CS4116 Week 5 Lab Guide

Software Development Project

Teaching Assistant Guide

Prepared by: Zia Ur Rehman
Course: CS4116 - Software Development Project
Professor: Conor Ryan

University of Limerick
2026

Contents

1 Overview (Week 5)

What this lab reinforces

This Week 5 lab connects three important ideas from the lectures:

- Writing and using **PHP functions** (with parameters and return values).
- Handling user input via **HTML forms** and the `$_POST` / `$_GET` arrays.
- Putting your PHP code on a **real web server** so it can be accessed from anywhere.

In this lab, students will:

- Set up a new project folder for Week 5.
- Reuse the associative array from Week 4.
- Write and modify simple PHP functions that increment numbers.
- Use those functions to update every value in an array.
- Create a PHP page that displays an HTML form, reads user input, and responds with a personalised message.
- Calculate a year of birth from an age entered in a form.
- Run their PHP scripts on an external web hosting service.

Prerequisite

You should already be comfortable with:

- Basic PHP syntax, variables and arrays.
- `for` / `foreach` loops from Week 4.
- The idea of HTML forms and the difference between GET and POST, from the lectures.

2 Task 1: Creating the `week05` Folder

2.1 Understanding the Requirements

Exactly as in previous weeks, we begin by organising our work.

For this task you must:

- Create a folder called `week05`.
- Place all the PHP and HTML files for this lab inside that folder.
- Keep this folder separate from `week04` so you can easily find Week 5 work later.

2.2 Step-by-Step Instructions

1. In your usual CS4116 workspace, create a new folder named `week05`.
2. Open this folder in VS Code.
3. Inside VS Code, create an empty file called `week05.php` (or multiple files later, such as `functions.php`, `form.php`, etc.).
4. Save the file(s).

You are ready

Once `week05.php` exists inside the `week05` folder and the folder is open in VS Code, you are ready to start the remaining tasks.

3 Task 2: Recreate the Key/Value Array

3.1 Understanding the Requirements

This task repeats the array structure from Week 4, but in your new `week05` project.

From the lab sheet:

Name	Key
John	23
Edward	16
Aidan	81
Sheraz	14
Atif	99

For this task you must:

- Create a PHP array that stores these key/value pairs.
- Use the number as the *key*.
- Use the name as the *value*.

3.2 PHP Concepts You'll Need

Concept 1: Associative arrays revisited

We again use the `[key => value]` syntax:

```
<?php
    $people = [
        23 => "John",
        16 => "Edward",
        81 => "Aidan",
        14 => "Sheraz",
        99 => "Atif"
    ];
?>
```

Concept 2: Quick debugging with `print_r`

To verify the array is correct:

```
echo "<pre>";
print_r($people);
echo "</pre>";
```

3.3 Step-by-Step Plan

1. Open `week05.php`.
2. Inside the main `<div class="container">`, add a PHP block.
3. Define the `$people` array with the five key/value pairs.
4. Optionally display it with `print_r`.

3.4 Complete Solution with Comments

Listing 1: Task 2 – associative array for Week 5

```
<?php
    echo "<h2>Task 2: Create the key/value array</h2>";

    // Associative array:
    //   - keys: student numbers from the lab sheet
    //   - values: corresponding names
    $people = [
        23 => "John",
        16 => "Edward",
        81 => "Aidan",
        14 => "Sheraz",
        99 => "Atif"
    ];

    // Optional: show the internal structure for checking.
    echo "<pre>";
    print_r($people);
    echo "</pre>";
?>
```

Common Mistakes to Avoid

- Swapping key and value (writing `"John" => 23`).
- Forgetting commas between entries.
- Missing quotes around the names.

4 Task 3: Function `incr()` and Increment by 1

4.1 Understanding the Requirements

For this task you must:

- Write a PHP function called `incr()`.

- It takes a single argument (a number).
- It returns that number increased by 1.
- Using your `$people` array, iterate through it and increment each value by 1 using this function.

Note: the keys are the numbers, so we will increment the *keys*, not the names.

4.2 PHP Concepts You'll Need

Concept 1: Simple functions with parameters and return values

General pattern:

```
function add1($x) {
    $result = $x + 1;
    return $result;
}
```

We will name our function `incr` and use exactly one parameter.

Concept 2: Building a new array with changed values

Because keys in associative arrays are special, it is usually clearer to build a new array rather than modifying keys in place.

Pattern:

```
$new = [];
foreach ($old as $key => $value) {
    $new[ incr($key) ] = $value;
}
```

4.3 Step-by-Step Plan

1. Define the `incr()` function near the top of your PHP code.
2. Test it quickly (e.g. `echo incr(5);` should print 6).
3. Create a new array called `$peoplePlus1 = []`.
4. Use `foreach ($people as $key => $name)`.
5. For each entry:
   - Compute `$newKey = incr($key)`.
   - Assign `$peoplePlus1[$newKey] = $name`.
6. Print both arrays so students can see the difference.

4.4 Complete Solution with Comments

Listing 2: Task 3 – incr() function and incrementing keys by 1

```
<?php
    echo "<h2>Task 3: Function incr() and increment by 1</h2>";

    // ----------------------------
```

```php
    // Step 1: Define the function
    // ----------------------------

    // incr() takes one argument ($x) and returns $x + 1.
    function incr($x) {
        $x = $x + 1;    // or simply: return $x + 1;
        return $x;
    }

    // Quick test (optional):
    // echo "<p>incr(5) gives " . incr(5) . "</p>";

    // ----------------------------------------
    // Step 2: Use incr() on all array keys
    // ----------------------------------------

    $peoplePlus1 = [];  // new array for incremented keys

    foreach ($people as $key => $name) {

        // Compute the new key by calling our function.
        $newKey = incr($key);

        // Store the same name under the new key.
        $peoplePlus1[$newKey] = $name;
    }

    echo "<p>Original array (keys before increment):</p>";
    echo "<pre>";
    print_r($people);
    echo "</pre>";

    echo "<p>New array (keys after incr()):</p>";
    echo "<pre>";
    print_r($peoplePlus1);
    echo "</pre>";
?>
```

Common Mistakes to Avoid

- Forgetting to `return` from `incr()` (then it always returns `null`).
- Modifying the name instead of the key.
- Reusing the same array (`$people`) while also iterating over it, which can be confusing.

5 Task 4: Modify `incr()` to Increment by 5

5.1 Understanding the Requirements

The lab now asks you to:

- Modify the function from Task 3 so that it increments its argument by 5 instead of 1.
- Use the updated function to increment all keys in the array by 5.

This is mainly to reinforce the idea that functions encapsulate behaviour: change the function and all callers get the new behaviour.

5.2 PHP Concepts You'll Need

Concept 1: Updating a function's body

We keep the same function name and parameter list, but change the calculation:

```
function incr($x) {
    return $x + 5;
}
```

Everything else (how we call `incr`) stays the same.

Concept 2: Keeping old results separate

To compare the effect of "+1" and "+5", it is useful to store results in different arrays (e.g. `$peoplePlus1` and `$peoplePlus5`).

5.3 Step-by-Step Plan

1. Update the body of `incr()` so it returns `$x + 5`.
2. Create a new array `$peoplePlus5 = []`.
3. Use `foreach ($people as $key => $name)` again.
4. For each entry:
   - Compute `$newKey = incr($key)` (now adding 5).
   - Store `$name` under `$newKey` in `$peoplePlus5`.
5. Print the new array for comparison.

5.4 Complete Solution with Comments

Listing 3: Task 4 – incr() updated to add 5

```
<?php
    echo "<h2>Task 4: Modify incr() to increment by 5</h2>";

    // ------------------------------------
    // Step 1: Change the incr() function
    // ------------------------------------

    // Now incr() adds 5 instead of 1.
    function incr5($x) {
        return $x + 5;
    }

    // ----------------------------------------
    // Step 2: Apply incr() to all array keys
    // ----------------------------------------

    $peoplePlus5 = [];

    foreach ($people as $key => $name) {
```

```php

        // New key is old key plus 5.
        $newKey = incr5($key);

        $peoplePlus5[$newKey] = $name;
    }

    echo "<p>Array with keys incremented by 5:</p>";
    echo "<pre>";
    print_r($peoplePlus5);
    echo "</pre>";
?>
```

Common Mistakes to Avoid

- Keeping two different versions of `incr()` in the same file (PHP will error if you define it twice).
- Forgetting that Task 3's behaviour has now changed; if you still need the +1 version, give it a different name (e.g. `incr1()` vs `incr5()`).
- Accidentally adding 6 (`$x++` inside and also `$x + 5`) instead of exactly 5.

6 Task 5: Running Your Code on a Web Server

6.1 Understanding the Requirements

For this task you must:

- Upload your Week 5 PHP files to a real web server (e.g. infinityfree or any other host your group uses).
- Run the scripts for Tasks 3 and 4 *through the server*, not using `php` on the command line only.

The goal is to experience the real client–server flow: browser → server → PHP script → HTML back to browser.

6.2 Concepts You'll Need

Concept 1: Local vs. remote execution

- **Local**: `php week05.php` in a terminal, or built-in server on `localhost`.
- **Remote**: upload `week05.php` to a hosting account and visit a URL like: `https://yourname.infinityfre`

Concept 2: Uploading files

Most free hosts provide:

- A file manager in the browser, or
- FTP/SFTP access using a client like FileZilla.

You simply copy your `week05` folder (or its contents) into the host's web root.

6.3 Step-by-Step Plan

1. Create an account on your chosen free PHP host (e.g. infinityfree).
2. Find the "file manager" or FTP details for your site.
3. Upload your `week05.php` and any related files into a public folder (often called `htdocs` or `htdocs/week05`).
4. In your browser, navigate to the URL given by the host for that file.
5. Verify that:
   - The page loads.
   - The output from Tasks 2–4 appears as expected.

7 Task 6: Hello Name Form

7.1 Understanding the Requirements

Now you will create a PHP program that:

- The first time a user visits, displays a form asking for their name.
- After they submit the form, displays a page saying: `Hello Name`, where `Name` is what they entered.

This combines HTML forms with PHP's `$_POST` or `$_GET` array.

7.2 PHP and HTML Concepts You'll Need

Concept 1: HTML forms and the `name` attribute

A simple form:

```
<form method="post" action="hello.php">
  Name: <input type="text" name="username">
  <input type="submit" value="Submit">
</form>
```

The `name="username"` attribute is the key used in PHP.

Concept 2: Reading form data in PHP

In the PHP file specified by `action`:

```
<?php
    $name = $_POST["username"];  // from the form field
    echo "<p>Hello $name</p>";
?>
```

Concept 3: Detecting first vs. subsequent visits

We can use `isset` to check if the form has been submitted:

```
if (!isset($_POST["username"])) {
    // show the form
} else {
    // show "Hello Name"
}
```

7.3 Step-by-Step Plan

1. Create a file, e.g. `hello.php`, in the `week05` folder.
2. Start with a normal HTML skeleton and Bootstrap container if you like.
3. Inside a PHP block:
   (a) Use `if (!isset($_POST["username"]))` to check if we have a name yet.

   (a) If not set: output the HTML form.

   (b) If set: read `$name = $_POST["username"];` and print `Hello Name`.

7.4 Complete Solution with Comments

Listing 4: Task 6 – simple Hello Name form

```php
<?php
    echo "<h2>Task 6: Hello Name</h2>";

    // Have we received the form yet?
    if (!isset($_POST["username"])) {

        // No: show the form that asks for the user's name.
        echo "<form method='post' action='hello.php'>";
        echo "Name: <input type='text' name='username'>";
        echo "<input type='submit' value='Submit'>";
        echo "</form>";

    } else {

        // Yes: read the value from the POST array.
        $name = $_POST["username"];

        // Output a personalised message.
        echo "<p>Hello $name</p>";
    }
?>
```

Common Mistakes to Avoid

- Using a different input `name` than the one you read in PHP.
- Forgetting to set `method="post"` and then wondering why `$_POST` is empty.
- Putting the PHP code before the `<!DOCTYPE html>` and then echoing a second full HTML document.

8 Task 7: Age and Year of Birth

8.1 Understanding the Requirements

You must extend the program from Task 6 so that it:

- Asks the user: `Hello Name, how old are you?`
- After they enter their age, prints a message like: `Name was born in year 2005.`

You may choose to do this in one or two steps (e.g. ask name and age together on one form, or first ask for name, then age).

8.2 PHP Concepts You'll Need

Concept 1: Adding more form fields

We can extend the form to have both name and age:

```
<form method="post" action="hello_age.php">
  Name: <input type="text" name="username"><br>
  Age:  <input type="number" name="age"><br>
  <input type="submit" value="Submit">
</form>
```

Concept 2: Simple arithmetic with the current year

We can compute an approximate birth year using:

```
$currentYear = date("Y");    // e.g. 2026
$birthYear   = $currentYear - $age;
```

Concept 3: Reusing the `isset` pattern

We again use:

```
if (!isset($_POST["username"]) || !isset($_POST["age"])) {
    // show the form
} else {
    // process the values
}
```

8.3 Step-by-Step Plan

1. Either modify `hello.php` from Task 6, or create a new file `hello_age.php`.
2. In the "no data yet" branch, echo a form with two inputs: `username` and `age`.
3. In the "data received" branch:
   (a) Read `$name` and `$age` from `$_POST`.
   (b) Compute the current year with `date("Y")`.
   (c) Compute `$birthYear = $currentYear - $age`.
   (d) Echo a message: `Hello Name, you were born in year BirthYear.`

8.4 Complete Solution with Comments

Listing 5: Task 7 – Hello Name with age and year of birth

```
<?php
echo "<h2>Task 7: Hello Name, how old are you?</h2>";

// Check if both name and age have been submitted.
if (!isset($_POST["username"]) || !isset($_POST["age"])) {

```

```php
    // Show a form asking for both name and age.
    echo "<form method='post' action='hello_age.php'>";
    echo "Name: <input type='text' name='username'><br>";
    echo "Age:  <input type='number' name='age'><br>";
    echo "<input type='submit' value='Submit'>";
    echo "</form>";

} else {

    // Read the values from the POST array.
    $name = $_POST["username"];
    $age  = (int) $_POST["age"];  // cast to integer for safety

    // Compute an approximate year of birth.
    $currentYear = date("Y");
    $birthYear   = $currentYear - $age;

    // Output the message.
    echo "<p>Hello $name, you are $age years old.</p>";
    echo "<p>$name was born in year $birthYear.</p>";
}
?>
```

Common Mistakes to Avoid

- Treating `$age` as a string and doing string concatenation instead of numeric subtraction.
- Forgetting to check that `$age` is actually set before using it.
- Using a hard-coded year (e.g. `2026`) instead of `date("Y")`, which will go out of date.

9 Task 8: Prepare Submission for Feedback

9.1 Understanding the Requirements

The final step is administrative but important. The lab sheet asks you to:

- Zip up your `week05` folder.
- Submit the zipped file if you would like feedback from the teaching team.

Summary and Key Takeaways (Week 5)

In this lab, you moved beyond simple loops and arrays and started to build more realistic, interactive PHP programs.

- You organised their work into a dedicated `week05` folder and recreated the associative array of names and numeric keys.
- You practised writing and modifying simple PHP functions (`incr()`), and used these functions to transform all the keys in an array, reinforcing the idea of "write once, reuse everywhere".

- You saw how to deploy PHP scripts to a real web server and access them through a browser, experiencing the full client–server cycle.
- You built basic HTML forms, read user input using `$_POST`, and generated personalised responses (`Hello Name`).
- They combined form handling with simple arithmetic to calculate and display a year of birth from a user's age.

By the end of Week 5, you should feel more comfortable with:

- Writing and calling PHP functions with parameters and return values.
- Working with associative arrays and transforming their keys and values.
- Creating and processing HTML forms using the POST method.
- Running and testing PHP code on both local and remote web servers.