Software Design and Development Coursework

This coursework consists of 3 parts and you should answer all parts.

Part 1: Module coupling and cohesion

For this part of the coursework, you are to analyse a program in terms of module coupling and cohesion. The output of this part is a short report containing extracts of source code along with descriptive writing.

Instructions

Consider the attached JavaScript program. This program implements the connect-4 game (taken from https://codingartistweb.com/2022/12/connect-4-javascript/). The source code is provided in a separate ZIP file.

In this game, players are allocated a set number of discs each. The objective is to strategically place the discs into unoccupied slots, aiming to create a sequence of four discs arranged either vertically, horizontally, or diagonally. The first player to achieve this configuration is declared the winner. This game is designed for two players. If all spaces are filled without a player forming a sequence of four discs, the game is considered to be over.

Carry out the following tasks:

* Examine the code and identify two different types of module coupling that are happening in the code. In your report, put in extracts of the code which illustrate the coupling and explain why you think that type of coupling is happening there.

* Examine the code and identify two different types of module cohesion that are happening in the code. In your report, put in extracts of the code which illustrate the cohesion and explain why you think that type of coupling is happening there.

Here is a checklist for part 1:

	Done	Marks
Module coupling example 1: code extract and explanation		2.5
Module coupling example 2: code extract and explanation		2.5
Module cohesion example 1: code extract and explanation		2.5
Module cohesion example 2: code extract and explanation		2.5
Total		10

Note 1: Please ensure that you type the report and submit it in PDF format. Failure to comply with this requirement may lead to a score of 0 for this question.

Note 2: The word limit for this task is approximately 500 words. Exceeding this limit by more than 10 percent will result in a deduction of 2 marks.

Part 2: Unit testing activity

In this part of the coursework, you are to develop a set of unit tests for a scientific calculator program written in JavaScript (taken from https://codepen.io/ggetchell/pen/KVgrYq). The output of this part is some source code and a testing report. The source code is provided in a separate ZIP file.

Instructions

Download the code. At first run the program. Try to use it to calculate some expressions. Then try to understand how does it work. Finally:

* Write a total of 9 tests, organized into three sets.

* Each test set should target one functional element of the program (You cannot choose +, -, / and *, as those have been covered in the course)

* Write the code that allows the program to pass your tests.

* Explain your approach in each test set. What was your strategy here?

* Capture screenshots or console logs of the code before and after it passed the tests, showing the output of the unit testing library. Shorten the output to the most important part, where it lists the number of tests passed and failed.

* Write everything up in a report which you should save out to PDF format.

*copy/paste your unit test codes as an appendix in the same PDF as a report

Here is a checklist you can use to ensure you have completed all the tasks:

	Done?	Marks
Test set 1 target function name		0
Test set 1 written explanation of strategy		1
Test set 1 test 1 coded, explained and run		0.5
Test set 1 test 2 coded, explained and run		0.5
Test set 1 test 3 coded, explained and run		0.5
Test set 2 target function name		0
Test set 2 written explanation of strategy		1
Test set 2 test 1 coded, explained and run		0.5
Test set 2 test 2 coded, explained and run		0.5

Test set 2 test 3 coded, explained and run		0.5
Test set 1 target function name		0
Test set 3 written explanation of strategy		1
Test set 3 test 1 coded, explained and run		0.5
Test set 3 test 2 coded, explained and run		0.5
Test set 3 test 3 coded, explained and run		0.5
Annotated output of running 9 tests before and after passing		0.5
Source code of all tests in a zip file (not a rar/7z, etc.)		1
Report in PDF format containing description of test strategy in each test set and annotated unit test output before and after passing, as described above. Aim for around 500 words (without considering the appendix). Copy/paste your unit test codes as an appendix in the same PDF as a report.		1 If it is not, a penalty would be considered (-2 marks)
Total		10

Part 3: Secure programming

For this part, you are to develop a strategy to improve the security of a piece of software using secure programming techniques. The output of this part is a report in PDF format.

Instructions

Consider one of the web-applications you have seen in the degree (or anywhere else) so far. It can be any web application. The chosen application should include a login page to check a username and password using a database. Additionally, the application should provide a separate registration page, allowing new users to sign up, to define a username, and to set a password. This application should then store these new usernames and passwords into the database. Assume the client sends everything as a text (without using any encryption algorithm) to the server using the GET method. we'll assume that the passwords are stored in the database as plain text. Other parts of the application can be anything. Referring to the strategies in "OWASP's secure coding guidelines" available at https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf, describe FIVE things you can do to the code in that application to improve its security. Report your secure programming plan. Note that you do not need to actually implement the secure code, just report on what you plan to do.

Here is a checklist for part 3.

Items	Done?	Marks
Secure programming recommendation 1: what is the problem? What should change and how?		2
Secure programming recommendation 2: what is the problem? What should change and how?		2
Secure programming recommendation 3: what is the problem? What should change and how?		2
Secure programming recommendation 4: what is the problem? What should change and how?		2
Secure programming recommendation 5: what is the problem? What should change and how?		2
Report in PDF format containing secure programming recommendations, as described above. Aim for around 500 words.		If it is not, a penalty would be considered (-2 marks)
Total		10

What to submit

You should submit the following:

Part 1: PDF report, approx 500 words.

Part 2: Source code for unit tests in a zip file and PDF report, approx 500 words.

Part 3: PDF report, approx 500 words.