UNIVERSITY OF LINCOLN

Lincoln School of Computer Science

Assessment Item Briefing Document

Title: CMP1127M Programming and Data Structures: Assessment 2	**Indicative Weighting: 50%**

Learning Outcomes:

On successful completion of this assessment item a student will have demonstrated competence in the following areas:

- [LO2] apply appropriate data structures in common programming solutions;
- [LO3] implement programs consisting of multiple procedures;
- [LO4] apply simple testing techniques.

Requirements

This assignment requires you to design and develop a 'module note taker' type application called "***modnote***". This could be either a Windows Forms application, or a console application. Your application should have the following essential requirements:

- It is designed as a desktop application – do not consider the application on a mobile platform, the Universal Windows Platform (UWP) etc.
- It could be developed as a console application that includes these essential requirements.
- Module information (title, summary content, learning outcomes, assignment due dates etc) will be held in a separate file. This information will need to be imported either by default, or if the user selects that module from a list/drop down box etc.
- As well as being added, modules could be 'deleted'. i.e. remove them from the application.
- Each module should have 'notes' associated with it. The user can create a new 'note' which can contain text and optionally, items such as images and links. Modules could have multiple notes. 'Notes' should be able to be titled (such as 'Lecture 1 notes' etc).
- Assignments should display whether their due date has been reached or not.
- Modules can either be displayed all together (probably a bit visually cluttered), or use other methods, such a tabbed panes, selection from a list of modules etc.
- The application should be tested to make sure it meets these requirements and that it is implemented correctly.

In addition to the essential characteristics, you could also implement:

- Have all assignments showing and sort them according to due date. (Note: this sorting algorithm should be YOUR code, not a standard algorithm accessed through the code)
- Assignments could have their own 'notes'.
- In the 'module notes' or 'assessment notes', links could open in a separate browser or a separate pane.

Your application will be written using C#, should make use of Windows Forms (for the GUI application) and demonstrate object oriented principles in the structure of your code.

The design and testing report should contain the following:

- A design for the application which should include:
  - A written description of the application
  - A top level Use-Case diagram
  - A class diagram for:
    - The Module class
    - The Assignment class
    - The Note class
  - A sequence diagram showing message passing involving at least one of the classes implemented.
- A simple testing strategy for the application which should include:
  - The 'black box' testing strategy and results
    - i.e Testing against the requirements, its description, and presenting the results in a tabular form.
  - A 'white box' testing strategy and results
    - Show testing results for 'method coverage, statement coverage and branch coverage' for at least one action of the code (e.g. creating a module/assessment note and adding content,
- The demonstration video URL.

A short (up to 30 seconds) video of your application running should also be produced and uploaded to YouTube.

Useful Information

This assessment is an individual piece of work. Your work must be presented according to the Lincoln School of Computer Science guidelines for the presentation of assessed written work. Please make sure you have a clear understanding of the grading principles for this component as detailed in the accompanying Criterion Reference Grid.

If you are unsure about any aspect of this assessment component, please seek the advice of a member of the delivery team.

Submission Instructions

The deadline for submission of this work is included in the School Submission dates on Blackboard.

You should submit your work as a single ZIP (it should have the *.zip. file* extension) to the *Assessment Item 2 – Supporting Documentation Upload* section, and the report to the *Assessment Item 2 Upload* section. Use of other compression formats such as RAR files will be penalised.

a) The ZIP file which is uploaded to *Assessment Item 2 – Supporting Documentation* should contain the project files, accompanying module input files, any output files, executable and source code files for your GUI application. The project should be able to be opened in Visual Studio (or any other IDE that you have used – *Mono* for example).

b) The pdf report to be uploaded to *Assessment Item 2 – Upload* should contain:

a. A contents page

b. A basic design for the application including:

i. A written description of the application **(~500 words)**

ii. Class diagrams for the application

iii. A top level Use-Case diagram

iv. A sequence diagram showing an example of message passing between two of the classes involved.

v. Where appropriate, diagrams should be used to accompany your design

c. A description of a simple testing strategy for the application (~500 words) including:

i. The 'black box' testing strategy and results

ii. The 'white box' testing strategy and results

d. The URL of the video

i. The short video should be captured with free software such as Screencast-O-Matic (http://www.screencast-o-matic.com). Download and install the software. Using the 'free version', follow the instructions to capture your video. When the video is complete (no greater than 1 minute in length), select 'Upload to YouTube'. Upload your video as an 'unlisted' video (this allows us to see the video, but only when you tell us of its URL). Full, illustrated instructions for this process will also be available. The video should show your application running while you describe what you have done, how you have implemented it and how it works. You **will not** be assessed on the quality of the recording.

e. A Reference list showing items you have used in your learning that are correctly cited in the body of the report

DO NOT include this briefing document with your submission.

Example marks:

Student A submits a Windows Form application in which:

- Module data is imported and placed on individual tabbed panes and collected in a list
- Multiple notes can be created for each module and are collected in a list, but these only display text
- Assessments are created for each module, but no notes are associated with these
- Assessment due dates are shown in a separate panel for the modules that are added but are not sorted according to due date.
- The application is laid out logically. The code shows some exception handling and commenting.
    - LO2 (Data structures): 75, LO3 (Implementation): 65
- The application is described fully with screenshots, UML class diagrams for Module, Assessment and Note are largely correct, a Use Case diagram is submitted which is correct, a sequence diagram is submitted, although this has some elements missing
    - LO3 (Design): 70
- Black Box testing is presented and tests the requirements of the application. White Box testing is made showing the three different 'coverage' tests for one action. Both strategies are described, and their results are presented in a table.
    - LO4 (Testing): 75
- Student A is awarded 70

Student B submits a Windows Form application in which:

- Module data is imported. This is displayed in a single panel in a table format.
- Notes can be created for each module, they are collected in an array, and they allow text and images to be entered.
- Assessments are created for each module, but no notes are associated with these.
- Assessments are not sorted or shown separately showing their due date.
- The application is shown with one pane, a separate window is created showing 'About' information. No exception handling is included, some commenting is present.
    - LO2 (Data Structures): 45, LO3 (Implementation): 50
- The application is described briefly, UML class diagrams for Module, Assessment and Note are shown, a Use Case diagram is shown but it has errors, and no sequence diagram is included.
    - LO3 (Design): 50
- Black Box testing is presented and tests the requirements of the application. White Box testing is made showing the three different 'coverage' tests for one action – although this is very brief. Both strategies are described briefly.
    - LO4 (Testing): 55
- Student B is awarded 50