**Exercise 02\_02\_01 – Step 1**

In this Exercise, we will learn how to code functions and how to implement flow of control in PHP.



1. Create a folder named Exercise 02\_02\_01 and open it with your IDE. Create a new file called ***TwoFunctions.php***. Scaffold a basic HTML code layout into it. Complete our standard opening documentation in the ***<head>*** element. Make sure to have the ***modernizr*** <script> linked in. Set the <title> content to ***Two Functions***:  
   ***<!DOCTYPE html>  
   <html lang="en">  
     
   <head>  
    <meta charset="utf-8">  
    <meta name="viewport" content="width=device-width">  
    <title>Two Functions</title>  
    <script src="modernizr.custom.65897.js"></script>  
   </head>  
     
   <body>  
     
   </body>  
     
   </html>***
2. In the <body>, create an ***<h2>*** element with content ***Two Functions***. Create a set of PHP standard script delimiters.  
    ***<h2>Two Functions</h2>  
    <?php  
     
    ?>***Copy the project folder into the appropriate spot on your Web Server and test it.
3. Add the first function to the script section. It will be using a ***parameter*** as follows:  
    ***function displayMessage($firstMessage) {  
    echo "<p>$firstMessage</p>";  
    }***
4. Below the function definition, add the main script body with a call to the new function:  
    ***// main script body  
    displayMessage("This message was displayed from within a   
    function.");***  
   Give this a server test.
5. Below the first function definition, add a new function. This function will not take a ***parameter***, but it will use a ***return*** statement:  
    ***function returnMessage() {  
    return "<p>This message was displayed from within a   
    function.</p>";  
    }***  
   Give this a server test.
6. At the end of the script code block, below the first function call, add a call to the new function:  
    ***$returnValue = returnMessage();  
    echo $returnValue;***  
   Give this a server test.

**Exercise 02\_02\_01 – Step 2**



1. Copy the file ***TwoFunctions.php*** to a file named ***PassParameters.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Passing Parameters.*** In the <body>, create an ***<h2>*** element with content ***Passing Parameters***. Create a set of PHP standard script delimiters.  
    ***<h2>Passing Parameters</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Add the first function to the script section. It will be using parameter that is passed by ***value***: ***function incrementByValue($countByValue) {  
    ++$countByValue;  
    echo "<p>incrementByValue() value is   
    $countByValue.</p>";  
    }***
3. Below the function definition, add the main script body with a call to the new function:  
    ***$count = 1;  
    echo "<p>Main program starting value is $count.</p>";  
    incrementByValue($count);  
    echo "<p>Main program after incrementByValue() value is   
    $count.</p>";***  
   Give this a server test.
4. Below the first function definition, add a new function. This function will take a ***parameter***, that is passed by ***reference***:  
    ***function incrementByReference(&$countByReference) {  
    ++$countByReference;  
    echo "<p>incrementByReference() value is   
    $countByReference.</p>";  
    }***  
   Give this a server test.
5. At the end of the script code block, below the previous code, add a call to the new function:  
    ***incrementByReference($count);  
    echo "<p>Main program after incrementByReference() value   
    is $count.</p>";***  
   Give this a server test.
6. Now let’s take a look at ***default*** ***arguments***. Open up another PHP script block below the first one. Place the following function definition and call inside the block:  
    ***echo "<h2>Default Arguments</h2>";  
    function paint($color) {  
    return "<p>The color of the room is {$color}.</p>";  
    }  
    echo paint("blue");***Give this a server test.
7. Now make a copy of the function call, removing the argument:  
    echo paint("blue");  
    ***echo paint();***  
   Give this a server test and it should throw an exception.
8. Modify the code as follows to give it a ***default*** ***argument***:  
    function paint($color***="red"***) {  
   Give this a server test and it should work for both calls.
9. Modify the code as follows to use multiple parameters:  
    ***function paint($room="office", $color="red") {  
    return "<p>The color of the {$room} is {$color}.</p>";  
    }  
    echo paint();***  
   Give this a server test and it should work using no arguments and both arguments.
10. Now let’s add another call in which we give it both arguments:  
     echo paint(); ***echo paint("bedroom", "blue");***  
    A server test shows that this works without throwing an exception, and both parameters go to the arguments.
11. Now let’s just add another call in which the second argument is ***null***:  
     echo paint();  
     echo paint("bedroom", "blue"); ***echo paint("bedroom", null);***A server test shows that this works without throwing an exception, but the second parameter does not go to the default.
12. Now let’s add another call containing just the first argument:  
     echo paint();  
     echo paint("bedroom", "blue"); echo paint("bedroom", null);  
     ***echo paint("bedroom");***A server test shows that this works without throwing an exception, the first parameter goes to the argument, and the second parameter does go to the default.
13. Now let’s add another call containing just the second argument:  
     echo paint();  
     echo paint("bedroom", "blue"); echo paint("bedroom", null);  
     echo paint("bedroom");  
     ***echo paint("blue");***A server test shows that this does not work as expected. It goes without throwing an exception, the first parameter goes to the argument, and the second parameter does go to the default. It will not work, all arguments that are defaulted must be at the end after required arguments.

**Exercise 02\_02\_01 – Step 3**



1. Copy the file ***TwoFunctions.php*** to a file named ***VariableScope.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Variable Scope.*** In the <body>, create an ***<h2>*** element with content ***Variable Scope***. Create a set of PHP standard script delimiters.  
    ***<h2> Variable Scope </h2>  
    <?php  
     
    ?>***Give this a server test.
2. Declare a variable outside the scope of any function, so it has global scope:  
    <?php  
    ***$globalVariable = "global variable";***  
    ?>
3. Add a function to the script section, that will be using local variable. Call the function at the bottom of the script: ***function scopeExample() {  
    $localVariable = "local variable";  
    echo "<p>This is a $localVariable</p>";  
    }  
    scopeExample();***Give this a server test and everything should work as planned.
4. Below the function call, display the global variable:  
    scopeExample(); ***echo "This is a $globalVariable</p>";***  
   Give this a server test and we should have gotten both variables, the local one displayed inside the function, and the global one displayed outside.
5. Below this code, try to display the local variable:   
    scopeExample();  
    echo "This is a $globalVariable</p>"; ***echo "This is a $localVariable</p>";***  
   Give this a server test and it should throw an exception. The local variable is out of scope, remove the offending code.
6. At the end of the script code block, let’s build a new function and call it:  
    ***function globalExample() {  
    echo "<p>This is a $globalVariable</p>";  
    }  
    globalExample();***  
   Give this a server test and it produces a syntax error that the global variable is undefined, because it is out of ***scope***. It then executes the function up to the offending variable.
7. Inside the function, let’s employ the global keyword to bring the variable into scope:  
    function globalExample() { ***global $globalVariable;*** echo "<p>This is a $globalVariable</p>";  
    }  
    globalExample();Give this a server test and everything should now work as planned. We have brought the global variable into the scope of the function.
8. Now let’s add another call containing just the second argument:  
    echo paint();  
    echo paint("bedroom", "blue"); echo paint("bedroom", null);  
    echo paint("bedroom");  
    ***echo paint("blue");***A server test shows that this does not work as expected. It goes without throwing an exception, the first parameter goes to the argument, and the second parameter does go to the default. It will not work, all arguments that are defaulted must be at the end after required arguments.

**Exercise 02\_02\_01 – Step 4**



1. Copy the file ***TwoFunctions.php*** to a file named ***DiceRoll.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll</h2>  
    <?php  
     
    ?>***Give this a server test.
2. We will start with declaring and populating a couple of arrays. Add the following code to the beginning of the script code block:  
    <?php  
    ***$faceNamesSingular = array("one", "two", "three", "four", "five",   
    "six");  
    $faceNamesPlural = array("ones", "twos", "threes", "fours", "fives",   
    "sixes");*** ?>
3. Now we will create a function ***checkForDoubles()*** that takes two parameters. It will display the roll of 2 dice correctly, using the arrays, if doubles were rolled. Let’s start scaffolding out the function with a test. Add it to the bottom of the script section: ***function checkForDoubles($die1, $die2) {  
    global $faceNamesSingular;  
    global $faceNamesPlural;  
    echo "Die 1: $die1<br>Die 2: $die2<br>";  
    }  
    checkForDoubles(2, 2);***Give this a server and browser test.
4. Now let’s display the roll of 2 dice correctly, using the arrays, if doubles were rolled. We can use one test for doubles and one test which is not doubles. Add the following to the function code: ***if ($die1 === $die2) {  
    echo "The roll was double ", $faceNamesPlural[$die1 -   
    1], ".<br>";  
    }  
    if ($die1 !== $die2) {  
    echo "The roll was a ", $faceNamesSingular[$die1 - 1],   
    " and a ", $faceNamesSingular[$die2 - 1], ".<br>";  
    }  
    }  
    checkForDoubles(6, 6);  
    checkForDoubles(1, 6);***Give this a server test.
5. We can remove our two function call tests. Now let’s simulate the roll of two dice using a PHP built-in function called ***rand()***. It will generate a random number between 2 values. We will get two random numbers and store them in another array. We will then add them together, print the score, and check for doubles:  
    ***$dice = array();  
    $dice[0] = rand(1,6);  
    $dice[1] = rand(1,6);  
    echo "<p>";  
    $score = $dice[0] + $dice[1];  
    echo "The total score for the roll was $score.<br>";  
    checkForDoubles($dice[0], $dice[1]);  
    echo "</p>";***Give this a server test and cycle the browser multiple times to see the results.
6. Now let’s write a function to display something descriptive for the score. Add the new function below the first function we built:   
    ***function displayScoreText($score) {  
    if ($score === 2) {  
    echo "You rolled snake eyes!<br>";  
    }  
    if ($score === 3) {  
    echo "You rolled a loose deuce!<br>";  
    }  
    if ($score === 5) {  
    echo "You rolled a fever five!<br>";  
    }  
    if ($score === 7) {  
    echo "You rolled a natural!<br>";  
    }  
    if ($score === 9) {  
    echo "You rolled a nina!<br>";  
    }  
    if ($score === 11) {  
    echo "You rolled a yo!<br>";  
    }  
    if ($score === 12) {  
    echo "You rolled boxcars!<br>";  
    }  
    }***
7. Add a call to the new function below the call to the previous function:  
    checkForDoubles($dice[0], $dice[1]); ***displayScoreText($score);*** echo "</p>";  
   Give this a server test and cycle the browser multiple times to see the results.

**Exercise 02\_02\_01 – Step 5**



1. Copy the file ***DiceRoll.php*** to a file named ***DiceRoll2.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 2.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 2***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 2</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Let’s modify the ***checkForDoubles()*** function to use an ***if…else*** structure:  
    if ($die1 === $die2) {  
    echo "The roll was double ", $faceNamesPlural[$die1 - 1],   
    ".<br>";  
    }  
    ***else {*** echo "The roll was a ", $faceNamesSingular[$die1 - 1], " and   
    a ", $faceNamesSingular[$die2 - 1], ".<br>";  
    }  
    }  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 6**



1. Copy the file ***DiceRoll2.php*** to a file named ***DiceRoll3.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 3.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 3***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 3</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Let’s modify the ***displayScoreText()*** function to use an ***else if*** and an ***elseif*** structure:  
    function displayScoreText($score) {  
    if ($score === 2) {  
    echo "You rolled snake eyes!<br>";  
    }  
    ***else if*** ($score === 3) {  
    echo "You rolled a loose deuce!<br>";  
    }  
    ***elseif*** ($score === 5) {  
    echo "You rolled a fever five!<br>";  
    }  
    ***elseif*** ($score === 7) {  
    echo "You rolled a natural!<br>";  
    }  
    ***elseif*** ($score === 9) {  
    echo "You rolled a nina!<br>";  
    }  
    ***elseif*** ($score === 11) {  
    echo "You rolled a yo!<br>";  
    }  
    ***elseif*** ($score === 12) {  
    echo "You rolled boxcars!<br>";  
    }  
    ***else {  
    echo "Your roll has no nickname!<br>";  
    }***  
    }  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 7**



1. Copy the file ***DiceRoll3.php*** to a file named ***DiceRoll4.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 4.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 4***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 4</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Let’s modify the ***checkForDoubles()*** function to return a Boolean value to indicate whether doubles were rolled:  
    function checkForDoubles($die1, $die2) {  
    global $faceNamesSingular;  
    global $faceNamesPlural;  
    ***$returnValue = false;***  
    echo "Die 1: $die1<br>Die 2: $die2<br>";  
    if ($die1 === $die2) {  
    echo "The roll was double ", $faceNamesPlural[$die1 - 1],   
    ".<br>";  
    ***$returnValue = true;***  
    }  
    else {  
    echo "The roll was a ", $faceNamesSingular[$die1 - 1], " and   
    a ", $faceNamesSingular[$die2 - 1], ".<br>";  
    ***$returnValue = false;***  
    }  
    ***return $returnValue;***  
    }  
   Give this a server and browser test.
3. Modify the ***checkForDoubles()*** call to assign its return to a variable. Modify the call to ***displayScoreText()*** to use this variable as another argument:  
    ***$doubles =*** checkForDoubles($dice[0], $dice[1]);  
    displayScoreText($score***, $doubles***);
4. Modify the ***displayScoreText()*** definition to accept a new Boolean parameter to match the modified call:  
    function displayScoreText($score***, $doubles***) {
5. Modify the code block of ***displayScoreText()*** to use a ***nested if*** structure based on whether doubles were thrown:  
    function displayScoreText($score, $doubles) {  
    ***if ($doubles) {  
    if ($score === 2) {  
    echo "You rolled snake eyes!<br>";  
    }  
    elseif ($score === 12) {  
    echo "You rolled boxcars!<br>";  
    }  
    }  
    else {  
    if ($score === 3) {  
    echo "You rolled a loose deuce!<br>";  
    }  
    elseif ($score === 5) {  
    echo "You rolled a fever five!<br>";  
    }  
    elseif ($score === 7) {  
    echo "You rolled a natural!<br>";  
    }  
    elseif ($score === 9) {  
    echo "You rolled a nina!<br>";  
    }  
    elseif ($score === 11) {  
    echo "You rolled a yo!<br>";  
    }  
    else {  
    echo "Your roll has no nickname!<br>";  
    }  
    }*** }  
   Give this a server test.

**Exercise 02\_02\_01 – Step 8**



1. Copy the file ***DiceRoll4.php*** to a file named ***DiceRoll5.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 5.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 5***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 5</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Modify the code block of ***displayScoreText()*** to use a ***switch*** statement instead of the ***nested if*** structure:  
    function displayScoreText($score**~~, $doubles~~**) {  
    ***switch ($score) {  
    case 2:  
    echo "You rolled snake eyes!<br>";  
    break;  
    case 3:  
    echo "You rolled a loose deuce!<br>";  
    break;  
    case 5:  
    echo "You rolled a fever five!<br>";  
    break;  
    case 7:  
    echo "You rolled a natural!<br>";  
    break;  
    case 9:  
    echo "You rolled a nina!<br>";  
    break;  
    case 11:  
    echo "You rolled a yo!<br>";  
    break;  
    case 12:  
    echo "You rolled boxcars!<br>";  
    break;  
    default:   
    echo "Your roll has no nickname!<br>";  
    break;  
    }*** }  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 9**



1. Copy the file ***DiceRoll5.php*** to a file named ***DiceRoll6.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 6.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 6***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 6</h2>  
    <?php  
     
    ?>***Give this a server test.
2. After the two global array declarations, let’s declare a couple of new variables that we will use in implementing loops:  
    ***$doublesCount = 0;  
    $rollNumber = 1;***
3. Convert the main program part of the script into a ***loop*** that will continually roll the dice until a number 5 is rolled:  
    ***$dice = array();  
    while ($rollNumber <= 5) {  
    $dice[0] = rand(1,6);  
    $dice[1] = rand(1,6);  
    echo "<p>";  
    $score = $dice[0] + $dice[1];  
    echo "The total score for the roll was $score.<br>";  
    $doubles = checkForDoubles($dice[0], $dice[1]);  
    displayScoreText($score, $doubles);  
    echo "</p>";  
    if ($doubles) {  
    ++$doublesCount;  
    }  
    ++$rollNumber;  
    }***Give this a server and browser test.
4. For a finishing touch, let’s display some totals. Add some code below the ***while*** loop:  
    ***echo"<p>Doubles occurred on $doublesCount of the five   
    rolls.</p>";***  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 10**



1. Copy the file ***DiceRoll6.php*** to a file named ***DiceRoll7.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 7.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 7***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 7</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Convert the ***while*** loop to a ***do…while*** loop as follows:  
    ***do {*** $dice[0] = rand(1,6);  
    $dice[1] = rand(1,6);  
    echo "<p>";  
    $score = $dice[0] + $dice[1];  
    echo "The total score for the roll was $score.<br>";  
    $doubles = checkForDoubles($dice[0], $dice[1]);  
    displayScoreText($score, $doubles);  
    echo "</p>";  
    if ($doubles) {  
    ++$doublesCount;  
    }  
    ++$rollNumber;  
    ***} while ($rollNumber <= 5);***  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 11**



1. Copy the file ***DiceRoll7.php*** to a file named ***DiceRoll8.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 8.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 8***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 8</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Convert the ***do…while*** loop to a ***for*** loop as follows:  
    ***for ($rollNumber = 1; $rollNumber <= 5; $rollNumber++) {*** $dice[0] = rand(1,6);  
    $dice[1] = rand(1,6);  
    echo "<p>";  
    $score = $dice[0] + $dice[1];  
    echo "The total score for the roll was $score.<br>";  
    $doubles = checkForDoubles($dice[0], $dice[1]);  
    displayScoreText($score, $doubles);  
    echo "</p>";  
    if ($doubles) {  
    ++$doublesCount;  
    }  
    ***}***  
   Give this a server and browser test.

**Exercise 02\_02\_01 – Step 12**



1. Copy the file ***DiceRoll8.php*** to a file named ***DiceRoll9.php***, and open it with your IDE. Update the standard opening documentation in the ***<head>*** element. Set the <title> content to ***Dice Roll 9.*** In the <body>, create an ***<h2>*** element with content ***Dice Roll 9***. Create a set of PHP standard script delimiters.  
    ***<h2>Dice Roll 9</h2>  
    <?php  
     
    ?>***Give this a server test.
2. Immediately after the ***$faceNamesPlural*** declaration, declare a new array named ***$faceValues***:  
    $faceNamesPlural = array("ones", "twos", "threes", "fours",   
    "fives", "sixes"); ***$faceValues = array(1, 2, 3, 4, 5, 6);***
3. Delete the declaration for the ***$dice*** array, add a new declaration for a variable $rollCount  
    ~~$dice = array();~~  
    ***$rollCount = 0;***
4. Directly beneath the new variable, declare a new empty array called ***$scoreCount***. Initialize it as follows with a ***for*** loop:  
    $rollcount = 0;  
    ***$scoreCount = array();  
    for ($possibleRolls = 2; $possibleRolls <= 12;   
    $possibleRolls++) {  
    $scoreCount[$possibleRolls] = 0;   
    }***  
   Give this a server and browser test, and everything should still be working.
5. Replace the old ***for*** loop with two nested ***foreach*** loops. Make sure to get the foreach ***closing*** ***braces*** correct. Delete the two ***rand()*** calls and the ***$score*** assignment, all as follows:  
    ***foreach ($faceValues as $die1) {  
    foreach ($faceValues as $die2) {***  
    echo "<p>";  
    echo "The total score for the roll was $score.<br>";  
    $doubles = checkForDoubles($dice[0], $dice[1]);  
    displayScoreText($score, $doubles);  
    echo "</p>";  
    if ($doubles) {  
    ++$doublesCount;  
    }  
    ***}  
    }***
6. Insert the following code directly beneath the opening of the second ***foreach*** loop:  
    foreach ($faceValues as $die2) {  
    ***++$rollCount;  
    $score = $die1 + $die2;  
    ++$scoreCount[$score];***
7. Modify the ***echo*** statement and the ***checkForDoubles()*** call to use the new variables as follows:  
    ***echo "The total score for the roll $rollCount was $score.<br>";  
    $doubles = checkForDoubles($die1, $die2);***  
   Give this a server and browser test.
8. Let’s give the last echo statement some of the new functionality with a modification as follows:  
    echo"<p>Doubles occurred on $doublesCount of the ***$rollCount***   
    rolls.</p>";  
   Give this a server and browser test.
9. Let’s implement one more ***foreach*** loop at the end of the code to find out how many times each score occurred:  
    ***foreach ($scoreCount as $scoreValue => $scoreCount) {  
    echo "<p>A combined value of $scoreValue occurred   
    $scoreCount of $rollCount times.</p>";  
    }***  
   Give this a server and browser test.