 Computer Science and Creative Technologies

**Coursework or Assessment Specification**

## Module Details

|  |  |
| --- | --- |
| **Module Code** | UFCFEU-30-1 |
| **Module Title** | Programming |
| **Module Leader** | Emma Littlefair |
| **Module Tutors** | Emma Littlefair |
| **Year** | 2022 |
| **Component/Element Number** | 2  |
| **Total number of assessments for this module** | 2 |
| **Weighting** | 60% |
| **Total Assignment Time** | 4 weeks |
| **Element Description** | Portfolio of practical exercises |

## Dates

|  |  |
| --- | --- |
| **Date issued to students** | 4th July 2022 |
| **Date to be returned to students** | 15th August. 2022 |
| **Submission Date** | 1st August 2022 |
| **Submission Place** | Blackboard |
| **Submission Time** | 14:00  |
| **Submission Notes** | If you anticipate difficulty in meeting the submission date or time contact the module leader immediately |

## Feedback

|  |  |
| --- | --- |
| **Feedback provision will be:** | Partly given prior to submission during the classroom sessions, and finally when retuning the submission. |

# Contents

[Module Details 1](#_Toc55241908)

[Dates 1](#_Toc55241909)

[Feedback 1](#_Toc55241910)

[Contents 2](#_Toc55241911)

[Section 1: Overview of Assessment 3](#_Toc55241912)

[Section 2: Task Specification 3](#_Toc55241913)

[Section 3: Deliverables 5](#_Toc55241914)

[Section 4: Marking Criteria 6](#_Toc55241915)

[Section 5: Feedback mechanisms 8](#_Toc55241916)

# Section 1: Overview of Assessment

This assignment assesses the following module learning outcomes:

* Write, test, and debug programs in high and low level languages and scripts
* Apply system engineering and software development methodologies and models
* Design and implement algorithms in different languages within a suitable Integrated Development Environment (IDE).

Students will demonstrate their grasp of core programming concepts through a number of practical exercises of increasing complexity and difficulty. For example, they will design algorithms and implement a simple program for three different scenarios, storing and retrieving data in a suitable system.
In each case there must be a technical description, code and evidence of testing and correct function. A short report will explain the development methodologies used in each case.

The assignment is worth 60***%*** of the overall mark for the module.

The assignment requires you to create a portfolio of evidence containing:

* A review and analysis of the requirements
* A design solution
* An explanation of the design choices made with reference to chosen paradigms and languages
* Implementations
* A test plan
* Test results, issues and resolutions
* An evaluation of the appropriateness and capability of each implementation with respect to the requirements and the factors influencing the final choice

The assignment is described in more detail in section 2.

This is an individual assignment. Working on this assignment will help you to understand how requirements are practically implemented using a development process. It also shows how theory and concepts are translated to a functional solution.

If you have questions about this assignment, please raise them with the module leader.

#

# Section 2: Task Specification

You are a small sme software development business. A client who is based in the finance sector, has requested for you to build 3 different prototypes of software using different frameworks including low- and high-level languages.

**The 3 prototypes required are**

1. To create a console line procedural program in Assembly Language to be able to complete 4 basic functions Add, Subtract, Divide and Multiply integer numbers.
2. A calculator desktop application with a GUI and the ability to store and retrieve data in a suitable file system, using a language of your choice. The requirements are:
* It must include four basic operations (addition, subtraction, multiplication, and division).
* It must be able to calculate and display correct answers to the GUI screen.
* It should also support real numbers (having digits following a decimal point) and be able to calculate percentages.
* Square root and powers operations
* A backspace to allow a user to "undo" input characters or clear the screen
* A function to save and retrieve a specific calculation locally stored to include date and time of transaction to enable traceability.
* Appropriate error handling if a user enters a non-valid key an appropriate error message is displayed.
1. To create a web-based cloud application to store customer details & billing information.

Requirements

* To create and deploy an application with a web-based GUI to include suitable datatype storage for customer name, customer address, date, invoice no, description and invoice totals.
* Appropriate error handling / validation is required.
* The web-based application needs to connect to a suitable server database that enables CRUD statements for the data inputted. Server-side scripting is required.
* The GUI must have the ability to update and delete the invoice total in the database.
* The application and database must be suitable for cloud deployment.

**Your portfolio should contain the following sections:**

* A review and analysis of the requirements
* The design methodology and models you will use
* A design solution
	+ What will be required?
	+ What is your choice of paradigm and language for each solution?
	+ Your algorithms and pseudocode (or modelling language)
* An explanation of the design choices made with reference to
	+ Meeting the requirements and factors influencing the final choice
	+ The advantages your design has over other possible solutions
	+ How you connect the web-based application to the cloud-based server
	+ A justification of the IDE you have chosen
* Documentation of the implementation
	+ Three functional prototypes (you will include all the code as an appendix)
	+ Annotated screen shots of the finished product and key elements of code annotated
* A test plan
	+ A test suite using the template used in class with test conditions and scripts including expected results
	+ Tests should demonstrate that the requirements have been met and the required functionality is covered
* Test results
	+ Actual results need to be recorded as well as pass or fail.
	+ All fails must be described, and their cause and possible solution recorded.
* Evaluation

Evaluate the success of the implementations with respect to the requirements and the factors influencing choices including memory efficiency, execution time, complexity, ease of use and portability.

#

# Section 3: Deliverables

Your submission should be a Microsoft Word document which includes this assessment specification at the beginning. Clearly add your full name and student number to the document header and the file name. There is no fixed word count, but as a guide you should not exceed 4000 words excluding the code listings.

# Section 4: Marking Criteria

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0-29 | 30-39 | 40-49 | 50-59 | 60-69 | 70-84 | 85-100 | Mark & Advice for Improvement |
| **Requirements analysis and design solution****(25%)** | Superficial analysis, unrealistic solutions, few advantages given | Some analysis, poor choice of solutions, limited advantages, no algorithms or models | Adequate analysis of solutions and methodology choices. Key advantages stated, some modelling. | Adequate analysis of solutions and methodology choices. Clear modelling. Comparisons of advantages.  | Good analysis justifying solutions and methodology choices. Full modelling. Comparisons of advantages. | Excellent analysis justifying solutions and methodology choices. Full modelling. Comparisons of advantages. | Excellent analysis justifying solutions and methodology choices. Full modelling. Comparisons of advantages across all three solutions |  |
| **Implementation****(30%)** | The solutions do not operate  | Some operation of the 3 prototypes is possible | Minimum requirements are met for the 3 prototypes. There is a fully functional GUI for the desktop application. | Most requirements are met. There is a fully functional GUI and local storage of data. Coding best practices are evident (comments, indentation, naming) | Most requirements are met, including error handling. There is a functional GUI, local storage of data and a functional cloud-based application transfer of data. Coding best practices are evident (comments, indentation, naming) | All requirements are met, including error handling. There is a functional GUI, local storage of data and a functional cloud-based transfer of data. Coding best practices are evident (comments, indentation, naming) | All requirements are met for all 3 prototypes demonstrating good quality, commented code and logic with a sound syntax in appropriate languages. High quality GUIs and ease of use where applicable.  |  |
| **Test****(20%)** | No test plan or results | Poor test coverage and documentation, some results | Basic functionality tested and results documented | Good coverage and documented results | Full coverage and documented results and fixes. | Full coverage and well organised and documented results and fixes.  | An excellent test plan and results including recommendations for avoiding errors in future code.  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Evaluation****(25%)** | No conclusion  | Superficial evaluation | Adequate evaluation | Requirements mapped to the solutions and evaluated | Requirements mapped to the solutions and evaluated. Reasons given for mismatches | Full evaluation of the suitability of the three solutions to meet the requirements and factors affecting the choice | Full evaluation of the suitability of the three solutions to meet the requirements and factors affecting the choice, including recommendations as to the preferred solution. |  |

# Section 5: Feedback mechanisms

Sufficient time has been allocated in the classroom for the designs to be implemented. During these sessions you can request assistance with any issues that are not related to your designs.

Feedback will be given during the sessions on your implementation. This will ensure you have the opportunity to maximise the use of the classroom resources in developing your solutions.

You will not receive feedback on any of the other sections until after the whole assignment has been marked and returned to you. The feedback will be added to the returned submission.