

Core – PHP Assignment

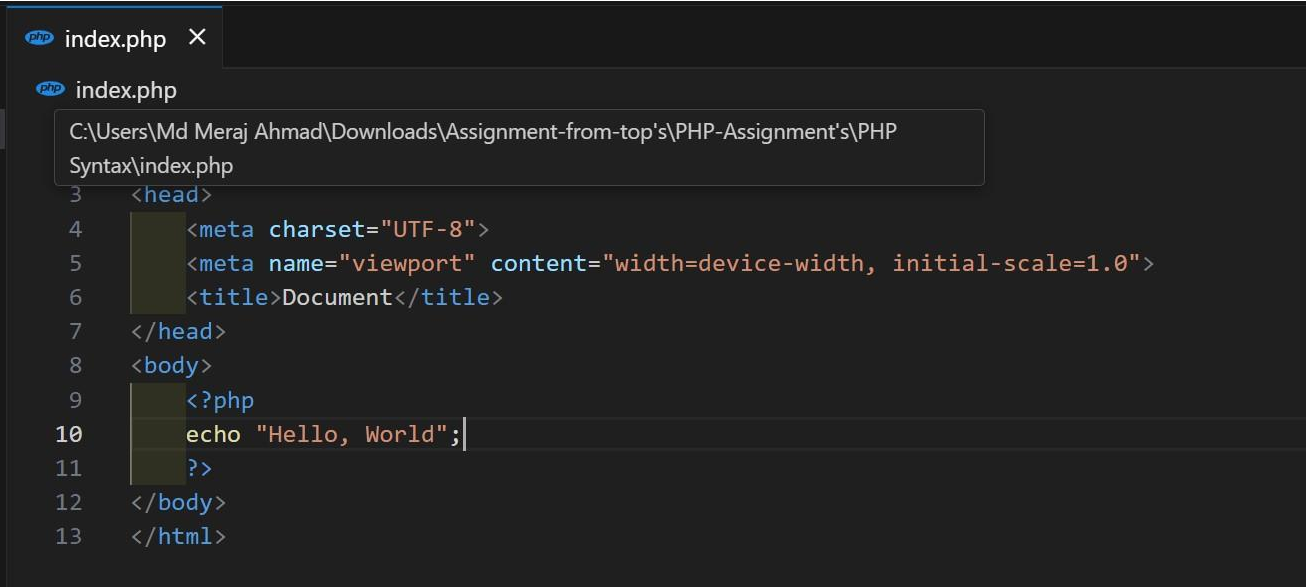
PHP Syntax

THEORY EXERCISE:

Q1. Discuss the structure of a PHP script and how to embed PHP in HTML?

Structure of a PHP script and embedding PHP in HTML:

- PHP scripts are written inside `<?php ?>` tags.
- PHP can be embedded within HTML to create dynamic content. Anything outside the PHP tags is treated as HTML.
- `echo` also has a shortcut syntax, which lets you immediately print a value.



```
index.php X
index.php
C:\Users\Md Meraj Ahmad\Downloads\Assignment-from-top's\PHP-Assignment's\PHP
Syntax\index.php
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Document</title>
7 </head>
8 <body>
9 <?php
10 echo "Hello, World";|
11 ?>
12 </body>
13 </html>
```

- The PHP code inside `<?php ?>` will be executed on the server, and the output will be inserted into the HTML content sent to the browser.

Output:

Hello, World

Q2. What are the rules for naming variables in PHP?

Rules for naming variables in PHP:

- Variables must begin with a \$ sign, followed by the name of the variable.
- The variable name must start with a letter or an underscore (_), followed by letters, numbers, or underscores.
- PHP variables are case-sensitive (e.g., \$Var is different from \$var).
- Variable names cannot contain spaces or special characters (other than underscores).
- Example of valid variables: \$name, \$age_2, \$user_name.

LAB EXERCISE:

- Write a PHP scripts to print “Hello, World!” on web page.

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

PHP Variables

THEORY EXERCISE:

Q1. Explain the concept of variables in PHP and their scope.

Concept of variables in PHP and their scope:

- Variables are used to store data in a memory location. Variables can store various types of data, such as integers, strings, arrays, and objects.

Scope: A variable's scope refers to where it can be accessed. There are three main types of variable scope in PHP:

- **Local scope:** Defined within a function and accessible only within that function.
- **Global scope:** Defined outside of functions and accessible globally in the script.
- **Static scope:** Variables defined with the static keyword retain their value across function calls.

LAB EXERCISE:

- Create a PHP script to declare and initialize different types of variables (integer, float, string, Boolean) and display them using echo.

```
variables.php X
PHP Variables > variables.php
1  <?php
2      $intVar = 10;
3      $floatVar = 10.5;
4      $stringVar = "Hello, World!";
5      $boolVar = true;
6
7      echo $intVar . "<br>";
8      echo $floatVar . "<br>";
9      echo $stringVar . "<br>";
10     echo $boolVar ? "True" : "False";
11 ?>
12 |
```

Output:

10

10.5

Hello, World!

True

Super Global Variables

THEORY EXERCISE:

Q1. What are super global variables in PHP? List at least five super global arrays and their use.?

Super global variables in PHP:

- ❖ Super global variables are built-in global arrays in PHP that are always accessible, regardless of scope. Some common super global arrays are:
 - `$_GET` – Used to collect form data after submitting an HTML form with `method="get"`.
 - `$_POST` – Used to collect form data after submitting an HTML form with `method="post"`.
 - `$_REQUEST` – Contains the contents of both `$_GET` and `$_POST`.
 - `$_SESSION` – Used to store session variables.
 - `$_COOKIE` – Used to read or set cookies.

LAB EXERCISE:

- Create a form that takes a user's name and email. Use the `$_POST` super global to display the entered data.

```
super_global_variables.php X
Super global variables > super_global_variables.php
1
2 <form method="post" action="">
3     Name: <input type="text" name="name">
4     Email: <input type="email" name="email">
5     <input type="submit" value="Submit" style="margin-left: 5px;">
6 </form>
7
8 <?php
9     if ($_SERVER["REQUEST_METHOD"] == "POST") {
10         echo "Name: " . $_POST['name'] . "<br>";
11         echo "Email: " . $_POST['email'];
12     }
13 ?>
14
```

Output:

Name: Email:

Name: md meraj ahmad
Email: hattorihanjo123@gmail.com

Conditions, Events, and Flows

THEORY EXERCISE:

- **Explain how conditional statements work in PHP:**
 - Conditional statements allow you to perform different actions based on different conditions. Common conditional statements are if, else, elseif, and switch.

Example:

```
condition_event_and_flow.php X
Conditions, Events, and Flows > condition_event_and_flow.php
1  <?php
2  $num = 10;
3  if ($num > 0) {
4      echo "Positive number";
5  } elseif ($num < 0) {
6      echo "Negative number";
7  } else {
8      echo "Zero";
9  }
10 ?>
```

Output:

Positive number

If Condition and If-Else If

LAB EXERCISE:

- ❖ **Write a PHP program to determine if a number is even or odd using if conditions.**

Example:

```
<?php
$num = 15;
if ($num % 2 == 0) { echo "$num is
    even.";
} else {
    echo "$num is odd.";
}
?>
```

Output:

15 is odd

Practical Example: Calculator and Day Finder

LAB EXERCISE:

❖ Simple Calculator:

- Create a calculator using if-else conditions that takes two inputs and an operator (+, -, *, /).

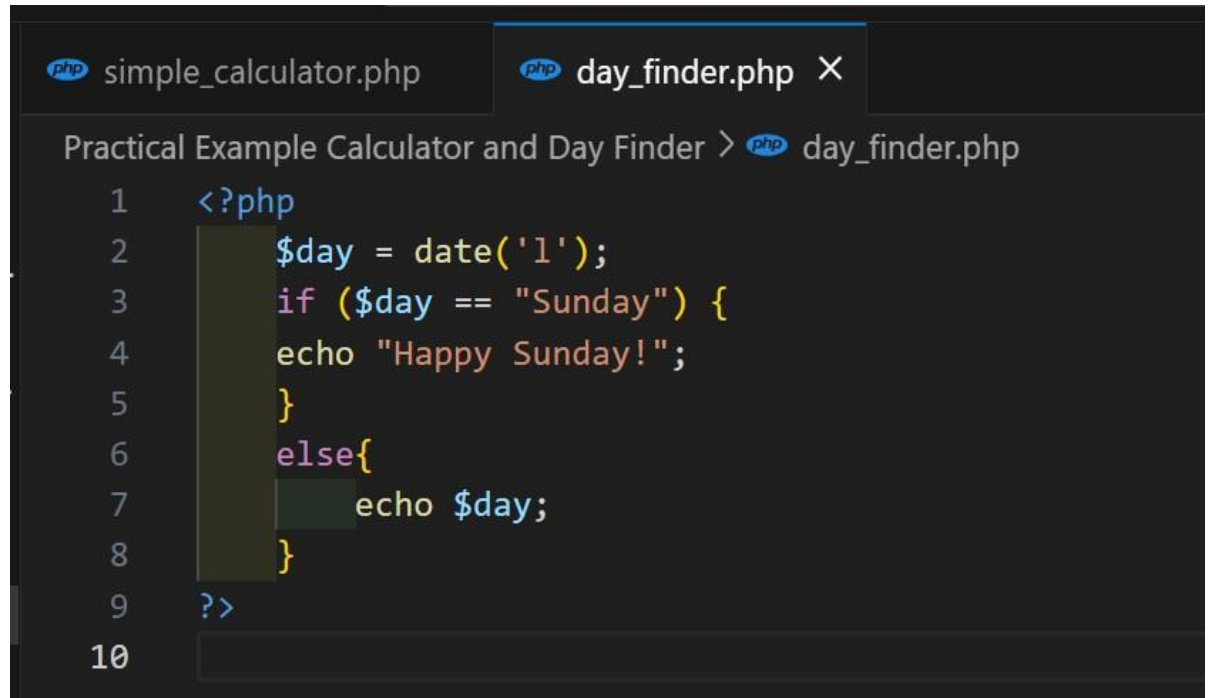
Example:

```
simple_calculator.php × day_finder.php
Practical Example Calculator and Day Finder > simple_calculator.php
1  <?php
2      $num1 = 10;
3      $num2 = 5;
4      $operator = '+';
5
6      if ($operator == '+') {
7          echo $num1 + $num2;
8      } elseif ($operator == '-') {
9          echo $num1 - $num2;
10     } elseif ($operator == '*') {
11         echo $num1 * $num2;
12     } elseif ($operator == '/') {
13         echo $num1 / $num2;
14     }
15  ?>
16
```

❖ **Day Finder:**

- Write a script that finds the current day. If it is Sunday, print “HappySunday”.

Example:

A screenshot of a code editor with two tabs at the top: 'simple_calculator.php' and 'day_finder.php'. The 'day_finder.php' tab is active. The editor shows a PHP script for a day finder. The script starts with a PHP opening tag, then assigns the current day to a variable. It uses an if-else statement to check if the day is Sunday. If it is, it echoes 'Happy Sunday!'. Otherwise, it echoes the day of the week. The script ends with a closing PHP tag. Line numbers 1 through 10 are visible on the left side of the editor.

```
1 <?php
2     $day = date('l');
3     if ($day == "Sunday") {
4         echo "Happy Sunday!";
5     }
6     else{
7         echo $day;
8     }
9     ?>
10
```

Switch Case and Ternary Operator

LAB EXERCISE:

1. Restaurant Food Category Program:

- Use a switch case to display the category (Starter/Main Course/Dessert) and dish based on user selection.

Example:

```
01_Restaurant_Food_Category_Program.php ×

Switch Case and Ternary Operator > 01_Restaurant_Food_Category_Program.php

1 <?php
2 $foodCategory = "Dessert";
3 switch ($foodCategory) {
4     case "Starter":
5         echo "Salad";
6         break;
7     case "Main Course":
8         echo "Pasta";
9         break;
10    case "Dessert":
11        echo "Cake";
12        break;
13    default:
14        echo "Invalid category";
15    }
16 ?>
17
```

2. Ternary Operator Example:

- Write a script using the ternary operator to display a message if the age is greater than 18.

Example:



The screenshot shows a code editor with a dark background. At the top, there is a tab labeled '02_Ternary_Operator_Example.php' with a small 'php' icon and a close button. Below the tab, the editor content is titled 'Switch Case and Ternary Operator > 02_Ternary_Operator_Example.php'. The code is as follows:

```
1 <?php
2     $age = 20;
3     echo $age > 18 ? "Adult" : "Minor";
4 ?>
5
```

3. Color Selector:

- Write a program to display the name of a color based on user input (red, green, blue).

Example:

```
03_Color_Selector.php X
Switch Case and Ternary Operator > 03_Color_Selector.php
1  <?php
2      $color = "red";
3      switch ($color) {
4          case "red":
5              echo "You chose Red";
6              break;
7          case "green":
8              echo "You chose Green";
9              break;
10         case "blue":
11             echo "You chose Blue";
12             break;
13         default:
14             echo "Unknown color";
15     }
16  ?>
17
```

Loops Do-While, For Each, For Loop

THEORY EXERCISE:

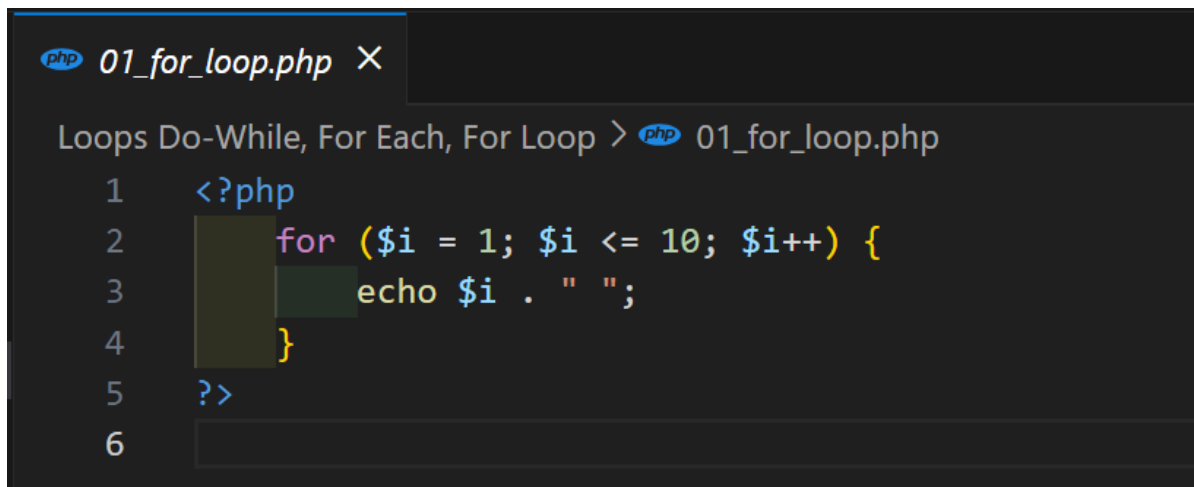
- **Difference between for loop, foreach loop, and do-while loop in PHP:**
 - **For Loop:** Used when you know the number of iterations in advance.
 - **Foreach Loop:** Used to iterate over arrays.
 - **Do-While Loop:** Executes at least once and then repeats based on the condition.
 - **Do-While Loop:** Executes at least once and then repeats based on the condition.

LAB EXERCISE:

1. For Loop:

- Write a script that displays numbers from 1 to 10 on a single line.

Example:

A screenshot of a code editor with a dark background. At the top, there is a tab labeled 'php 01_for_loop.php' with a close button 'X'. Below the tab, the editor shows the title 'Loops Do-While, For Each, For Loop' followed by a prompt '>' and the file name 'php 01_for_loop.php'. The code is written in PHP and is as follows:

```
1  <?php
2      for ($i = 1; $i <= 10; $i++) {
3          echo $i . " ";
4      }
5  ?>
6
```

2. For Loop (Addition):

- Add all integers from 0 to 30 and display the total.

Example:

```
php 02_for_loop_addition.php X
Loops Do-While, For Each, For Loop > php 02_for_loop_addition.php
1  <?php
2      $total = 0;
3      for ($i = 0; $i <= 30; $i++) {
4          $total += $i;
5      }
6      echo "Total: $total";
7  ?>
8
```

3. Chessboard Pattern:

- Use a nested loop to create a chessboard pattern (8x8 grid).

Example:

```
php 03_chessboard_pattern.php X
Loops Do-While, For Each, For Loop > php 03_chessboard_pattern.php
1  <?php
2      for ($i = 0; $i < 8; $i++) {
3          for ($j = 0; $j < 8; $j++) {
4              if (($i + $j) % 2 == 0) {
5                  echo "X ";
6              } else {
7                  echo "0 ";
8              }
9          }
10     echo "<br>";
11 }
12 ?>
```

4. Various Patterns:

- Generate different patterns using loops.

Example:

```
php Various_Patterns.php X
Loops_Do-While_For_Each_For_Loop > php Various_Patterns.php
1  <?php
2  echo "\nPyramid Pattern:\n";
3  $n = 5;
4  for ($i = 1; $i <= $n; $i++) {
5      for ($j = $i; $j < $n; $j++) {
6          echo " ";
7      }
8      for ($k = 1; $k <= (2 * $i - 1); $k++) {
9          echo "* ";
10     }
11     echo "\n";
12 }
13 ?>
```


PHP Array and Array Functions

THEORY EXERCISE:

- **Arrays in PHP:**

- Arrays are used to store multiple values in a single variable. There are three types of arrays:
 1. **Indexed arrays** – Arrays with numeric indices.
 2. **Associative arrays** – Arrays with named keys.
 3. **Multidimensional arrays** – Arrays containing other arrays.

LAB EXERCISE:

1. **Display the value of an array:**

Example:

```
php 01_Display_the_values_of_an_array.php X
PHP_Array_and_Array_Functions > php 01_Display_the_values_of_an_array.php
1  <?php
2  // $_array = [1, 2, 3, 4, 5, 6];
3  $_array = array(1, 2, 3, 4, 5, 6);
4  print_r($_array);
5  ?>
```

2. Find and display the number of odd and even elements in an array:

Example:

```
Find_and_display_the_number_of_odd_and_even_elements_in_an_array.php X
PHP_Array_and_Array_Functions > Find_and_display_the_number_of_odd_and_e
1  <?php
2  $x = [1, 2, 3, 4, 5, 6, 7];
3      $odd = 0;
4      $even = 0;
5      foreach ($x as $num){
6          if ($num % 2 == 0) {
7              $even++;
8          }else{
9              $odd++;
10         }
11     }
12     echo "Odd: $odd, Even: $even";
13 ?>
```

3. Create an associative array for user details (name, email, age) and display them:

Example:

```
03_Create_an_associative_array_for_user_details_name_email_age_and_display_them.php X
PHP_Array_and_Array_Functions > 03_Create_an_associative_array_for_user_details_name_email_age_and_display_
1  <?php
2      $user = ["name" => "Meraj", "email" => "hattorihanjo123.com", "age" => 22];
3      echo "Name: " . $user["name"] . "<br>";
4      echo "Email: " . $user["email"] . "<br>";
5      echo "Age: " . $user["age"];
6  ?>
7
```

4. Shift all zero values to the bottom of an array:

Example:

```
04_Shift_all_zero_values_to_the_bottom_of_an_array.php X
PHP_Array_and_Array_Functions > 04_Shift_all_zero_values_to_the_bottom_of_an_array.php
1  <?php
2      $x = [0, 1, 2, 0, 3, 0, 4];
3      $x = array_filter($x, fn($value) => $value != 0);
4      $x = array_merge($x, array_fill(0, 3, 0));
5      print_r($x);
6  ?>
7
```

PHP Date-Time Function

THEORY EXERCISE:

- **PHP Date-Time Function:** The date() function in PHP is used to format the current date and time based on a format string. It can display different parts of the date or time, such as year, month, day, hour, minute, second, etc.

Lab Exercise:

- What a script to display the current date and time in different formats

Example:

```
current_date-time_in_diff_formats.php X
11_PHP_Date-Time_Function > current_date-time_in_diff_formats.php
1  <?php
2  // Current date in Y-m-d format
3  echo "Current Date (Y-m-d): " . date("Y-m-d") . "\n";
4
5  // Current time in H:i:s format
6  echo "Current Time (H:i:s): " . date("H:i:s") . "\n";
7
8  // Full date and time
9  echo "Full Date and Time: " . date("l, F j, Y g:i A") . "\n";
10
11 // Unix timestamp
12 echo "Unix Timestamp: " . time() . "\n";
13
14 // Date and time in custom format
15 echo "Custom Format (d/m/Y H:i): " . date("d/m/Y H:i") . "\n";
16 ?>
17
```

Header Function

THEORY EXERCISE:

- **PHP Header Function:** The `header()` function in PHP sends raw HTTP headers to the browser. It is commonly used for things like redirecting users to another page or controlling cache settings.

LAB EXERCISE:

Include and Require

THEORY EXERCISE:

- **Difference between include and require in PHP:**
 - **Include:** If the file cannot be found or loaded, it will emit a warning, but the script will continue executing.
 - **Require:** If the file cannot be found or loaded, it will emit a fatal error and stop script execution.

LAB EXERCISE:

- Use include and require to insert common header and footer file into multiple PHP pages.

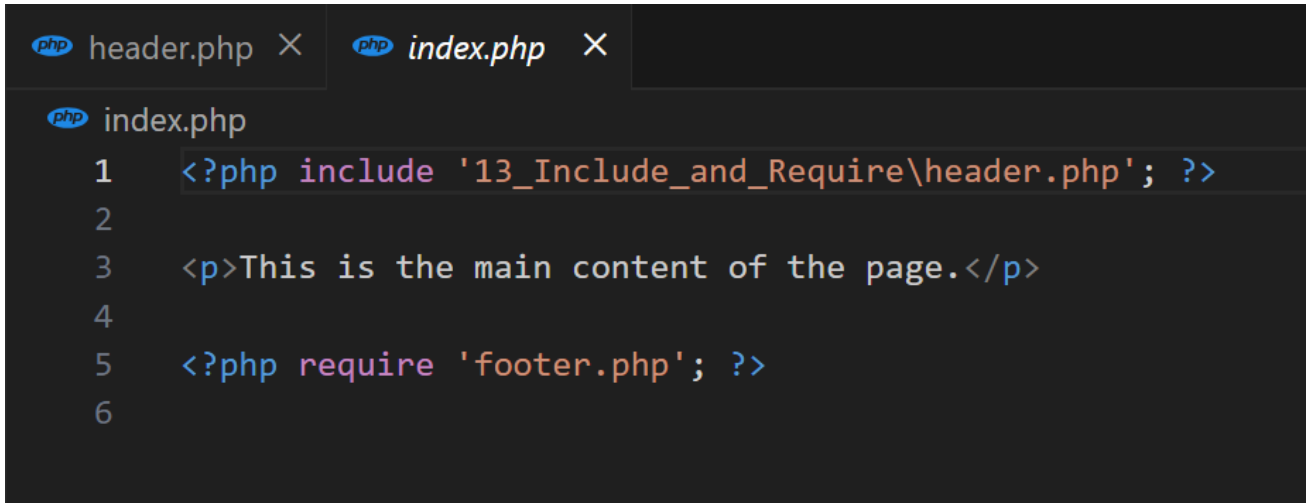
Example: Header.php

```
php header.php X
13_Include_and_Require > php header.php
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=\, initial-scale=1.0">
6      <title>My Website</title>
7  <style>
8      h1{
9          color: blue;
10     }
11 </style>
12 </head>
13 <body>
14     <h1>Welcome To My Website</h1>
15 </body>
16 </html>
```

Example: Footer.php

```
php header.php  php footer.php X
php footer.php
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <footer>© 2024 Hattori Hanjo</footer>
10 </body>
11 </html>
```

Example: Index.php



```
php header.php ×  php index.php ×  
php index.php  
1  <?php include '13_Include_and_Require\header.php'; ?>  
2  
3  <p>This is the main content of the page.</p>  
4  
5  <?php require 'footer.php'; ?>  
6
```

Practical Example: Calculator, Factorial, String Reverse

LAB EXERCISE:

1. **Calculator:** Create a calculator using user-defined functions:

Example: Calculator

```
01_Create-a-calculator-using-user-define-function.php X
14_Practical_Example__Calculator_Factorial_String_Reverse > 01_Create-a-calculator-using-user-define-function.php
1  <?php
2  function add($a, $b) {
3      return $a + $b;
4  }
5
6  function subtract($a, $b) {
7      return $a - $b;
8  }
9
10 function multiply($a, $b) {
11     return $a * $b;
12 }
13
14 function divide($a, $b) {
15     if ($b == 0) {
16         return "Cannot divide by zero";
17     } else {
18         return $a / $b;
19     }
20 }
21
22 echo "Addition: " . add(5, 3) . "<br>";
23 echo "Subtraction: " . subtract(5, 3) . "<br>";
24 echo "Multiplication: " . multiply(5, 3) . "<br>";
25 echo "Division: " . divide(5, 3) . "<br>";
26 ?>
27
```


2. **Factorial:** Write a function that finds the factorial of a number using recursion:

Example: Factorial

```
php 02_Write-a-function-that-find-the-factorial-number.php ×
14_Practical_Example__Calculator_Factorial_String_Reverse > php 02_Write-a-fun
1  <?php
2  function factorial($n) {
3      if ($n == 0) {
4          return 1;
5      } else {
6          return $n * factorial($n - 1);
7      }
8  }
9
10 echo "Factorial of 5 is: " . factorial(5);
11 ?>
12
```

3. **String Reverse:** Reverse a string without using built-in functions:

Example: String Reverse

```
03_Reverse-a-string-without-using-bulit-in-function.php ×
14_Practical_Example__Calculator_Factorial_String_Reverse > 03_Reverse-a-string-witho
1  <?php
2  function reverseString($str) {
3      $reversed = '';
4      for ($i = strlen($str) - 1; $i >= 0; $i--) {
5          $reversed .= $str[$i];
6      }
7      return $reversed;
8  }
9
10 echo "Reversed string: " . reverseString("Hello World");
11 ?>
12
```

4. **Download File:** Create a button that allows users to download a file:

Example: Download File

```
04_create-button-that-allow-user-to-download-file.php X
14_Practical_Example__Calculator_Factorial_String_Reverse > 04_create-button-that-allow-user-to-download-file.php
1  <?php
2  if (isset($_POST['download'])) {
3      $file = 'path/to/your/file.txt'; // specify file path
4      if (file_exists($file)) {
5          header('Content-Description: File Transfer');
6          header('Content-Type: application/octet-stream');
7          header('Content-Disposition: attachment; filename="' . basename($file) . '"');
8          header('Content-Length: ' . filesize($file));
9          readfile($file);
10         exit;
11     }
12 }
13 ?>
14
15 <form method="post">
16     <button type="submit" name="download">Download File</button>
17 </form>
```

PHP Expressions, Operations, and String Functions

Theory Exercise:

- **PHP Expressions:** PHP expressions are combinations of variables, operators, and values that PHP can evaluate to produce a result. Examples include:
 - Arithmetic Operations: +, -, *, /
 - Logical Operations: &&, ||, !

LAB EXERCISE:

- Write a scripts to perform various string operations like concatenation, substring extraction, and string length determination.

Example:

```
04_create-button-that-allow-user-to-download-file.php X
14_Practical_Example_Calculator_Factorial_String_Reverse > 04_create-button-that-allow-user-to-download-file.php
1  <?php
2  if (isset($_POST['download'])) {
3      $file = 'path/to/your/file.txt'; // specify file path
4      if (file_exists($file)) {
5          header('Content-Description: File Transfer');
6          header('Content-Type: application/octet-stream');
7          header('Content-Disposition: attachment; filename="' . basename($file) . '"');
8          header('Content-Length: ' . filesize($file));
9          readfile($file);
10         exit;
11     }
12 }
13 ?>
14
15 <form method="post">
16     <button type="submit" name="download">Download File</button>
17 </form>
```