**Q.2** a. How HTML is used to Create a Web Document?

**Answer:**

HTML is a set of codes that a website author inserts into a plain text file to format the content. The author inserts HTML tags, or commands, before and after words or phrases to indicate their format and location on the page. HTML tags are also used to add tables, lists, images, music, and other elements to a webpage.

Web documents contain three main sections: the head, title, and body. The head includes the webpage's identifying information, including the website's title and important keywords. The viewer sees the site's title, but any other information is hidden. The site's title appears in the browser's tab and is what appears when a user tries to bookmark a site. The body section is the main portion of the webpage visible to the viewer, including the text and graphics. HTML tags are also used in two additional ways that are not visible to the viewer: as meta tags and comments. Meta tags indicate keywords associated with the webpage to search engines. Comments are intended as explanation or additional information for other writers or readers of HTML code.

b. How a simple web page is created? Explain using suitable example.

**Answer:**

To create a simple web page, the first step is to learn a few HTML tags. And copy and paste, the HTML tags into a file using a text editor. Then save the file as DOS text file with the name index.html

Here are some basic HTML tags that show how a web page is created.

```
<html>
   <head>
           <title> favorites / bookmark title goes here </title>
   </head>
   <body bgcolor="white" text="blue">
           <h1> My first page </h1>
            This is my first web page and I can say anything I want in here
            by putting text or images in the body section
   </body>
</html>
```

c. What is XHTML? Why it is known as evolutionary?

**Answer:**

XHTML is a family of current and future document types and modules that reproduce, subset, and extend HTML 4. XHTML family document types are XML based, and ultimately are designed to work in conjunction with XML-based user agents.

XHTML is a reformulation of the three HTML 4 document types as applications of XML. It is intended to be used as a language for content that is both XML-conforming and, if some simple guidelines are followed, operates in HTML 4 conforming user agents.

Developers who migrate their content to XHTML will realize the following benefits:

- XHTML documents are XML conforming. As such, they are readily viewed, edited, and validated with standard XML tools.
- XHTML documents can be written to operate as well or better than they did before in existing HTML 4-conforming user agents as well as in new, XHTML conforming user agents.
- XHTML documents can utilize applications (e.g. scripts and applets) that rely upon either the HTML Document Object Model or the XML Document Object Model .
- As the XHTML family evolves, documents conforming to XHTML will be more likely to interoperate within and among various XHTML environments.

**The XHTML family is the next step in the evolution of the Internet.** By migrating to XHTML today, content developers can enter the XML world with all of its attendant benefits, while still remaining confident in their content's backward and future compatibility.

**Q.3** b. What are the improved features in CSS2 as compared to CSS?

**Answer:**

Font selection

When choosing which font to use, CSS2 offers both the standard "name matching" system that CSS1 uses, plus three other methods for defining fonts. These are: intelligent font matching, where the user agent uses a font that is the closest match to the requested font. Font synthesis, where the user agent creates a font that matches the metrics of the requested font. And font download, where the user agent retrieves a font over the Web.

**Tables**

CSS2 recognizes that there might not be a table element (and related elements) in an XML document - but to display tabular data, it is important to have this as a style. So CSS2 allows you to define any element as a table element (and all the related table elements).

**Positioning**

While CSS1 had some aspects of positioning, CSS2 takes it to the next level. Relative and absolute positioning determine their location based on their placement within the document or based on the user agent. But along with absolute positioning is the concept of fixed positioning. This acts as a sort of "watermark" in continuous media. In paged media, an element with fixed position is repeated on every page. This allows you to create frame-like documents or place a signature on every page of a document.

**Cursors**

Now you can define how you want your cursor to respond to various actions. For example, you might want the default behavior over a link to be changed over some of the links in your document. With CSS2 you can define how the cursor should look over any element.

There are many other features that are new with CSS2, but the above are some of the most exciting ones. There are also elements like text-shadows, new pseudo-classes, the use of system colors, and dynamic outlines.

**Q.4** a. How to use strings as array indexes using JavaScript?

**Answer:**

Javascript does not have a true hashtable object, but you can use the array as a hashtable.

```
<script type="text/javascript">
var days = ["Sunday","Monday","Tuesday","Wednesday",
"Thursday","Friday","Saturday"];

for(var i=0; i < days.length; i++) {
days[days[i]] = days[i];
}
document.write("days[\"Monday\"]:"+days["Monday"]);
</script>
```

This produces

```
days["Monday"]:Monday
```

b. What is JavaScript Function? Illustrate through a suitable example.

**Answer:**

Java script Function is nothing but it is a reusable code-block that is execute when the function is called. Function is defined in the head section of the code.

**The syntax of JavaScript function is as follows:**

```
function fname(prameter1,parameter2, ...)
{
JavaScript code 1
JavaScript code 2
....
}
```

**Example:**

```
<html>
<head>
<script type="text/javascript">
function displaymessage()
{
alert("Hello World!");
}
</script>
```

```
</head>
<body>
<form>
<input type="button" value="Click me!" />
</form>
</body>
</html>
```

If the line: alert("Hello world!!") in the example above had not been put within a function, it would have been executed as soon as the page was loaded. Now, the script is not executed before a user hits the input button. The function displaymessage() will be executed if the input button is clicked.

**The return Statement**

The return statement is used to specify the value that is returned from the function. Functions that are going to return a value must use the return statement. The example below returns the product of two numbers (a and b):

**Example:**

```
<html>
<head>
<script type="text/javascript">
function product(a,b)
{
return a*b;
}
</script>
</head>
<body>
<script type="text/javascript">
document.write(product(4,3));
</script>
</body>
</html>
```

c. “Events are the beating heart of any JavaScript application.” Explain how.

**Answer:**

Without events there are no scripts. Take a look at any web page with JavaScript in it: in nearly all cases there will be an event that triggers the script. The reason is very simple. JavaScript is meant to add interactivity to your pages: the user does something and the page reacts.

Therefore JavaScript needs a way of detecting user actions so that it knows when to react. It also needs to know which functions to execute, functions that do something that you, the web developer, have judged likely to increase the appeal of your pages. These pages describe the best way to write such scripts. It isn’t easy, but it is very satisfying work.

When the user does something an event takes place. There are also some events that aren't directly caused by the user: the load event that fires when a page has been loaded, for instance.
JavaScript can detect some of these events. From Netscape 2 onwards it has been possible to attach an event handler to certain HTML elements — mostly links and form fields in the early days. The event handler waits until a certain event, for instance a click on a link, takes place. When it happens it handles the event by executing some JavaScript you have defined. When the user takes action he causes an event. When your script makes the page react to this event, interactivity is born.

**Q.5** a. What is a Rollover Button? What are the types and uses of Rollover Buttons?

**Answer:**

A rollover button is a web button that changes its appearance with the move of your mouse. It contains three states: normal, over and down. Normal state applies to when your mouse is off the button, over state applies to when your mouse rolls over the button and down state applies to when you click on the button. Unlike static web buttons, rollover buttons are dynamic in nature, allowing you to change the color or style of the button with the move of your mouse.

**Types of Rollover Buttons**

Rollover buttons come in different shapes, sizes, colors and styles. Although most rollover buttons are rectangular in shape, they can be customized to the shape of a circle, star or polygon. Since images can be converted to rollover buttons, a rollover button can also contain images.
A rollover button can be raised, sunken, flat or animated when a user clicks on it.

**Uses of Rollover Buttons**

Rollover buttons are used primarily as navigational buttons on a web page to direct people to other locations. They are also used in drop-down and pop-up menus. Other rollover buttons are used for animated effects and sounds, so an image, color, shape, text or sound can change as the user rolls over the button on the web.

b. What are the pros and cons in deciding whether to use JavaScript Library or to Code yourself?

**Answer:**

One disadvantage in the past to using JavaScript libraries is that the libraries tend to be rather large and perhaps include quite a lot of code that your individual web pages don't actually use. It could therefore be argued that writing your own code that does exactly what you need and nothing more would be more efficient. The introduction by Google of a common repository that everyone using one of the more popular JavaScript libraries can link to negates this argument since if you link to the Google copy of the library instead of using your own copy then a large number of your visitors will already have the library cached in their browser and so will only need to actually download the part of the code that is unique to your page. Doing this may make using a library faster than doing it yourself.

One advantage to writing the code yourself and not relying on libraries is that there is less to learn. To be able to write your own JavaScript you just need to learn JavaScript. To be able to use one of the libraries you need to learn both JavaScript and that library. There may be a few things that you will be able to do if you just learn the library without also learning JavaScript but to be able to use the library in the best possible way you will need a greater understanding of JavaScript than that of the typical person who just uses JavaScript to write their own code.

An advantage to using a library that already handles 90% of the processing that you need to have run means that you only need to write that remaining 10%. This may save you time in writing the code since there is a lot less code to write. It may or may not save you time overall since in addition to writing the code you also need to test it and unless you have an in depth understanding of exactly how your chosen library does what it does the testing time using a library could end up being significantly longer than it would have been had you written the code yourself.

Another thing you need to consider is just how much of what a given library contains is code that you will actually need to use to do what you want to achieve as well as whether the library actually includes all of the processes you want. If you can't find one library that does everything you want then you will either need to write the rest yourself or try to incorporate a second library that includes the missing functionality. Depending on how the libraries work it may or may not be possible to use two of them together in the same web page.

How much JavaScript you have actually written yourself before considering using a library may also influence your decision. If you have already written all your own processing that performs all of the different functions you need then that may put you off the idea of switching to using a library that provides all those same functions but written in a different way. Given that those who have actually written these libraries in the first place are generally people with a thorough knowledge of JavaScript, you'd expect that there would be a reasonable likelihood of the library providing the functionality in a more efficient manner than that which you have written yourself (unless you too are a JavaScript expert).

What sort of JavaScript processing you want to add to a web page will also affect whether using a library is appropriate or not. If you are only adding minimal effects into your page that will only require a few lines of JavaScript to add anyway then using a library could result in your having to write more code to interface to the way the library does things than if you wrote all the code yourself. Of course the opposite applies if the processing you want to add into your page is more significant and matches closely to the type of functionality that your chosen library handles for you.

**Q.6** a. What is the main function of a regular expression in perl? Write a code demonstrating the use of a variable as a regular expression.

**Answer:**

Regular expressions are almost a language in themselves. Their main function is to provide a flexible syntax for finding patterns in text, but the advanced features they now provide virtually make them a language in themselves (for instance someone wrote a regular expression which would find prime numbers!).

Here's the code demonstrating the use of a variable as a regular expression.

```
#!/usr/bin/perl -w
# use strict;

sub test($$)
        {
        my $lookfor = shift;
        my $string = shift;
        print "\n$lookfor ";
        if($string =~ m/($lookfor)/)
                {
                print " is in ";
                }
        else
                {
                print " is NOT in ";
                }
        print "$string.";
        if(defined($1))
                {
                print "     <$1>";
                }
        print "\n";
        }

test("ra.h.", "radha was here");
test("ra.h.", "i love ramesh ");
test("ra.h.", "ready to leave");
```

The preceding code produces the following output.

```
ra.h.  is in radha was here.      <radha>

ra.h.  is in i love ramesh      <ramesh>

ra.h.  is NOT in ready to leave.
```

b. How do you define and call a subroutine in perl? How are the parameters passed? Write a sample code demonstrating the syntax.

**Answer:**

**Define and call subroutine**

Like other computer programming languages, Perl allows you to write your own user-defined function which is called subroutine. Subroutine allows you to reuse a chunk of code in program over and over again. You can place subroutine anywhere in program but it is recommended that subroutine should be placed at the beginning or at the end of program. To define subroutine you use the keyword sub followed by subroutine name and its body.

**Here is the syntax of subroutine definition.**

```
1 sub subroutine_name{
2 #subroutine body
3 }
```

**For example, here is a simple subroutine to print a message.**

```
1 sub print_message{
2 print "Perl subroutine tutorial";
3 }
```

To invoke or call subroutine, you use the subroutine name with the prefix is ampersand (&) as follows:

```
1 &print_message;
```

**Parameters in subroutine**

Parameters are passed to a subroutine as a list which is available in a subroutine as a @_ - an array variable.

For example:

```
1 sub print_list{
2 print "@_\n";
3 }
4
5 #print: this is Perl subroutine tutorial
6 &print_list("this","is","Perl","subroutine","tutorial");
7 #print: subroutine tutorial
8 &print_list("subroutine","tutorial");
```

In the line 2, we printed the list variable which was the input parameters of the subroutine. In line 5 and 7, we called subroutine with different list and we got different output.

You can refer to each parameter in a list by using scalar $_[index] for example:

```
sub max{
  if($_[0] > $_[1]){
     $_[0];
  }
  else{
     $_[1];
  }
}

$m = max(10,20);
$m2 = max(50,30);
print "$m\n";
print "$m2\n";
```

In the above example, we defined a subroutine which is used to return the maximum value of two input parameters. First we compared the first array elements to the second and return the maximum one. We referred to the parameters by using scalar $_[0] and $_[1]. Then we called the subroutine to return the maximum values of two input parameters. Let's go to the next section to see how subroutine returns values.

**Returning values**

Subroutine returns values which is the last expression evaluated. For example :

```
1  sub cal_sum{
2    print "sum of two numbers\n";
3    $_[0] + $_[1];
4  }
5  $result = &cal_sum(10,20); # result is 30
6  print $result;
```

In the *cal_sum* subroutine above, in the line 3 we added two input parameters and returned the result. If you want to return value immediately from subroutine, you can use *return*operator, for example:

```
sub min{
 if($_[0] < $_[1]){
    return $_[0];
  }
  else{
    return $_[1];
  }
}

$m = &min(10,20); # result is 10
print $m;
```

Subroutine returns not only scalar value but also a list of values. Let's take a look at an example below:

```
sub get_list{
  return 5..10;
}
@list = &get_list;
print "@list"; #5 6 7 8 9 10
```

**Q.7** a. What is CGI? Explain.

**Answer:**

CGI, or Common Gateway Interface, is the standard programming interface between web servers and external programs. It is almost one of the most exciting and fun areas of programming today. The CGI standard lets web browsers pass information to programs written in any language. If you want to create a lightning-fast search engine, then your CGI program will most likely be written in C or C++. However, most other applications can use Perl.

The CGI standard does not exist in isolation, it is dependent on the HTML and HTTP standards. HTML is the standard that lets web browsers understand document content. HTTP is the communications protocol that, among other things, lets web servers talk with web browser.

b. How to save a cookie after the header has been sent in Perl CGI? Explain.

**Answer:**

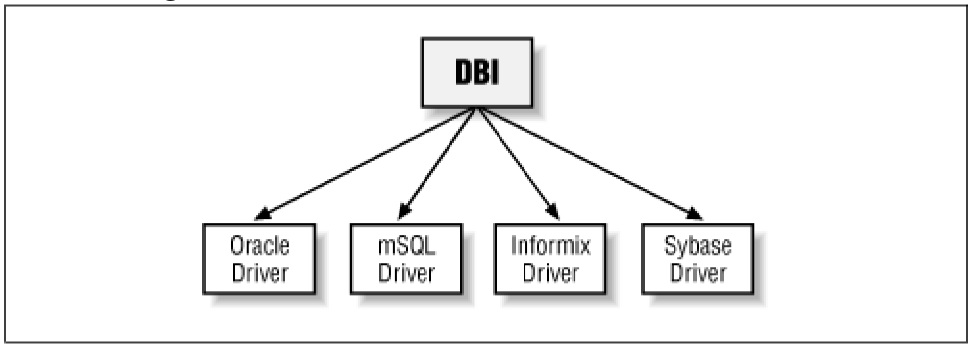
You can't. Cookies are only set in the header. HTTP provides no way to set them elsewhere. If you want to set a cookie based on form data, then you do it in the response to the form submission request.

If you want to use data both for generating a cookie and generating a form, then get that data into a variable before you send the header and use it in both locations. You could generate JavaScript to set the cookie in the HTML body but that would be unnecessarily complex and unreliable.

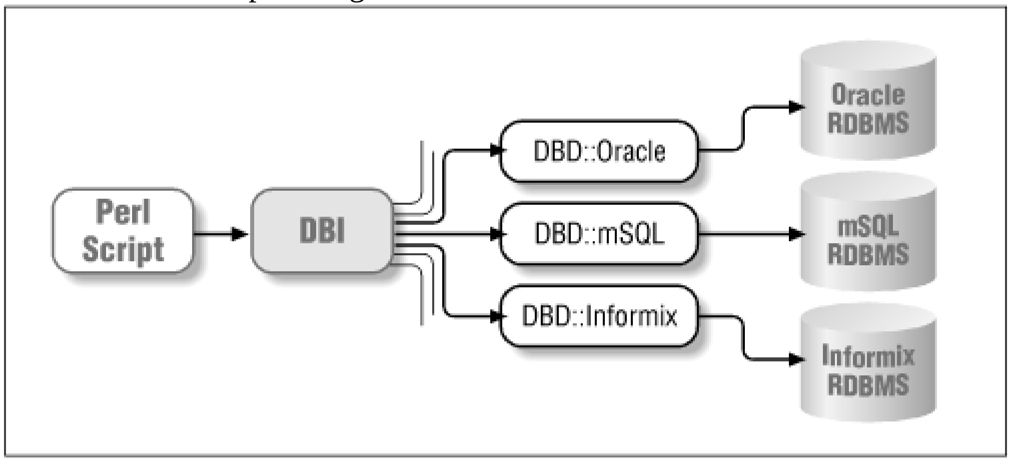
c. Describe the architecture of Perl DBI. How does the separation of the drivers from the DBI itself help?

**Answer:**

The DBI architecture is split into two main groups of software: the DBI itself, and the drivers. The DBI defines the actual DBI programming interface, routes method calls to the appropriate drivers, and provides various support services to them. Specific drivers are implemented for each different type of database and actually perform the operations on the databases. Figure below illustrates this architecture.



Therefore, if you are authoring software using the DBI programming interface, the method you use is defined within the DBI module. From there, the DBI module works out which driver should handle the execution of the method and passes the method to the appropriate driver for actual execution. This is more obvious when you recognize that the DBI module does not perform any database work itself, nor does it even know about any types of databases whatsoever. Figure below shows the flow of data from a Perl script through to the database.



Under this architecture, it is relatively straightforward to implement a driver for any type of database. All that is required is to implement the methods defined in the DBI specification as supported by the DBI module, in a way that is meaningful for that database. The data returned from this module is passed back into the DBI module, and from there it is returned to the Perl program. All the information that passes between the DBI and its drivers is standard Perl datatypes, thereby preserving the isolation of the DBI module from any knowledge of databases.

**The separation of the drivers from the DBI itself makes the DBI a powerful programming interface that can be extended to support almost any database available today.** Drivers currently exist for many popular databases including Oracle, Informix, mSQL, MySQL, Ingres, Sybase, DB2, Empress, SearchServer, and PostgreSQL. There are even drivers for XBase and CSV files.
These drivers can be used interchangeably with little modification to your programs. Couple this database-level portability with the portability of Perl scripts across multiple operating systems, and you truly have a rapid application development tool worthy of notice.
Drivers are also called database drivers, or DBDs, after the namespace in which they are declared. For example, Oracle uses DBD::Oracle, Informix uses DBD::Informix, and so on. A useful tip in remembering the DBI architecture is that DBI can stand for DataBase Independent and DBD can stand for DataBase Dependent.

**Q.8** a. Write simple PHP script for sieve of Eratosthenes (finding all prime numbers up to a specified integer).

**Answer:**

```
<?php
   // sieve or eratosthenes in PHP
  function eratosthenes($n)
  {
     $all=array();
     echo 1," ",2;
     $i=3;
     while($i<=$n)
     {
        do
        {
           if(!in_array($i,$all))
           {
              echo " ",$i;
              $j=$i;
              while($j<=($n/$i))
              {
                 do
                 {
                    array_push($all,$i*$j);
                 } while(false);
                 $j+=1;
              }
           }
        } while(false);
        $i+=2;
     }
     echo "\n";
     return;
  }
  eratosthenes(50);
  ?>
```

b. Explain the procedure to Use Sessions in your PHP Scripts.

**Answer:**

To use sessions in your script you need to do the following.

**Starting a Session**

At the beginning of your script, make a call to the session_start() function. This call should be in every script that needs to utilise the session data. For example, if you have a script that creates a CAPTCHA image and needs to store the secret word for the session, you will need to put session_start() at the beginning of the script. If you have another script that takes the user input for the form and checks the secret word entered by the user against what you stored earlier, you will also need to put session_start() in that script.

The function session_start() takes no parameters. It always returns TRUE, so you don't have to bother to check its return value.

When session_start() is first called, PHP sets a cookie (yes, a cookie) in your visitor's browser, containing a session identifier ("session ID"). It also creates a session data file to store variables related to that particular session. If the same script, or another script on your site, calls session_start() later, the PHP interpreter will receive the session ID cookie from the browser and load the variables from the session data file it created earlier.

**Storing and Accessing Variables**

To store variables relevant to the session, assign what you want to a member of the $_SESSION array. For example, the following snippet assigns "ABC123" to $_SESSION["secretword"] and a colour to $_SESSION["theme"]:

```
$_SESSION["secretword"] = "ABC123" ;
$_SESSION["theme"] = "purple" ;
```

You can assign as many variables as you wish.

To access those variables, simply reference it as you would any PHP array.

**Ending a Session**

Ending a session is not as easy as starting one, since there is no simple function to cleanly end it. If you really need a way to end a session yourself (other than by the user simply quitting his/her browser), PHP provides the session_destroy() to destroy the data associated with a session. However, this in itself does not clean up everything. For example, the session cookie is not unset. The $_SESSION array is also still available until your script ends.

To remove the cookie, manually delete it using the usual method one uses to delete a cookie in PHP. To get the name of the cookie to delete, call the session_name() function, which returns a string that is also the name of the cookie set by the PHP session handler.

**Q.9** a. "Reading XML can be more complicated than writing it, if only because we have so many options for reading XML. There are basically two approaches, using an XML parser or using XSLT". Elaborate.

**Answer:**

XSLT shines at transforming one type of XML markup into another type of markup, or into HTML, using standardized syntax and vocabulary. Generally, to process XML using XSLT you must run a program called an XSLT processor, such as Saxon or MSXSL.
In contrast, parsers are good at picking out specific elements in XML files and are generally bound to programming languages, which means that you must write your parser in Java, Perl, PHP, etc.
So, in general, you would choose to use XSLT if you were batch converting XML to HTML (or another XML markup) and you would choose a parser if you were writing a script and wanted to access selected elements in your source XML.

b. What do you mean by DOM? Explain how XML DOM is used?

**Answer:**

A DOM for XML is an object model that exposes the contents of an XML document. The W3C's Document Object Model (DOM) Level 1 Specification currently defines what a DOM should expose as properties, methods, and events. Microsoft's implementation of the DOM fully supports the W3C standard and has additional features that make it easier for you to work with XML files from your programs.

c. Why do applications use a DTD when the XML document follows the same structure?

**Answer:** You use the XML DOM by creating an instance of an XML parser. To make this possible, Microsoft exposes the XML DOM via a set of standard COM interfaces in Msxml.dll. Msxml.dll contains the type library and implementation code for you to work with XML documents. If you're working with a scripting client, such as VBScript executing in Internet Explorer, you use the DOM by using the Create Object method to create an instance of the Parser object.

**Text Book**

Web Programming – Building Internet Applications, Chris Bates, Third Edition, Wiley Student Edition, 2006