Homework 4

100 Points

Structures and Strings

22B_H4A_Str_Move.cpp (c-strings)
22B_H4B_Ptr_Sort.cpp (find and fix errors)
22B_H4C_Performers.cpp **Project: Star Search** (See next pages)

Grading

Program 4A	– 15
Program 4B	– 15
Program 4C	
1. Display welcome/farewell messages	– 5
2. Read names and scores from file	– 15
3. Insertion Sort	– 15
4. Write sorted array to file	– 15
5. Display the highest score, winners' names and scores	– 15
Self Assessment Report	– 5

Run the program as required and save the output at the end of the source file as a comment. Compress the source file, input and output files (if any), and upload the compressed file: 22B_LastName_FirstName_H4.zip

Project: Star Search

Project Star Search: A particular talent competition has five judges, each of whom awards a score between 0 and 10 to each performer. Fractional scores, such as 8.3, are allowed. A performer's final score is determined by dropping the highest and the lowest score received then averaging the three remaining scores.

Write a program that does the following:

1. Reads names and scores from an input file into a dynamically allocated array of structures. The first number in the input file represents the number of performers.
2. Calculates the final score of each performer.
3. Sorts the array in descending order by the final score using the insertion sort algorithm.
4. Writes the sorted array to a file, in a table format with a header of your choice and data nicely aligned (strings to the left, numbers to the right, aligned by the decimal point). The output file's name is created by adding "**out**" to the input file's name. For instance, if the input file's name is **performers.txt**, the output file's name will be **performersout.txt**
5. Displays highest score, the winner's names and scores. If two or more performers have the same highest score, all are considered winners. Display the output in a readable format of your choice.

The **Performer** structure has three fields:

name (string)	such as	**John Lee**
scores (an array)		**7.5 7.5 7.5 7.5 7.5**
final (double)		**7.5**

Input: Read data from an input file. Assume the input files have been validated. However, your program should detect if a file is empty and display an error message. First create the input files: copy and paste the following data into new text files or use the existing text files.

performers.txt

```
8
John Lee; 7.0 7.8 7.1 7.9 7.5
David T. Ng; 8.3 2.9 9.8 9.2 9.7
Mary Johnson; 8.3 8.0 8.9 7.9 8.5
Andy V. Garcia; 9.1 8.9 9.0 8.7 9.1
Ann Peterson; 9.0 8.8 9.1 9.7 9.3
Lucy A. Smith; 8.2 8.4 8.9 9.3 8.5
Dan Nguyen; 9.0 5.6 8.9 9.9 7.3
Sue K. Warren; 7.9 8.2 7.6 8.1 8.0
```

contestants.txt

```
11
Linda Johnson; 9.3 9.6 9.9 9.9 9.0
Steve Chen; 8.3 2.9 9.8 9.2 9.7
Mary Livingston; 8.3 8.0 8.9 7.9 8.5
Tom A. Retter; 9.1 8.9 9.0 8.7 9.1
Dana Wu; 9.0 8.8 9.1 9.7 9.3
James Michael Owen; 9.9 9.6 9.9 9.0 9.3
Marie Kondo; 9.6 9.0 9.9 9.9 9.3
Bob Schuster; 8.2 8.4 8.9 9.3 8.5
Kevin Yu; 8.3 9.2 9.8 2.9 9.7
Tina Queen; 9.0 9.6 9.9 9.9 9.3
Jonathan Edwards; 9.9 9.6 9.0 9.9 9.3
```

test.txt

```
7
John Lee; 8.3 7.5 8.9 9.9 8.6
David T. Ng; 8.3 8.9 9.9 8.6 7.5
Mary Johnson; 8.3 7.5 8.9 8.6 9.9
Andy V. Garcia; 7.5 8.3 8.9 9.9 8.6
John Lee; 9.9 8.3 7.5 8.9 8.6
Ann Peterson; 8.3 8.9 9.9 7.5 8.6
Dan Nguyen; 8.3 9.9 7.5 8.9 8.6
```

testempty.txt