PHP	5	CO4	L2
7. a)	Write a CGI-PERL program to use a cookie to remember the day of the last login, from a user and display it when executed	5	CO4	L3
7.b)	Develop a complete XHTML document with proper headings, a table with four rows and three columns, a form with two labels, two text box, three check box, three radio buttons, a submit and a reset button.	5	CO1	L3
8.	With example, explain simple input and output functions in ruby. Explain code blocks and iterators.	10	CO4	L2

5.a)	Describe the logical internal structure of an array in PHP	5	CO2	L2
5.b)	Create a DTD for a catalog of cars, where each car has the child elements make, model, year, color, engine, number-of-doors, transmission type and accessories. The engine element has the child elements, number of cylinders and fuel system. The accessories element has the attributes radio, air-conditioning, power-windows and power-brakes, each of which is required and has the possible values yes and no. Entities must be declared for the names of popular car make.	5	CO5	L3
6.a)	What is Document Type Definition? Describe the approach to declare elements, entities and attributes.	5	CO5	L2
6.b)	Explain any 5 string functions in PHP	5	CO4	L2
7. a)	Write a CGI-PERL program to use a cookie to remember the day of the last login, from a user and display it when executed	5	CO4	L3
7.b)	Develop a complete XHTML document with proper headings, a table with four rows and three columns, a form with two labels, two text box, three check box, three radio buttons, a submit and a reset button.	5	CO1	L3
8.	With example, explain simple input and output functions in ruby. Explain code blocks and iterators.	10	CO4	L2

#132, AECS Layout, IT Park Road, Kundalahalli, Bangalore – 560 037
T:+9180 28524466 / 77

CMR
INSTITUTE OF
TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Improvement Test Solution – Nov' 2017

10CS73 – Programming the WEB

1. A) What is hypertext? Explain HTTP phases. Mention various methods and status codes of HTTP.

HyperText Transfer Protocol:

The protocol used by ALL Web communications. It has a Request Phase with a Form: HTTP method domain part of URL HTTP ver.
Header fields blank line
Message body

An example of the first line of a request: GET /degrees.html HTTP/1.1

HTTP Methods:

GET - Fetch a document
POST - Execute the document, using the data in body
HEAD - Fetch just the header of the document
PUT - Store a new document on the server
DELETE - Remove a document from the server

HTTP Headers:

There are four categories of header fields: General, request, response and entity.

Common request fields: Accept: text/plain, Accept: text/*,
If-Modified_since: date

Common response fields: Content-length: 488, Content-type: text/html

- Can communicate with HTTP without a browser
- > telnet blanca.uccs.edu http GET /respond.html HTTP/1.1 Host: blanca.uccs.edu
  HTTP Response Form:
  Status line
  Response header fields blank line

Response body

Status line format:
HTTP version status code explanation

Example: HTTP/1.1 200 OK (Current
version is 1.1)
Status code is a three-digit number; first digit specifies the general status

Status Code
1 => Informational
2 => Success
3 => Redirection
4 => Client error
5 => Server error

HTTP Response: Example
HTTP/1.1 200 OK
Date: Tues, 18 May 2004 16:45:13 GMT
Server: Apache (Red-Hat/Linux)
Last-modified: Tues, 18 May 2004 16:38:38 GMT
Accept-ranges: bytes
Content-length: 364
Connection: close
Content-type: text/html, charset=ISO-8859-1

B) Write an XHTML document to describe an ordered list of four states. Each element of the list must have an unordered list of atleast two cities in the state

```
<html>
    <head>
        <title>Ordered List</title>
    </head>
    <body>
        <ol>
            <li>Karnataka</li>
                <ul>
                    <li>Bengaluru</li>
                    <li>Hassan</li>
                </ul>
                <li>Tamilnadu</li>
                <ul>
                    <li>Chennai</li>
                    <li>Coimbatore</li>
                </ul>
                <li>Kerala</li>
                <ul>
                    <li>Calicut</li>
                    <li>Trissur</li>
                </ul>
                <li>Andhra Pradesh</li>
                <ul>
                    <li>Vishakapatnam</li>
                    <li>Nellore</li>
                </ul>
        </ol>
    </body>
</html>
```

2. A) Explain syntactic differences between HTML and XHTML.

PARAMETERS	HTML	XHTML
Case Sensitivity	Tags and attributes names are case insensitive	Tags and attributes names must be in lowercase
Closing tags	Closing tags may be omitted	All elements must have closing tag
Quoted attribute values	Special characters are quoted. Numeric values are rarely quoted.	All attribute values must be quoted including numbers
Explicit attribute values	Some attribute values are implicit. For example: <table border>. A default value for border is assumed	All attribute values must be explicitly stated
***id* and *name* attributes**	Both *id* and *name* attributes are encouraged	Use of *id* is encouraged and use of *name* is discouraged
Element nesting	Rules against improper nesting of elements (for example: a form element cannot contain another form element) are not enforced.	All nesting rules are strictly enforced

B) Write an XHTML document to display an image and three buttons. The buttons should be labeled 1,2 and 3. When pressed, each button should change the content of the image to that of a different image.

```
<html>
    <head>
        <title>Button with images</title>
        <script type="text/javascript">
            function img(abc)
            {
                switch(abc)
                {
                    case 1:
                        document.write("<img src='table11.jpeg' width='500' height='500' />");
                        break;
                    case 2:
                        document.write("<img src='table31.jpeg' width='500' height='500' />");
                        break;
                    case 3:
                        document.write("<img src='table43.jpeg' width='500' height='500' />");
                        break;
                }
            }
        </script>
    </head>
    <body>
        <form action="">
            <img src="cot23.jpeg" width="500" height="500" />
            <input type="button" value="1" id="1" onclick="img(1)"/>
            <input type="button" value="2" id="2" onclick="img(2)"/>
            <input type="button" value="3" id="3" onclick="img(3)"/>
        </form>
    </body>
</html>
```

3. A) Explain all selector forms with example for each.

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 ;}
```

Selectors can also specify that the style should apply only to elements in certain positions in the document .This is done by listing the element hierarchy in the selector.

- Contextual selectors: Selectors can also specify that the style should apply only to elements in certain positions in the document .
- In the eg selector applies its style to the content of emphasis elements that are descendants of bold elements in the body of the document.

```
body b em {font-size: 24pt ;}
```

Also called as descendant selectors. It will not apply to emphasis element not descendant of bold face element.

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>...
 …

<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>

Apple is the common name for any tree of the genus Malus, of the family Rosaceae. Apple trees grow in any of the temperate areas of the world. Some apple blossoms are white, but most have stripes or tints of rose. Some apple blossoms are bright red. Apples have a firm and fleshy structure that grows from the blossom. The colors of apples range from green to very dark red. The wood of apple trees is fine-grained and hard. It is, therefore, good for furniture construction. Apple trees have been grown for many centuries. They are propagated by grafting because they do not reproduce themselves.

 APPLES ARE GOOD FOR YOU </body>
</html>

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>
```

B) Create XHTML document that has 2 frames. The left frame displays content.html and right frame displays cars.html where the second frame is a target of link from the first frame. Note : contents.html is a list of links to the cars description.

```html
<html>
    <head>
        <title>frames</title>
    </head>
    <frameset cols=30,70>
        <frame src="content.html" />
        <frame src="cars.html" name="n1" />
    </frameset>
</html>
```

Content.html

```html
<html>
    <head>
        <title>content</title>
    </head>
    <body>
        <ol>
            <li><a href="ferrari.html"  target="n1"/>Ferrari</li>
            <li><a href="volvo.html"  target="n1"/>Volvo</li>
            <li><a href="cooper.html" target="n1" />Cooper</li>
        </ol>
    </body>
</html>
```

Cars.html

```html
<html>
    <head>
        <title>cars</title>
    </head>
    <body>
        <p> These are luxury budget cars</p>
    </body>
</html>
```

Ferrari.html

```html
<html>
    <head>
        <title>ferrari</title>
    </head>
    <body>
        <p>Ferrari N.V. is an Italian sports car manufacturer based in Maranello. Founded by Enzo Ferrari in 1939 out of Al
    </body>
</html>
```

Volvo.html

```html
<html>
    <head>
        <title>volvo</title>
    </head>
    <body>
        <p>The Volvo Group is a Swedish multinational manufacturing company headquartered in Gothenburg. While it
    </body>
</html>
```

Cooper.html

```html
<html>
    <head>
        <title>cooper</title>
    </head>
    <body>
        <p>The 2018 Mini Cooper is a jaunty little car, with playful styling and spirited performance.</p>
    </body>
</html>
```

4. A) Explain the purposes of XML processors

THE PURPOSES OF XML PROCESSORS

- First, the processor must check the basic syntax of the document for well-formedness.
- Second, the processor must replace all references to entities in an XML document with their definitions.
- Third, attributes in DTDs and elements in XML schemas can specify that their values in an XML document have default values, which must be copied into the XML document during processing.

- Fourth, when a DTD or an XML schema is specified and the processor includes a validating parser, the structure of the XML document must be checked to ensure that it is legitimate.

B) Create a XHTML document with PHP to display the number, square root, square, cube and quadruple using sqrt and pow functions

```php
<?php
$a=20;
print "The number is $a<br />";
print "The square of the number is :".sqrt(20)."<br />";
print "The square of 2 is :".pow(2,2)."<br />";
print "The cube of 2 is :".pow(2,3)."<br />";
print "The quadruple of 2 is :".pow(2,4)."<br />";
?>
```

5. A) Describe the logical internal structure of an array in PHP

Arrays in PHP combine the characteristics of regular arrays and hashes

• An array can have elements indexed numerically. These are maintained in order

• An array, even the same array, can have elements indexed by string. These are not maintained in any particular order

• The elements of an array are, conceptually, key/value pairs

• Two ways of creating an array

• Assigning a value to an element of an array

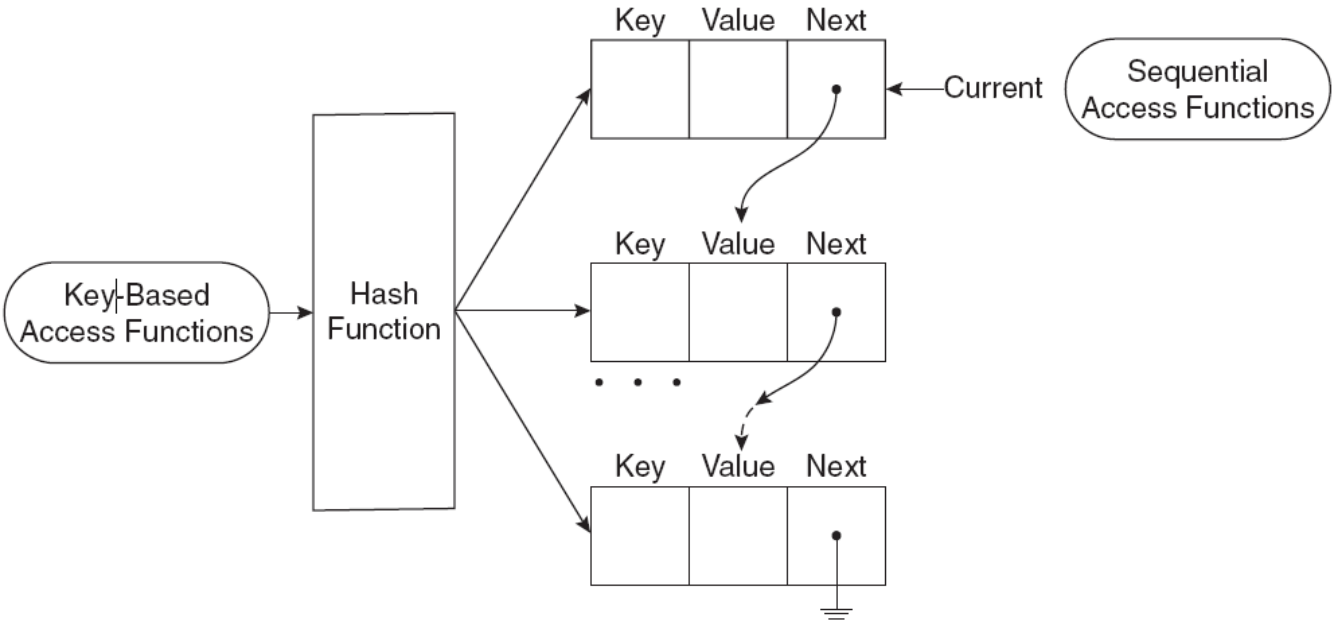
• Using the array function

• Create a numerically indexed array

• array(23, _xiv‘, —bob‖, 777);

• Create an array with string indexes

• array(—x‖ => —xerxes‖, —y‖ => —ytrbium‖)
• Array elements are accessed by using a subscript in square brackets
• An array can be assigned to a list of variables
• list($x, $y, $z) = array($y, $z, $x)
• The unset function can be used to remove an array or an element of an array
• The array_keys function returns a list of the keys of an array
• The array_values returns a list of values in an array
• The array_key_exists function returns true if a given key is actually present in a given array
• is_array determines if its argument is an array
• implode converts an array of strings to a single string, separating the parts with a specified string
• explode converts a string into a list of strings by separating the string at specified characters



B) Create a DTD for a catalog of cars, where each car has the child elements make, model, year, color, engine, number-of-doors, transmission type and accessories. The engine element has the child elements, number of cylinders and fuel system. The accessories element has the attributes radio, air-conditioning, power-windows and power-brakes, each of which is required and has the possible values yes and no. Entities must be declared for the names of popular car make.

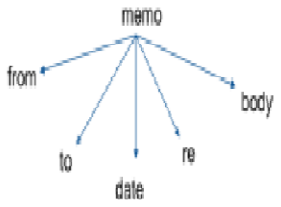
```
<!ELEMENT car (make, model, year, color, engine, number_of_doors, transmission_type, accessories)>

<!ELEMENT make (#PCDATA)>

<!ELEMENT model (#PCDATA)>

<!ELEMENT year (#PCDATA)>

<!ELEMENT color (#PCDATA)>

<!ELEMENT engine (number_of_cylinders, fuel_system)>

<!ELEMENT number_of_doors (#PCDATA)>

<!ELEMENT transmission_type (#PCDATA)>

<!ELEMENT accessories (#PCDATA)>

<!ATTLIST accessories radio CDATA #REQUIRED>

<!ATTLIST accessories air_conditioning CDATA #REQUIRED>

<!ATTLIST accessories power_windows CDATA #REQUIRED>

<!ATTLIST accessories power_steering CDATA #REQUIRED>

<!ATTLIST accessories power_brakes CDATA #REQUIRED>

<!ENTITY c "Chevrolet">

<!ENTITY f "Ford">
```

6. A) What is Document Type Definition? Describe the approach to declare elements, entities and attributes.

- A Document Type Definition (**DTD**) is a set of structural rules called **declarations**, which specify a *set of elements* that can appear in the document as well as *how and where these elements may appear*
- DTDs also provide **entity definitions**
- Not all XML documents need a DTD, use of a DTD is related to the use of an external style sheet for XHTML documents
- A DTD can be embedded in the XML document whose syntax rules it describes, in which case it is called an **internal** DTD
- The alternative is to have the DTD stored in a separate **file**, in which case it is called an **external** DTD
- Syntactically, a ***DTD is a sequence of declarations.***
- Each declaration has the form of a markup declaration:
- **<!keyword ...>**
- **Four** possible keywords can be used in a declaration:
- **ELEMENT** used to define **tags**;
- **ATTLIST** used to define **tag attributes**;
- **ENTITY** used to define **entities**; and
- **NOTATION** used to define **data type notations.**

Element Declarations:

- Each **element declaration** in a DTD specifies the structure of ***one category of elements***
- The declaration provides the **name of the element** whose structure is being defined, along with the specification of the **structure of that element**
- The form of an element declaration for elements that contain elements is as follows:
- **<!ELEMENT element_name (list of names of child elements)>**
- For ex., consider the following:
- **<!ELEMENT memo (from, to, date, re, body)>**



- It is necessary to specify the *number of times that a child element may appear*
- This can be done by adding a modifier to the child element specification
- **Child element specification modifiers**

Modifier	Meaning
+	One or more occurrences
*	Zero or more occurrences
?	Zero or one occurrence

- Consider the following DTD declaration:
- **<!ELEMENT person(parent+, age, spouse?, sibling*)>**
- In this, the person element is specified to have the following children elements: one or more *parent* elements, one *age* element, possibly a *spouse* element, and zero or more *sibling* elements
- The form of leaf element declaration is:
- **<!ELEMENT element_name (#PCDATA)**

Attribute Declarations:

- An **attribute declaration** must include the *name of the element to which the attribute belongs, the attribute's name, and its type*
- It may also include a **default value**
- The general form of an attribute declaration is as follows:
- **<!ATTLIST element-name attribute-name attribute-type [default-value]>**
- If more than one attribute is declared for a given element, the declarations can be combined, as follows:

```
<!ATTLIST element_name
   attribute_name_1 attribute_type default_value_1
   attribute_name_2 attribute_type default_value_2
   ...
   attribute_name_n attribute_type default_value_n
>
```

B) Explain any 5 string functions in PHP

Function	Parameter Type	Returns
strlen	A string	The number of characters in the string
strcmp	Two strings	Zero if the two strings are identical, a negative number if the first string belongs before the second (in the ASCII sequence), or a positive number if the second string belongs before the first
strpos	Two strings	The character position in the first string of the first character of the second string if the second string is in the first string; false if it is not there
substr	A string and an integer	The substring of the string parameter, starting from the position indicated by the second parameter; if a third parameter (an integer) is given, it specifies the length of the returned substring
chop	A string	The parameter with all white-space characters removed from its end
trim	A string	The parameter with all white-space characters removed from both ends
ltrim	A string	The parameter with all white-space characters removed from its beginning
strtolower	A string	The parameter with all uppercase letters converted to lowercase
strtoupper	A string	The parameter with all lowercase letters converted to uppercase

7. A) Write a CGI-PERL program to use a cookie to remember the day of the last login, from a user and display it when executed

```perl
#!C:\xampp1\perl\bin\perl.exe
use CGI ":standard";
@last_day = cookie('last_time');
$day_of_week = (qw(Sunday Monday Tuesday Wednesday Thursday Friday Saturday))[(localtime)[6]];
$month = (qw(January February March April May June July August September October November December)) [(localtime)[4]];
$day_of_month = (localtime)[3];
@day_stuff = ($day_of_week, $day_of_month, $month);
$day_cookie = cookie(-name => 'last_time',
                     -value => \@day_stuff,
                     -expires => '+60s');
print header(-cookie => $day_cookie);
print start_html('This is day_cookie.pl');
if (scalar(@last_day) == 0)
{
   print "Welcome to you on your first visit to our site <br />";
}
else
{
   ($day_of_week, $day_of_month, $month) = @last_day;
   print "Your last visit was on $day_of_week, $month $day_of_month <br />";
}
print end_html;
```

B) Develop a complete XHTML document with proper headings, a table with four rows and three columns, a form with two labels, two text box, three check box, three radio buttons, a submit and a reset button.

```
tml>
  <head>
      <title>complete form</title>
  </head>
  <body>
      <table border="4" cellspacing="5" cellpadding="5">
      <caption>CMRIT</caption>
          <tr>
              <th>Dept</th>
              <th>CSE</th>
              <th>ISE</th>
          </tr>
              <td>Sem-4</td>
              <td>50</td>
              <td>60</td>
          <tr>
          </tr>
              <td>Sem-6</td>
              <td>60</td>
              <td>50</td>
          <tr>
          </tr>
              <td>Sem-8</td>
              <td>70</td>
              <td>70</td>
          <tr>
      </table>
  <form action="">
      <br />
      <label>Enter your name: <input type="text" id="n1" /> </label><br /><br />
      <label>Enter your USN: <input type="text" id="n2" /> </label><br /><br />
      <label>Enter your Semester : Sem-4 <input type="checkbox" id="n3" />
                                   Sem-6 <input type="checkbox" id="n3" />
                                   Sem-8 <input type="checkbox" id="n3" />
      </label><br /><br />
      <label>Elective Subject : JAVA <input type="radio" id="n4" />
                                   C# <input type="radio" id="n4" />
                                   SAN <input type="radio" id="n4" />
      </label><br /><br />
      <input type="submit" value="SUBMIT FORM" />
      <input type="reset" value="RESET FORM" /><br />
  </form>
body>
>
```

8. A) With example, explain simple input and output functions in ruby. Explain code blocks and iterators.

Output is directed to the screen with the puts method (or operator). We prefer to treat it as an operator. The operand for puts is a string literal. A newline character is implicitly appended to the string operand. If the value of a variable is to be part of a line of output, the #{...} notation can be used to insert it into a double-quoted string literal, as in the following interactions:
>> name = —Fudgy‖ => —Fudgy‖

>> puts —My name is #{name}‖

My name is Fudgy => nil The value returned by puts is nil, and that is the value returned after the string has been displayed. The print method is used if you do not want the implied newline that puts adds to the end of your literal string. The way to convert a floating-point value to a formatted string is with a variation of the C language function sprintf. This function, which also is named sprintf, takes a string parameter that contains a format code followed by the name of a variable to be converted. The string version is returned by the function. The format codes most commonly used are f and d. The form of a format code is a percent sign

(%), followed by a field width, followed by the code letter (f or d). The field width for the f code appears in two parts, separated by a decimal point. For example, %f7.2 means a total field width of seven spaces, with two digits to the right of the decimal point—a perfect format for money. The d code field width is just a number of spaces—for example, %5d. So, to convert a floating-point value referenced by the variable total to a string with two digits to the right of the decimal point, the following statement could be used: str = sprintf(―%5.2f‖, total) **Keyboard Input** Because Ruby is used primarily for Rails in this book, there is little need for keyboard input. However, keyboard input is certainly useful for other applications, so it is briefly introduced here. The gets method gets a line of input from the keyboard. The retrieved line includes the newline character. If the newline is not needed, it can be discarded with chomp: >> name = gets apples => ―apples\n‖ >> name = name.chomp => ―apples‖ This code could be shortened by applying chomp directly to the value returned by gets: >> name = gets.chomp apples => ―apples‖
If a number is to be input from the keyboard, the string from gets must be converted to an integer with the to_i
method, as in the following interactions: >> age = gets.to_i 27 => 27 If the number is a floating-point value, the conversion method is to_f: >> age = gets.to_f 27.5 => 27.5