

Tutorial letter 103/0/2024

Internet Programming ICT2613


Year Module

School of Computing

Important information:

This tutorial letter contains information about Assignment 4.

BAR CODE



1. INTRODUCTION

This tutorial letter contains Assignment 4.

For Assignment 4, you are expected to complete four tasks containing programming exercises and upload them to a web server. You should also submit your assignment 4 on the Assessment 4 portal, which will be a CodeGrade submission portal, on myUnisa. The due date for Assignment 4 is August 19, 2024.

Please read this tutorial letter carefully. Make sure that you understand all the rules and requirements of Assignment 4.

2. COMPLETING ASSIGNMENT 4

- a. Complete all the assignment tasks on your PC.
- b. For each task (`.php`) create a text file with the same name. For example, for `task1.php` you will have an extra file named `task1.txt`. This text file will contain the PHP code for that task (you may leave out any HTML code that is not part of the PHP code). For each task you will thus have a `.php` and a `.txt` file. Because it is a text file, the server will not parse (interpret) the PHP code in the file, allowing the marker to view your code for that task.
- c. Create a file named `menu.inc`. Inside this file, write HTML code that will produce a **horizontal menu** with links to all the different task pages.

Task 1 | Task 2 | Task 3 | Task 4

- d. Create a page named `index.php`. Just after the `<body>` tag, include the following line of PHP code:


```
<?php include 'menu.inc'; ?>
```
- e. Include this line of PHP code in every `.php` task page. If you want to change a link for whatever reason, then you only need to change it once in `menu.inc`, as opposed to every task page.

3. MAKING YOUR ASSIGNMENT 4 AVAILABLE FOR MARKING

- a. Open an account on a free web hosting server **of your choice**. Make sure this web hosting service supports PHP and MySQL. Examples of such free hosting servers are <https://www.000webhost.com/> and <https://infinityfree.net/>.

<https://www.000webhost.com/> worked well in the past five years. When you register a site on the server of your choice, you must use a site name that contains your student number **and** any other word/s of your choice. Here is an example: 8785634-wanghor.000webhost.com. If your site contains only your student number, it is relatively easy for other students to guess and gain access to your code. **For this**

reason, if your site name contains only your student number, it will not be marked.

Transfer your .php and .txt files to your web site. Learning how to upload and make your files available on your web site forms part of the learning curve. For some guidance, we will upload a document that outlines how to upload on <https://www.000webhost.com/> in *Additional Resources* folder during the year. Note that you will upload your .php and .txt files and not .htm files. Also make sure that you have read through the account details as displayed/sent to you after registration. Remember to upload the index.php page as well, overwriting the one that may already be in there. Alternatively, if there is an index.html, index.htm, default.htm page in the directory, then delete it.

If you have an existing account on a free web hosting server, we suggest you create a new website for your assignment. Please refrain from reusing an existing website or URL.

- b. You will also be required to recreate your database on the free web hosting server using phpMyAdmin, which you will be familiar with by the time you upload your files.
- c. You will submit the URL of your web site to us for marking (see the ASSIGNMENT 4 SUBMISSION PROCEDURE section). When we enter this URL in a browser, your web site should display. In particular, we want to see the web page with the horizontal menu (home page) you have created as stated in Section 2, step c.
- d. At the bottom of each web page of a task, include the relevant text file in a centered iframe with an exact width of 1200px and a height of 400px. Here is how you do it:

```
<iframe src="task1.txt" height="400" width="1200">
  Your browser does not support iframes. </iframe>
```

If required, and in the text file, use a series of forward slashes across the page to separate sections of code, e.g.:

```
////////////////////Task 1(a)////////////////////
                Your code for task 1(a)
////////////////////Task 1(b)////////////////////
                Your code for task 1(b)
```

IMPORTANT:

- i. **A task will be awarded 0 marks if the PHP code that produced the output for that task is not displayed in an iframe on the web page of the task.**
- ii. **If the menu link to a task page or the task page itself is not available, then a task cannot be marked (0 mark awarded). It cannot be left to the marker to figure out how to access a task page.**

- iii. If the text file cannot be loaded in the iframe then the task page will start loading and then go blank (or diverted to advert pages). In such a case that task will be awarded 0 marks. One reason why this happens is because file names hosted on some servers are case-sensitive. Task1.txt is not the same as task1.txt. The same holds true for .php file names.
- iv. We mark the output of your code and use the code in the iframe to verify the output was generated as per the task requirement. We do not mark code only. If the task page does not display any output, then that task will be awarded 0 marks.

4. ASSIGNMENT 4 SUBMISSION PROCEDURE

This section outlines the **EXACT** steps to follow when submitting Assignment 4 for marking via *myUnisa*. Read it slowly. Then read it again.

STEP 1: Create a folder, **StudentSurname_StudentInitials_StudentNumber**, on your PC. Provide your surname, initials and student number in the folder name.

STEP 2: To access your site for marking, we will require a working URL. This URL will load the default launch page (index.php - which contains the menu).

- Inside the folder created in step 1, create a new text file and save it as `url.txt`. **Copy (do not type)** the URL of your web site into this file. Test the URL and make sure we can access your site. **If we cannot access your site because of a wrong URL or because you forgot to include the URL/text file, then your Assignment 4 cannot be marked and thus will be awarded 0 marks.**
- Create another text file, and name it `db.txt`. Inside this file, provide your **live** site's database name, username and password (note it is NOT the database connection details as used on your own computer, but the connection details used on the site that you are currently submitting for marking). Place `db.txt` in the folder created in step 1.
- Create another text file named `login_details.txt`. In this file provide all the necessary login details required to access the file manager you have used to upload your files. If you used a FTP client, provide the required details. Place `login_details.txt` in the folder created in step 1.
- Copy all your code (php files) and the i-frame text files into the folder you have created in step 1. Do not include any other "clutter" e.g. images you may have used. There should only be `.php` and `.txt` files in this folder. Your zip file should not be more than 100KB in size.

STEP 3: Using WinZip (or other suitable utility), zip this folder. WinZip will automatically give the zip file the name of your folder (e.g. StudentSurname_StudentInitials_StudentNumber.zip). Note that *myUnisa* only accepts `.zip` files for Assignment 4 submission.

STEP 4: Upload this zip file to *myUnisa* under *Assessment 4*. Note that the submission portal will be CodeGrade. Ensure you upload before the due date in case you have internet connection problems etc. We will not allow late submissions and we will only accept

submissions made via *myUnisa*.

5. WEB SITE AVAILABILITY & OTHER RULES AND NOTES.

- a. If your menu is not available and we are required to use a browser back-button to access a previous page, you will be penalized 10% of the final mark awarded. **If a task page is not found because of an error you made in the menu link to that page, that task will not be marked.** Note there is a difference between a page not loading because the server is slow and a page not found. We revisit sites/pages that load slowly.
- b. Many free web hosting servers delete sites that are not accessed regularly. You should, therefore, visit your site at least once a week until you have received your mark. If your site is not available, restore it as quickly as possible using the same server site name and notify your lecturer that your site was down and that it has been restored. **Your original URL from the submitted zip file will be used to access the restored site i.e. we will not accept a different URL.**
- c. You are not allowed to use any PHP frameworks or software including AI tools that generates (complete) code on your behalf. You must code the entire application by yourself.
- d. You are expected to submit your own work. If we suspect plagiarism, we will award 0 marks for copied work.
- e. Web hosting sites can have down times and if so, please try later. For this reason, please do not wait for the due date of the assignment to upload files on the web server. Try to do it in advance.
- f. Marks for Assignment 4 are awarded solely based on the website (navigation, output, code in iframes, etc) you have created. Note that the code that you have submitted on *myUnisa* is used only for checking plagiarism and record keeping.
- g. CodeGrade is used to receive Assignment 4 submissions for plagiarism checking and to provide feedback on the submissions.

6. SUPPORT PROVIDED

- a. Tasks in this assignment can be done mainly using the prescribed chapters of the textbook.
- b. Try and make use of Google to find solutions to errors. As a PHP-developer you WILL spend hours on Internet forums seeking solutions to problems.
- c. Make use of PHP documentation available online (see chapter 2 on how to access it)
- d. Participate in the *myUnisa* forum.
- e. E-mail the e-tutor/lecturer for help.

7. ASSIGNMENT 4

1. Make use of comments in your code
2. All the tasks and subtasks (excluding code in the iframe) will produce output of some sort to the screen. We first consider the output produced, and then look at the code to see how the output was produced.
3. When a task has subtasks (marked using (a) and (b)), label the subtasks clearly in the output.

Task 1, page name: task1.php, chapter: 7, marks: 25

Create a form to facilitate the registration of school learners for national Olympiads. The form should contain the following:

- A drop-down list to select a school name (populate the list with at least five school names)
- Two text fields to input the name and the surname of the learner who is being registered for the Olympiad.
- A drop-down list to select the grade of the learner (populate the list with grades 1 to 9)
- A set of checkboxes for different subjects a student can register for. Include at least five subjects. Examples of subjects are mathematics, robotics, life skills, general knowledge, arts, and Bible study. A student should be able to register for more than one subject.
- A submit button.

All input fields should be appropriately labelled.

You should use the POST method to send the form data to the same script, task1.php.

When processing the form data, first check to make sure that all the fields are filled in. Specifically, a school is selected, both text fields have entries, a grade is chosen and at least one subject (checkbox) is selected. If the form is incomplete, inform the user that the form is incomplete and that all fields need to be filled.

If the form is completed correctly, then present a paragraph below the form to indicate that the student is registered with the data entered by the user using an echo statement. The paragraph must be presented in a neat, readable format.

Additionally, if the form is completed correctly, display the total amount to be paid for the learner's registration. The total registration fee is calculated by including an administrative fee of R40 plus R20 per subject. For example, if a student is registered for 3 subjects, the total fee would be $R40 + (R20 \times 3)$, which is R100.

Task 2, page name: task2.php, chapter: 8, marks: 30

(a) (10 marks)

Generate and display the following table using a while loop. Use an appropriate constant in the code as this a 5 times table.

1	5
2	10
3	15
4	20
5	25
6	30
7	35
8	40
9	45
10	50
11	55
12	60

Note: No HTML form is allowed to obtain input from the user for this task.

(b) (10 marks)

Code a function that accepts two parameters; one to represent proctoring status of a student (that can be of two values `pass` or `fail`) and the other to represent whether there is any suspicious activity noted for the student (that can be a Boolean value of either `true` or `false`).

Code the function so that it displays:

- Release marks, if the proctoring status is `pass`.
- Disciplinary case, if the proctoring status is `fail` and the suspicious activity status is `true`.
- Supplementary/cancel exam, if the proctoring status is `fail` and the suspicious activity status is `false`.

Code the logic using ternary operator (conditional assignment operator).

Invoke the function with the following parameters and display the outcome:

Proctoring status	Suspicious activity status
<code>fail</code>	<code>false</code>
<code>fail</code>	<code>true</code>
<code>pass</code>	<code>false</code>

Note: No HTML form is allowed to obtain input from the user for this task.

(c) (10 marks)

Refer to the table below that lists hello in different South African languages (<https://frenchside.co.za/greetings-in-the-11-official-south-african-languages/>):

English	Hello
Afrikaans	Hallo
Northern Sotho	Dumela
IsiZulu	Sawubona
IsiXhosa	Molo
Sesotho	Dumela
IsiNdebele	Lotjhani/Salibonani
Setswana	Dumela
SiSwati	Sawubona
Tshivenda	Ndaa!/Aa!
Xitsonga	Xewani

Use a simple HTML form with a drop-down list containing the eleven South African languages as listed in the table above. Use a switch statement that will test the option selected by the form user against cases and that will print the hello in the selected language.

Task 3, page name: task3.php, chapters: 9, 10 & 11, marks: 30

(a) (10 marks)

Write a function that accepts an email address (you can assume it is a valid one) and displays the local and domain parts of it. For example, if the function is invoked with ICT2613@unisa.ac.za, the function must display the following:

The local part is ICT2613, and the domain parts are za, ac and unisa.

Note the domain parts must be displayed in reverse order to how it is in the email address.

You must make use of the string functions in Chapter 9.

Invoke the functions using the following email addresses:

ICT2613@unisa.ac.za
someone@gmail.com
test_email@isa.it.uk

Note: No HTML form is allowed to obtain input from the user for this task.

(b) (5 marks)

Write code to display the current date in three different formats. Also, display the date of next Monday (calculated from the current date).

Note: No HTML form is allowed to obtain input from the user for this task.

(c) (10 marks)

Write code to create an array and populate the array with the eleven official languages of South Africa (see Table in Task 2 (c)).

- Using a `foreach` loop, iterate through the array and display each item in a separate line.
- Invoke the relevant function so that the array contents are displayed alphabetically. Display the content of the array to demonstrate that the sorting has been done.
- Iterate the array and display the official language that is the longest string, i.e., the string name of the language is the longest string.

Note: No HTML form is allowed to obtain input from the user for this task.

(d) (5 marks)

Rewrite the Task 2 (c) so that the switch statement is replaced with an associative array, which is then used to retrieve the correct greeting for the user-chosen language.

Task 4, page name: task4.php, chapters: all prescribed chapters, marks: 55

Educational institutions often organise events for their stakeholders. In this task, you will implement a booking system for a small pre-school to facilitate the booking of various events hosted by the pre-school.

You will design and implement a web-based database application that has the following functionalities:

- One should be able to enter, modify, delete, store and view information about learners in the school.
You need to design at least one database table and HTML forms to achieve this.
- One should be able to enter, modify, store and view information about events held by the school.
You need to design at least one database table and forms to achieve this. An event will have a title (for example, art exhibition), a description (for example, display of art by learners on the theme seasons), a date on which it will take place (3 May 2024), starting and ending times.
- One should be able to book to attend an upcoming event organised by the school.
Bookings should only be allowed for upcoming events. Each individual booking is associated with a learner at the school. One should be able to specify who is making the booking, their

email address, cell phone number and the number of people attending the event, as the school encourages family and friends of a child's parents or guardians to attend such events.

- One should be able to view a summary of the number of people attending or having attended an event.

Allow the user to choose (past or upcoming) events organised by the school and a summary of the event along with the total number of people attended or attending the event should be displayed.

You need to structure the application code using the model view controller principle.

You are expected to give attention to the overall view, ease of navigation and usability of your application.

When this application is made available for marking, each database table should be populated with at least 5 rows of data.

To display the code in an iframe, you can include the code in all the files in one txt file for this task.