UNIVERSITY OF LINCOLN

Lincoln School of Computer Science

Assessment Item Briefing Document

Title: CMP1124M Algorithms and Complexity: Assessment 2	**Indicative Weighting: 70%**

Learning Outcomes:

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

- [LO1] Understand the time and space efficiency of algorithms and how to calculate/estimate/evaluate and improve them.
- [LO2] Determine an appropriate algorithmic approach to a problem.
- [LO3] Ability to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context.

Requirements

This assignment asks you to design and implement a Search and Sort application. In particular, you are to create a Console Application, which will help with the analysis of share prices. This assignment has mandatory (1-3) and additional (4-7) tasks, which will allow you to achieve higher marks.

A set of files is provided: “Close_128.txt”, “Change_128.txt”, “Open_128.txt”, “High_128.txt”, “Low_128.txt”, “Close_256.txt”, “Change_256.txt”, “Open_256.txt”, “High_256.txt”, “Low_256.txt”, “Close_1024.txt”, “Change_1024.txt”, “Open_1024.txt”, “High_1024.txt”, and “Low_1024.txt”.

The files correspond to real share prices of a major bank taken from the London Stock Exchange. The Close_*.txt, Open_*.txt, High_*.txt, Low_*.txt, and Change_*.txt respectively correspond to the Close, Open, High, and Low prices of the stocks, as well as their Percentage Change of share value traded on the day. The 128, 256 and 1024 numbers correspond to the number of stocks stored in the files.

Initially, read the files “Close_128.txt”, “Change_128.txt”, “Open_128.txt”, “High_128.txt”, and “Low_128.txt” into individual Arrays.

Your Console Application should be able to provide the following functionality to the user:

1. Select which individual Array is to be analysed.

2. Sort in ascending or descending order and display the selected Array.

3. Search the selected Array for a user-defined value, if the value exists, then provide its location (if it appears more than once then provide ALL the locations) otherwise provide an error message.

4. Repeat the previous task, but if the value does not exist then provide the value(s) and location(s) of its nearest value.

5. Your Console Application should be in position to input the files with length 256 and 1024. Then repeat Tasks 2 to 4 and display the corresponding values for all the selected arrays.

6. For additional marks, **Merge** the Close_128.txt and High_128.txt files. Then repeat Tasks 2 to 4 and display the corresponding values.

7. For top marks repeat task 6 using the files with length 256 and 1024.

Enhancing your submission for top marks.

Undertake a comparative evaluation for the all the searching and sorting tasks by using different Searching or Sorting algorithms. You should display the number of steps that each algorithm performed. You **should NOT use any built-in sorting and searching functions** from any built-in or external C# library.

A short (up to 1 minute) 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" file to the "*Assessment 2 – Source Code Upload"* section, and a report (in .PDF format only) to the "*Assessment 2 – Report Upload* section". Use of other compression formats such as RAR files will be penalised.

a) The ZIP file which is uploaded to *Assessment 2 – Source Code* should contain the project files, accompanying input files, any output files, executable and source code files for your 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 2 – Report Upload* should contain:
    a. A contents page
    b. A basic design for the application (1 page) including:
        i. A written description of the application.
        ii. Comment about the implementation of tasks 2 to 7.
    c. A description of the algorithmic choices you made for the application (1 page) including:
        i. Justification of selecting and implementing particular searching and sorting algorithms for your application.
        ii. An evaluation of the time and space efficiency of the searching, and sorting parts, as well as your program overall.
        iii. Provide tables of the number of steps for the searching and sorting algorithms you used in respect to the size of the arrays.

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.