Project #06 (20 points): File Input/Output

➔10% grading bonus if your program output matches the sample solution exactly ➔

Submit Your Solution Using ANGEL by Noon on Friday, 02/15/13

(A late submission drop box will be available on 02/15/13 from Noon to 2pm)

Obtaining Project 6 Input Order and Output Requirements:

(Sample Solution and Comparison Script)

To view the order and style of the input and output information for the project, run the provided solution by **typing the following at a terminal window prompt.** (**Note**: Input files must be present in the directory that the terminal window is currently in.)

Sample Solution path

/home/work/cpe112/Executables/Project_06/Project_06_solution

Provided sample input files are zipped in the file P6_in.zip

Note: do not use input redirection with these input files

Comparison Script Path

/home/work/cpe112data/Project_06/CompareSolution.bash Project_06.cpp

Project 6 Restrictions

Only material from Chapters 1 through 4 and any extra code in this handout is allowed.
You cannot use any C++ techniques that are covered in Chapters 5 and higher.

You are not allowed to use any global variables. Global variables are declared above the int main() line in a program. Global constants can be used

Starting Project 6:

- Open a terminal window and move (cd) into the Project_06 directory created in the CPE112_SPR13 directory (This is the directory structure created in project 1.) The command for this is: **cd CPE112_SPR13/Project_06 (typing cd ~/CPE112_SPR13/Project_06 works as well)**. You will need to modify the names as necessary to match your capitalization style for the two directories.
- Download all needed files from ANGEL into this directory. Using your favorite text editor (i.e. gedit), open the existing **Project_06.cpp** file (this is the header file created in Project 2) and add code to it to complete this project. Once you have finished with the program and it compiles without syntax errors, run the executable and verify that the output for your program matches the output from the provided solution executable
- Remember to use the firefox browser when viewing ANGEL. Also, to view a pdf file saved in a directory, type the command **evince filename.pdf** at the prompt in a terminal window that has the same working directory as the directory in which the file is present.
- **Once you are satisfied with your solution, submit Program_06.cpp via Angel**.

NOTE: make sure that you do not change the order in which the information is entered. An automatic script is used to process all lab submissions, and if the order of the input information is modified, the script will not work properly with your program.

<Project 6 Description>

For this project, you will write a complete C++ program that performs ***all of the following tasks***. **Use variables of Data Type *float* to contain all numerical values**.

(1) Open a user specified (user enters in the name) input file. Verify that the file opened successfully. If it did not open successfully, output an error message and terminate. – See the **Opening Input/Output Files section** on page 3 for the code to perform the check on the success of opening a file
(2) If the input file is successfully opened, open a user specified output file. Verify that the file opened successfully. If it did not open successfully, output an error message and terminate. Use the filename "**`Bad/file`**" to cause the open function to fail for the output file.
(3) For steps 1 and 2 echo print out the name of the file entered by the user.

(4) Read in the title line from the input file and write this line to the output file.
(5) Write the column headings to the output file (see the sample solution output)

(6) For the first person, read in their first name, last name and four test scores. (check the project 6 input file information on page 3)
(7) Find the average of the four test scores and add this average to an overall sum.
(8) Write the information for the first person to the output file. Output the first 9 characters of the last name, the first 10 characters of the first name and the test average of the person

(9) Repeat steps 6, 7 and 8 for the second and third person information in the input file

(10) Calculate the overall average of the exams for all three people
(11) Output the overall average information line followed by the overall average value
(12) Output of the program shall match that of the sample solution

<Project 6 Hints>

- Make sure all output is to the output file – including the output statements that set up the formatting.
- Use the ignore function to remove the new line character from the input stream when a getline is used after an extraction operation.

- **The name columns are in a field width of 12 and left justified. The average column is just output (no setw used for it)**
- Print the comment line and the headers to the output file before processing the first person

- One set of variables only are needed to hold the first name, last name, four test scores and average. Do all calculations and outputs for one person and then copy the code for use on the second and third persons.
- Output the first 9 characters of the last name and the first 10 characters of the first name – use the substring function

- Run the sample solution (not the comparison script) to see what the program writes to the terminal and to the output file
- Look at the contents of an input file to see what has to be read. Note the use of commas to separate name parts.

Project 6: Opening Input/Output Files

The first action taken by the program is to prompt for the input file name (i.e. P6_in1.txt). Once that name has been read into a string variable, use it to open the input file stream that is associated with that file. **Immediately after the file stream open statement, put the following if code segment in the program**. This code segment verifies the status of the file stream. If the file stream is valid, then the input file exists. If the file stream is not valid, then the input file does not exist, and the program prints the message and terminates. **Note:** ***FileStreamVar*** should be replaced with your input file stream variable associated with the input file specified by the user and ***FileName*** should be replaced with your string variable containing the name of the input file entered by the user.

```
if(!FileStreamVar)
{
     cout << endl << string(12,'*') << "  File Open Error  ";
     cout <<  string(12,'*')  << endl;
     cout << "==> Input file failed to open properly!!\n";
     cout << "==> Attempted to open file: " << FileName << endl;
     cout << "==> Terminating program!!!\n";
     cout << string(43,'*') << endl << endl;
     return 1;
}
```

A similar procedure is then used to verify the open of the user specified output file – replace all instances of input above with output.

<Project 6 Input File Format>

The Input file format is similar to the information shown below. Look at all of the provided input files to determine the exact format.

- **On all lines, a comma separates the last name from the first name and the scores from the last name. All scores are separated by spaces.**
- **The first line of the input file is a header line which is to be copied to the output file**

Note, use the getline function terminated on a comma to read in the first and last name of the person. Use the extraction operator to read the scores. Remember that first and last names can consist of multiple words.

```
// Class Roll For CPE112
Ron,Bowman, 100.75 100.3 100 99.43
Johnny B.,Quick, 54 53 52 51
Mary,Francis Beacon, 75 96 32 68
```