Experiment 06

Aim: Develop a web page using PHP functions.

Learning Objective: Students should be able to develop a web page using PHP.

Tools: XAMPP, IDE(VS Code / Sublime Text3 / Notepad++), Web Browser.

Theory:

PHP

PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages. PHP can create, open, read, write, delete, and close files on the server, collect form data, add, delete, modify data in your database.

Set Up PHP

1. install a web server
2. install PHP
3. install a database, such as MySQL

   The official PHP website (PHP.net) has installation instructions for PHP: http://php.net/manual/en/install.php

Basic PHP Syntax:

A PHP script starts with <?php and ends with ?>:

```
<?php
// PHP code goes here
?>
```

The default file extension for PHP files is ".php".A PHP file normally contains HTML tags, and some PHP scripting code.

```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

PHP Variables :

In PHP, a variable starts with the $ sign, followed by the name of the variable.

```
<?php
$txt = "Hello world!";
$x = 5;
$y = 10.5;
?>
```

Rules for PHP variables:

- A variable starts with the $ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive ($age and $AGE are two different variables)

PHP Data Types:
PHP supports the following data types:

- String
- Integer
- Float (floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

PHP Conditional Statements:

In PHP we have the following conditional statements:

1. **if statement** - executes some code if one condition is true.
   Syntax:
   ```
   if (condition) {
     code to be executed if condition is true;
   }
   ```
   Example :
   ```
   <?php
   $t = date("H");
   if ($t < "20")
   {
         echo "Have a good day!";
   }
   ?>
   ```

2. **if...else statement** - executes some code if a condition is true and another code if that condition is false.

   Syntax:
```
if (condition) {
  code to be executed if condition is true;
} else {
  code to be executed if condition is false;
}
```
   Example :
```
$t = date("H");

if ($t < "20") {
  echo "Have a good day!";
} else {
  echo "Have a good night!";
}
?>
```

3. **if...elseif...else statement** - executes different codes for more than two conditions.

   Syntax:
```
if (condition) {
  code to be executed if this condition is true;
} elseif (condition) {
    code to be executed if first condition is false and this
condition is true;
} else {
  code to be executed if all conditions are false;
}
```
   Example :
```
<?php
$t = date("H");

if ($t < "10") {
  echo "Have a good morning!";
} elseif ($t < "20") {
  echo "Have a good day!";
} else {
  echo "Have a good night!";
}
?>
```

PHP Loops

In PHP, we have the following loop types:

1. **while** - loops through a block of code as long as the specified condition is true.

   Syntax:
```
while (condition is true) {
  code to be executed;
}
```

Example :
```
<?php
$x = 1;

while($x <= 5) {
  echo "The number is: $x <br>";
  $x++;
}
?>
```

2. **do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true.
   Syntax:
```
do {
  code to be executed;
} while (condition is true);
```
   Example :
```
<?php
$x = 1;

do {
  echo "The number is: $x <br>";
  $x++;
} while ($x <= 5);
?>
```
3. **for** - loops through a block of code a specified number of times.
   Syntax:
```
for (init counter; test counter; increment counter) {
  code to be executed for each iteration;
}
```
   Example :
```
<?php
for ($x = 0; $x <= 10; $x++) {
  echo "The number is: $x <br>";
}
?>
>
```
4. **foreach** - loops through a block of code for each element in an array.
   Syntax:
```
foreach ($array as $value) {
  code to be executed;
}
```
   Example :
```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
  echo "$value <br>";
}
```

```
?>
```

PHP Functions

PHP has over 1000 built-in functions that can be called directly, from within a script, to perform a specific task.
Please check PHP reference for a complete overview of the PHP built-in functions.

A user-defined function declaration starts with the word function:

Syntax:

```
function functionName() {
  code to be executed;
}
```

Note: A function name must start with a letter or an underscore. Function names are NOT case-sensitive.
Example :

```
<?php
function writeMsg() {
  echo "Hello world!";
}

writeMsg(); // call the function
?>
```

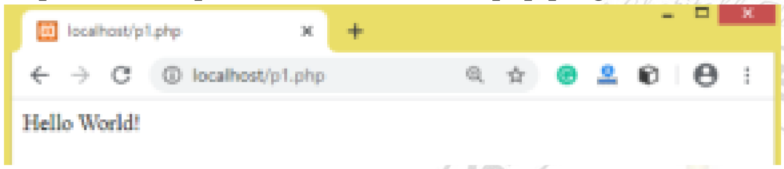
PHP with HTML

```
<!DOCTYPE html>
<html>
<head>
<title>How to put PHP in HTML - foreach Example</title>
</head>
<body>
<?php
$employees = array('John', 'Michelle', 'Mari', 'Luke', 'Nellie');
$total = count($employees);
?>
<h1>List of Employees</h1>
<ul>
<?php
$i = 0;
?>
<?php while ($i < $total) { ?>
<li><?php echo $employees[$i] ?></li>
<?php ++$i ?>
<?php } ?>
```

```
    </ul>
    </body>
    </html>
```

How to run PHP programs in XAMPP

Step 1: Create a simple PHP program like hello world.
Step 2: Save the file with hello.php name in the htdocs folder, which resides inside the xampp folder. (Eg: "C:\xampp\htdocs")
Step 3: Run the XAMPP server and start Apache and MySQL.
Step 4: Now, open the web browser and type localhost http://localhost/hello.php on your browser window.
Step 5: The output for the above hello.php program will be shown as the screenshot below:



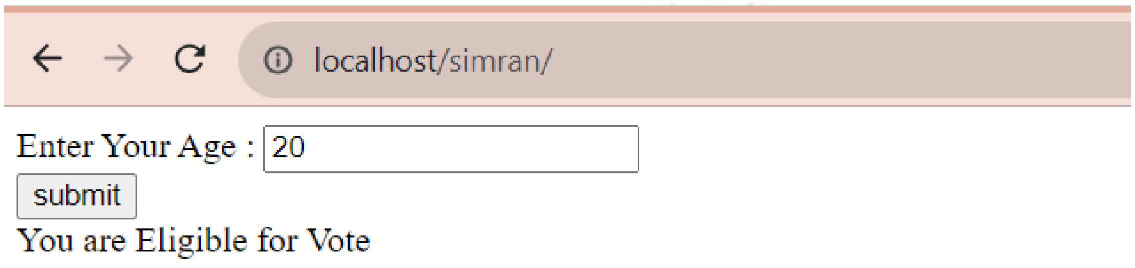
Hello World!

<u>Programs :</u>

<u>Program 1:</u>

Write a PHP program using a function to check if a person is eligible to vote.

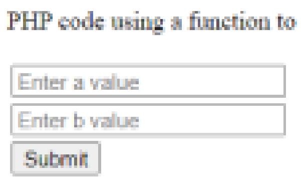
```
<!DOCTYPE html>
<html>
<head>
<title>Eligible to Vote or not</title>
</head>
<body>
<form action="" method="post">
Enter Your Age :
<input type="text" name="t1" placeholder="Enter a
number">
<br><input type="submit" name="submit"
value="submit">
</form>
<?php
if (isset($_POST['submit'])) {
vote();
}
function vote() {
$a = $_POST['t1'];
intval($a);
if($a>=18){
echo "You are Eligible for Vote";
}
else{
echo "You are not Eligible for Vote";
}
}
?>
</body>
</html>
```



Enter Your Age : 20
submit
You are Eligible for Vote

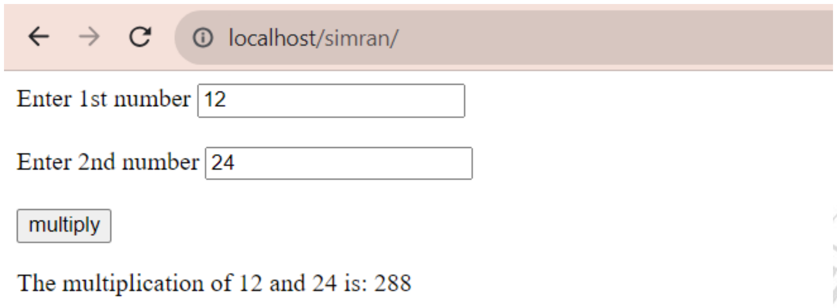
Program 2:

Write a PHP code using a function to find a product of 2 numbers.


PHP code using a function to find product of 2 numbers.

```
<form method = "post">
Enter 1st number
<input type = "number" name="number1"/><br><br>
Enter 2nd number
<input type = "number" name="number2"/><br><br>
<input type = "submit" name="submit" value="multiply"/><br>
</form>
<?php
if(isset($_POST['submit']))
{
    $number1=$_POST['number1'];
    $number2=$_POST['number2'];
    $mul=$number1 * $number2;
    echo "The multiplication of $number1 and $number2 is: ".$mul;

}
?>
```



```
localhost/simran/
Enter 1st number 12
Enter 2nd number 24
multiply
The multiplication of 12 and 24 is: 288
```

Result and discussion: We successfully illustrated use of functions in PHP.

Learning Outcomes: Students should have the ability to

LO1: To process input taken from the user with PHP functions.

LO2: To build a simple webpage using PHP functions.

Conclusion:

For Faculty Use

Correction Parameters	Formative Assessment	Timely completion of Practical	Attendance / Learning Attitude	
Marks Obtained				