# Lab 01: Triangle class

**Problem Definition**

Objective: Upon completing this assignment, you should be able to implement a simple class, as well as gain a better understanding of the building and use of classes and objects.

### Task:

An equilateral triangle is a triangle whose sides are equal. You are to write a class called Triangle, using filenames triangle.h and triangle.cpp, that will allow the creation and handling of equilateral triangles, whose sides are integers in the range 1-39.

### Details:

1. The single constructor for the Triangle class should have 3 parameters: an integer size (required), which is the length of a side; a border character (optional, with a default of '#'); and a fill character (optional, with a default of '\*'). If the size provided is less than 1, set the size to 1. If the size provided is greater than 39, set the size to 39. The class will need to provide internal storage for any member data that must be kept track of.
2. There should be member functions GetSize, Perimeter, and Area, which will return the size of a side, the perimeter of the triangle, and the area of the triangle, respectively. The first 2 should return integer results. The Area function should return its result as a double.
3. There should be member functions Grow and Shrink, which will increase or decrease (respectively) the size of the triangle's sides by 1, unless this would cause the size to go out of bounds (out of the 1-39 range); in the latter case, Grow and Shrink should make no change to the size.
4. There should be member functions SetBorder and SetFill, which each allow a new border or fill character (respectively) to be passed in as a parameter. There is a chart of ASCII characters in an appendix of the textbook. The characters that should be allowed for the border or fill characters are any characters from the '!' (ascii 33) up through the '~' (ascii 126). If an attempt is made to set the border or fill characters to anything outisde the allowable range, the function should set the border or fill back to its original default (the ones listed for the constructor -- the border default is '#' and the fill default is '\*').

1. There should be a member function called Draw that will display a picture of the triangle on the screen. You may assume that the cursor is already at the beginning of a line when the function begins, and you should make sure that you leave the cursor on the line following the picture afterwards (i.e. print a newline after the last line of the triangle). Use the border character to draw the border of the triangle, and use the fill character to draw the internal characters. Separate the characters on a line in the picture by a single space to make the triangle look more proportional (to approximate the look of an equilateral triangle). You may not use formatting functions like setw to draw the triangle. This must be handled with loops. (You will only print out the newline, spaces, the border character, and maybe the fill character on any given line).
2. Provide a member function called Summary that displays all information about a triangle: its size, perimeter, area, and a picture of what it looks like. When displaying the area (decimal data), always show exactly 2 decimal places. Your output should be in the exact same format as mine (seen in the linked sample run below)
3. I am providing a sample driver program (called driver.cpp) that uses objects of type Triangle and illustrates sample usage of the member functions.
4. I have also provided the output from the sample execution of the driver.cpp program. Your class declaration and definition files must work with my main program, as-is (do not change the driver program to make your code work!). You are encouraged to write your own driver routines to further test the functionality of your class, as well. Most questions about the required behavior of the class can be determined by carefully examining my driver program and the sample execution. Keep in mind, this is just a sample. Your class must meet the requirements listed above in the specification -- not just satisfy this driver program. (For instance, I haven't tested every illegal fill character in this driver program -- I've just shown a sample). Your class will be tested with a larger set of calls than this driver program represents.
5. General Requirements
   * No global variables, other than constants!
   * All member data of your class must be private
   * You will need to use the <iostream> library for output. You may use the <iomanip> library for formatting your decimal output to two places, if you wish to use the parameterized stream manipulators, but you may not use setw() or other output formatting functions for drawing the actual triangle. You may use the <cmath> library
   * Do not use language or library features that are C++11-only
   * When you write source code, it should be readable and well-documented.
   * Your triangle.h file should contain the class declaration only. The triangle.cpp file should contain the member function definition

**What to Submit:**

On canvas submit the following files:

* triangle.h
* triangle.cpp