2E3 Lab 5

In this Lab, you will write a program that reads student data from a file, sort it alphabetically using a linked list and then write the sorted data to an output file.

The input file is called student176.txt. It contains one **Student** record per line comprising lastName, firstName, student ID and exam mark. The first 4 lines of student.txt are as follows:

COAKLEY	Thomas	08000010	80
NORTON	Helen	08000059	56
MANNION	Shawn	08000015	73
O'TOOLE	Nicholas	08000088	47

The records can be read from the input file using the >> operator in the same way that input can be read from the console. The only difference is that a named file is explicitly opened instead of using **cin**. The code below creates an input file stream **in**, opens it specifying a file name, reads data from the file until the end of file (eof) is reached and finally closes the file. The input file should be placed in the top level directory of your Visual Studio project and you can assume that the input file is correctly formed. You will need to include `<fstream>` in order to use file input/output.

```
#include "studentList.h"      // StudentList
#include <iostream>           // cout
#include <fstream>            // ifstream

using namespace std;
…


ifstream in;
in.open("student2015.txt");
if (in.fail()) {
     cout << "unable to open student2015.txt" << endl;
     return 0;
}


while (!in.eof()) {
     Student *student = new Student();
     in >> student->lastName >> student->firstName;
     in >> student.id >> student.mark;
     …
}
in.close();
```

Each input record should be read into a **Student** object which should then be added to a **StudentList** object. **The StudentList object must maintain a linked list of Student objects.** The list should be sorted in ascending alphabetic order by surname. To ensure this, when you are inserting a new Student into the list, make sure to insert it into its correct alphabetical position, by comparing it to Students already in the list. If 2 students have the same surname, list them in the alphabetical order of their first names.

To compare 2 string variables you can use method **compare** provided in <string> library. To compare strings a and b: int r = a.compare(b); The function returns 0 if all the characters in the compared contents compare equal, a negative value if the first character that does not match compares to less in a than in b (i.e., if the letter occurs earlier in the alphabet), and a positive value in the opposite case.

The program should read in the student data and output the sorted data to studentSorted.txt.

Data can be written to an output file using the << operator in the same way that data can be written to the console. The only difference is that a named file has to be explicitly opened instead of using **cout**. The code below creates an output file stream **out**, opens it specifying a file name, writes data to the file and finally closes the file. The output file should be created in the top level directory of your Visual Studio project.

```
ofstream out;
out.open("student2015Sorted.txt");
if (out.fail()) {
     cout << "unable to open student2015Sorted.txt" << endl;
     return;
}
out << "COAKLEY   Thomas      08000010    80" << endl;
out << "NORTON    Helen       08000059    56";
out.close();
```

The student2015.txt input file can be found on the 2E3 web site.

Note that the example above writes fixed strings to the output file, while you will have to write the description of each student.

EXTRA CHALLENGES

1. Use the function setw (http://www.cplusplus.com/reference/iomanip/setw/) and setfill (http://www.cplusplus.com/reference/iomanip/setw/) in (#include <iomanip>) to format the output.
2. Make sure that your output file can be used as input to your program
3. Implement different modes of sorting, such as by student id or by the mark. Enable user to select which sorting option to use.

Suggestions:

First make sure you can read/write to/from file. Write the code for this in your main function and test it with the input file.

Decide which data members, constructors and methods will you need in your student class. Define and implement Student class (student.h, student.cpp). The class should include a method to compare 2 students which you will need to use when inserting new students into the list. Test the class behaves as expected by creating 2 instances of the class in your main function and comparing the Students using the compare method.

After that, define and implement the linked list that stores the list of students (studentList.h, studentList.cpp). Decide which data members and methods will this class need (check out lecture notes on linked lists – note we are now using *Student instead of *Node)

Marking schema:

0 = did not attend the lab session
---
1 = attended the lab session but have no or very little working code or **THE CODE DOES NOT COMPILE. Your code MUST compile in order to get a mark higher than 1.**
1= file in/out
3 = student.h/.cpp (1 of 3 for comparing students correctly)
2 = studentList.h
3 = studentList.cpp (2 out of 3 go for add method)

Max 10 points

The program needs to be DEMONSTRATED for marks to one of the demonstrators before the end of your lab session on December 2nd/3rd AS WELL AS SUBMITTED ONLINE by Friday December 4rd on blackboard (**submit all .h and .cpp as well as student*.txt files**).