HLL PART 3 - PHP

Exercise Sheet 1 – Introducing the PHP Interpreter and more…

Coding PHP scripts is a breeze because it uses exactly the same setup as you have been using with perl, except for a different shebang line at the top of your files.To compile a simple program using PHP you simply type the url for that page into the web browser and keep your fingers crossed for good results. For example

http://scarlet.cs.nott.ac.uk/~username/cgi-bin/myscript.php

Important things to remember are that university cgi-bin pages are not available on the world wide web – so remember that it is **scarlet.cs.nott.ac.uk**. Also make sure that your permissions for the file, as well as your cgi-bin, are set correctly and you have the shebang at the top of the script.

Also make sure you are writing your scripts in unix using Nedit. If you don't you are lining yourself up for end of line errors caused by windows text formatting, which can be a nightmare to debug**. I recommend to always use Nedit.**

1. Using *your* favourite text editor (you do have a favourite don't you?), create a script containing the following code for a "Hello World" style program. Save it as a file called **hello.php** and save it somewhere inside your cgi-bin.

```
#!/usr/local/bin/php

<HTML>
<BODY>

<?
    print "I'm a HLL student, get me out of here";
?>

</HTML>
<BODY>
```

Display the script in your browser. If the result produces any error messages check the common errors mentioned above, correct the source file, save it, and try again. Remember unlike perl you don't need any pre-processing such as CGI objects.

2. Now your off and running, write a script which displays the times tables.

I.e. 1x1=1, 2x1=2, 3x1=3…

..up to 12x12=144

(*Hint – you will need to use two loops nested within each other combined with some print statements*)

3. Display the times tables formatted in an HTML table.

How you do this is up to you, but your going to need to print <TD> and <TR> tags somewhere in those loops – and don't forget to close your table with a </TABLE> at the end.

4. We're going to move straight on to handling user input (all common structures are exactly the same as all other languages that you've learnt so there is no point dwelling on them.)

Write an HTML form page (*you will have to save it in your public_html folder remember*) that prompts the user to input their firstname, surname and age. When the user clicks submit on this HTML page it will send the information to a feedback php script.

Write this feedback page (in your cgi-bin remember) that processes the form simply by displaying the form fields on the page.

5. Now combine these two pages (the HTML and PHP) into one php page that does different things depending on the data sent to it. If no data is sent, it is clearly the first time the user has arrived so output the HTML form.

However if a firstname variable exists for example, we have clearly been sent data and it is time to display the forms results on screen as we did before.

(Hints – you may want to check the name variable using an if statement with the ***empty*** *or* ***isset*** *commands to decided which action to undertake. Don't forget to that this time when the user submits his info the button will call itself – you can hard code this or better still use the* ***$PHP_SELF*** *variable as the target of the form)*

6. Finally lets store this user information in a **session**. Start the session at the top of your script and register the session variables firstname, lastname and age.

Then in the appropriate section of your code fill these session variables with the information sent from the HTML form section of you script as it calls itself.

(Hint. check the syntax of this in PHP lecture 4)

7. To test if this has all worked add a hyperlink at the bottom of your output section, to a new PHP script called session_check.php. This new script should look something like and again be saved in your cgi-bin:

```
#!/usr/local/bin/php
<?
   session_start();
   print "Session variable Firstname=$firstname <BR>";
   print "Session variable Lastname=$lastname <BR>";
   print "Session variable Age=$age <BR>";
?>
```

Don't forget the important session start line at the top. If this new page outputs the correct information that the user originally typed in you can claim victory – how much easier is that than messing about with perl and cookies!?

8. Update your current files so that the HTML form part of your main PHP script takes in one input field "Size of your christmas tree".

Your output part of that script should save this $size variable as a session variable and display it to screen in the format "Your christmas tree is SIZE high – click here to see it", where SIZE is the number the user typed into the form.

The click here link will then take you to the session_check.php as before which will this time actually draw the christmas tree in the following format, using stars and a loop, depending on the size session variable that it can of course see:

```
   *
  **
 ***
****
*****
   *
```

This would be a christmas tree of size 5. *(Hint. To draw a space in HTML use the special character identifier)*