

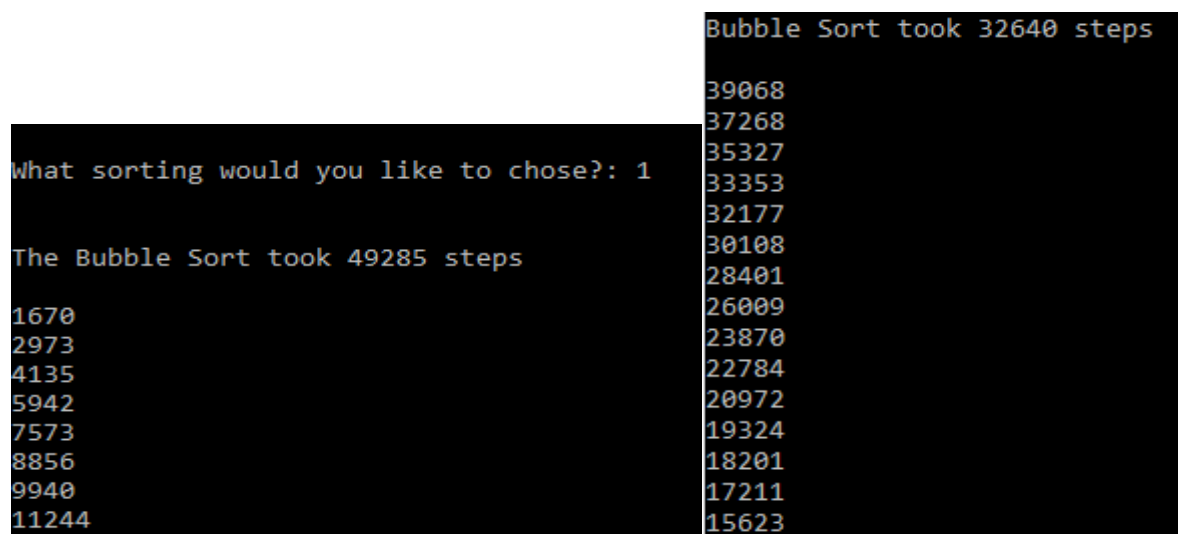
TABLE OF CONTENTS

Basic design for the application.....	2
Task 1.....	2
Task 2.....	2
Task 3.....	2
Task 4.....	3
Task 5.....	3
Task 6.....	3
Task 7.....	3
Justifying algorithms, I choose.....	3
Evaluating time and space efficiency.....	3 - 5
Number of steps taken.....	4
References.....	4 - 5

BASIC DESIGN FOR THE APPLICATION

The program initializes the class and main function following which it starts to load each of the individual text files provided to us into separate individual Arrays. Next, the user is presented with which array they would like to select to load and send it to sort and search functions. I have used Userchoice in Switchcase statements to record the users input when the menu is presented to them. In the second Switchcase I have used it to link each array to the sort and searching variables that have been introduced below. I created the Sorting algorithm that imports the selected array and asks the user if they want to use the bubble sort or Insertion sort. Next, the code will filter the 256 arrays and 2048 arrays so that the code can display every 10th or 50th element respectively, after sorting it. It will display the sorted array elements first in ascending order and upon pressing the enter key it will display it in descending order. The Searching function is where I have placed the Search algorithm, using Switchcase which lets the user select if they want to search with linear search or Binary search. I have separate classes for implementing each individual searching and sorting algorithms.

I also implemented Step counter variable inside the loops for all the sorting and searching algorithms used in my code to count the steps performed by them. This is displayed before the program prints out the sorted array for both ascending and descending



```
What sorting would you like to chose?: 1

The Bubble Sort took 49285 steps
1670
2973
4135
5942
7573
8856
9940
11244
39068
37268
35327
33353
32177
30108
28401
26009
23870
22784
20972
19324
18201
17211
15623
```

Task 1- I created 6 separates with variables with names corresponding to the filename and loaded the elements of the text file into the arrays in the variables.

Task 2- I used switch case statements to present the user with the menu where they are presented with the sorted array, both for Ascending and descending sequences. I have also included a, for loop that will determine if the array selected has 256 elements in the array and if it is then only display every 10th value in the sorted array.

Task 3- I have used the Search function to allow the user to declare what value they want the search function. I have connected the linear search. function class to the main file by calling the class and function and it give the index number and if else statement throws an error if the number does not exist.

Task 4- I did not manage to implement the code to show the nearest values to the user specified value

Task 5- My code uses Switchcase to ask the user which sorting algorithms they want to use. Once the user choses the array will be sorted. My code will check if the user chose text file with 2048 elements and with the help of for loop will sort the values and inly display every 50th value in the array in ascending and descending order to the user.

Task 6- I was not able to merger the text files mentioned in the brief for task 6

Task7- I was not able to merge the 2048 text file arrays and despite numerous attempts to get it to work.

ALGORITHM CHOICES

Bubble sort: The logic behind how Bubble sort code works was easy to understand by referring to some online resources. With help I was able to implement it in my own way. The data was sorted in place, so a little so very little memory space is used for doing it. Due the array sizes not being very bid the compilation time is not that bad.

Insertion sort: When the arrays values are close and easy to sort, this algorithm is very efficient, so this was the perfect for me to use to sort my arrays. It also sorts out the values in place so the impact on memory is minimal. The code for Insertion sort was easy to understand and adapt for my use case. Also, I was not dealing with big arrays, so it is very efficient when it comes to sorting out the elements of my arrays.

Linear search: The code to implement linear search algorithm was easy to understand and I was able to adjust the algorithm I found online to implement in my code and allow me to count steps taken by it and find the index location of the number entered by the user, if it exists in the selected array. It is also beneficial for sorting fixed small size arrays that I have.

EVALUATING THE TIME AND SPACE EFFICIENCY

Algorithm	Space Complexity	Time complexity (Average)
Linear Search	$O(1)$	$O(1)$
Binary Search	$O(1)$	$\log_2(n)$
Insertion Sort	$O(1)$	$O(n^2)$
Heap Sort	$O(1)$	$O(n \log(n))$
Quick Sort	$O(\log(n))$	$O(n \log(n))$
Bubble Sort	$O(1)$	$O(n^2)$

Bubble sort is the slowest of all the existing sort algorithms and its time complexity depends on the length of the array $\{o(n)$ for the best case and $o(n^2)$ for the worst case}. Since my 256 arrays are short so they are sorted very quickly but the 2048 arrays take more time to be sorted as the number of fields in it are many times more.

Insertion sort is many times quicker than bubble sort, this is because it has $O(n^2)$ complexities and therefore are more time efficient. Even so, Insertion sort still can suffer if the array is bigger in size

{ $O(n)$ for the best case and $O(n^2)$ for the worst case}. Since my 256 arrays are short so they are sorted very quickly but the 2048 arrays take a few more steps to be sorted as the number of fields in it are many times more.

NUMBER OF STEPS TAKEN

Search or Sort Algorithm	Number of steps taken by the Algorithm			
	256 Array size		2048 Array size	
	For Ascending	For Descending	For Ascending	For Descending
Bubble sort	49,285 steps	32,640 steps	31,51,493 steps	20,96,080 steps
Insertion sort	255 steps	1206 steps	2047 steps	14890 steps
Linear search	250 steps for value 1158		2000 steps for value 1074	

References

Halpern, P., 2017. *What Are The Advantages And Disadvantages Of Binary Search?* - Quora. [online] Quora.com. Available at: <<https://www.quora.com/What-are-the-advantages-and-disadvantages-of-binary-search>> [Accessed 12 March 2020].

Nayak, A., 2018. [online] Quora. Available at: <<https://www.quora.com/What-are-the-advantages-and-disadvantages-of-linear-search-over-binary-search>> [Accessed 12 March 2020].

Wandy, J., 2020. *The Advantages & Disadvantages Of Sorting Algorithms*. [online] Sciencing.com. Available at: <<https://sciencing.com/the-advantages-disadvantages-of-sorting-algorithms-12749529.html>> [Accessed 12 March 2020].

n.d. [ebook] Available at: <https://blackboard.lincoln.ac.uk/bbcswebdav/pid-2629732-dt-content-rid-4805342_2/courses/CMP1124M-1920/CMP1124M_Workshop_5_wc17022020.pdf> [Accessed 12 March 2020].

n.d. [ebook] Available at: <https://blackboard.lincoln.ac.uk/bbcswebdav/pid-2629734-dt-content-rid-4814728_2/courses/CMP1124M-1920/CMP1124M_Workshop_6_wc24022020.pdf> [Accessed 12 March 2020].

Choudaiah, S., Chowdary, P. and Kavitha, M., 2017. *Evaluation Of Sorting Algorithms, Mathematical And Empirical Analysis Of Sorting Algorithms*. [online] Ijser.org. Available at: <<https://www.ijser.org/researchpaper/Evaluation-of-Sorting-Algorithms-Mathematical-and-Empirical-Analysis-of-sorting-Algorithms.pdf>> [Accessed 12 March 2020].

Bodnar, J., n.d. *Arrays In C# - Working With Arrays In Csharp*. [online] Zetcode.com. Available at: <<http://zetcode.com/lang/csharp/arrays/>> [Accessed 12 March 2020].

Docs.microsoft.com. 2020. *Built-In Types - C# Reference*. [online] Available at: <<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/built-in-types>> [Accessed 12 March 2020].

Eddiejackson.net. 2020. *C# Text File And Arrays – Lab Core / The Lab Of Mrnettek*. [online] Available at: <<http://eddiejackson.net/wp/?p=11817>> [Accessed 12 March 2020].

Eddiejackson.net. n.d. *C# Text File And Arrays – Lab Core / The Lab Of Mrnettek*. [online] Available at: <<http://eddiejackson.net/wp/?p=11817>> [Accessed 12 March 2020].

GeeksforGeeks. n.d. *Binary Search - Geeksforgeeks*. [online] Available at: <<https://www.geeksforgeeks.org/binary-search/>> [Accessed 12 March 2020].

GeeksforGeeks. n.d. *Linear Search - Geeksforgeeks*. [online] Available at: <<https://www.geeksforgeeks.org/linear-search/>> [Accessed 12 March 2020].

Randall, M., n.d. *Adding A Counter To A Quick Sort Algorithm To Display The Total Number Of Swap Actions*. [online] Stack Overflow. Available at: <<https://stackoverflow.com/questions/49631416/adding-a-counter-to-a-quick-sort-algorithm-to-display-the-total-number-of-swap-a>> [Accessed 12 March 2020].

Tripathi, P., n.d. *Binary Search Implementation Using C#*. [online] C-sharpcorner.com. Available at: <<https://www.c-sharpcorner.com/blogs/binary-search-implementation-using-c-sharp1>> [Accessed 12 March 2020].

youtube. 2014. 5. *How To Program In C# - SWITCH STATEMENTS - Beginner Tutorial*. [online] Available at: <<https://www.youtube.com/watch?v=Qs-LAYkp9YU>> [Accessed 12 March 2020].