Programming Assignment #6C: Sorting an Array of Structures

Adding sorting services into Programming 6B:

We want to extend programming assignment #6B further to allow the user to sort the student records based on either (i) the scores in the records or (ii) the last names and the first names.

Sorting by scores:

When we do sorting based on the scores, we want to sort the records from the ones with the lowest score to the ones with the highest score in ascending order. (It will be accepted too if you want to implement it as sorting from the highest to the lowest by score instead.)

Sorting by names:

When we do sorting based on names, we want to sort the records according to the alphabetical ascending order of first names and last names described in the following.

- **Note on string comparison**: In C++, you can directly compare **the alphabetical order** of two strings X and Y using the relational operators such as <, >, ==, and so forth. For example, if *stdRecords[i].lastName* has the contents "Lin" and *stdRecords[j].lastName* has the contents "Johnson", then the logic expression (*stdRecords[i].lastName* > *stdRecords[j].lastName*) is true while the logic expression (*stdRecords[i].lastName* < *stdRecords[j].lastName*) is false. See more about the string class in Section 10.7 of the textbook.
- **Comparing the last names first**: If the last names are different, the order of two student records is determined by the **alphabetical order of last names**.
- **Comparing the first names further if the last names are the same**: If the last names are the same, the order of two student records is determined by the **alphabetical order of first names**.
- **Example:** Consider four records with the following names (first name followed by last name): *David Lin*, *Linda Johnson*, *Cathy Johnson*, and *Jonny Cash*. According to the **alphabetical order** of names, the records should appear as *Jonny Cash*, *Cathy Johnson*, *Linda Johnson*, and *David Lin*.

Requirements:

- **Implement two sorting functions**: Refresh your understanding of the implementation of either bubble sort or selection sort in Section 8.3 of the textbook for sorting numbers in **an array of integers**. Now adapt the code in the book to implement (i) a function ***sortByScore*** that can sort **an array of student records** by scores and (ii) another function ***sortByName*** that can sort **an array of student records** by names respectively as show below before the main function.

```
// …
void sortByScore ( stdRecord myArray[ ], int numOfRecords )
{       // Fill in your code to sort the numOfRecords stfRcords
//      in myArray based on the scores stored in these records
}
void sortByName ( stdRecord myArray[ ], int numOfRecords )
{       // Fill in your code to sort the numOfRecords stfRcords
//      in myArray based on the last names and first names
//      stored in these records
}

int main( )
{       // …
}
```

- **Provide the two sorting services under the service menu**: In your program you should add two more options under the service menu for the user, corresponding to these two sorting services. When the user selects one of these two new options, call the corresponding sorting function to sort the array first and then display the contents in the new order.