The University of Alabama in Huntsville
Electrical and Computer Engineering
Project 7 (30 points + 10% for perfect match)

Submit Your Solution Using Canvas by Noon, Friday October 5, 2018 (late submissions will be accepted on 10/5/18 from Noon to 2:00pm)

<Project 7 Description>

For this project, you will write a complete C++ program that performs ***all of the following tasks***. Remember, your program must contain appropriate comments in order to receive full credit:

(1) Prompt for, read, echo print and open a user defined input file. If the user inputs the name of a file that will not open, a loop is entered that prints out an appropriate error message, resets the input stream (see the end of the hints section), prompts for the name of the input file again and tries to open the new file name. The loop is to continue executing until the user successfully enters the name of a file or types ctrl-c to terminate the program.
(2) Print out a 5 column heading for the data to be printed
(3) Read every character (tabs, spaces and new-lines are characters) on a line-by-line basis from the input file and count them as indicated:
   a. Count the number of characters on a line that are letters only
   b. Count the number of characters on a line that are digits only
   c. Count the number of other characters that are neither a letter or a digit
(4) Output the character counts on a line by line basis (counting stops for a line when the new line character is encountered - be sure to count the new line character)
(5) After reading all lines, output a row of 50 dashes, then output a total line for each column of data printed. Below this line, print out a line indicating the percent **(shown to two decimal places)** of the total characters that each column represents. Use the same field widths as the columns for the lnes.
(6) The columns of information printed are: Line number, number of letters, number of digits, number of other characters and the total number of characters for the line.

Your output is to be identical to the output of the sample solution (i.e. 5 columns).

You may run the sample solution **Project_07_solution** by typing the following at a command prompt in a terminal window**:**

Sample Solution – for determining vertical spacing differences

/home/work/cpe211/Executables/Project_07/Project_07_solution

Comparison Script – for determining line differences

/home/work/cpe211data/Project_07/CompareSolution.bash Project_07.cpp

<Project 7 Directions>

On Project 7, you may only use concepts presented in Chapters 1 - 7 of your textbook!!
Do not use arrays. Global variables are not allowed

Using your favorite text editor, type your solution and save it as a file named **Project_07.cpp** within your **CPE211_FALL18/Project_07** directory. If there are syntax errors, correct them and compile again. Once your program successfully compiles, run it and verify that the output for your modified program matches the output from the solution executable.

Once you are satisfied with your solution, submit Project_07.cpp via Canvas.

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 7 Hints>

- Go over the slides for this project
- Run the sample solution (not just the comparison script)

- Use a priming read to read the first character of the input file (outside of outer loop)
- Verify if the input file is empty or not. For an empty input file print out the appropriate message
- Use 2 nested loops to read in the characters.
  - The outer loop is looking for the end of the file and is checked on a line by line basis
  - The inner loop is reading a line character by character and looking for the new line character that signals the end of the line
- Be careful with the termination of the inner loop when a new line character is read – make sure that it is counted in the other category.
- Set up of the nested loop logic is important. Make sure that all characters read are counted and that no characters are skipped. This looping structure requires some thought to be done correctly

- Use the header file **cctype** to make use of some functions that tell the type of character stored in a char variable. See Below.
- Technically correct programs will have at least 5 header files.

- Use integer variables to hold the counted values. Need two sets of variables to hold the character type counts. One set is for the characters on each line. The other one is for holding the totals of each type of character for the entire input file.
- **The sample output is <u>left justified</u> with a field width of 15 for the line number column and 10 for the next 4 columns.**
- For an empty input file, the empty input file message only is printed.

- After a new line character is read, the individual line counts are output and the total amounts for the program are updated with the counts from the individual lines.
- Look at the slides for further information on the project

- **To reset a file stream variable, place the following statement as shown into your program at the location where you want to reset the stream:**

  ```
  File_stream_var.clear();
  ```

  Where File_stream_var is the name of the file stream variable you use in your program (i.e. inFile, outFile, etc).

- **There are 50 dashes** separating the line counts from the totals line

- **cctype** header file contains the following functions. These functions return true or false values and each requires a character as the argument in the function call
  - isalpha(charVar) – returns true if the character in charVar is a letter
  - isdigit(charVar) – returns true if the character in charVar is a digit(number)