EECE 1080C / Programming for ECE

Summer 2019

Laboratory 9: C++ Structures

Plagiarism will not be tolerated:

all students who share files will receive a 100% penalty on this assignment

Topics covered:

- Namespace
- File Input/Output
- Excel Plotting

Objective:

- To become familiar with the utility of namespaces and file input/output type objects in C++. The student will create two short programming assignments to build a better understanding.
- The main goals of this laboratory are as follows:
  - Gain hands on experience with creation and usage of Files in C++.
  - Learn to use namespaces to store special functions

Collaboration:

- As with most laboratory assignments in this course this laboratory assignment is to be performed by an individual student. You can help each other learn by reviewing assignment materials, describing to each other how you are approaching the problem, and helping each other with syntax errors. You can also get help from teaching assistants and instructors. Having slightly similar code for some assignments is expected but most assignments have multiple different solutions. Your code is expected to be different.
- Please document any help you receive from teaching assistants or instructors. Just add the names to the top of your source file.
- Having someone else code for you, sharing code with other students, or copy-pasting code from the internet or previous terms, **is cheating**. A helper should "teach you to fish, not feed you the fish". Laboratory assignments prepare you for exams so be smart.

Highlights:

- Please consider the following:
  - Review namespaces.
  - Review file streams.

- Avoid using magic numbers in these functions. Use constants when appropriate.
- Add a main comment section at the beginning of each program you submit, specifying your name in Pinyin, the date it was last updated and a brief description of what it does.
- Read problem statements carefully in order to fulfill all instructed requirements.
- Remember to validate inputs accordingly. It is also a good programming practice.
- Run your program for all probable outcomes to confirm its functionality. If possible, have other people test your code. This can help identify and troubleshoot problems.
- Comment important sections of your code to easily recognize your logic and approach to solving the tasks.

- To receive full credit for this laboratory please sign the attendance sheet.
- Please access the laboratory assignment via the canopy/blackboard link. The descriptions for each problem are contained within this document.
- Each part should be worked on separately. You will need a separate project for each part of this assignment when working within your IDE.
- **Submit your .cpp file on Blackboard using the assignment dropbox.**

Rubric: 50 points

- Part A = 20
- Part B = 20
- Part C = 60

Tasks:

A. Object height over time, (part 2)

- This program will start with the program that you created in Lab 03 Part D (Wk 14)
- Complete the following:
    - Create a global variable namespace
        - Move the gravitational acceleration of 9.80665 m/s^2 into the namespace
        - Move the height function to the namespace (and add an initial height)
            1) $y = y_0 + V_0 t - 0.5gt^2$
            2) $y_0,\ V_0,\ t$ should all be input variables
    - Create an input file that contains two items (and remove the user prompt for the initial velocity)
        - initial velocity of 60 m/s
        - initial height of 50 m.
    - Duplicate the on-screen output and send it to an output file.
    - Transfer the data to Excel and create a plot.
- The remaining elements of the program should stay the same as the original.

B. Metal Expansion Table, (part 2)

- This program will start with the program that you created in Lab 03 Part E (Wk 14)
- Complete the following:
    - Create a global namespace
        - Move the equation ($x = w + 0.0001(T - 70)$) into the namespace
            1) w, T should be input variables
    - Replace the user prompt for the Width and Tolerance with an input file containing these two items.
        - Width = 10, Tolerance = 0.0050
    - In addition, add the temperature boundaries to the input file $T_{min} = 40°F$ and $T_{max} = 125°F$
    - Replace the cout commands for the table with ofstream variable commands.
    - Transfer the data to Excel and create a plot

C. Computational Fluid Mechanics – Data Probe

- You will need the VelocityProbe.txt data file from Blackboard:
    - This file contains the velocity information of four probe locations of a simulation at the times listed in the file.
    - **Be Mindful of the structure**
        - **Lines 1, 2, 3:** Contains 5 columns
            1) placement holders (row 1: x, row 2: y, row 3: z) – throw away

2) x, y, and z coordinates of the first probe
3) x, y, and z coordinates of the second probe
4) x, y, and z coordinates of the third probe
5) x, y, and z coordinates of the fourth probe

- You will find the distance from the origin and use the numerical value as a header in the output file
- **Line 4:** Contains one word 'Time' – throw way
- **Lines 5 to eof:** contains the following set of information

$$t, \quad V_{1x}, \quad V_{1y}, \quad V_{1z}, \quad V_{2x}, \quad V_{2y}, \quad V_{2z}, \quad V_{3x}, \quad V_{3y}, \quad V_{3z}, \quad V_{4x}, \quad V_{4y}, \quad V_{4z}$$

- Create a program that will successfully read in to the computer **all lines** from the file and properly store them into the appropriate vectors
- Once the vectors data is stored, compute the magnitude of each vector
  - Distance from Origin: $D = \sqrt{x^2 + y^2 + z^2}$
  - Velocity: $V_{magnitude} = \sqrt{V_x^2 + V_y^2 + V_z^2}$
- Create a tab-delimited output file that will store all the results
  - In the header line: you will create five columns
    - 'Time'
    - The value of each "distance from the origin" calculated using the distance formula
  - Then, send the stream of time and velocity magnitude data to the file
- Transfer to Excel and plot the results for each velocity versus time

Hint: While you could count the number of lines, try the **getline()** function. It will read an entire line of the file. Then the counter will show the number of lines in the file.
Once the counter is known, you can initialize the vectors, reopen the file, and transfer the information to C++.

```
ifstream file;
file.open("findMe.txt");
string temp;
int counter = 0;
while (!file.eof()) {
      getline(file, temp);
      counter++;
}
file.close();
file.open("findMe.txt");
```