Lab G

Goals-
Implement simple algorithms for searching and sorting

1. Create data files.

You will need at least two text files filled with random integer values. You don't need a large number of values. Around 20. This will make testing easier. Create a file with a random collection of the values; 1, 2, 3, 4, 5, 6, 7, 8, & 9. Save it. Make a copy. In one copy add a single value of 0. Keep one without a 0.

NOTE- We use 20 values only for testing in this lab. Your code should accept input of much larger size, so you must use a dynamic array or possibly a linked list.

NOTE: For each file you use you should use exceptions to catch file errors and report them.

2. For each program you will prompt the user for the input file name. For #4 you will also prompt them for the output file name. You will implement each with a separate source and header file. That will make it easier to add to the "toolkit" of software tools you should be maintaining. Then you will need 1 program file with a menu for testing.

3. Search for the target value (i.e. 0).

Implement a linear search in a program that searches for a user-specified target value in your data files. You cannot use binary search yet.

4. Sort a set of values.

Find an algorithm for bubble sort. Implement it in a program. Your program should write the sorted results into a file using a name the user provides. Sort both input files and save the results.

5. Search for the target value, redux.

Find an algorithm for binary search. Implement it in a program that searches for a user specified target value in your data file. Remember that you cannot use any of the original files, as you need a sorted file. Search for a 0 in both output files you created in #4. Does it correctly report not finding the 0? Search for another value.

WARNING: You are tasked with finding an algorithm for each search or sort. Do NOT borrow any code! For # 3, 4, & 5 put your algorithm in the file as a comment. Include the URL or other citation as to where you found it. Pseudoocode is fine as long as it is not C or C++ code that the author refers to as "pseudocode".

NOTE: You can create separate programs or create a single program with a menu for the user.

What to submit-

You will submit the following files to TEACH-

Code to implement your searches, both header and source files (4 files)

Code to implement your bubble sort, both header and source files (2 files)

Code to demonstrate the operation of your searches and sort.
(1 or 3 files)

Makefile

All files in a zip archive

Grading

Programming style- 1 point

Implement and test the linear search algorithm- 2 points

Implement and test the bubble sort algorithm - 3 points

Implement and test the binary search algorithm- 3 points

Correctly use dynamic memory- 1 point