Tutorial letter 103/2/2018

Internet Programming
ICT2613

Semester 1

School of Computing

IMPORTANT INFORMATION:

This tutorial letter contains information about Assignment 2

BAR CODE

1. INTRODUCTION

Dear Student

This tutorial letter contains Assignment 2. The due date is 9 April 2018.

For Assignment 2, you are expected to complete 6 tasks containing programming exercises and upload them on a Web server.

Please read this tutorial letter carefully. Make sure that you understand all the rules and the requirements of Assignment 2.

2. COMPLETING ASSIGNMENT 2

a. Complete all the assignment tasks on your PC. See Appendix A (Windows) or B (for MAC) in the prescribed handbook on how to set up your PC, as well as the chapter 1 section "How to edit and test a PHP application". TL102 also has information to assist you with the installation of the required software.

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 | Task 5 | Task 6

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 2 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.

   https://www.000webhost.com/ worked well in 2017. When you register a site on the server of your choice, you must use a site name that contains your student number as well as 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 later in the semester. 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.

b. You will also be required to recreate your database on this 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 2 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 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 6(a) ////////////////////////

Your code for task 6(a)

////////////////////////////Task 6(b) ////////////////////////

Your code for task 6(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 2 SUBMISSION PROCEDURE

This section outlines the EXACT steps to follow when submitting Assignment 2 for marking via *my*Unisa. Read it slowly. Then read it again.

STEP 1: Create a folder, **Assignment2**, on your PC.

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 2 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. Assignment2.zip). Note that *my*Unisa only accepts .zip file for Assignment 2 submission.

STEP 4: Upload this zip file to *my*Unisa. 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 *my*Unisa.

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 - but this is a

rare event. Nevertheless, you should 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 thomaa@unisa.ac.za 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 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.

6. SUPPORT PROVIDED

a. Tasks in this Assignment can be easily done with 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. Participate in *my*Unisa forum
d. E-mail the lecturer for help

7. ASSIGNMENT 2

1.	Throughout your code, make use of comments to explain the 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: 2, marks: 20

Write a program to compute and display the final module mark of a student in ICT2613. The program should also inform the user whether the student has passed or failed (with or without admission to a supplementary exam admission).

The program should take into account the following information regarding ICT2613:

- There are only two assignments in a semester.
- Assignments 1 and 2 contribute 10% and 90% respectively to the semester mark.
- Semester mark contributes 30% toward the final mark.
- Exam mark contributes 70% toward the final mark.
- You need to obtain a minimum of 40% in the exam for the semester mark to be included in the final mark. If you obtain less than 40% in the exam, your final mark (%) will be equal to the exam mark (%).
- One obtains a pass in ICT2613 if the final mark is equal to or above 50%.
- If a student obtains less than 40% final mark, then the student fails the module.
- If a student obtains between 40% and 49% final mark, then the student gets admission to the supplementary exam.

The program should contain camel case variables to store a student number, assignment 1 and 2, and exam mark percentages. Use constants to store each assignment contribution to the semester mark, contribution of semester mark to the final mark and contribution of exam mark to the final mark. Calculate semester mark, final mark and the final outcome for the given marks. Display in a format of your choice the student number, individual assignment marks, exam mark, calculated semester mark, calculated final mark, and the final outcome for the given student number.

Given below is a sample calculation for a student who obtained 100% and 60% for assignments 1 and 2, and 40% in the exam:
Semester mark is (100 X 0.1) + (60 X 0.9), which is 64
Final mark is (40 X 0.7) + (64 X 0.3), which is 47.2
Outcome based on the final mark is supplementary exam

Task 2, page name: task2.php, chapter: 2 & 7, marks: 10

Create a form for question 1, so that the user can enter the student number, assignment 1, 2 and exam marks. Submitting the form should calculate the semester mark, final mark and the outcome for the student. Display in a format of your choice the student number, individual assignment marks, exam mark, calculated semester mark, calculated final mark, and the final outcome for the given student number.

You should use the GET method to send the form data to the same script, task2.php.

Prerequisite for Tasks 3 & 4, chapter: 4

Create a database with a table `emergency`. Create the following columns for the `emergency` table:

- `employeenum`, which is the primary key for storing unique 7 digit employee numbers
- `name`, which stores the names of the employees
- `icename`, which stores names of people that the employer can contact in case of an emergency
- `icenum`, which stores the contact numbers of the people that the employer can contact in case of an emergency.

Populate the table with at least three employees' information.

You need to write a PHP script named, `connection.php` that connects to the database using PDO. If a connection cannot be made to the database, display an appropriate error message by handling the exception thrown by the connecting statement.

Task 3, page name: task3.php, chapter: 2, 3 & 4, marks: 13

Task3.php must include code to make use of `connection.php` to connect to the database.

(a) Using **a** SELECT query extract data from the `emergency` database table and present the data on the task page in a HTML table format.

(b) Present a form where the user can enter an employee number. When the form is submitted, display the name and the contact number of the person that the employer can contact for an emergency.

Task 4, page name: task4.php, chapter: 2, 3, 4, 5 & 7 marks: 25

Task4.php must include code to make use of `connection.php` to connect to the database.

Present the data in the `emergency` database table in a HTML table format on the web page.

Design and code appropriate HTML forms to add, to update and to delete from the `emergency` database table. The user should be able to add a row to the database, to update everything in a row except the employee number and delete a row.

Note that the general design of the forms is up to you but the forms should be easy to use.

You should use the POST method to send the form data to the same script, task4.php.

Update the HTML table on the task page whenever adding, updating and deleting of data in the `emergency` database table is successful.

Task 5, page name: task5.php, chapter: 8, marks: 17

(a) Use any two of the following equality, relational and logical operators together in a control statement of your choice. An output of some sort must be produced.

==
!=
<=
>=
&&
||

(4)

(b) Use a while loop, an if statement and the modulus operator together to generate the following row of numbers (note the boldfaced and underlined numbers, as well as those not boldfaced and underlined). (8)

1 2 3 4 5 6 7 8 9 10 **11 12 13 14 15 16 17 18 19** 20 **21 22 23 24 25 26 27 28 29** 30 **31 32 33 34 35 36 37 38 39** 40

(c) Repeat, but use a for loop in the place of a while loop to generate the following row of numbers. (5)

1 2 3 4 **5** 6 7 8 9 10 11 12 13 14 **15** 16 17 18 19 20 21 22 23 24 **25** 26 27 28 29 30 31 32 33 34 **35** 36 37 38 39 40

Task 6, page name: task6.php, chapter: 9, 10 & 11 marks: 15

(a) Explain the differences between a HEREDOC and a NOWDOC. Run code of your choice to illustrate the differences. (4)

(b) Use the following string assignment `$greet="Hi, my name is Vusi!";` with 7 string functions that will produce exactly the following output. (7)

```
20
Vusi!
Hi, my name is Blessing!
```

```
hi, my name is Vusi!
Hi, My Name Is Vusi!
HI, MY NAME IS VUSI!
!isuV si eman ym ,iH
```

(a) Explain the differences between an array and an associative array. Write code to declare and display both types of arrays. (4)