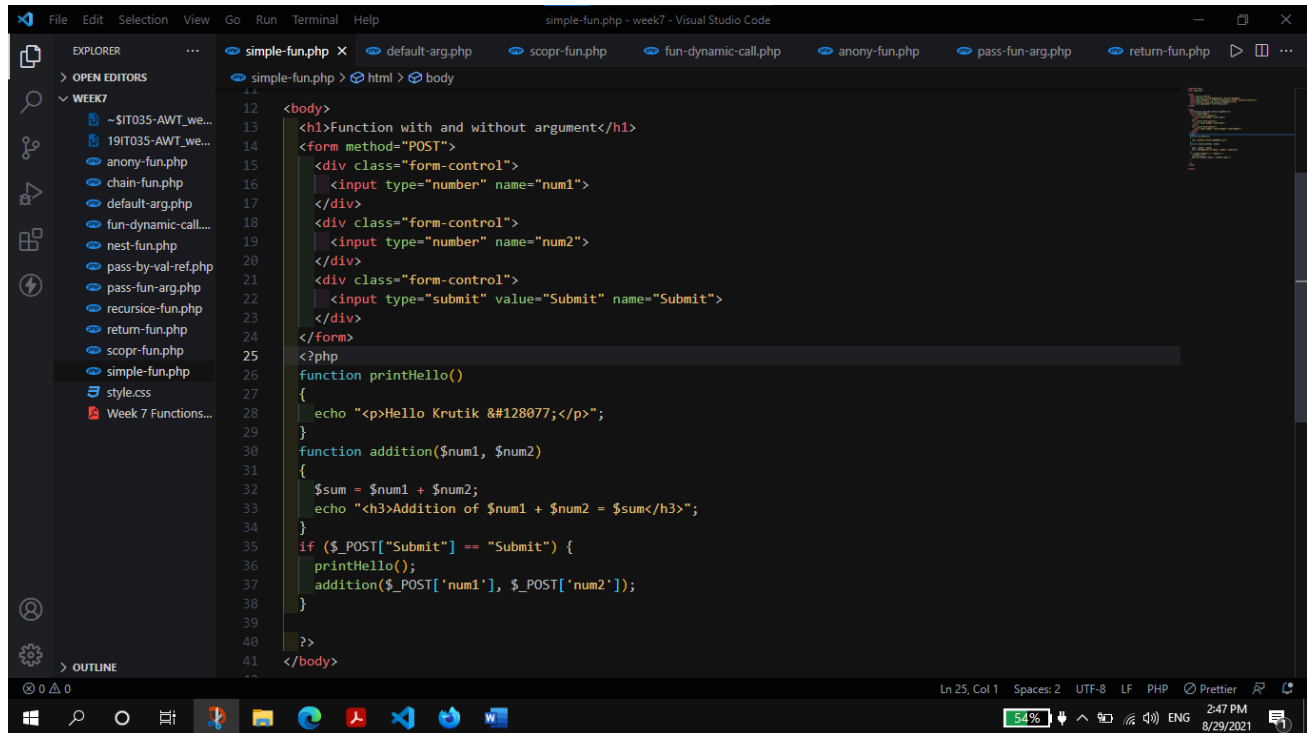


AWT Practical Week - 7 Task

Functions in PHP

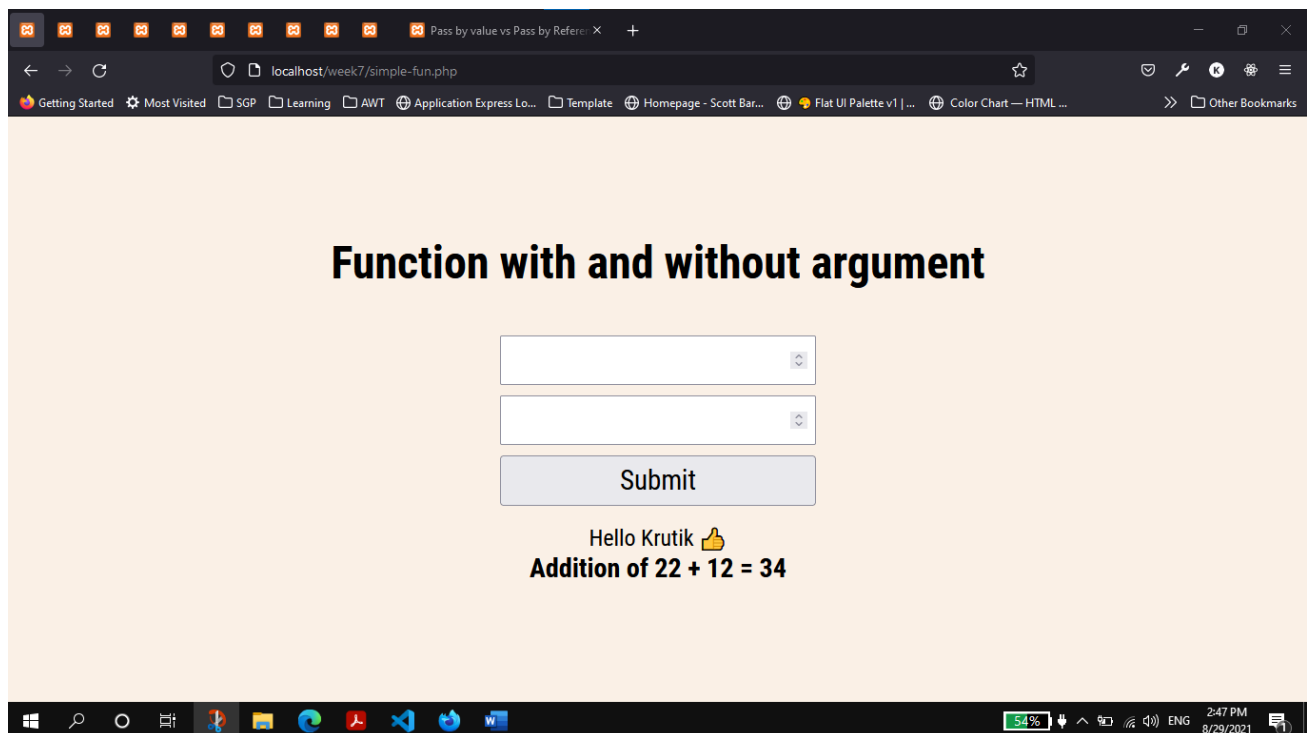
1. Function with and without argument

Code:



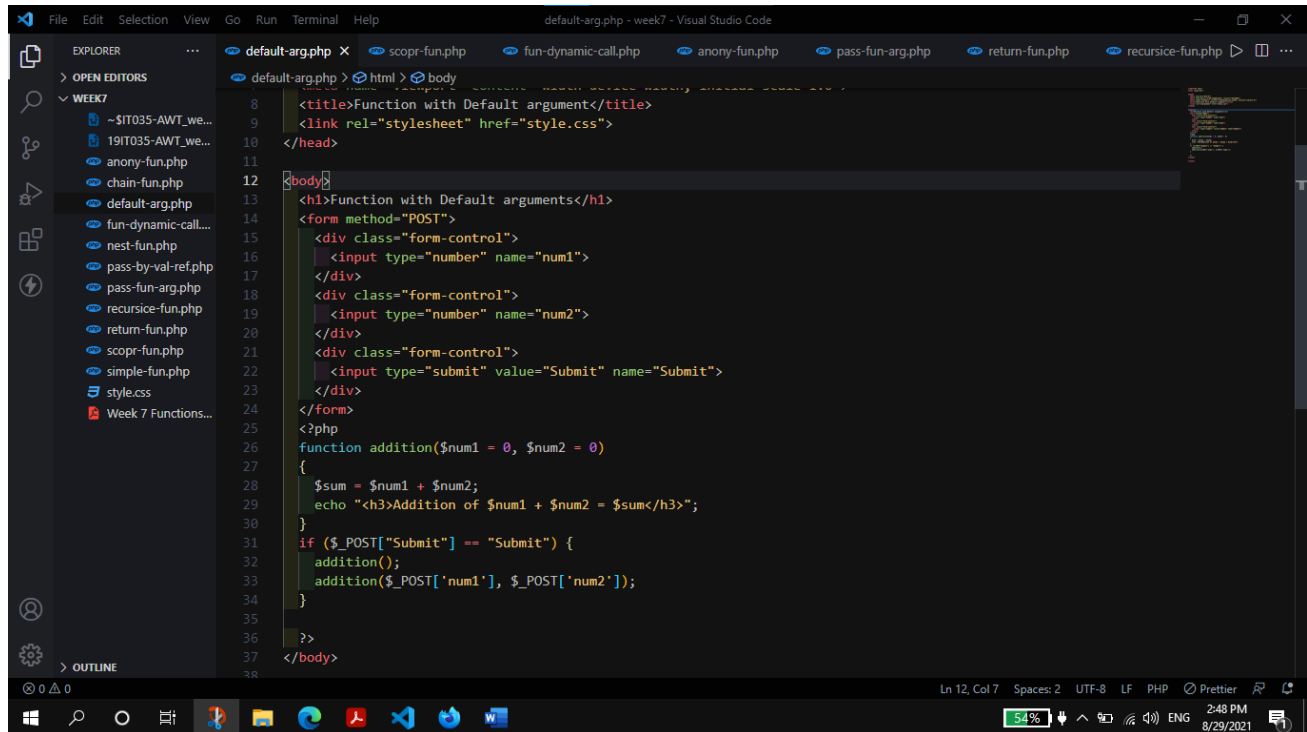
```
12 <body>
13 <h1>Function with and without argument</h1>
14 <form method="POST">
15   <div class="form-control">
16     <input type="number" name="num1">
17   </div>
18   <div class="form-control">
19     <input type="number" name="num2">
20   </div>
21   <div class="form-control">
22     <input type="submit" value="Submit" name="Submit">
23   </div>
24 </form>
25 <?php
26 function printHello()
27 {
28   echo "<p>Hello Krutik &#128077;</p>";
29 }
30 function addition($num1, $num2)
31 {
32   $sum = $num1 + $num2;
33   echo "<h3>Addition of $num1 + $num2 = $sum</h3>";
34 }
35 if ($_POST["Submit"] == "Submit") {
36   printHello();
37   addition($_POST['num1'], $_POST['num2']);
38 }
39
40
41 >>
42 </body>
```

Output:



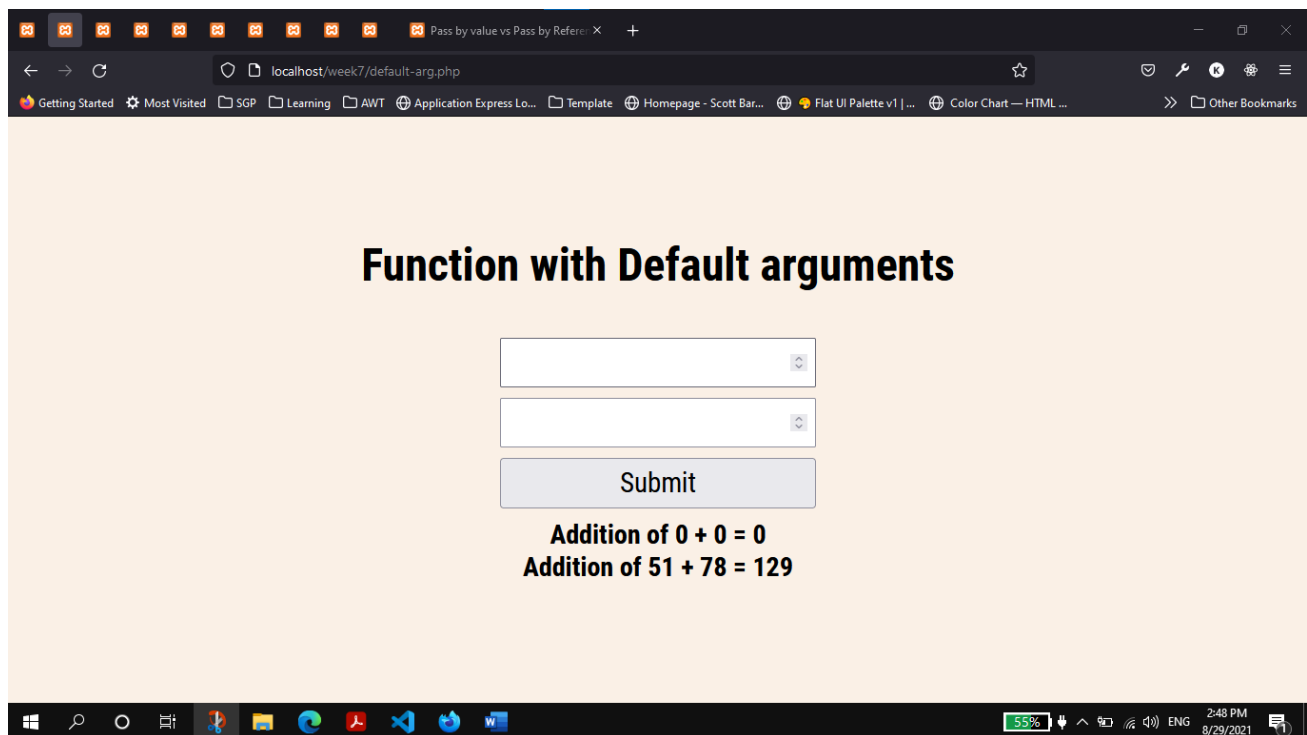
2. Function with Default argument.

Code:



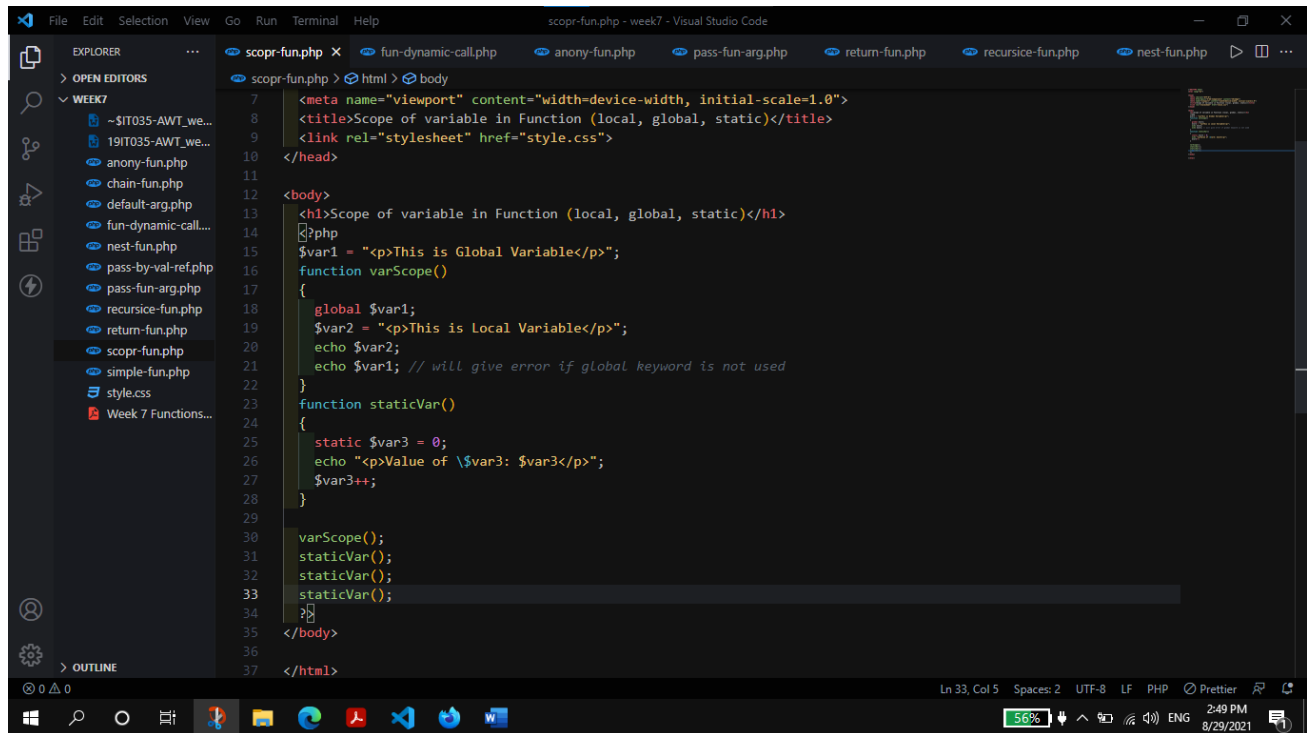
```
8 <title>Function with Default argument</title>
9 <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13 <h1>Function with Default arguments</h1>
14 <form method="POST">
15 <div class="form-control">
16 <input type="number" name="num1">
17 </div>
18 <div class="form-control">
19 <input type="number" name="num2">
20 </div>
21 <div class="form-control">
22 <input type="submit" value="Submit" name="Submit">
23 </div>
24 </form>
25 <?php
26 function addition($num1 = 0, $num2 = 0)
27 {
28     $sum = $num1 + $num2;
29     echo "<h3>Addition of $num1 + $num2 = $sum</h3>";
30 }
31 if ($_POST["Submit"] == "Submit") {
32     addition();
33     addition($_POST['num1'], $_POST['num2']);
34 }
35
36 ?>
37 </body>
```

Output:



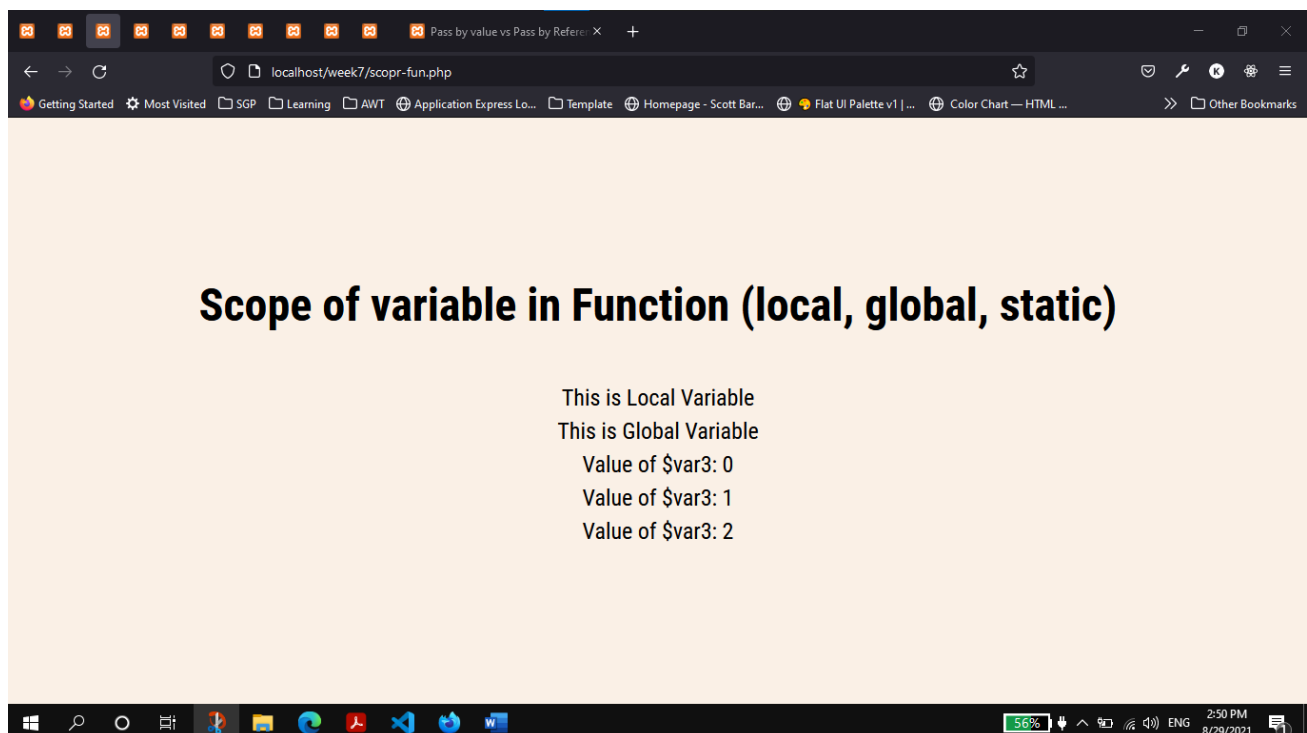
3. Scope of variable in Function (local, global, static)

Code:



```
7 <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 <title>Scope of variable in Function (local, global, static)</title>
9 <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13 <h1>Scope of variable in Function (local, global, static)</h1>
14 <?php
15 $var1 = "<p>This is Global Variable</p>";
16 function varScope()
17 {
18     global $var1;
19     $var2 = "<p>This is Local Variable</p>";
20     echo $var2;
21     echo $var1; // will give error if global keyword is not used
22 }
23 function staticVar()
24 {
25     static $var3 = 0;
26     echo "<p>Value of \$var3: $var3</p>";
27     $var3++;
28 }
29
30 varScope();
31 staticVar();
32 staticVar();
33 staticVar();
34 }
35 </body>
36
37 </html>
```

Output:

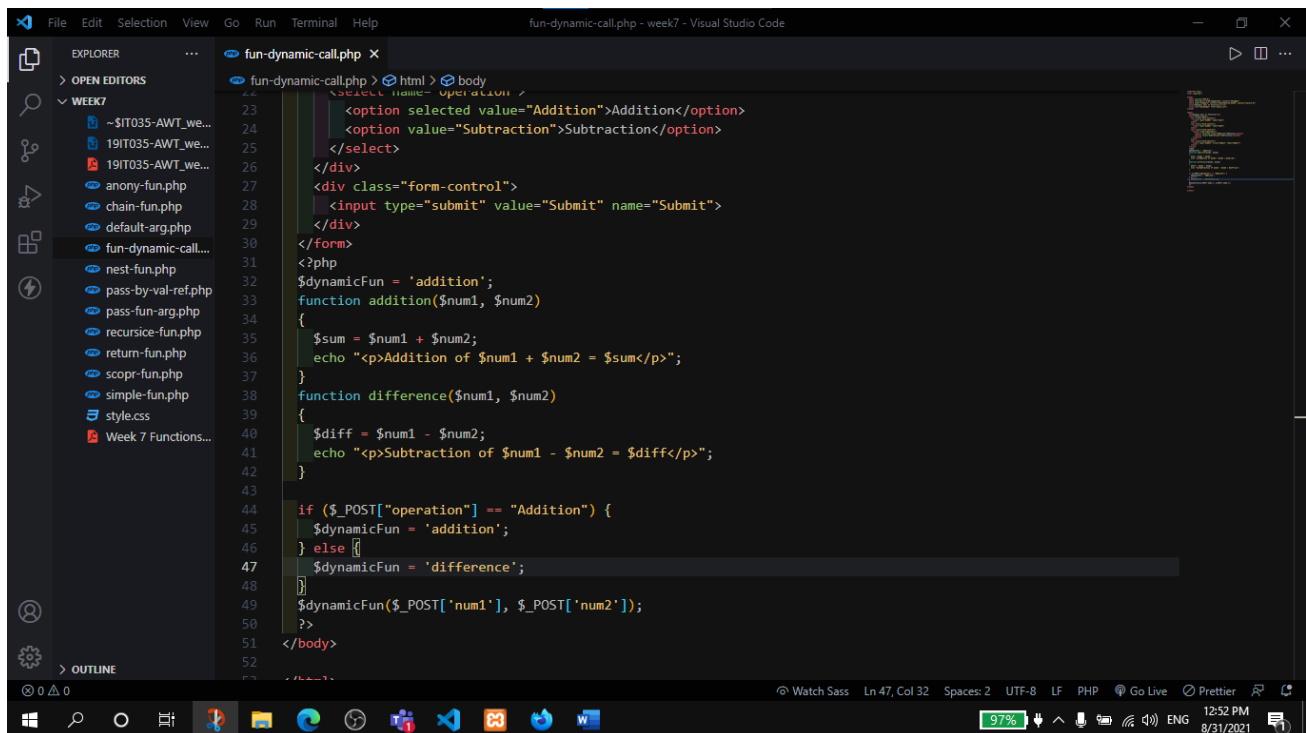


Scope of variable in Function (local, global, static)

This is Local Variable
This is Global Variable
Value of \$var3: 0
Value of \$var3: 1
Value of \$var3: 2

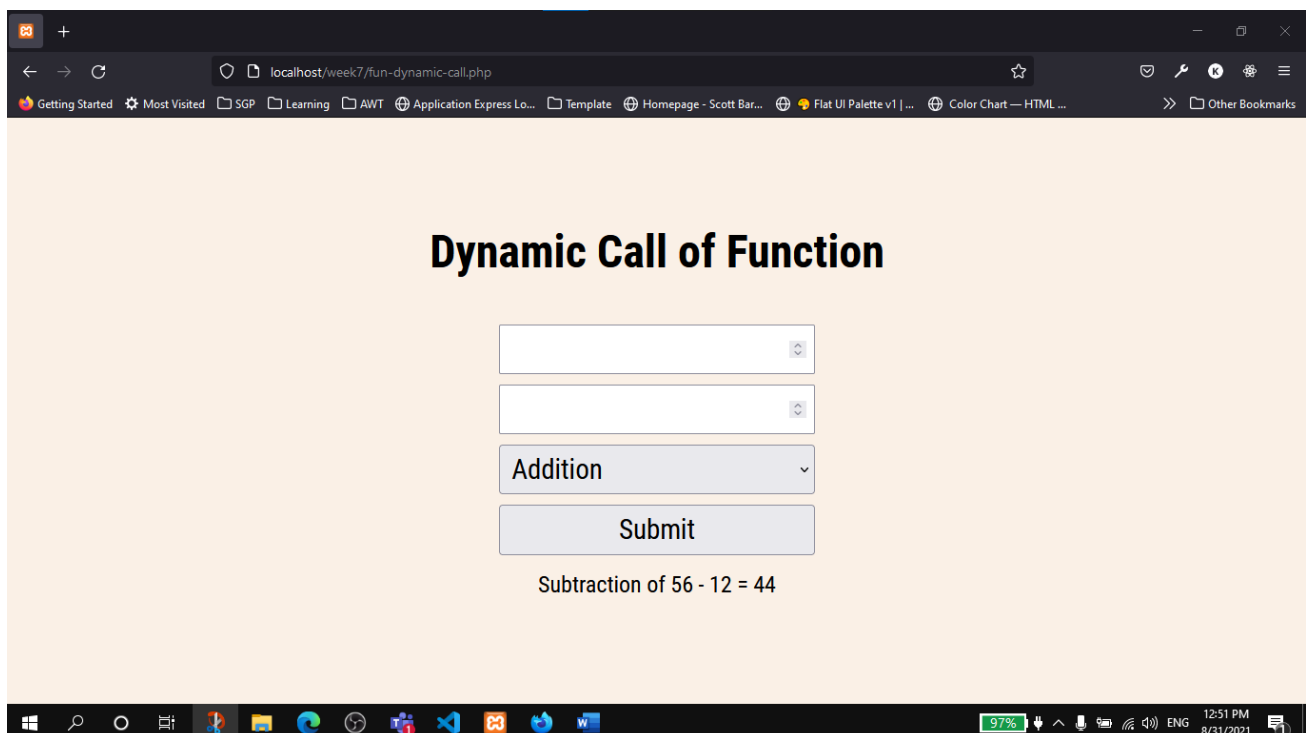
4. Dynamic Call of Function

Code:



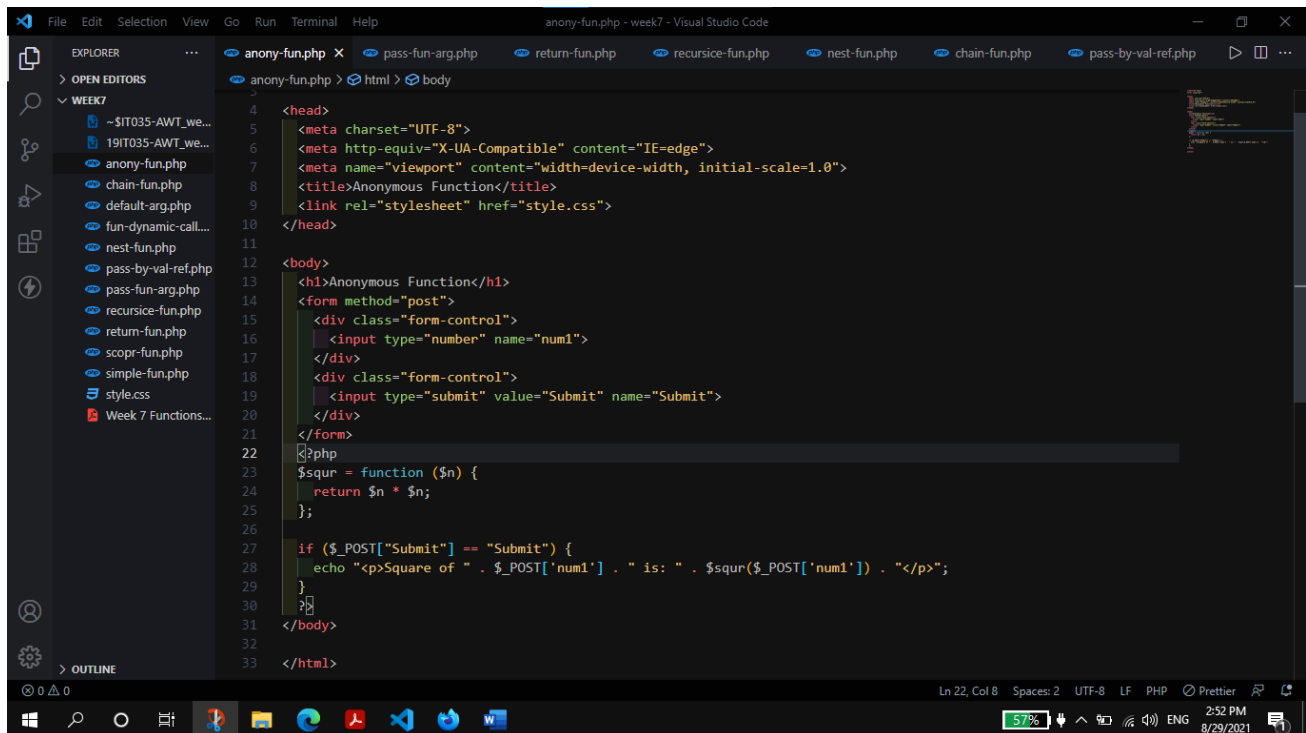
```
fun-dynamic-call.php > <html> <body>
  <select name="operation">
    <option selected value="Addition">Addition</option>
    <option value="Subtraction">Subtraction</option>
  </select>
</div>
<div class="form-control">
  <input type="submit" value="Submit" name="Submit">
</div>
</form>
<?php
$dynamicFun = 'addition';
function addition($num1, $num2)
{
    $sum = $num1 + $num2;
    echo "<p>Addition of $num1 + $num2 = $sum</p>";
}
function difference($num1, $num2)
{
    $diff = $num1 - $num2;
    echo "<p>Subtraction of $num1 - $num2 = $diff</p>";
}
if ($_POST["operation"] == "Addition") {
    $dynamicFun = 'addition';
} else {
    $dynamicFun = 'difference';
}
$dynamicFun($_POST['num1'], $_POST['num2']);
?>
</body>
```

Output:



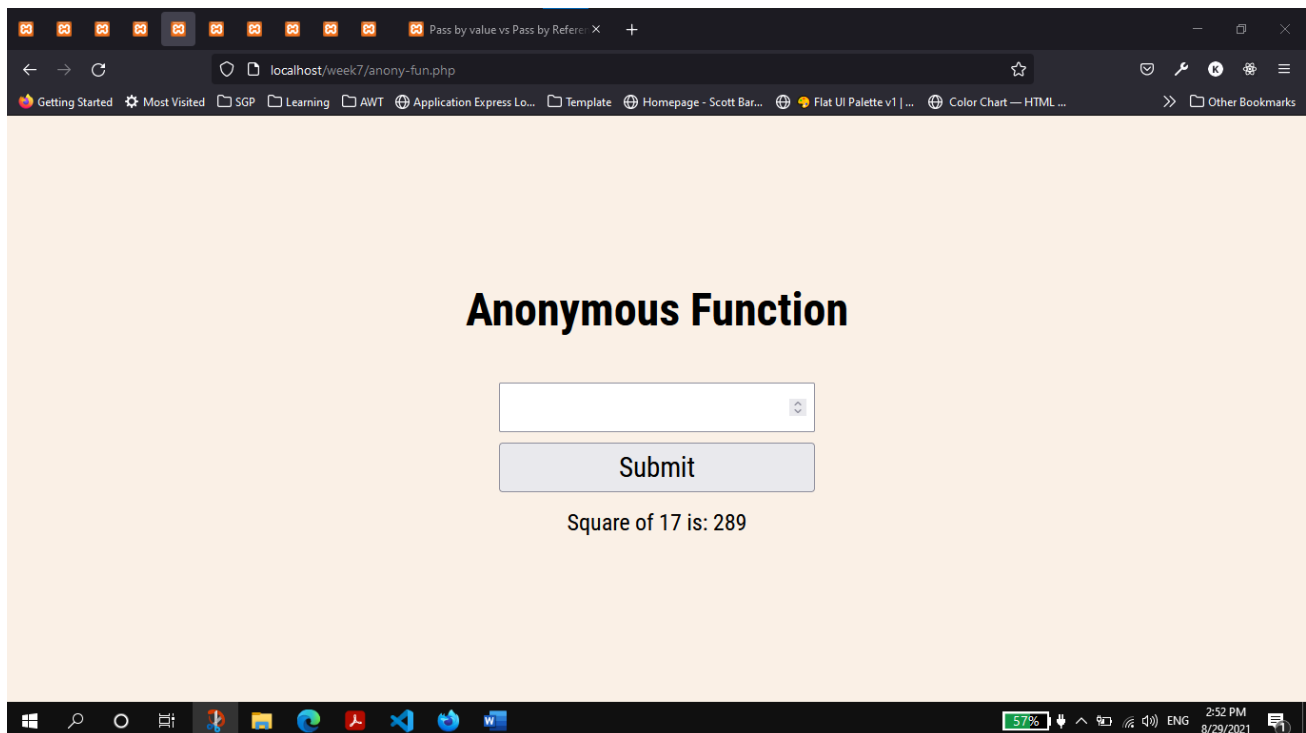
5. Anonymous Function

Code:



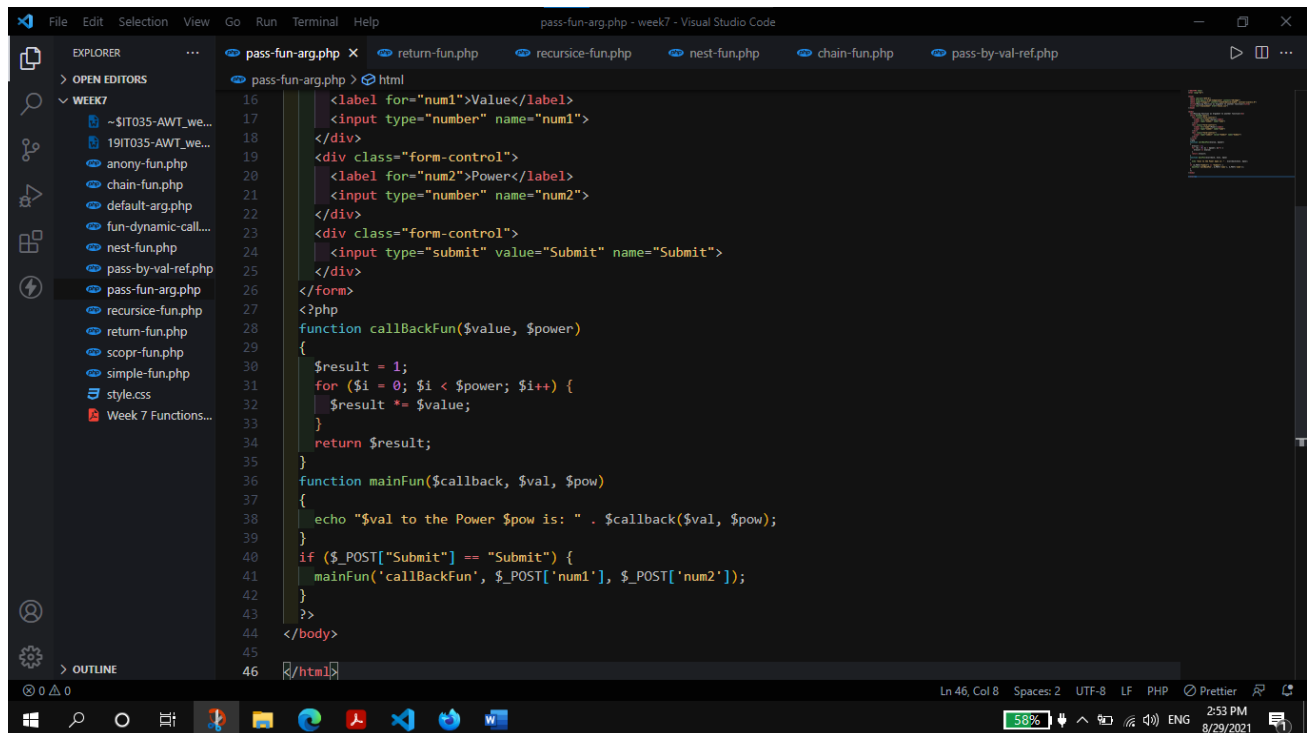
```
4 <head>
5 <meta charset="UTF-8">
6 <meta http-equiv="X-UA-Compatible" content="IE=edge">
7 <meta name="viewport" content="width=device-width, initial-scale=1.0">
8 <title>Anonymous Function</title>
9 <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13 <h1>Anonymous Function</h1>
14 <form method="post">
15 <div class="form-control">
16 <input type="number" name="num1">
17 </div>
18 <div class="form-control">
19 <input type="submit" value="Submit" name="Submit">
20 </div>
21 </form>
22 <?php
23 $squr = function ($n) {
24     return $n * $n;
25 };
26
27 if ($_POST["Submit"] == "Submit") {
28     echo "<p>Square of " . $_POST['num1'] . " is: " . $squr($_POST['num1']) . "</p>";
29 }
30
31 </body>
32
33 </html>
```

Output:



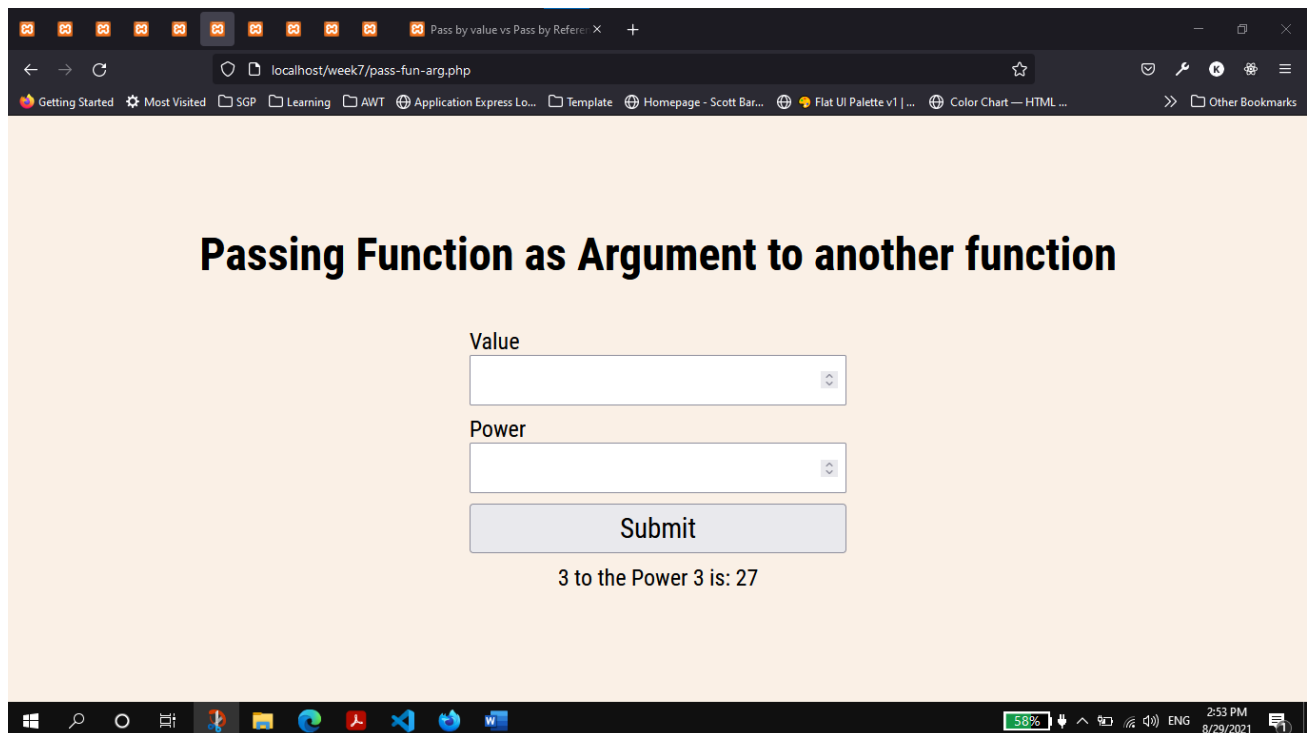
6. Passing Function as Argument to another function

Code:



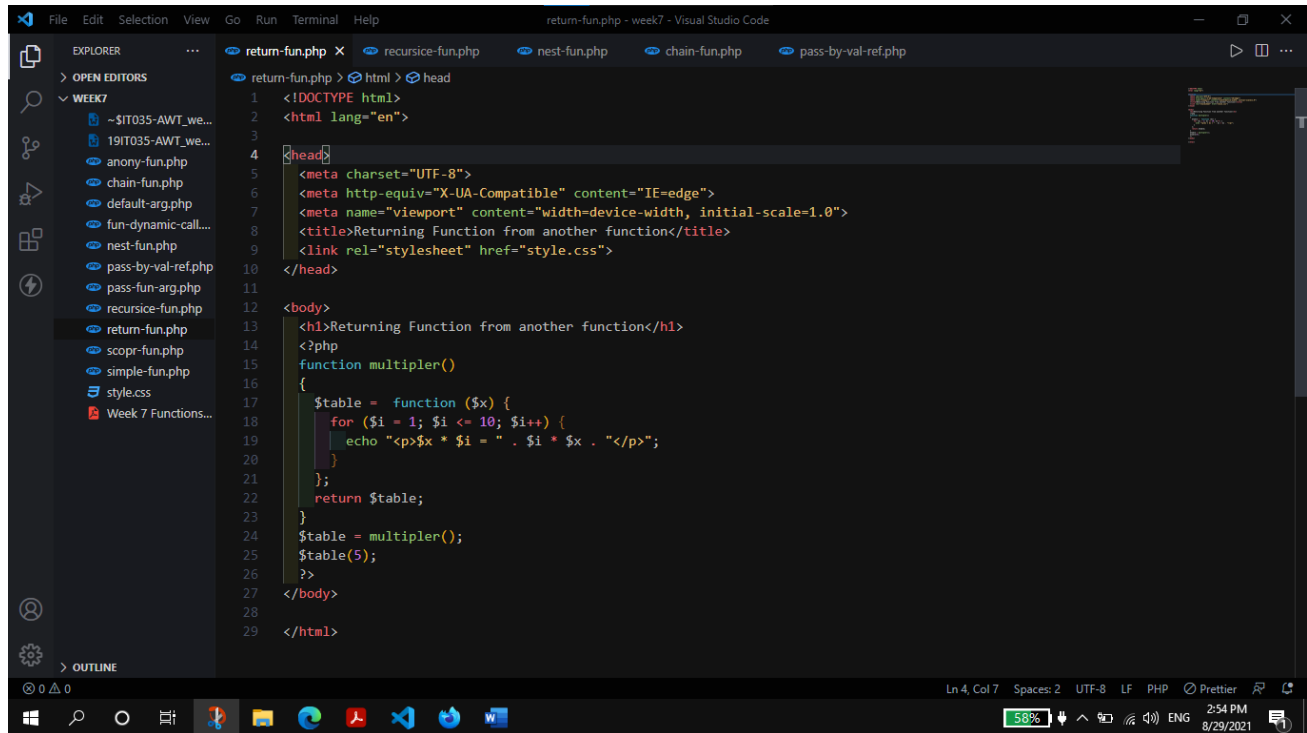
```
pass-fun-arg.php > html
16 <label for="num1">Value</label>
17 <input type="number" name="num1">
18 </div>
19 <div class="form-control">
20 <label for="num2">Power</label>
21 <input type="number" name="num2">
22 </div>
23 <div class="form-control">
24 <input type="submit" value="Submit" name="Submit">
25 </div>
26 </form>
27 <?php
28 function callBackFun($value, $power)
29 {
30     $result = 1;
31     for ($i = 0; $i < $power; $i++) {
32         $result *= $value;
33     }
34     return $result;
35 }
36 function mainFun($callback, $val, $pow)
37 {
38     echo "Val to the Power $pow is: " . $callback($val, $pow);
39 }
40 if ($_POST["Submit"] == "Submit") {
41     mainFun('callBackFun', $_POST['num1'], $_POST['num2']);
42 }
43 ?>
44 </body>
45
46 </html>
```

Output:



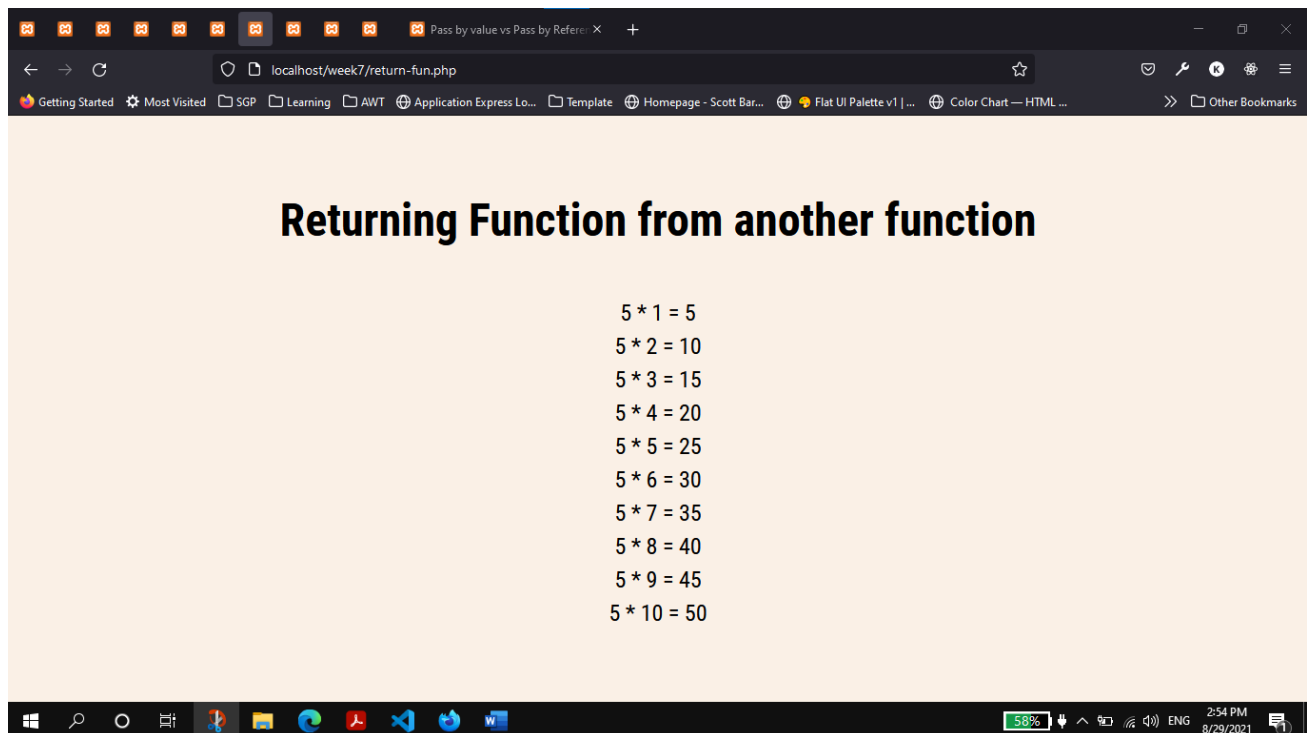
7. Returning Function from another function

Code:



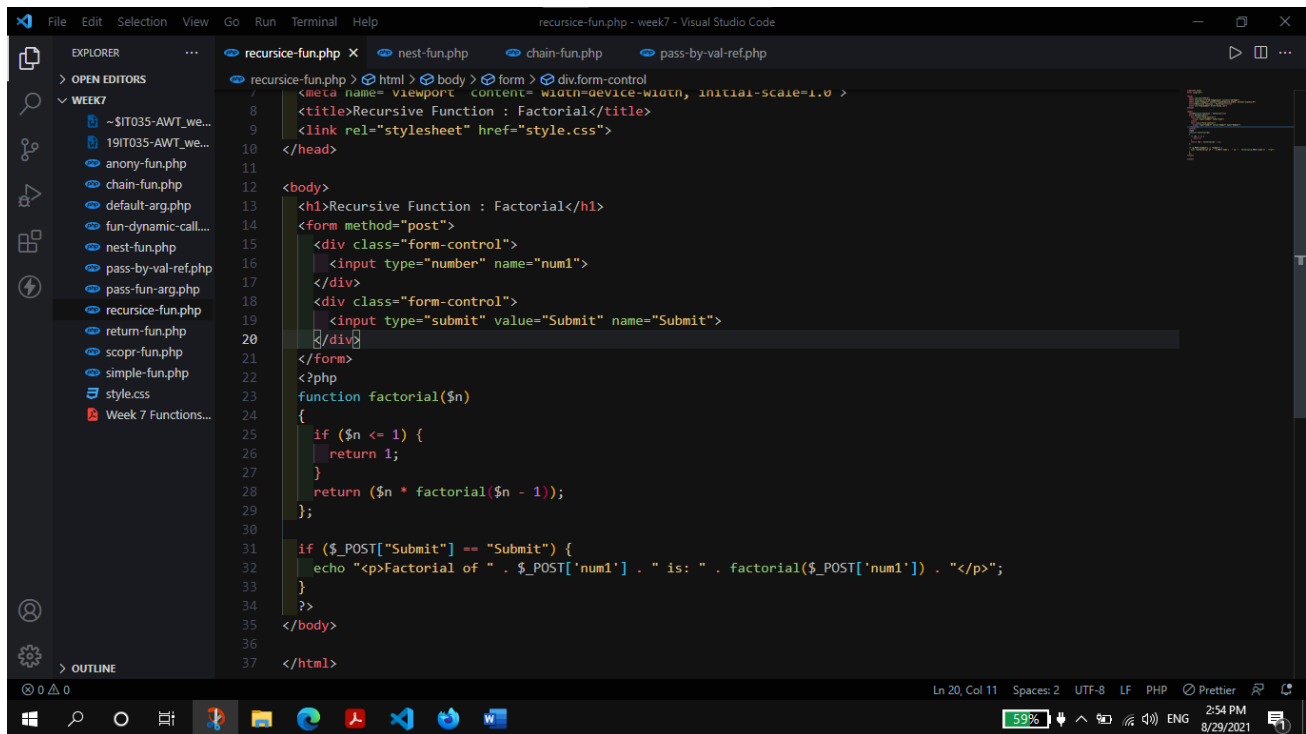
```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Returning Function from another function</title>
9   <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13   <h1>Returning Function from another function</h1>
14   <?php
15     function multiplier()
16     {
17       $table = function ($x) {
18         for ($i = 1; $i <= 10; $i++) {
19           echo "<p>$x * $i = " . $i * $x . "</p>";
20         }
21       };
22       return $table;
23     }
24     $table = multiplier();
25     $table(5);
26   ?>
27 </body>
28
29 </html>
```

Output:



8. Recursive Function : Factorial

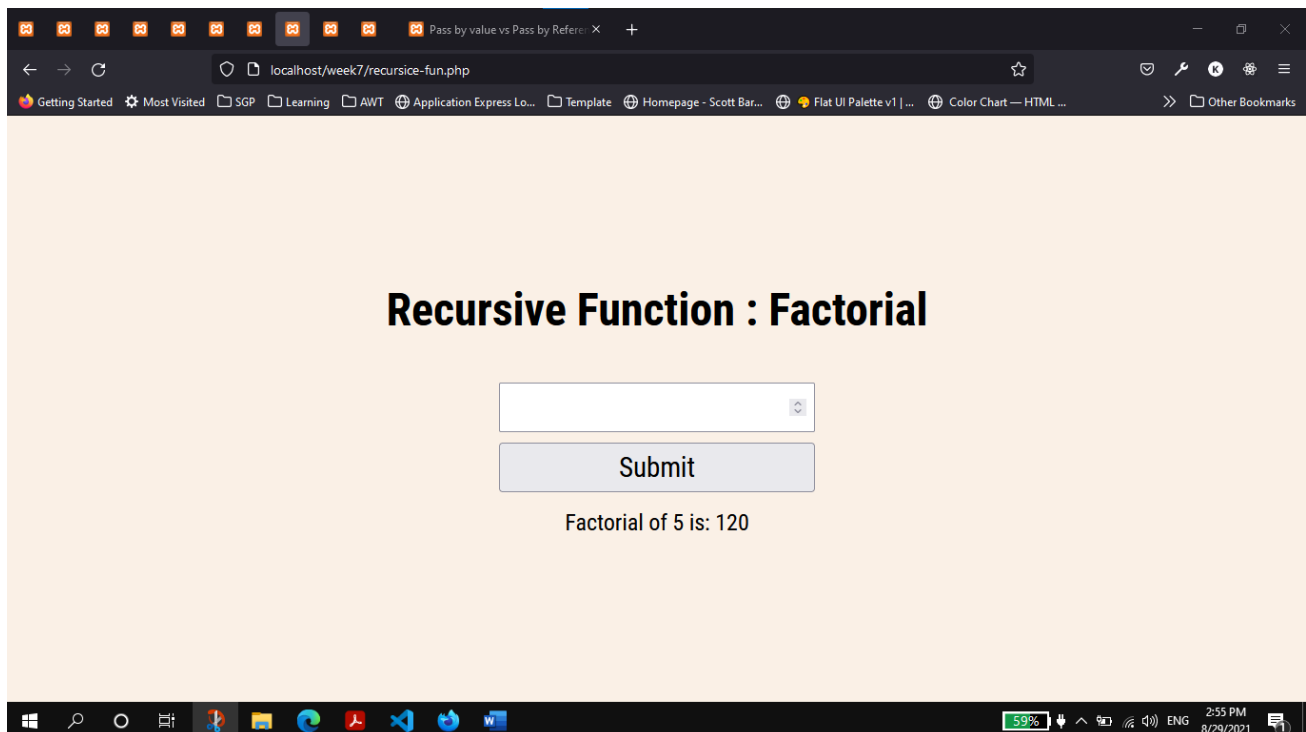
Code:



```
recursice-fun.php <html> <body> <form> <div.form-control>
<meta name= viewport content= width=device-width, initial-scale=1.0 >
<title>Recursive Function : Factorial</title>
<link rel="stylesheet" href="style.css">
</head>
<body>
<h1>Recursive Function : Factorial</h1>
<form method="post">
<div class="form-control">
<input type="number" name="num1">
</div>
<div class="form-control">
<input type="submit" value="Submit" name="Submit">
</div>
</form>
<?php
function factorial($n)
{
    if ($n <= 1) {
        return 1;
    }
    return ($n * factorial($n - 1));
};

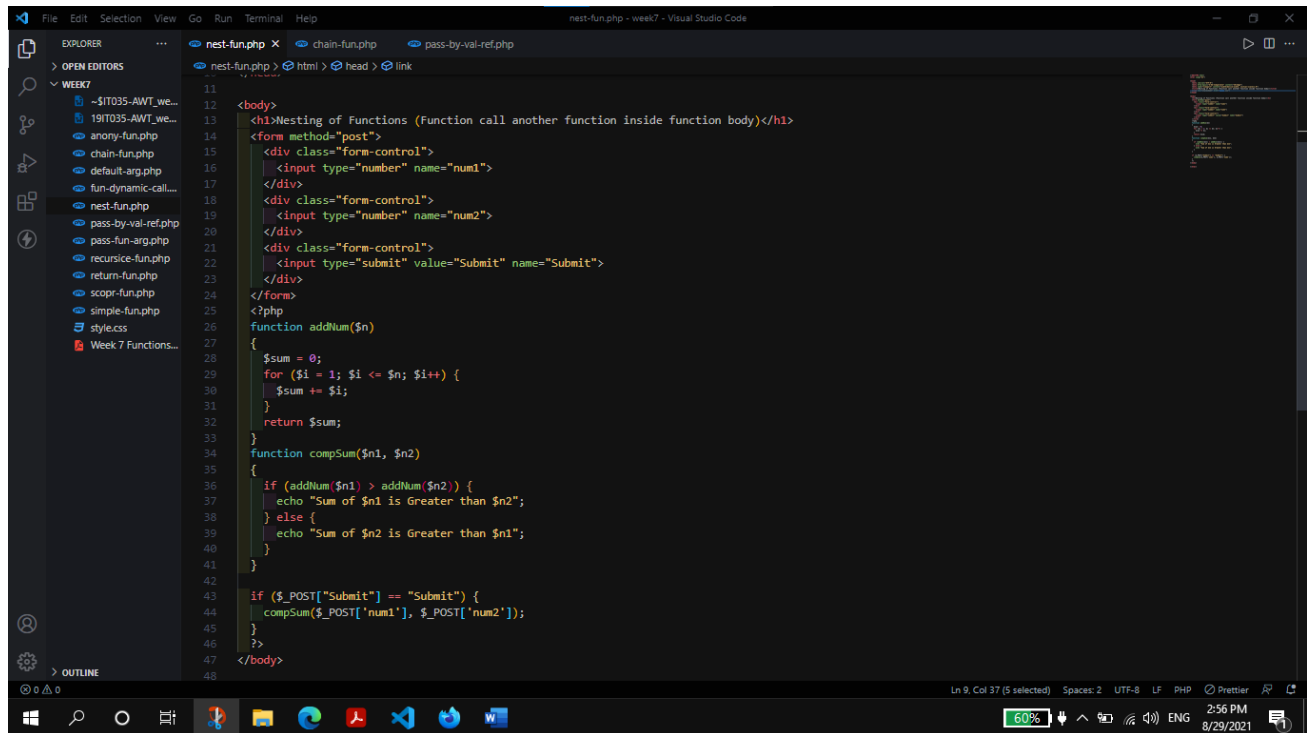
if ($_POST["Submit"] == "Submit") {
    echo "<p>Factorial of " . $_POST['num1'] . " is: " . factorial($_POST['num1']) . "</p>";
}
?>
</body>
</html>
```

Output:



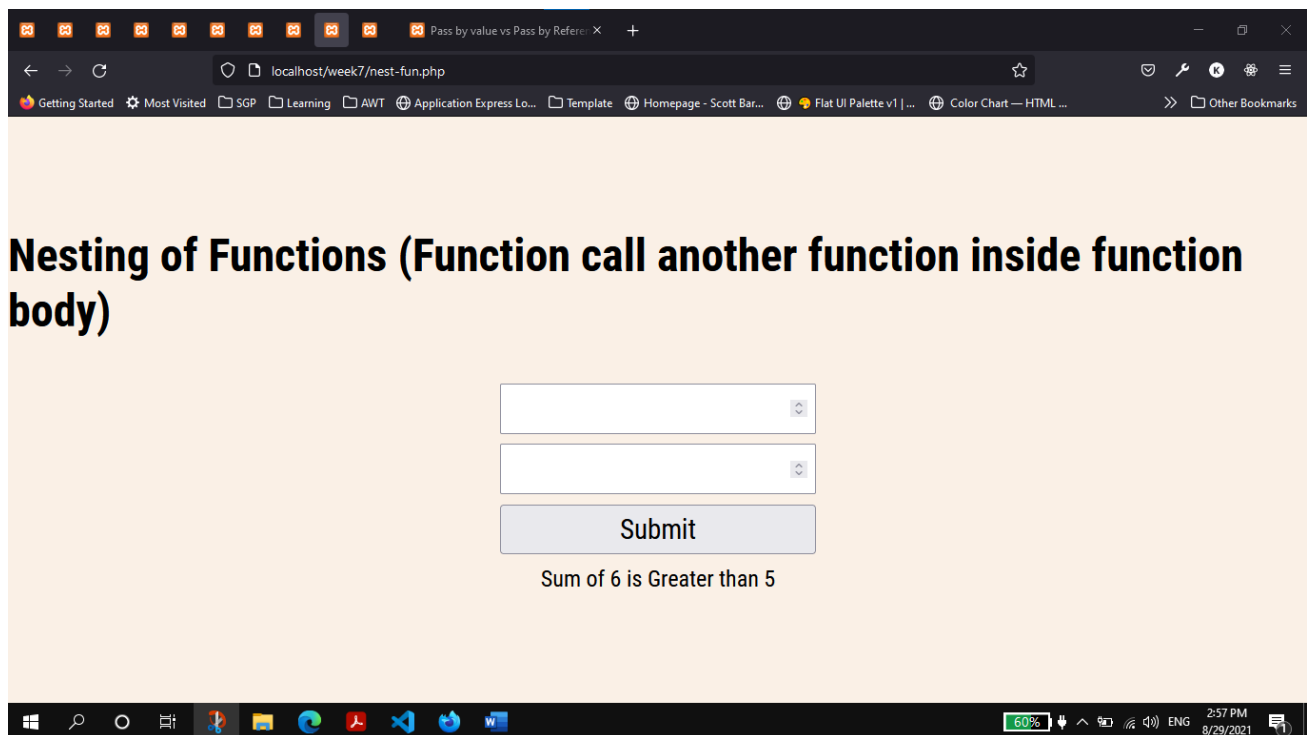
9. Nesting of Functions (Function call another function inside function body)

Code:



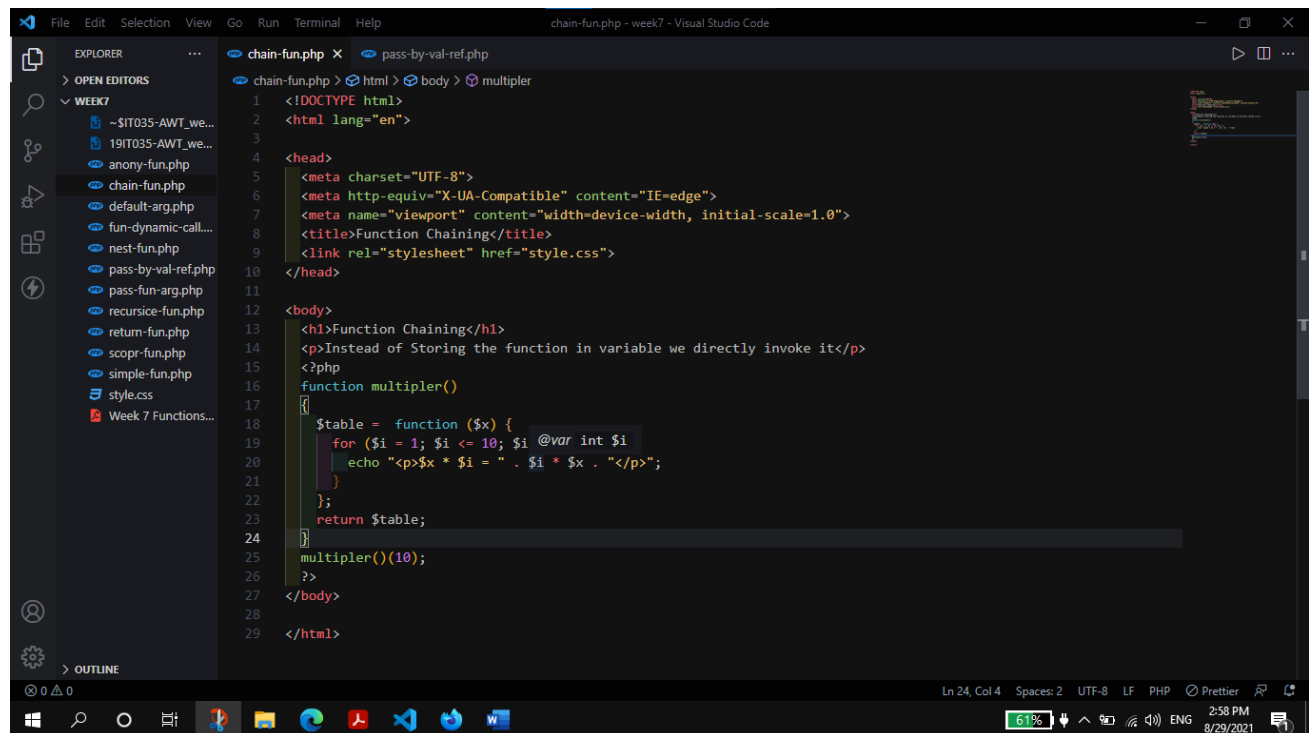
```
11 <body>
12 <h1>Nesting of Functions (Function call another function inside function body)</h1>
13 <form method="post">
14 <div class="form-control">
15 <input type="number" name="num1">
16 </div>
17 <div class="form-control">
18 <input type="number" name="num2">
19 </div>
20 <div class="form-control">
21 <input type="submit" value="Submit" name="Submit">
22 </div>
23 </form>
24 <?php
25 function addNum($n)
26 {
27     $sum = 0;
28     for ($i = 1; $i <= $n; $i++) {
29         $sum += $i;
30     }
31     return $sum;
32 }
33 function compSum($n1, $n2)
34 {
35     if (addNum($n1) > addNum($n2)) {
36         echo "Sum of $n1 is Greater than $n2";
37     } else {
38         echo "Sum of $n2 is Greater than $n1";
39     }
40 }
41
42 if ($_POST["Submit"] == "Submit") {
43     compSum($_POST["num1"], $_POST["num2"]);
44 }
45 ?>
46 </body>
47
```

Output:



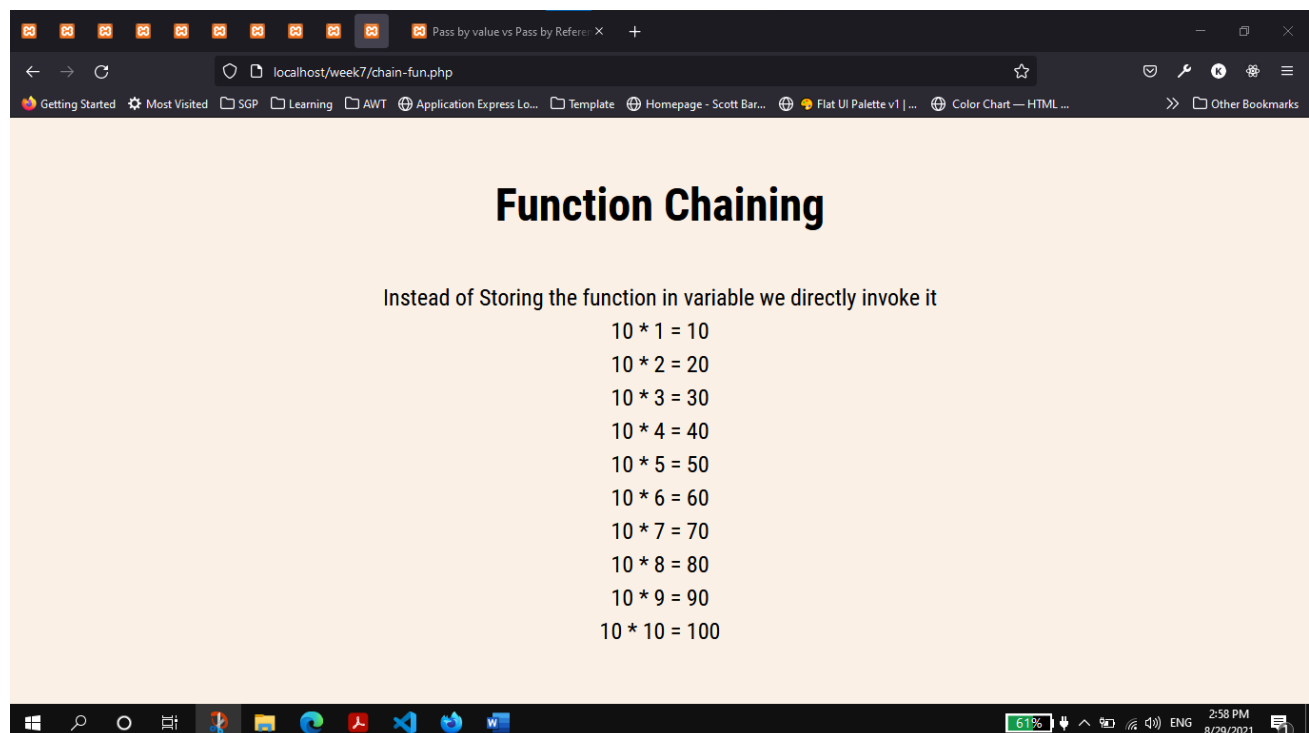
10. Function Chaining

Code:



```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Function Chaining</title>
9   <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13   <h1>Function Chaining</h1>
14   <p>Instead of Storing the function in variable we directly invoke it</p>
15   <?php
16     function multiplier()
17     {
18       $table = function ($x) {
19         for ($i = 1; $i <= 10; $i @var int $i
20           echo "<p>$x * $i = " . $i * $x . "</p>";
21       };
22       return $table;
23     }
24     multiplier(10);
25   ?>
26
27 </body>
28
29 </html>
```

Output:



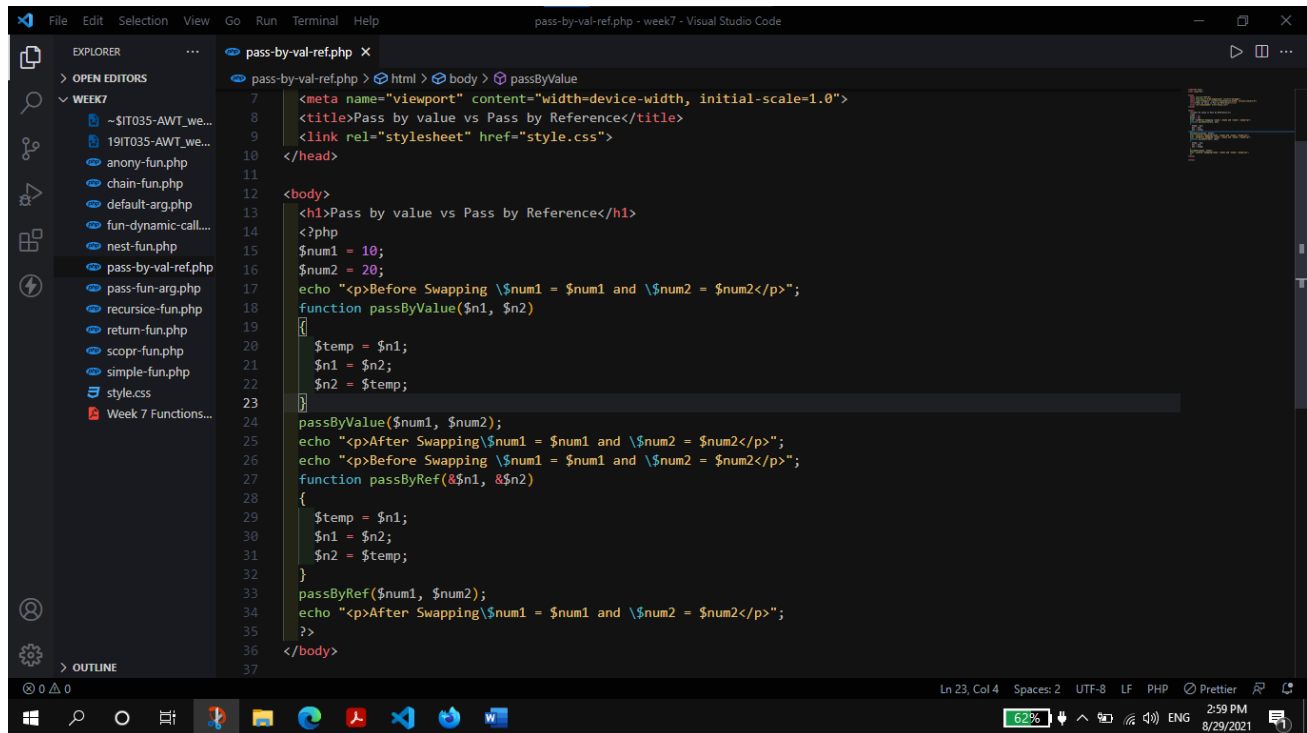
Function Chaining

Instead of Storing the function in variable we directly invoke it

10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100

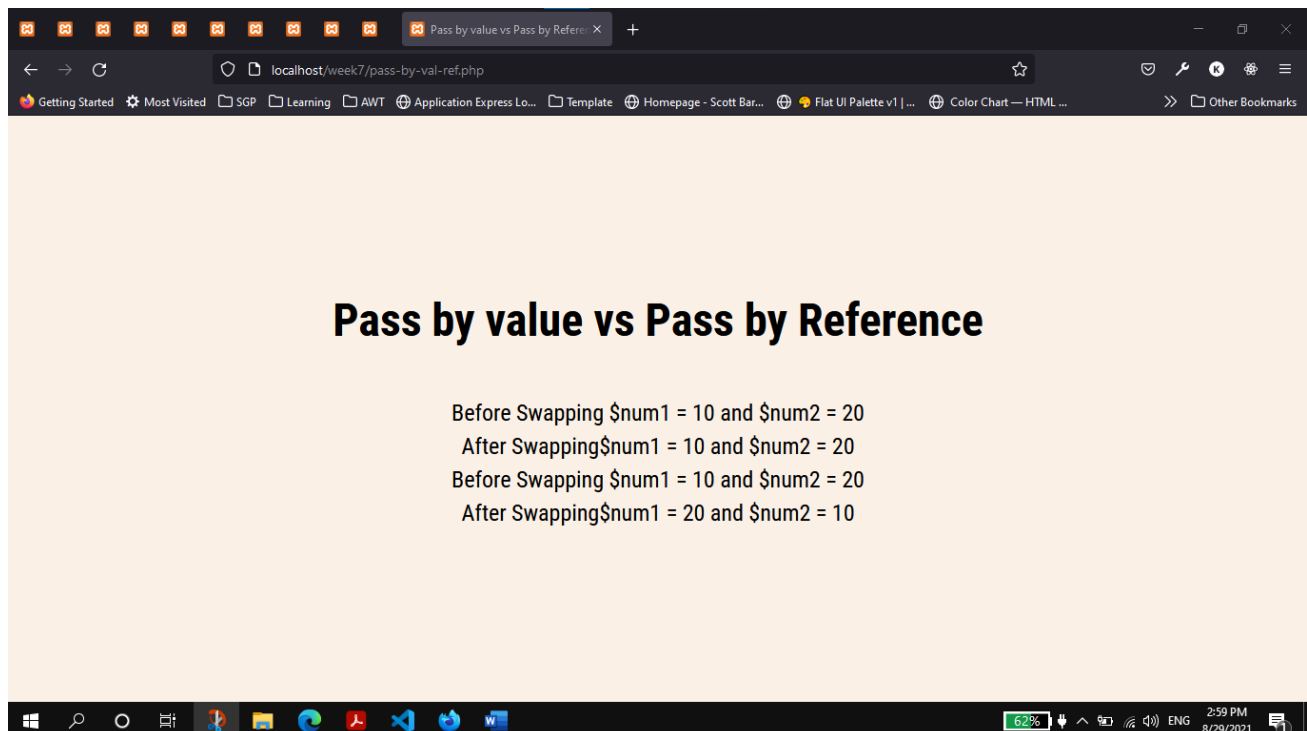
11. Pass by value vs Pass by Reference

Code:



```
pass-by-val-ref.php - week7 - Visual Studio Code
7  <meta name="viewport" content="width=device-width, initial-scale=1.0">
8  <title>Pass by value vs Pass by Reference</title>
9  <link rel="stylesheet" href="style.css">
10 </head>
11
12 <body>
13 <h1>Pass by value vs Pass by Reference</h1>
14 <?php
15 $num1 = 10;
16 $num2 = 20;
17 echo "<p>Before Swapping \\\$num1 = \$num1 and \\\$num2 = \$num2</p>";
18 function passByValue($n1, $n2)
19 {
20     $temp = $n1;
21     $n1 = $n2;
22     $n2 = $temp;
23 }
24 passByValue($num1, $num2);
25 echo "<p>After Swapping \\\$num1 = \$num1 and \\\$num2 = \$num2</p>";
26 echo "<p>Before Swapping \\\$num1 = \$num1 and \\\$num2 = \$num2</p>";
27 function passByRef(&$n1, &$n2)
28 {
29     $temp = $n1;
30     $n1 = $n2;
31     $n2 = $temp;
32 }
33 passByRef($num1, $num2);
34 echo "<p>After Swapping \\\$num1 = \$num1 and \\\$num2 = \$num2</p>";
35 ?>
36 </body>
37
```

Output:



```
Pass by value vs Pass by Reference
Before Swapping $num1 = 10 and $num2 = 20
After Swapping $num1 = 10 and $num2 = 20
Before Swapping $num1 = 10 and $num2 = 20
After Swapping $num1 = 20 and $num2 = 10
```