Debug I

Download Debug I

Lab I

Background

String streams are an excellent way to convert strings into other types of data, such as an integer, a double, and so forth, or to handle lines where some fields are optional. For example, in Lab I, you will be reading a student's first name and last name and a set of scores. Each student may have no scores, or they may have a large number of scores. If you used the traditional extraction operators (fin >> first >> last >> scores), you would wrap to the next line and start reading another student's information. Instead, with a string stream, we can get an entire line and then break it apart. This keeps us safe by knowing we're only operating on one student at a time.

Problem

You will be writing a program that inputs a set of students and their scores from a file. Then, your program will generate a report to a file summarizing the average score for each of the student. However, the Dean gave you the list of students in reverse order. So, you will be required to read in the entire student report into an array (up to a maximum of 10 students), reverse the array, and then generate the report.

Student Structure (Class)

To start this problem, create a structure called **Student** that contains three public member variables:

A student's first name, their last name, and their average score.

There are no member functions.

Functions

Your program makes use of 3 functions: 1) reading the input file into an array, 2) reversing the array, and 3) generating the report. You must create three functions to complete these three phases.

Reading Input Function

Returns: an integer representing the number of students that were read from the file (up to the maximum number of students)
Parameters: an input file stream object of an already opened and checked file, an empty array of **Student** structures to store the file information into
Purpose: this function will read the input file stream parameter into the Student structures array. You will need to read the input file line-by-line and break apart the fields using a string stream. You do not know how many students will be input so be sure to count and not to go past the capacity of the array.

The input file has the following format:

```
StudentFirstName StudentLastName score1 score2 score3 score4 ... scoreN
```

Store StudentFirstName and StudentLastName into a Student structure. Then, input each score, and when there are no more scores on the given line, calculate the average of the scores and store it into the Student structure (the third member variable in the Student structure is the grade, which is the average of the scores).

Sometimes a line will be blank, in which you must skip it and move onto the next line. Furthermore, a student may have zero or more scores, which you must be able to handle using the input string stream (istringstream).

Finally, copy the student structure onto the Student structures array.

Reversing Array Function

Returns: nothing
Parameters: an array of Student structures previously filled using the reading input function and an integer representing the number of students in the array
Purpose: this function will reverse the Student structures in the array so that the very first student will be the very last student, the second student will be the second to last student, and so forth. You may not declare/use a second array.

Printing Student Report Function

Returns: nothing
Parameters: an output file stream object of an already opened output file, the array of Student structures, and an integer representing the number of students in the array.
Purpose: this function will output a report using the output file stream object and the reversed array of student structures using the following format:

```
LastName, FirstName              =  XXX.YY
```

The LastName, FirstName will be in a left-justfied field of 20 characters. The XXX.YY represents the calculated average scores and will be in fixed-decimal notation and precise to 2 decimal places.

To place both LastName and FirstName in a single left-justified, 20 character field, use an output string stream.

If there are no students to report, output "No students to report.\n" to the output file.

int main()

Now that you have your functions, it is time to put it all together. Your int main function will be responsible for opening the input and output files along with generating the appropriate error messages, and calling the necessary functions . **This means that you will need to pass the file streams, the array of Student structures, and the actual size of the array as parameters. Do not open an ifstream or ofstream in any of the functions...it must be done inside of int main!**

You will also need to declare your student array in int main to pass to the functions.

Inputs

You will need to ask the user for the input file and the output file using the following prompts:

```
"Enter input filename: "
"Enter output filename: "
```

If the input file could not be opened, output "Unable to open input file.\n"

Additional Requirements

1. You must prototype and define the three functions listed in the Functions category
    1. These functions must take the parameters specified and return the values specified in the Functions category
2. You must use an input string stream and getline to read the input file
3. You must use an output string stream to join the last name and first name into a single string
4. You must use a constant that stores the maximum number of elements in the Student structures array (10)

C++ Topics Covered

Arrays
Input file streams
Output file streams
Classes/structures
Passing objects to functions
Input/output string streams

Textbook Chapters Covered

Chapter 8.1 (Line oriented input)

Relevant Reading