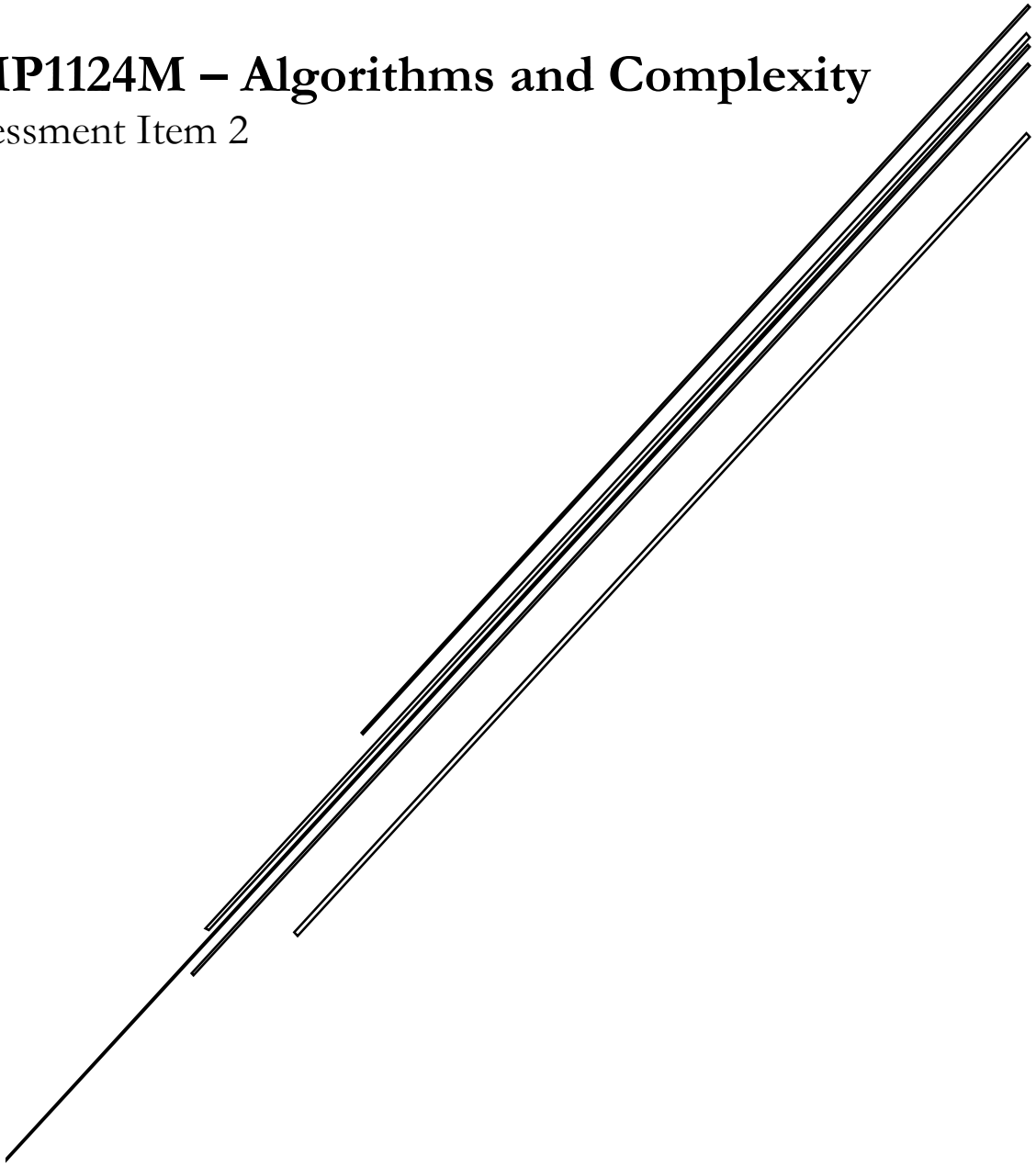
# CMP1124M – Algorithms and Complexity

Assessment Item 2



University Of Lincoln

Computer Science

Ling Kan

KAN13489110

# Contents



# Application

The Weather Data program is used to search and sort data of two weather stations; Lerwick and Ross on Wye. This information is taken from the Met Office Historical Data web site. It includes a range of data including the Mean Daily Temperature Maximum (in C), Mean Daily Temperature Minimum (in C), Days of Air Frost, Total Rainfall (in mm) and Total Sunshine duration (in hours).

In this application there is a simple menu allowing users to choose specific functions and data they would like to search and sort information from. Users are also able to see the range of data in one table, enabling you to see the corresponding information to any specific row of data given. This program also includes a range of algorithms, which is used to solve problems by using a step-by-step procedure.

Users are able to sort data of any chosen data range, in either ascending or descending order. This helps organise and find values quickly. It also makes it easier to observe the range of data, making it easier to identify patterns. This information can also be found by using the maximum, minimum and median function.

Another option for users to find data is using the search function. This filters the amount of data and shows all the corresponding values, making it easier to look for specific information. To ensure there are no human errors when typing in the search function, a while loop statement has been implemented. This will show an error message and repeats certain commands if the incorrect value is entered.

During the implementation of this program, I made sure the code was organised by creating several classes for each search and sort algorithm. I also inserted switch statements, which allows users to specify function and data type they would like to use.

## Choice of Algorithms

***Sorting***

Before implementing a sorting algorithm within the program, I considered the different options that to make sure I use the most efficient algorithm. I first considered BubbleSort, as it was the easiest to implement. However, it cannot deal with a large list of data. Meaning that is a sorting method is unsuitable for this program as there are many arrays of data. Similarly, with selection and insertion sort they are also good for sorting small lists, as they do not need additional temporary storage. Therefore, these sorting methods are also unsuitable due to the unreliable time in which it runs at.

Finally I considered quick and merge sort. These are very similar as they both use divide-and-conquer recursive algorithm, and can sort large amounts of data quickly. However, the MergeSort requires more memory and requires secondary storage. While as QuickSort partitions in place by using two pointers and swapping elements within the array. Therefore, even though the MergeSort is faster in terms of speed. As the worst time complexity for QuickSort is O(N^2), while MergeSort is O(NlogN). Based on this information I have chosen to apply the QuickSort, as it is one of the most efficient algorithms for this program, as it can handle a large list of data. I also implemented this sort method to find the median values of each array.

***Searching***

When choosing a search algorithm I decided not to choose interpolation search as it is difficult to implement, in the time given for completion. Even though the runtime complexity is potentially quicker, as its average case is O(log(log(N)). However the worst case is O(N), meaning that it can be slow. This is similar to the linear search, with a best case of O(1). This search method works well in particular when the list is not listed in order. However, the drawback is that it can be slower than any other search algorithm.

While as for binary search and interpolation search, the average time complexity is quicker, although the data will have to be sorted before searching. Therefore, I have chosen to use the linear search algorithm. As the current array of data is not sorted. To improve on this further I should use the binary or interpolation sort as the runtime is quicker.

## Video URL

https://youtu.be/AodeBu25VWc

(Annotations will need to be turned on when playing this video.)

# Bibliography

Softwareandfinance.com. (2016). *C# Sorting Algorithm - QuickSort Recursive*. [online] Available from: http://www.softwareandfinance.com/CSharp/QuickSort_Recursive.html [Accessed 20 April 2016].

Oxfordmathcenter.com. (2016). *Searching and Sorting Arrays*. [online] Available from: http://www.oxfordmathcenter.com/drupal7/node/31 [Accessed 5 April 2016].

Snipd.net. (2016). *Quicksort algorithm in C#*. [online] Available from: http://snipd.net/quicksort-in-c [Accessed 5 April 2016].

***Report***

Wandy, J. and Wandy, J. (2016). *The Advantages & Disadvantages of Sorting Algorithms | eHow*. [online] eHow. Available from: http://www.ehow.com/info_8446142_advantages-disadvantages-sorting-algorithms.html [Accessed 5 April 2016].

Puntambekar, A.A. (2008). *Data Structures And Algorthims*. 2nd edition. Pune, India: Technical Publications Pune.

Includehelp.com. (2016). *Searching (Linear/ Sequential, Binary and Interpolation Searching) Data Structure Tutorial with C & C++ Programming | includehelp.com*. [online] Available from: http://www.includehelp.com/data-structure-tutorial/linear-sequential-binary-interpolation-%20searching-in-data-structure-tutorial.aspx [Accessed 5 April 2016].