Lab 03: Processing Delimiter Separated Files

CSci 515 Spring 2015

CSci 515 Spring 2014 *<2015-01-23 Fri>*

Dates:

Due: In Lab, Wednesday February 4, by 4 pm (lab end time)

Objectives

- Be able to open a serial text file for reading.
- Learn about and process simple text files of delimiter separated values.
- Use I/O formatting manipulators, for writing formatted data output.
- Use loops to read and process data files.
- Practice using control structures for looping and conditional execution in C.

Description

Plain text files containing tables of information are very common, minimal representations of data sets needed for processing. The simplest type of formatting of a table of values, is to separate the columns or features of the table using spaces, tabs or commas. In general these are known as delimiter separated value files (DSV). The most common is to use commas (known as a comma separated value file or CSV files). The C++ IOStream operators make it trivial to process space separated and tab separated value files, since it uses whitespace (spaces and tabs) by default as the delimiter when breaking apart a stream to automatically parse and convert into input variables. In a space or tab separated file, the values separated by space

are the features, and each row or line is an individual record or trial in an experiment.

For this lab you are to read in records from a space separated file, and do some processing of the data. I will give you a space separated file to use. The file you are to open and process looks like this:

```
 1.03925 -0.0466664  6.78488
 5.733837 -7.16068  -2.792169
-4.5309 -0.1812 6.10121759
  1.437688 2.798135755   4.12021
1.22328  -1.5012   8.99924
-1.53216  -5.395889   4.0939
  8.6847   5.44601  -7.70818
-4.76181 8.362  -0.389249517
 2.298 4.148659  -0.757
  3.3200104   3.953700223 -5.8364352
```

You need to perform the following tasks:

- Open the file and process it one line at a time.
- Output the original contents of the file, but with cleaned up formatting. You will output each column in a field of width 15 characters (`setw()`). All numbers will be printed with exactly 5 decimal point digits of precision (`setprecision()`). All of the floating point numbers should be displayed using scientific notation (`scientific`). See the example output to get details on what this should look like.
- You will add an index as a new first column of each line, in a field of width 5.
- You will add a header to the original data, as shown in the example output.
- In addition to cleaning up and formatting the data, you will do some processing of the data. You need to determine the minimum value of all values seen in column 1, the maximum of the values in column 2 and the average of the values in column 3. You will report these values after displaying the cleaned contents of the file, with some others. Again see the example output to determine exactly which summary information you must produce and in what format.

Here is an example of the correct, cleaned up and summarized data for the previously shown input file:

```
Trial      Feature-1      Feature-2      Feature-3
    1    1.03925e+00   -4.66664e-02    6.78488e+00
    2    5.73384e+00   -7.16068e+00   -2.79217e+00
    3   -4.53090e+00   -1.81200e-01    6.10122e+00
    4    1.43769e+00    2.79814e+00    4.12021e+00
    5    1.22328e+00   -1.50120e+00    8.99924e+00
    6   -1.53216e+00   -5.39589e+00    4.09390e+00
    7    8.68470e+00    5.44601e+00   -7.70818e+00
    8   -4.76181e+00    8.36200e+00   -3.89250e-01
    9    2.29800e+00    4.14866e+00   -7.57000e-01
   10    3.32001e+00    3.95370e+00   -5.83644e+00

        Number of Trials:         10
    Minimum of Feature 1:   -4.76181
    Maximum of Feature 2:    8.36200
    Average of Feature 3:    1.26164
```

Lab Submission

An eCollege dropbox has been created for this lab. You should upload your version of the lab by the end of lab time to the eCollege dropbox named `Lab 03 Process DSV File`. Work submitted by the end of lab will be considered, but after the lab ends you may no longer submit work, so make sure you submit your best effort by the lab end time in order to receive credit.

Requirements and Grading Rubrics

Program Execution, Output and Functional Requirements

1. Your program must compile, run and produce some sort of output to be graded. 0 if not satisfied.

2. Your program must successfully open the file from the current working directory. 10 or more points.

3. Your program must successfully read the lines from the file in the correct order and attempt to process them. 20 points.

4. Your program must use I/O manipulators to achieve the correct output format of the cleaned up original data. 30 or more points.

5. Your program must include a header for the original data, properly formatted. 10 points.

6. Your program must produce the correct summary information values. 30 or more points.

7. Your program must format the summary information as required, again using I/O formatting manipulators. 20 or more points.

Program Style

Your programs must conform to the style and formatting guidelines given for this course. The following is a list of the guidelines that are required for the lab to be submitted this week.

1. The file header for the file with your name and program information and the function header for your main function must be present, and filled out correctly.

2. You must indent your code correctly and have no embedded tabs in your source code. (Don't forget about the Visual Studio Format Selection command).

3. You must not have any statements that are hacks in order to keep your terminal from closing when your program exits.

4. You must have a single space before and after each binary operator.

5. You must have a single blank line after the end of your declaration of variables at the top of a function, before the first code statement.

6. You must have a single blank space after , and ; operators used as a separator in lists of variables, parameters or other control structures.

7. You must have opening { and closing } for control statement blocks on their own line, indented correctly for the level of the control statement block.

Failure to conform to any of these formatting and programming practice guidelines for this lab will result in at least 1/3 of the points (33) for the

assignment being removed. Failure to follow other class/textbook programming guidelines may result in a loss of points, especially for those programming practices given in our Deitel textbook that have been in our required reading so far.