Lab 10

PHP Basics

Objective:

To familiarize students with the basic programming constructs in PHP**.**

Activity Outcomes:

After this lab the students should be able to use

- PHP constructs to solve basic programming problems

Instructor Note:

As pre-lab activity, read Chapter 1 from the textbook "Beginning PHP and MySQL: From Novice to Professional 5th ed. Edition by by Frank M. Kromann, Year 2018"

1) Useful Concepts

PHP is a powerful server side scripting language. It is used to create dynamic web pages. It is mainly
used to manage database at server. PHP can be embedded with HTML in three ways

- We can add blocks of PHP code in HTML
- We can use HTML instructions in PHP
- We can write standalone PHP script

Following are some of the basic rules for writing PHP code

- Blocks of PHP code can be added in HTML code with opening and closing tags, as follows:

<?php and ?>.
Similarly, we can use simply <? or ?> or <script language="PHP">……
</script> to add PHP blocks in HTML

- PHP statements end with a semicolon
- Comments can be added as: // for one line comment and /* and */ for multiple lines comment PHP provides several instructions to write output on the browser screen. Most commonly we use echo to write output on the browser screen.

The syntax of the echo is
echo ("Welcome to PHP");

We can also use echo command as
echo “Welcome to PHP”;
print command can also be used to write out put on the browser. The syntax of writing print command is
print(“Welcome to PHP”); or print “Welcome to PHP”;
echo is marginally faster as compared to print as echo does not return any value. printf function can also be used to output a formatted string.
We can also write HTML instructions in PHP. The echo statement outputs whatever it’s told to the browser. It can output not only plain text but also HTML tags. We have to write HTML tags in quotation marks. In PHP single and double quotation marks are used interchangeably but their logical sequence is must to maintain.
A constant is a placeholder for a value that you reference within your code that is formally defined before using it. The name of a constant in PHP begins with a letter or an underscore. Names of constants are case sensitive. Typically they are named using all capital letters. PHP function define() is used to assign a value to a constant.
In PHP, variable names begin with $ sign. First character must be a letter or underscore while remaining characters may be letters, numbers or underscores. Variable names are case sensitive. We don’t need to declare or initialize variables. PHP is a loosely typed language it means that data types does not require to be declare explicitly.

An operator is a symbol used to perform a specific operation on variables or operands. PHP provides several types of operators to represent different operations. Following are the operators available in PHP
Arithmetic operators: + (addition), - (subtraction), / (division), * (multiplication) and % (remainder)

Assignment operator: = (assignment), += (Operator adds and assigns a value. For example $a += $b, here += adds the value of $b in $a and then assigns the result to $a. -=, *= and/= operators can also be used), .= operator concatenates and assign the value. For example $.=$b, here .= concatenates the value of $b with $a and then assigns this value to $a)
String operators: . (Concatenate two strings), .= (concatenates; and assign the value).

Increment and decrement operators: ++ (increment), -- (decrement). These operators can be used either in prefix or postfix form.
Logical operators: AND or && (logical and), OR or || (logical or), Not or! (logical not) and XOR (logical exclusive-or).
Comparison operators: = = (equality), = = = (checks both types and values of variables), != (inequality), > (greater), < (less), >= (greater or equal) and <= (less or equal).
Conditional statements: Conditional statements make it possible for your computer program to respond accordingly to a wide variety of inputs, using logic to discern between various conditions based on input value. In PHP following conditional statements are available.

If statement: if statements allow code to be executed when the condition specified is met; if the condition is true then the code in the curly braces is executed. Here is the syntax for an if statement:

```
if (condition)
{ statement }
```

if….else statement: When you have two possible situations and you want to react differently for each, you can use an if...else statement. This means: “If the conditions specified are met, run the first block of code; otherwise run the second block.” The syntax is as follows:

```
if (condition)
{
code to be executed if condition is true
}
else
{
code to be executed if condition is false
}
```

Switch statement: You can think of the switch statement as a variant of the if-else combination, often used when you need to compare a variable against a limited number of values called cases in switch statement terminology. The syntax of the switch statement in PHP is

```
switch(variable)
{
case option:
action break;
.
.
}
```

Looping statements in PHP: Looping statements are used to execute the same block of code a specified number of times. Following are the basic loops in PHP.

A while loop runs the same block of code while or until a condition is true. Syntax of for loop is given below

while loop:

```
while(condition)
{
Statements Increment/decrement
}
```

A do while loop runs once before the condition is checked. If the condition is true, it will continue to run until the condition is false. (The difference between a do and a do while loop is that do while runs once whether or not the condition is met.).

```
do
{
```

Statements Increment/decrement
}
while(condition)

A for loop runs the same block of code a specified number of times (for example, five times).
for (init counter; test counter; increment counter) { code to be executed;
}
A foreach loop is used to read an array.
foreach (array_expression as $value)
{ statement}

Arrays in PHP: An array is traditionally defined as a group of items that share certain characteristics. Each item consists of two components; the key and a value. Key is the index of the item in the array and value is the value of the item. PHP doesn't require that you assign a size or type to an array at creation time.
Declaring an array: PHP doesn't even require that you declare the array before using it, although you're free to do so. We just add items to the array. We can add an item to a list as
$array-name[index of the element]= value;
For example we can start and add an item in an array as given below
$players[0] = 'Muhammad Yousuf';
We can add more items in the similar way but use a different index
$players[1]="Ricky Ponting";
Accessing an array: we can read an array item just by entering the array name and the index of the item. For example if we want to read and display the value of second item in the $players array we can do this as given below
echo $players[1];

Associative array: PHP allow us to create arrays where items are declared by name instead of index or key number. Such an array is called an associative array. An item in an associative array can be added as
$array-name[item name] = value For example
$players['yousuf']="Muhammad Yousuf";
Items in associate array are accessed by name. For example, we can read the value of the above array as
echo $players['yousuf'];
array() function: we can also use array function to create an array. By using this function we can add multiple items in one line. The syntax of the array function is
$array_name=array(item_1,item_2,…item_n);
Example is $players=array("M.Yoursuf","Imran Khan");
We can also use array function to create associative arrays such as
$players=array("Yousuf"=>"M.Yoursuf","imran"=>"Imran Khan");

Sorting an array: PHP provides us sort() to sort an array in ascending order while rsort() function to sort an array in descending order.
A function is a block of statements that can be used repeatedly in a program. In PHP, a function can be defined as;
function functionName(list of arguments) { code to be executed;
}
To call the function, just write its name along with the required arguments.

2) Solved Lab Activities

Sr. No	*Time Allocated*	*Complexity*	*CLO-Mapping*
Activity 1	***15 Minutes***	***Medium***	***CLO-4***
Activity 2	***15 Minutes***	***Medium***	***CLO-4***
Activity 3	***15 Minutes***	***Medium***	***CLO-4***
Activity 4	***15 Minutes***	***Medium***	***CLO-4***

Activity 1:

Write the PHP code that creates a table as given below

Subject	**Total Marks**	**Obtained Marks**
Web Engineering	100	80
Database Systems	100	90

Solution:

```
<?php
echo "<table border=1 >
<tr> <th><font color=blue>Subject</th> <th>Total Marks</th><th>Obtained
Marks</font></td></tr>
<tr> <td>Web Engineering</td> <td>100</td><td>80</td></tr>
<tr> <td>Database Systems</td> <td>100</td> <td>90</td></tr>
</table>";
?>
```

Activity 2:

Write the PHP code that creates a table as given below

Write a PHP script to calculate and display the sum, average, and five lowest and highest numbers from the given list.
123, 160, 62, 153, 345, 128, 387, 825, 666, 614, 723, 163, 811, 176, 732, 628, 722, 733, 755, 765,

794, 613, 627

Solution:

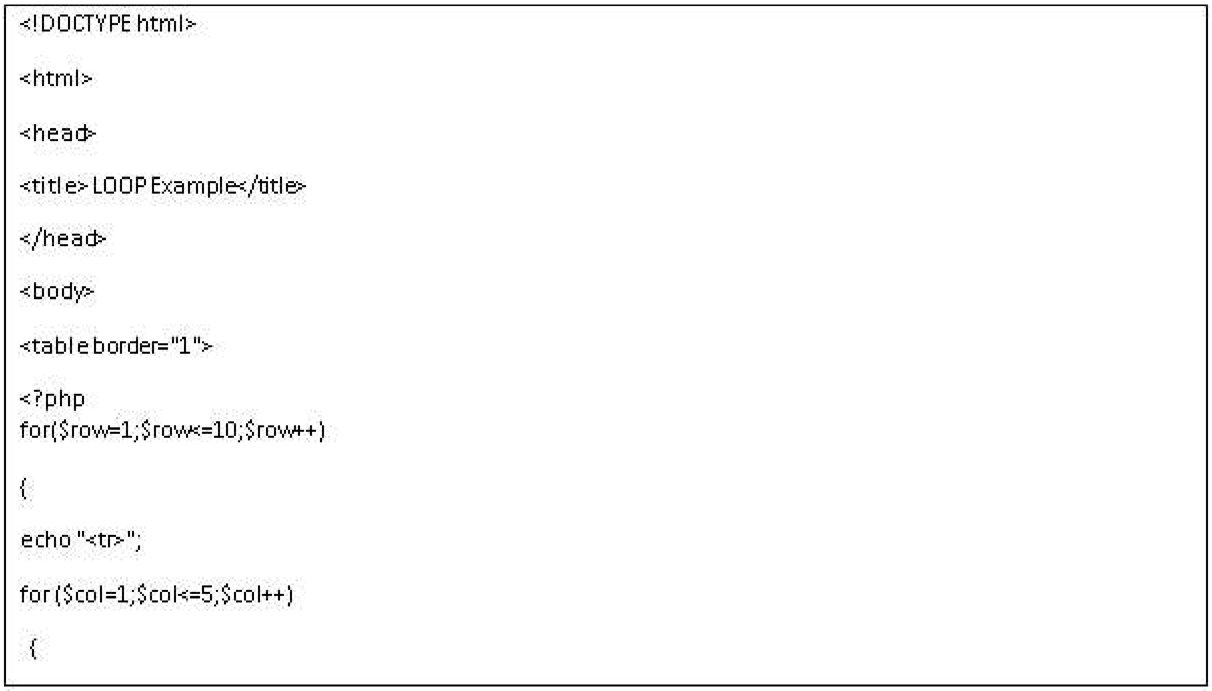
```
<?php
$nums = "123, 160, 62, 153, 345, 128, 387, 825, 666, 614, 723, 163,
811, 176, 732,628, 722, 733, 755, 765, 794, 613,
627";
$nums_array = explode(',', $nums);
$tot_num = 0;
$nums_array_length =
count($nums_array);
foreach($nums_array as
$num)
{
 $tot_num += $num;
}
echo "Sum of Number is : ".$tot_num."
<br>";
$avg_num =
 $tot_num/$nums_array_len
 gth; echo "Average Number
 is : ".$avg_num."
<br>";
sort($nums
_array);
echo "
 List of Five Lowest
Numbers : <br>"; for
($i=0; $i< 5; $i++)
{
echo $nums_array[$i].", ";
}
echo "<br>
List of Five Highest Numbers : <br>";
for ($i=($nums_array_length-5); $i< ($nums_array_length); $i++)
{
echo $nums_array[$i].", ";
}
?>
```

Activity 3:

Write a PHP script to create the following table (using looping statement).

1+1=2	1+2=3	1+3=4	1+4=5	1+5=6
2+1=3	2+2=4	2+3=5	2+4=6	2+5=7
3+1=4	3+2=5	3+3=6	3+4=7	3+5=8
4+1=5	4+2=6	4+3=7	4+4=8	4+5=9
5+1=6	5+2=7	5+3=8	5+4=9	5+5=10
6+1=7	6+2=8	6+3=9	6+4=10	6+5=11
7+1=8	7+2=9	7+3=10	7+4=11	7+5=12
8+1=9	8+2=10	8+3=11	8+4=12	8+5=13
9+1=10	9+2=11	9+3=12	9+4=13	9+5=14
10+1=11	10+2=12	10+3=13	10+4=14	10+5=15

Solution:



```
<!DOCTYPE html>

<html>

<head>

<title> LOOP Example</title>

</head>

<body>

<table border="1">

<?php
for($row=1;$row<=10;$row++)

{

echo "<tr>";

for ($col=1;$col<=5;$col++)

 {
```

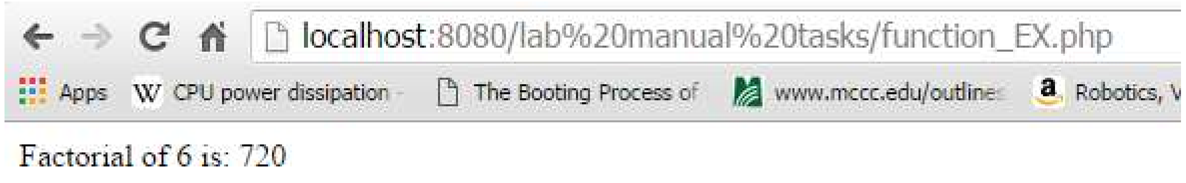
Activity 4:
Write a PHP function that gets a number as an argument and calculates and displays its factorial.

Solution:

```
<?php
function findFact($n)
{
if($n<0)
echo "Please enter a positive number";

 elseif($n ==0)
  {
        return 1;
  }
 else
  {
        return $n * findFact($n-1);
  }
       }
       $num=6;
echo "Factorial of $num is: ".$factorial=findFact($num);
?>
```

Output:



```
Factorial of 6 is: 720
```

3) Graded Lab Tasks

Note: The instructor can design graded lab activities according to the level of difficult and complexity of the solved lab activities. The lab tasks assigned by the instructor should be evaluated in the same lab.

<u>Lab Tasks</u>

1. *Write a PHP program that finds the sum of all prime numbers from an array.*
2. *Write a nested for loop in the PHHP that prints the following output:*

```
                    1
                 1  2   1
              1  2  4   2 1
           1  2  4  8   4 2  1
        1  2 4  8  16  8  4  2 1
      1 2 4  8 16  32 16  8  4 2 1
    1 2 4 8 16 32  64 32 16  8 4 2 1
  1 2 4 8 16 32 64 128 64 32 16 8 4 2 1
```