CMP1124M Algorithms and Complexity: Assessment 2

Table of Contents

Application Description

For this Assessment, I needed to design and create a console application to import two station's weather data from text files and allow the user to analyse the data via searching and sorting. The application I have created allows the user to import the data from the text files for both weather stations; the user can choose whether to import just one weather station's data, or both station's data, and for ease of use even import everything together.

The application allows the user to select each set of data to view, or manipulate. The user can sort the data in multiple ways, such as in ascending or descending order; the user can search for collections of data too, e.g. search for a specific year or month, and retrieve all data from the selected sets of data for that year, or month. The user could also retrieve statistics of the data, for example the minimum and maximum, or the median values of the selected data sets.

The application I have created implements all 1-9 tasks; I implemented task 3 by converting a dictionary's values to a list, and then using a binary search to find if the dictionary contained the value the user was searching for. The 4th task is implemented by using the array contains method, which uses a linear search, this is because the array containing the months is not sorted, whereas the year's dictionary is. For task 6, I used IEnumerable's min and max methods, which utilize a single for loop each to find the minimum and maximum value in the collection. To implement task 7, I first sorted the data using a dual-pivot quicksort (or insertion sort if the data set is small) and then took the middle value of the data set.

The application accomplishes task 8 by retrieving an array of ordered keys from the sorted data which allows for looping through and retrieving each selected data sets data for that specific key rather than changing the data stored in the dictionaries.

The Application also implements the enhanced task, of outputting to a HTML webpage, rather than the console, allowing for easier viewing; the program will try to open the webpage upon creation in the users default program.

Algorithms

For my application I decided to use dictionaries for storing the weather data, because I would mostly be searching for specific years and months of data; I chose a dictionary because it implements a hash table, which has an average time complexity of O(1), and a worst case time complexity of O(n) on searching. This mean using a hash table was more ideal than binary or linear searches because a binary search with best case performance is on par with the hash tables O(1) time complexity. However, I implemented the binary search when searching for a specific year, because the years already sorted and the binary search performs better than the linear search with a O(log n) time complexity compared to linear searches O(n); I also utilized the linear search to check if the month's array contains the month the user input, I used this because the array is not sorted, and is small, so sorting the array first would not be beneficial.

My application utilizes two different sorting algorithms, the insertion sort and the quicksort (although a dual-pivot version), I chose to use the insertion sort because for small collections of data it outperforms the more complex algorithms like merge-sort and quicksort. I chose to use the quicksort because the merge-sort requires more auxiliary space of O(n) whereas the quicksort only requires O(log n). While the merge-sort is a stable algorithm, I didn't need the input order to stay the same because I was retrieving the dictionaries keys, and not the values in order. The dual-pivot quicksort I chose to implement is faster than the classical quicksort by utilizing 2 pivots, and as a result uses less swaps. (Yaroslavskiy, 2009)

In conclusion, I believe the sorting and searching methods I have used provide very good time efficiency due to the fast performances on the hash tables, and quicksort algorithm. The algorithms for searching and sorting are also very space efficient, with the largest space complexity being O(n). These have allowed the program to be very fast in sorting and searching through the data.

Video URL

https://youtu.be/49-IUuR-Mn8

References

Yaroslavskiy, V. (2009, September 11). *Replacement of Quicksort in java.util.Arrays with new Dual-Pivot Quicksort*. Retrieved from core-libs-dev@openjdk.java.net: http://permalink.gmane.org/gmane.comp.java.openjdk.core-libs.devel/2628