

Allegheny International Manufacturing

Cone/Frustum Regional Data File Processing Version

Background

Your programming work for the Allegheny International Manufacturing (AIM) has been a great success. As is often the case, success brings with it new system requirements. All computations and validation rules from Project #1 and Project #2 remain the same. The data file format is the same as for Project #2; however, now there are separate files for each sales region. Senior management decided that critical data elements from the input file(s) must be stored in memory in the following system of parallel vectors:

```
vector<string> vRegion;  
vector<char> vShapeCode;  
vector<char> vColorCode;  
vector<double> vRadius2;  
vector<double> vRadius1;  
vector<double> vHeight2;  
vector<char> vClosedBase;  
vector<char> vClosedTop;
```

You may add more vectors, but do so with caution. Those listed above should be all that are required for calculations and any additions must enhance your solution.

For the new software release, we must also incorporate a repeating menu so the application continues running until the user elects to quit. Since the menu must repeat until the user elects to quit, some error conditions and messages need to be modified. Selecting an incorrect menu option or entering an invalid filename should not end the program. The user should be informed of the error and the main menu shall be displayed again. Note also, that the summary reports shown in the menu cannot run unless data file(s) have been loaded. If the user selects any report option without previously loading one or more data files, they should be informed of the issue and the menu shall be displayed again.

When a data file is loaded, selected data elements from each row of the file shall be stored in parallel vectors previously listed. Validation tests shall be done within the function that loads data. Only records having no errors shall be appended to the vectors. Once file data have been stored in the parallel vectors; all subsequent calculations shall be made by iterating through the vector contents. No further file input shall take place until such time as the user requests to load another data file. If a data file has already been loaded, and the user selects the option to load another file, the new file contents shall be appended to the existing data. The same data file could potentially be loaded multiple times. A production system would require additional safeguards to prevent such data duplication. For our purposes in this project we will ignore the risk and simply assume that it will not happen.

Your program must be divided into multiple functions. Specifically, every menu option shall correspond to a call to a user-defined function that performs the operations specific to that menu option. There are other functions as well, such as a function to actually display the menu. The table below lists prototypes for the functions that your program must include. This list of functions must be implemented exactly as specified to include the parameters listed in each prototype. You are free to use other functions, but do so with caution and do not add any function that would replace the operation(s) implied by any of the functions listed below. When your project is complete, function `main()` should contain variable declarations, control structures, and function calls. With few exceptions, all calculations and other "work" shall be done within the functions that are called from `main()` depending upon the user's menu selections. You must use the function identifiers listed below. You are allowed to change the identifiers of the function parameters to match your variable naming scheme from previous projects. Identifiers for function parameters that are listed below have been abbreviated but it should be clear from the data type and the abbreviation what data element they represent.

```
char showMenu();

bool loadData( string fileName, vector<string> &reg,
               vector<char> &sC, vector<char> &cC,
               vector<double> &r2, vector<double> &r1, vector<double> &h2,
               vector<char> &cBase, vector<char> &cTop );

double getGC( double r2, double r1, double h2 );

double getGF( double r2, double r1, double h2 );

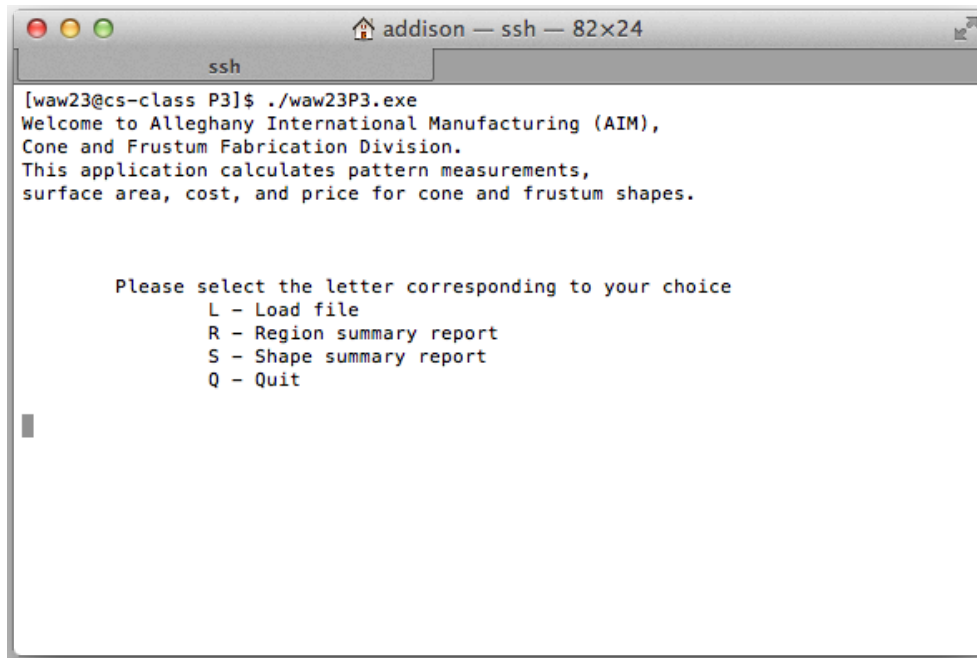
double getTheta( double r2, double r1, double h2 );

double getTotalSurfaceArea(double r2, double r1, double h2, char sCode,
                           char closedBase, char closedTop);

void summaryByRegion( const vector<string> &reg, const vector<char> &sC,
                     const vector<char> &cC,
                     const vector<double> &r2, const vector<double> &r1,
                     const vector<double> &h2,
                     const vector<char> &cBase, const vector<char> &cTop );

void summaryByShape( const vector<string> &reg, const vector<char> &sC,
                    const vector<char> &cC,
                    const vector<double> &r2, const vector<double> &r1,
                    const vector<double> &h2,
                    const vector<char> &cBase, const vector<char> &cTop );
```

The screen capture below shows the required menu items. Screen captures of the output from these menu items are not provided. However, there is a sample executable that you may download from Blackboard. This executable file will only run on the class server. Do not attempt to run the executable file on your computer, open the file in an IDE, or otherwise use the file except to run it on the class server. To receive full credit, your program's operation should be similar to that of the example program.



```

[waw23@cs-class P3]$ ./waw23P3.exe
Welcome to Allegheny International Manufacturing (AIM),
Cone and Frustum Fabrication Division.
This application calculates pattern measurements,
surface area, cost, and price for cone and frustum shapes.

Please select the letter corresponding to your choice
L - Load file
R - Region summary report
S - Shape summary report
Q - Quit

```

How to approach this project

There is no design submission, but you are strongly encouraged to create a pseudocode or flowchart design of the algorithm(s) you plan to implement. Pseudocode terms that may be useful are listed below.

- **start**
- **input**
- **output**
- **calculate**
- **if condition, then statement(s)**
- **if condition, then statement(s); otherwise, statement(s)**
- **switch value:**
 - case value: statement(s)
 - case value: statement(s)
 - ...
 - case value: statement(s)
 - otherwise: statement(s)
- **end switch**
- **while condition**
 - statement(s)
- **end while**
- **stop**

To group multiple statements together, such as part of an **if** statement, use

```

begin
    statement
    ...
    statement
end

```

For this project several milestones are recommended, however you are NOT required to turn anything in or to meet these milestones. Always make sure that your code compiles and runs before starting the next milestone. Much of the code for the new functions already exists in your Project #2 program. Do not duplicate any code. Move code that already exists to the appropriate function and then use a call to that function in place of the code that was moved. See additional details below.

Milestone 1 – NLT July 26th

- Subscribe to the Project 3 Forum
- Create an empty project with just function main
- Add the required function prototypes above main (but after preprocessor directives and using namespace std;)

Milestone 2 – NLT July 28th

- Add function stubs for each required function
- Implement the showMenu function
- Modify function main so that it calls this function
- Add function calls to main that call the correct function stubs based on the value returned by the showMenu function

Milestone 3 – NLT July 29th

- Implement the loadData function
 - this function is passed a string that stores the full path and name of an input data file
 - the loadData function shall open the specified file, read all contents, perform validation checks, and output details to the terminal screen (with some prior planning, you should be able to copy and paste your code from Project #2 to get the core of this function working)
 - add code to append selected data elements, from error-free rows, to the parallel vectors
 - if the data load successfully this function returns the Boolean value true, otherwise false

Milestone 4 – NLT July 30th

- Implement the functions getGF, getGC, getTheta, and getTotalSurfaceArea
- Plan carefully, most of the code for these function can be cut and pasted from code that will still be in the loadData function after the previous milestone
- Take each function one at a time and move the code out of loadData and into the new function, then add a function call within loadData to call the function just implemented

Milestone 5 – NLT August 1st

- Implement the functions summaryByRegion and summaryByShape
- Plan carefully, the code for summaryByShape can be cut and pasted from code that should still be in the loadData function, summaryByRegion is new, but will be very similar

Milestone 6 – NLT August 2nd

- Make final checks and a final confirmation that your program compiles on the class server
- Turn your project in on time!

Program Source Code

Important: Your output and input should be very similar to that shown in the sample program. Some content must also be included in your program **exactly** as specified.

Include the following comments at the start of your source code file:

```
/*
 * <FileName>.<file extension>
 *
 * COSC 051 Summer 2016
 * Project #3
 *
 * Due on: AUG 2, 2015
 * Author: <your name>
 *
 *
 * In accordance with the class policies and Georgetown's
 * Honor Code, I certify that, with the exception of the
 * class resources and those items noted below, I have neither
 * given nor received any assistance on this project.
 *
 * References not otherwise commented within the program source code.
 * Note that you should not mention any help from the TAs, the professor,
 * or any code taken from the class textbooks.
 */
```

These comments must appear **exactly** as shown above. The only difference will be values that you replace where there are "place holders" within angle brackets such as <netID> which should be replaced by your own netID. For example, I would replace <netID>P3.cpp with waw23P3.cpp.

Submission Details

Post to Blackboard a .cpp file containing your source code. Locate the assignment Project 3 on Blackboard and attach/upload your file. Do **not** post your executable file. You should ensure that your source file compiles on the server and that the executable file runs and produces the correct output. Use the following file name for your file: <netID>P3.cpp

Due date is no later than end-of-day (11:59 pm) on August 2nd. Late submissions will be penalized 2.5% for each 15 minutes late. If you are over 10 hours late you may turn in the project to receive feedback but the grade will be zero. In general requests for extensions will not be considered.

The value for this project is 100 points.

Grading

A grade rubric will be published separately.

Academic Integrity

This is an individual project and all work must be your own. Refer to the guidelines specified in the *Academic Honesty* section of this course syllabus or contact me if you have any questions.