University of Central Lancashire UCLan **School of Psychology and Computer Science**	**UCLan Coursework Assessment Brief**	2021 - 2022
	Module Title: Computational Thinking Module Code: CO2412	Level 5
	Alternative Assessment 1: Evaluating the Efficiency of Algorithms	This assessment is worth 60% of the overall module mark

THE BRIEF/INSTRUCTIONS

Module Learning Outcomes

On successful completion of this module a student will be able to:	
1.	Use appropriate methods including logic and probability to reason about algorithms and data structures.
2.	Compare, select and justify algorithms and data structures for a given situation
3.	Analyse the space and time complexity of simple algorithms
4.	Use a range of appropriate notations to represent and analyse problems
5.	Implement and test algorithms and data structures

Assignment Context

You have been employed as a software developer at CompT Enterprises who have asked you to create a custom **sorting function** called `comptSort`, which can be utilised within their own applications to quickly apply different sorting algorithms.

As the lead developer has forbidden the use of any pre-written search functions **you must implement each of the sorting algorithms manually**.

Deliverables

D1

A program, written in the language of your choice, which allows data to be sorted using different sorting algorithms using a function called `comptSort`.

Your implementation of `comptSort` should accept the following parameters:

- `uData` – the original, unsorted data.
- `sort` – a string specifying the sorting algorithm to be used as described in the marking scheme below.
- `asc` – If true, sort is performed in ascending order. If false, sort is performed in descending order (only required for 2:1 or higher).

Your function should return the sorted data once complete.

For lower grade bands your function only needs to be able to sort integers.
For 2:2 and higher your function should be able to sort both characters and integers which have been read in from a file.

D2

A report which acts as a user guide for the function. It should explain in detail how each of the sorting algorithms work and how algorithms can be compared using their asymptotic complexity (Big O Notation).
There is no specific word/page count for this report. Rather, marks will be awarded for depth of explanations and points being supported with appropriate academic references.
You may use diagrams, code snippets and screenshots to support your explanations.

You should use the **APA** referencing format for your report.

Marking Scheme

Marking bands are indicative and can be overridden at the marker's discretion with justification.

40 – 49% (Pass Criteria)

- Suitable data structures are created and initialised with appropriate unsorted integers.
  At this level it is acceptable for the unsorted data to be hard coded.

  (10 marks)
- `comptSort` function has been created with specified parameters.

  (10 marks)
- Bubble Sort has been implemented correctly and returns the sorted data.

  (10 marks)
- Report contains a short description of why Big O notation is a useful tool for comparing the efficiencies of algorithms which is not supported by any references.

  (5 marks)
- A brief description of how Bubble Sort works is included within the report but does not include any discussion of its asymptotic complexity.

  (5 marks)

- Up to an additional **10 marks** will be awarded for good programming practices such as commenting, naming conventions and correct indentations.

50 – 59% (2:2 Criteria)

- Unsorted data is imported into the program from unsorted text files which may contain either integers or characters.
  Depending on what language you choose to implement your solution in you may need to make some modifications to your comptSort function to accommodate characters as well as integers.
  You should explain any changes you make (or why you do not need to make any) within your report.

  (2 marks)
- Insertion Sort has been implemented correctly and returns the sorted data.

  (3 marks)
- Brief discussion of why Big O is a useful tool for comparing algorithms.
  Brief explanations are included of how both Insertion Sort and Bubble Sort operate with the best, worst and average time complexities being listed but not explained.
  Space complexity for both algorithms is also listed.

  (3 marks)
- Up to three appropriate references have been used from either academic (papers, books, etc.) or non-academic (websites) sources.

  (2 marks)

60 – 69% (2:1 Criteria)

- Binary Insertion Sort is implemented in addition to the previously listed algorithms.

(2 marks)

- The comptSort function allows for sorting to be performed in both ascending and **descending order**

(3 marks)

- Good discussion of why Big O notation is a useful tool for comparing algorithm.
  Good discussion of how each of the algorithms work.
  The time and space complexity have been listed and compared with appropriate conclusions about which is the most efficient algorithm being made.

(3 marks)

- Up to five appropriate references have been used from either academic (papers, books, etc.) or non-academic (websites) sources.

(2 marks)

70 – 79% (1:1 Criteria)

- Merge Sort has been implemented in addition to the previously listed algorithms.

(2 marks)

- Very detailed discussion of why Big O notation is a useful tool for comparing algorithm.

(3 marks)

- Detailed discussion of how each of the algorithms work **which is supported by diagrams. Note:** You should produce your own diagrams which use the same data you are using to test your program.

  The time and space complexity have been listed and compared with appropriate conclusions about which is the most efficient algorithm being made.

(3 marks)

- At least 6 references from academic sources (papers, journals, etc.)

(2 marks)

80% + (High 1st)

In addition to 1:1 criteria…

- Quick Sort has been implemented and explained in the report with the support of diagrams.

(5 marks)

- Detailed discussion and analysis of the algorithms with discussion about limitations of each algorithm.
  I.e., consider different use cases for sorting data – is the data completely random, or is already sorted data being updated? Make clear recommendations based on asymptotic complexity.

(10 marks)

- All points made are supported by academic literature.

(5 marks)

PREPARATION FOR THE ASSESSMENT

Before attempting this assessment you should refresh yourself with the following topics:

- Reading files (first year content)
- Big O notation
- Sorting algorithms

The following book is available online and from the library which can be a starting point for your reading:
Introduction to Algorithms, Third Edition

You should also decide on which language you wish to implement your solution in. Python or C++ is recommended.

RELEASE DATES AND HAND IN DEADLINE

Assessment Release date: 18th November 2022 Assessment Deadline Date and time: 22nd April 2022, 23.59 hours.

Please note that this is the latest time you can submit – not the time to submit!
Your feedback/feed forward and mark for this assessment will be provided on 18th January 2021 when feedback can be accessed.

SUBMISSION DETAILS

You should compress your work into a single zip file and submit using the assignments tab on Blackboard. **Do not use other compression formats**.

HELP AND SUPPORT

- Questions regarding this assessment should be asked through the CO2412 Module Teams Channel.
- For support with using library resources, please contact Bob Frost, RSFrost@uclan.ac.uk or SubjectLibrarians@uclan.ac.uk. You will find links to lots of useful resources in the My Library tab on Blackboard.
- If you have not yet made the university aware of any disability, specific learning difficulty, long-term health or mental health condition, please complete a Disclosure Form. The Inclusive Support team will then contact to discuss reasonable adjustments and support relating to any disability. For more information, visit the Inclusive Support site.
- To access mental health and wellbeing support, please complete our online referral form. Alternatively, you can email wellbeing@uclan.ac.uk, call 01772 893020 or visit our UCLan Wellbeing Service pages for more information.
- If you have any other query or require further support you can contact The <i>, The Student Information and Support Centre. Speak with us for advice on accessing all the University services as well as the Library services. Whatever your query, our expert staff will be able to help and support you. For more information , how to contact us and our opening hours visit Student Information and Support Centre.
- If you have any valid mitigating circumstances that mean you cannot meet an assessment submission deadline and you wish to request an extension, you will need to apply online prior to the deadline.

Version: 1