CS 135

Design Assignment 3 (DA3-09/26)

Programming Assignment 3 (PA3-09/30)

As specified in your syllabus, you must turn your assignments in by 6:00 pm on the due date specified. If it is turned in late, but prior to 12:00 midnight the day it is due, credit will be reduced by 50% of the earned score. Any laboratories turned in more than 6 hours late will not earn any credit.

Objectives:

1) You will use a systematic development process to create a program
2) You will use a set of standardized functions to implement command-line I/O operations in a formatted command line system
3) You will use global constants to assist with program clarity
4) You will create simple functions as part of applying program modularity, and to allow for reuse of functions when code components are repeated
5) You will implement mathematical operations in order to solve problems
6) You will develop and implement decision-making actions in your program

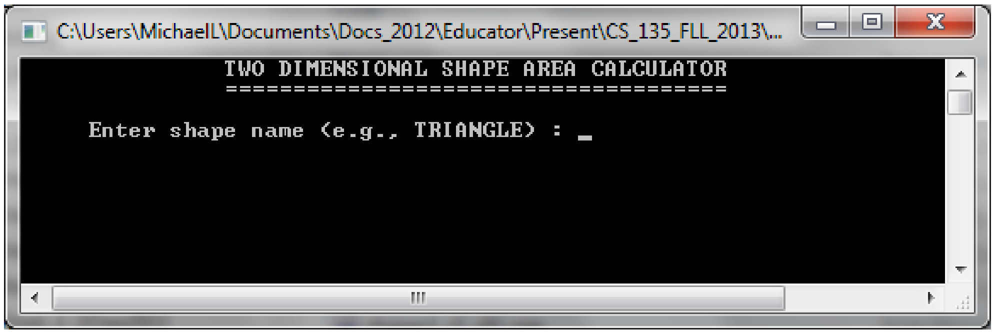
Tasks:

Calculating Two-Dimensional Shape Areas, The Prequel

1) The program, called **`shapes3.cpp`** will continue to require the use of the **`formatted_cmdline_io_v10.h`** so that nice looking tables can be displayed. You may not use any of the other I/O tool sets (e.g., **`iostream`** or formatted console I/O), and you may not use any other header files or tools for this assignment.

2) The program will accept the kind of two-dimensional shape that is to be calculated and then the data necessary to do the calculations. The output table will then show the results related to the specific shape entered.

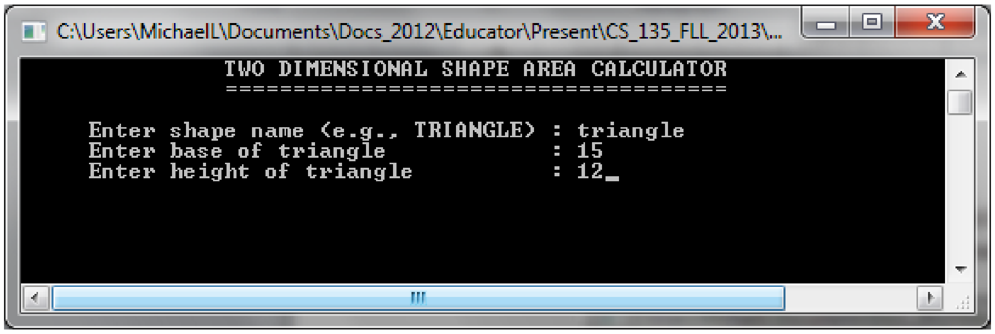
3) The initial part of the program will show a title and prompt for the shape name. The program accepts triangles, rectangles, and parallelograms.



```
C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_2013\...
                 TWO DIMENSIONAL SHAPE AREA CALCULATOR
                 =====================================

     Enter shape name (e.g., TRIANGLE) : _
```

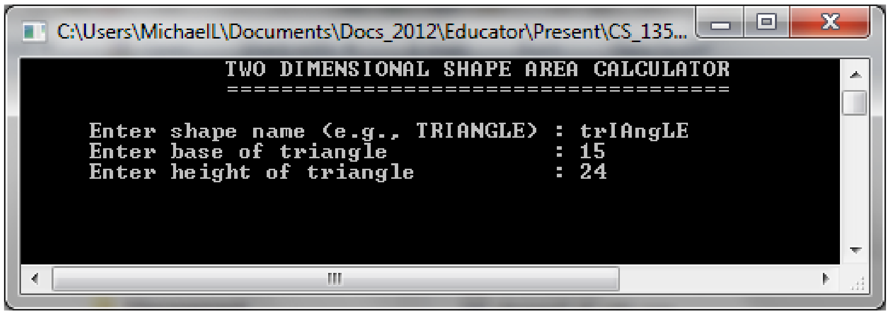
4) After the user enters the shape name, the program then shows prompts appropriate to the shape provided.



```
C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_2013\...
                 TWO DIMENSIONAL SHAPE AREA CALCULATOR
                 =====================================

     Enter shape name (e.g., TRIANGLE) : triangle
     Enter base of triangle            : 15
     Enter height of triangle          : 12_
```

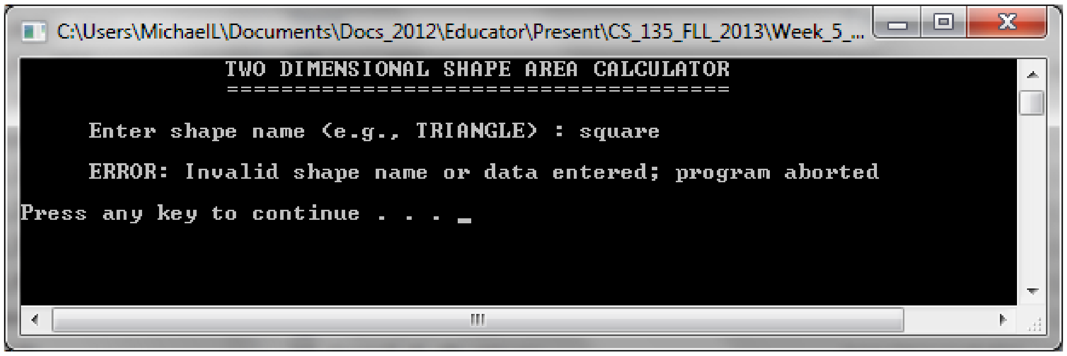
5) The user may enter the name of the shape with any letter case (i.e., upper or lower case).



```
C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135...
                 TWO DIMENSIONAL SHAPE AREA CALCULATOR
                 =====================================

     Enter shape name (e.g., TRIANGLE) : trIAngLE
     Enter base of triangle            : 15
     Enter height of triangle          : 24
```

6) If the user does not enter one of the three correct shape names, the program will show an error message and abort. It must shut the program down immediately after identifying the program, and it must not conduct any further actions such as calculations or other display prior to the aborted shut down.



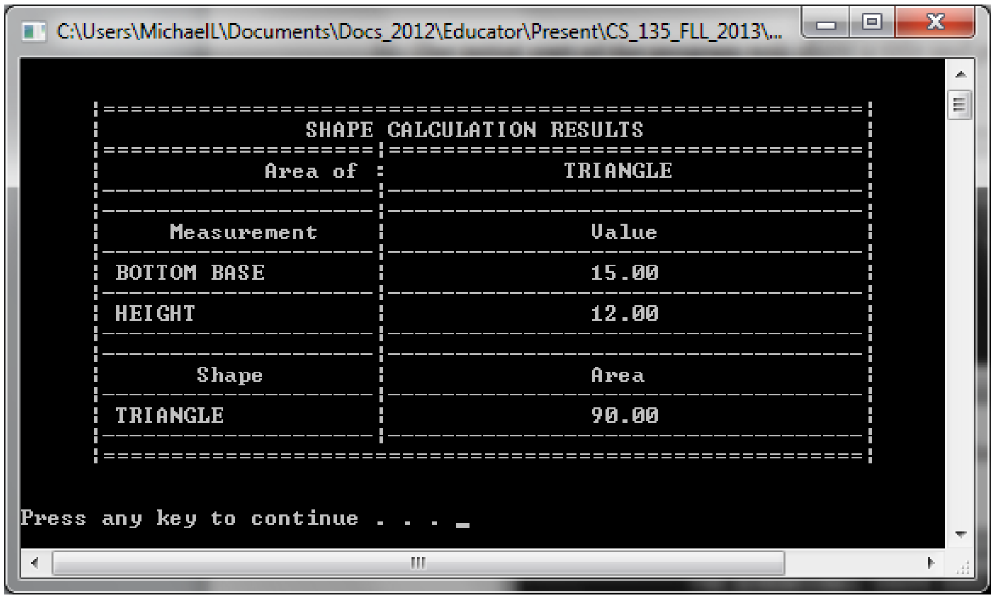
```
C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_2013\Week_5_...
                    TWO DIMENSIONAL SHAPE AREA CALCULATOR
                    =====================================

      Enter shape name (e.g., TRIANGLE) : square

      ERROR: Invalid shape name or data entered; program aborted

Press any key to continue . . . _
```

7) If the data entered is correct, the program must show the results specific to the shape and data entered. The input and results will be displayed in table form.



```
C:\Users\MichaelL\Documents\Docs_2012\Educator\Present\CS_135_FLL_2013\...

       |=========================================================|
       |                SHAPE CALCULATION RESULTS                |
       |====================|====================================|
       |           Area of : |             TRIANGLE               |
       |--------------------|------------------------------------|
       |--------------------|------------------------------------|
       |     Measurement     |                Value               |
       |--------------------|------------------------------------|
       | BOTTOM BASE         |                15.00               |
       |--------------------|------------------------------------|
       | HEIGHT              |                12.00               |
       |--------------------|------------------------------------|
       |--------------------|------------------------------------|
       |       Shape         |                Area                |
       |--------------------|------------------------------------|
       | TRIANGLE            |                90.00               |
       |--------------------|------------------------------------|
       |=========================================================|

Press any key to continue . . . _
```

8) The challenges for this program are that you need to decide which shape you have BEFORE you can accept the input and calculate the area and you must make decisions as to which measurement components to display when showing the output. For example, the rectangle has the measurements length and width, and triangle has base and height. Your output display must appropriately show these measurement names along with the measurements themselves. Before starting this assignment, make sure you thoroughly analyze the example program so you can understand what your program must be able to do. Your program must work the same as the example program.

9) It will be very easy to create several simple functions to support this program; the example program has twelve. However, you are only required to create a minimum of five functions. That said, your program will be easier to manage if you use more (appropriate) functions. Your program must contain at least the following functions:

   a) one function that displays: 1) one double solid (no divider) line such as the one at the top of the display table; 2) one double divided (one pipe divider) lines such as the one under the top title of the display table; or 3) one single divided (one pipe divider) line such as the one under the "Measurement" and "Value" subtitles in the display table. This function will obviously be called several times in order to display the table shown previously in this document. No code may be used outside this function to create the horizontal divider lines

   b) one function that calculates the area of the shape depending on its name; no shape area calculations may be conducted outside this single function

   c) one function that displays one table data line. This function will display appropriate text (left justified) in the left column, and the appropriately related value (center justified) in the right column. For the table result displayed previously in this document, this could be the line that shows "BOTTOM BASE" and "15.00" under the "MEASUREMENT" and "VALUE" subtitles, or the line that shows "TRIANGLE" and "90.00" under the "SHAPE" and "VALUE" subtitles. Obviously this function would be called at least three times with each program. No code may be used outside this function to display these table data lines

10) It will cause you a large amount of grief if you try to arbitrarily write five functions, and it is not necessary. This program lends itself to many supporting functions. If you follow the Six Step Programming Process carefully and thoroughly, you will almost certainly find that you have created more functions than are required (and your programming process will be easier and take less time).

11) You will also need another pair of functions to support your program. As mentioned previously, the user may enter the shape name with any case (i.e., lower or upper) she wishes to use although she still has to spell the name correctly. In order for your program to handle this and make good decisions, you will have a function called **`toUpperCaseWord`** which will depend on another function called **`toUpperCaseLetter`**. Both of these functions are provided for you in a separate text file. You can copy them as needed into your program but make sure you put the right parts into the right places. Remember that even though you have been given the implementation code, you must not show any code in your Design Assignment (i.e., steps 1 – 5). You will need to use the specifications, prototypes, and stubs in appropriate places but remember that you will lose credit if you show any code in supporting functions in step 5. Also note that since you will be given the specifications, it is expected that you will know how to appropriately use the **`toUpperCaseWord`** function (you will not use the **`toUpperCaseLetter`** function but **`toUpperCaseWord`** depends on it). As final note, these two functions may not be counted as part of your five functions.

12) If you have any questions for following the six step process, review the pertinent sections of the online reference and/or the online code examples and videos, or check with the CS 135 Instruction Team.

13) You will develop a Design Assignment which includes the step 1 through 5 source code files by Thursday at 6:00 pm, and then you must develop the program code for next Monday at 6:00 pm. Further explanation of the required components and uploading process are provided near the end of this document.

14) You must also acquire three screen shots of the input and resulting output of your program operation (two screen shots each for a total of six). The first two pairs of screen shots should show the input used and the output displayed in the examples shown previously in this document. The other two should show examples of operations on the remaining shapes (e.g., rectangle and parallelogram). Again remember to annotate your screen shot document.

Turning in your Design Assignment:

Information:

Week: 5
Laboratory: 5
Design Assignment: 3
Due Date: 09/26, 6:00 pm

To turn in:

The first five steps of the Six Step Programming Process, including:

1. shapes3_s1.cpp
2. shapes3_s2.cpp
3. shapes3_s3.cpp
4. shapes3_s4.cpp
5. shapes3_s5.cpp

Upload these as separate files. Note that following the instructions for uploading your work is critical; you will lose points if you do not follow these instructions. Do not upload any files or in any format other than that specified here.

For information on how to turn in Design Assignments, refer to the "How to Turn in Design Assignments" in the "General Course Information" folder

Turning in your Programming Assignment:

Information:

Week: 5
Laboratory: 5
Programming Assignment: 3
Due Date: 09/30, 6:00 pm

To turn in:

1. The Word file containing the following:

   a. There should be at least three pairs screen shots (i.e., a total of six screen shots) for the programs as specified in item #14 previously in this document

   b. Remember to clearly annotate every displayed result

2. The executable file:
   a. shapes3.exe (shapes3_s6.exe is also acceptable)

3. The source code file:
   a. shapes3_6.cpp

These files <u>must</u> be compressed and uploaded as one zip file. To do this, select all of the required files, right click on them, and select “Send To”, then select “Compressed (zipped) Folder”.

Once the folder is created, it will be placed in the same folder in which you are working. Change the name of the zipped folder to “LastnameFirstname_PAX” (where ‘X’ is the number of the Programming Assignment) as shown in the following example: “LeveringtonMichael_PA03” (no quotes). After you have renamed the zipped folder, double click on it to verify that it has all the files it is supposed to have.

Note that following the instructions for uploading your work is critical; you will lose points if you do not follow these instructions. Do not upload any files or in any format other than that specified here.

For information on how to turn in Programming Assignments, refer to the "How to Turn in Programming Assignments" in the "General Course Information" folder