

# 2023

PHP Logbook



NAME

Stuart Jeetoo  
2210\_22803

BATCH

BSE21BFT2

DATE SUBMITTED

??-??-2023

Server-Side scripting with

# Web Services

## WAT2124C

## Contents

<b>INTRODUCTION .....</b>	<b>2</b>
<b>CREATING DYNAMIC WEBSITES WITH PHP .....</b>	<b>1</b>
<b>PART I.....</b>	<b>1</b>
<b>SERVER(S) AND DATABASE SYSTEM(S) SUPPORTS PHP? .....</b>	<b>2</b>
<b>ON WHICH PORT NUMBER DOES A WEB SERVER OPERATE? .....</b>	<b>2</b>
<b>PART II.....</b>	<b>2</b>
<b>DESCRIBE YOUR INSTALLATION AND CONFIGURATION OF XAMPP SERVER .....</b>	<b>2</b>
<b>TUTORIAL 1(A): VARIABLES, DATA TYPES &amp; OPERATORS.....</b>	<b>5</b>
<b>LAB EXERCISE 1: VARIABLES, DATA TYPES &amp; OPERATORS .....</b>	<b>6</b>
<b>DIFFICULTIES EXPERIENCED .....</b>	<b>11</b>
<b>LEARNING OUTCOME .....</b>	<b>11</b>
<b>TUTORIAL 2(A): INTRODUCING ARRAYS.....</b>	<b>12</b>
<b>TUTORIAL 2(B): INTRODUCING ARRAYS.....</b>	<b>12</b>
<b>LAB 2: INTRODUCING ARRAYS .....</b>	<b>12</b>
<b>DIFFICULTIES EXPERIENCED .....</b>	<b>17</b>
<b>LEARNING OUTCOME .....</b>	<b>17</b>
<b>LAB3,4:.....</b>	<b>18</b>
<b>LAB EXERCISE 3,4: DIFFICULTIES EXPERIENCED .....</b>	<b>26</b>
<b>LAB EXERCISE 3,4: LEARNING OUTCOME .....</b>	<b>26</b>

## INTRODUCTION

1. PHP file uses extension .php
2. PHP code can be defined using <?php as opening and ?> as closing

Example:

```
<?php
```

```
// PHP code to be executed
```

```
?>
```

3. Each PHP statements should end with a semicolon ( ; ) like in java
4. Keyword like echo, if, else, while, etc... are not case sensitive (example: ECHO, Echo, eCHO, echo is valid)
5. However, all variables name are case sensitive (example: color, Color, cOLOR, etc... is not same)

# Creating Dynamic Websites with PHP

Date:02/05/23

## Part I

List five server side scripting languages and their corresponding servers used to develop dynamic websites (apart from PHP)

1. **Python:**

- Server-Side Scripting Language: Python
- Corresponding Server: Django (framework), Flask (micro-framework)

2. **Ruby:**

- Server-Side Scripting Language: Ruby
- Corresponding Server: Ruby on Rails (Rails)

3. **Java:**

- Server-Side Scripting Language: Java
- Corresponding Server: Apache Tomcat, Jetty, WildFly (formerly known as JBoss), Spring Boot

4. **Node.js:**

- Server-Side Scripting Language: JavaScript (Node.js runtime)
- Corresponding Server: Express.js, Koa.js, Hapi.js

5. **ASP.NET:**

- Server-Side Scripting Language: C# (ASP.NET framework)
- Corresponding Server: Microsoft Internet Information Services (IIS)

## Server(s) and database system(s) supports PHP?

### **Servers:**

1. Apache
2. Nginx
3. LiteSpeed
4. Microsoft Internet Information Services (IIS)
5. Caddy

### **Database Systems:**

1. MySQL
2. MariaDB
3. PostgreSQL
4. SQLite
5. Microsoft SQL Server

## On which port number does a Web Server operate?

Port 80

## Part II

## Describe your installation and configuration of XAMPP Server

### **Download XAMPP:**

- Visit the official XAMPP website (<https://www.apachefriends.org>) and download the appropriate version of XAMPP for your operating system


## What is XAMPP?


XAMPP is the most popular PHP development environment

XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.



 XAMPP for **Windows**  
8.2.4 (PHP 8.2.4)

 XAMPP for **Linux**  
8.2.4 (PHP 8.2.4)

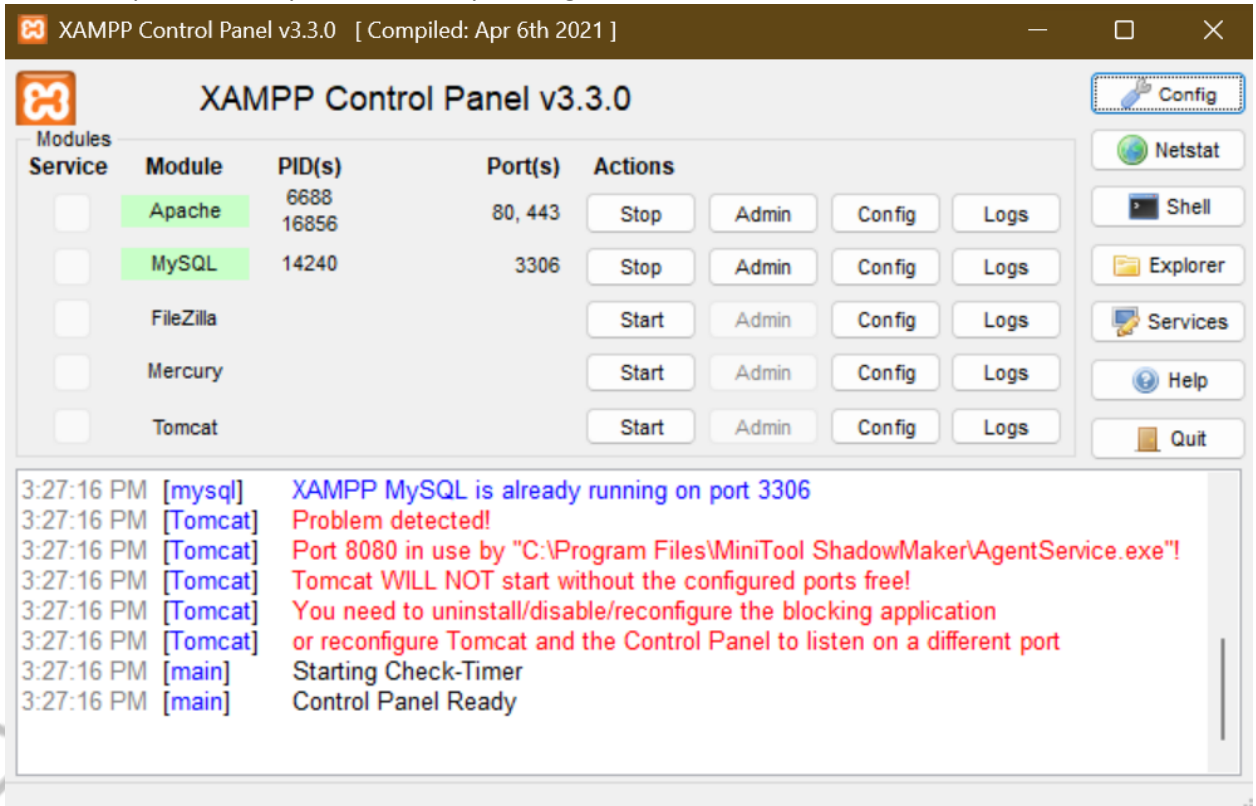
 XAMPP for **OS X**  
8.2.4 (PHP 8.2.4)

### Install XAMPP:

- Run the installer that you downloaded.
- Follow the installation prompts to select components to install. You can choose components like Apache, MySQL, PHP, phpMyAdmin, etc.
- Choose the installation directory

## Start XAMPP:

- After installation, run the XAMPP control panel.
- Start the Apache and MySQL services by clicking the "Start" buttons next to them.



### Test the Installation:

- Open a web browser and navigate to `http://localhost`

## Tutorial 1(a): Variables, Data Types & Operators

**Date:**09/05/23

Q.1

The following are valid variable names:

- \$first\_name
- \$name3
- \$name\_3

The following are invalid variable names:

- \$\_name
- \$5name
- \$name?
- \$first+name
- \$first.name

Q2. Can you use keywords for variable names?

No, keywords cannot be used as variable names and using it will result as a syntax error. It is crucial to choose variables name that are not keywords to ensure that codes execute properly, in order to avoid confusion and make the code more readable



## Lab Exercise 1: Variables, Data Types & Operators

Solution answers/code and screen shots (where applicable) for all exercises in lab 1

Q1:

1. Create the following Table in HTML

Table Heading Cell Spanning 4 Columns			
Normal cell	Cell spanning 2 columns		Normal cell
Cell spanning 3 rows with a gray background	Normal cell	Normal cell	Normal cell
	Normal cell	Cell spanning 2 rows and 2 columns	
	Normal cell		

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="utf-8">
  <title></title>
</head>
<body>
  <table border="1">
    <tr>
      <td colspan="4" style="justify-content:center">Table Heading Cell spanning 4 columns</td>
    </tr>
    <tr>
      <td>Normal cell</td>
      <td colspan="2">Cell spanning 2 columns</td>
      <td></td>
    </tr>
    <tr>
      <td rowspan="3">Cell spanning 3 rows with a gray background</td>
      <td>Normal cell</td>
      <td>Normal cell</td>
      <td>Normal cell</td>
    </tr>
    <tr>
      <td>Normal cell</td>
      <td colspan="2" rowspan="2">Cell spanning 2 rows and 2 columns</td>
    </tr>
    <tr>
      <td>Normal cell</td>
    </tr>
  </table>
</body>
</html>
```

```

<td>Normal cell</td>

</tr>

<tr>

<td style="background-color:grey" rowspan="3">Cell spanning 3 rows with a gray background</td>

<td>Normal cell</td>

<td>Normal cell</td>

<td>Normal cell</td>

</tr>

<tr>

<td>Normal cell</td>

<td rowspan="2" colspan="2">Cell spanning 2 rows and two columns</td>

</tr>

<tr>

<td>Normal cell</td>

</tr>

</table>

</body>

</html>

```

Table Heading Cell spanning 4 columns			
Normal cell	Cell spanning 2 columns		Normal cell
Cell spanning 3 rows with a gray background	Normal cell	Normal cell	Normal cell
	Normal cell	Cell spanning 2 rows and two columns	
	Normal cell		

Q2:

2. Find the errors in this PHP program:

```

<? php
print 'How are you?';
print 'I'm fine.';
??>

```

-<?PHP

Print "How are you?";

Print "I', fine.";

?>

Q3:

3. Which of the following are not valid variables:

a. \$blah

b. \$f11

c. \$\_11f

☒ d. \$11f

e. None of the above

4. What will \$foo be set to in this expression: **\$foo = "wombat" \* 2**?

5. What will \$bar be set to in this expression: **\$bar = 5 \* 5 + 5**?

6. What does the =< operator do?

7. How does **OR** differ from **||**?

Q4:

Wombatwombat

Q5:

30

Q6:

Less than or equal to

Q7:

**||** has a higher precedence than 'or' which means it is taken in consideration before 'or'

Q8:

8. Write a PHP program that computes the total cost of this restaurant meal: two hamburgers at \$4.95 each, one chocolate milk shake at \$1.95, and one cola at 85 cents. The sales tax rate is 15%.

```
<?php
$hamburger = 4.95;

$CMS = 1.95;

$cola = 0.85;

DEFINE("taxrate",0.15);

$total = ($hamburger+$CMS+$cola) *(1+taxrate);

Echo $total;

?>
```

9. Write a PHP program that sets the variable **\$first\_name** to your first name and **\$last\_name** to your last name. Print out a string containing your first and last name separated by a space. Also print out the length of that string.

```
<?php
$FN = "Stuart";
$LN = "Jeetoo";
Echo $FN." ".$LN;
Echo strlen($FN." ".$LN);

?>
```

10. Write a script that creates a variable and assigns an integer value to it, then adds 1 to the variable's value three times, using a different operator each time. Display the final result to the user.

```
<?php
$number = 10;

$number = $number +1;

$number += 1;

$number++;

Echo "final result=".$number;

?>
```

11. Write a script that creates two variables and assigns a different integer value to each variable. Now make your script test whether the first value is

- a) equal to the second value
- b) greater than the second value
- c) less than or equal to the second value
- d) not equal to the second value

And output the result of each test to the user.

a)

```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title></title>
```

```
</head>
```

```
<body>
```

```
<?php
```

```
$num1 = 5;
```

```
$num2 = 2;
```

```
$result="";
```

```
If ($num1==$num2){
```

```
$result = $result."same";
```

```
}
```

```
If ($num1>$num2){
```

```
$result = $result.'greater than second value';
```

```
}
```

```
If ($num1<=$num2){
```

```
$result = $result.'less or equal than second value';
```

```
}
```

```
If ($num1!=$num2){
```

```
$result = $result.'not equal to second value';
```

```
}
```

```
Echo $result;
```

```
?>
```

```
</body>
```

12. For this PHP exercise, write a script using the following variable:

```
$around="around";
```

Single quotes and double quotes don't work the same way in PHP. Using single quotes ('') and the concatenation operator, echo the following to the browser, using the variable you created: **What goes around, comes around.**

```
</html>
```

```
<?php
```

```
$around = "around";
```

```
Echo 'what goes '.$around.', '.'comes'.$around.'.';
```

```
?>
```

## Difficulties Experienced

Explain all the difficulties experienced with this tutorial

No difficulty has been experienced in Lab 1 & Tutorial 1.

## Learning Outcome

Briefing overview of what you have learned in this tutorial

1. Comprehension of how constants are employed.
2. Manipulating strings.
3. Handling numerical data.
4. Implementing accurate syntax in coding.
5. Acquired knowledge concerning variables, adhering to appropriate rules and recommendations for naming variables.
6. Working with operators.

-Working with arrays, sort one or multiple array.

## Tutorial 2(a): Introducing Arrays

```
$tv = array();
```

```
$tv[] = 'Flight of the Eagles';
```

The content of `$tv[0]` is "Flight of the Eagles".

## Tutorial 2(b): Introducing Arrays

### Lab 2: Introducing Arrays

LAB2

QU1:

```
<?php
```

```
$cities = array("New York","Los Angeles","Chicago","Houston","Philadelphia","Phoenix","San  
Diego","dallas","San Antonio","Detroit");
```

```
$sacro = array("NY","CA","IL","TX","PA","AZ","CA","TX","TX","MI");
```

```
$population  
array("8008278","3694820","2896016","1953631","1517550","1321045","1223400","1188580","11446  
46","951270");
```

```

$total = 0;
for ($i = 0;$i<10;$i++){
    echo nl2br("$cities[$i], $accro[$i] ($population[$i]) \n");

    $total = $total + $population[$i];
}

echo nl2br("\n total is $total")

?>

```

Sorting by Population

City	Population
New York, NY	8008278
Los Angeles, CA	3694820
Chicago, IL	2896016
Houston, TX	1953631
Philadelphia, PA	1517550
Phoenix, AZ	1321045
San Diego, CA	1223400
Dallas, TX	1188580
San Antonio, TX	1144646
Detroit, MI	951270

Sorting by City name

City	Population
Chicago, IL	2896016
Dallas, TX	1188580
Detroit, MI	951270
Houston, TX	1953631
Los Angeles, CA	3694820
New York, NY	8008278
Philadelphia, PA	1517550
Phoenix, AZ	1321045
San Antonio, TX	1144646
San Diego, CA	1223400

QU 2:

```
<?php
```

```

$cities = array("New York","Los Angeles","Chicago","Houston","Philadelphia","Phoenix","San
Diego","dallas","San Antonio","Detroit");

```

```

$accro = array("NY","CA","IL","TX","PA","AZ","CA","TX","TX","MI");

```

```

$population
array("8008278","3694820","2896016","1953631","1517550","1321045","1223400","1188580","11446
46","951270");

```

```

$total = 0;

```

```

array_multisort($population,$accro,$cities);

```



```

for ($i = 0;$i<10;$i++){
    echo nl2br("$cities[$i], $accro[$i] ($population[$i]) \n");
    $total = $total + $population[$i];
}

array_multisort($cities,$population,$accro);
echo nl2br("\n total is $total\n");
for ($j = 0;$j<10;$j++){
    echo nl2br("$cities[$j], $accro[$j] ($population[$j]) \n");
    $total = $total + $population[$j];
}
?>

```

QU3:

Message 3.Age: 12. Shoe Size: 14

QU4:

```

<!DOCTYPE html>
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title></title>
    </head>
    <body>
        <?php
$first = -50;
$far = 0;
echo '<table border="1">';
while ($first!=50){
echo "<tr>";

```

```

echo "<td>$first</td>";

$far = round(((( $first - 32)*5)/9);

echo "<td>$far</td>";

echo "</tr>";

    $first = $first +5;

}

echo "</table>";

?>

</body>

</html>

```

Code:

Fahrenheit	Celcius
-50 °F	-45.56 °C
-45 °F	-42.78 °C
-40 °F	-40 °C
-35 °F	-37.22 °C
-30 °F	-34.44 °C
-25 °F	-31.67 °C
-20 °F	-28.89 °C
-15 °F	-26.11 °C
-10 °F	-23.33 °C
-5 °F	-20.56 °C
0 °F	-17.78 °C
5 °F	-15 °C
10 °F	-12.22 °C
15 °F	-9.44 °C
20 °F	-6.67 °C
25 °F	-3.89 °C
30 °F	-1.11 °C
35 °F	1.67 °C
40 °F	4.44 °C
45 °F	7.22 °C
50 °F	10 °C

QU 5.

```

<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

    <meta charset="utf-8">

    <title></title>

```

```
</head>
<body>
    <?php
$first = -50;
$far = 0;
    echo '<table border="1">';
    for ($i = 0;$i<21;$i++){
echo "<tr>";
echo "<td>$first</td>";
$far = round(((( $first - 32)*5)/9);
echo "<td>$far</td>";
echo "</tr>";
        $first = $first +5;
    }
echo "</table>";
    ?>
</body>
</html>
```

Output:

Fahrenheit	Celcius
-50 °F	-45.56 °C
-45 °F	-42.78 °C
-40 °F	-40 °C
-35 °F	-37.22 °C
-30 °F	-34.44 °C
-25 °F	-31.67 °C
-20 °F	-28.89 °C
-15 °F	-26.11 °C
-10 °F	-23.33 °C
-5 °F	-20.56 °C
0 °F	-17.78 °C
5 °F	-15 °C
10 °F	-12.22 °C
15 °F	-9.44 °C
20 °F	-6.67 °C
25 °F	-3.89 °C
30 °F	-1.11 °C
35 °F	1.67 °C
40 °F	4.44 °C
45 °F	7.22 °C
50 °F	10 °C

## Difficulties Experienced

Explain all the difficulties experienced with this tutorial

No difficulty has been experienced in Lab 2 & Tutorial 2.

## Learning Outcome

Briefing overview of what you have learned in this tutorial

1. Utilizing conditional statements like if and switch.
2. Sorting arrays.
3. Gained insight into superglobal variables.
4. Exploring iterative structures such as for and while loops.
5. Familiarity with arrays, including associative arrays and multi-dimensional arrays, and learning how to access and manipulate data from them.
6. Mastering the `isset()` function for checking variable existence.
7. -Working with arrays, sort one or multiple array.

### Lab3,4:

Q1:

```
<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

<meta charset="utf-8">

<title></title>

</head>

<body>

<?php
function display($src,$alt,$height,$width){
    $imgtag="";
    if (!empty($src)){
        $imgtag = '';
```

```
return $imgtag;
```

```
}
```

```
    $src="https://mdn.github.io/learning-area/html/multimedia-and-embedding/tasks/images/larch.jpg";
```

```
    $alt="imgtest";
```

```
    $height = 50;
```

```
    $width = 50;
```

```
echo display($src,$alt,$height,$width);
```

```
    ?>
```

```
    </body>
```

```
    </html>
```

Q2:

```
<!DOCTYPE html>
```

```
<html lang="en" dir="ltr">
```

```
    <head>
```

```
        <meta charset="utf-8">
```

```
        <title></title>
```

```
    </head>
```

```
    <body>
```

```
        <?php
```

```
function display($src,$alt,$height,$width){
```

```
    $imgtag="";
```

```
    if (!empty($src)){
```

```

    $imgtag = '';
return $imgtag;
}

$src="larch.jpg";
$alt="imgtest";
$height = 50;
$width = 50;
echo display($src,$alt,$height,$width);
?>

```

</body>

</html>

Q3:

3. What does the following code print out?

```
$cash_on_hand = 31;
$meal = 25;
$tax = 10;
$tip = 10;

while(($cost = restaurant_check($meal,$tax,$tip)) < $cash_on_hand) {
    $tip++;
    print "I can afford a tip of $tip% ($cost)\n";
}

function restaurant_check($meal, $tax, $tip) {
    $tax_amount = $meal * ($tax / 100);
    $tip_amount = $meal * ($tip / 100);
    return $meal + $tax_amount + $tip_amount;
}
```

I can afford a tip of 11% (\$30) I can afford a tip of 12% (\$30.25) I can afford a tip of 13% (\$30.5) I can afford a tip of 14% (\$30.75)

Q4:



4. What does \$\_POST look like when the following form is submitted with the third option in the Braised Noodles menu selected, the first and last options in the Sweet menu selected, and 4 entered into the text box?

```
<form method="POST" action="order.php">
    Braised Noodles with: <select name="noodle">
        <option>crab meat</option>
        <option>mushroom</option>
        <option>barbecued pork</option>
        <option>shredded ginger and green onion</option>
    </select>
    <br/>
    Sweet: <select name="sweet[ ]" multiple>
        <option value="puff"> Sesame Seed Puff
        <option value="square"> Coconut Milk Gelatin Square
        <option value="cake"> Brown Sugar Cake
        <option value="ricemeat"> Sweet Rice and Meat
    </select>
    <br/>
    Sweet Quantity: <input type="text" name="sweet_q">
    <br/>
    <input type="submit" name="submit" value="Order">
</form>
```

order

(

[noodle] => shredded ginger and green onion

[sweet] => Array

(

[0] => puff

[1] => cake

```

    )
    [sweet_q] => 4
    [submit] => Order
)

<form method="POST" action="order.php">
    Braised Noodles with: <select name="noodle">
        <option>crab meat</option>
        <option>mushroom</option>
        <option>barbecued pork</option>
        <option>shredded ginger and green onion</option>
    </select>
    <br/>
    Sweet: <select name="sweet[ ]" multiple>
        <option value="puff"> Sesame Seed Puff
        <option value="square"> Coconut Milk Gelatin Square
        <option value="cake"> Brown Sugar Cake
        <option value="ricemeat"> Sweet Rice and Meat
    </select>
    <br/>
    Sweet Quantity: <input type="text" name="sweet_q">
    <br/>
    <input type="submit" name="submit" value="Order">
</form>

```

5. Write a **process\_form()** function that prints out all submitted form parameters and their values. You can assume that form parameters have only scalar values.

```

function process_form() {
    echo "Submitted form parameters and their values:<br/>";

    foreach ($_POST as $param => $value) {
        echo "$param: $value<br/>";
    }
}

```

```

    }
}

// Call the function to process the form data
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    process_form();
}

```

- Write a program that does basic arithmetic. Display a form with text box inputs for two operands and a <select> menu to choose an operation: addition, subtraction, multiplication, or division. Validate the inputs to make sure that they are numeric and appropriate for the chosen operation. The processing function should display the operands, operator, and the result. For example, if the operands are 4 and 2 and the operation is multiplication, the processing function should display something like "4 \* 2 = 8".

```

<!DOCTYPE html>

<html>

<head>

    <title>Unique Arithmetic Calculator</title>

</head>

<body>

<?php
function perform_calculation($num1, $num2, $operation) {
    switch ($operation) {
        case 'add':

```

```

        return $num1 + $num2;
    case 'subtract':
        return $num1 - $num2;
    case 'multiply':
        return $num1 * $num2;
    case 'divide':
        if ($num2 != 0) {
            return $num1 / $num2;
        } else {
            return "Cannot divide by zero";
        }
    default:
        return "Invalid operation";
    }
}

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $num1 = $_POST["num1"];
    $num2 = $_POST["num2"];
    $operation = $_POST["operation"];

    if (is_numeric($num1) && is_numeric($num2)) {
        $result = perform_calculation($num1, $num2, $operation);
    } else {
        $result = "Invalid input";
    }

    echo "Result: $num1 $operation $num2 = $result";
}

```

?>

<h2>Unique Arithmetic Calculator</h2>

<form method="post" action="">

Number 1: <input type="text" name="num1"><br>

Number 2: <input type="text" name="num2"><br>

Operation:

<select name="operation">

<option value="add">Addition (+)</option>

<option value="subtract">Subtraction (-)</option>

<option value="multiply">Multiplication (\*)</option>

<option value="divide">Division (/)</option>

</select><br>

<input type="submit" value="Calculate">

</form>

</body>

</html>

### Lab Exercise 3,4: Difficulties Experienced

No difficulties have been experienced in Lab Exercise 3 & tutorial 3,4.

### Lab Exercise 3,4: Learning outcome

1. Explored variable reach across distinct tiers: global, function, and class scopes.
2. Gained an understanding of the fundamental structure of functions and how they yield results.
3. Acquired knowledge regarding the concept of functions.
4. Discovered the flexibility of Variable-Length Argument Lists; utilizing functions like `func_num_args` to count argument quantities, `func_get_arg` to access specific arguments by position, and `func_get_args` to group all arguments into an array.
5. Examined the transmission of arguments using both By Reference and By Value approaches.
6. Employed techniques such as `is_numeric()`, `isset()`, and `empty()` to ensure form validity, examining their advantages and drawbacks.
7. Explored the distinctions between the GET and POST methods, and determining their appropriate usage.

8. Acquired knowledge in the creation of HTML forms.
9. Integrated multiple PHP files using `include()`, `include_once()`, `require()`, and `require_once()`, comprehending their purposes and recognizing the constraints of `include_once()` and `require_once()`.
10. Understood the concept of Sticky Forms and its practical application in the provided exercise from lecture 4.
11. Navigated through the management of HTML forms, utilizing powerful global variables: `$_REQUEST`, `$_SERVER`, `$_GET`, `$_POST`, and `$GLOBALS`.