Programming Assignment #1

Due: 11:59pm, Wednesday, Jan. 30th

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 all three sides are equal. If two equilateral triangles are "glued" together along a common side, this will form a **diamond**. You are to write a class called **Diamond**, using filenames **`diamond.h`** and **`diamond.cpp`**, that will allow the creation and handling of diamonds based on the following description, whose sides are integers in the range 1-39.

Details:

1. The single constructor for the Diamond class should have **3 parameters**: an integer size (required), which is the length of a side; a border character (optional, with a default of `'#'`, whose ASCII code is 35); and a fill character (optional, with a default of `'*'`, whose ASCII code is 42). 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 needs 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 diamond, and the area of the diamond, 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 Diamond'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 that can be found online: https://www.ascii-code.com. 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 outside 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 `'*'`);

5. There should be a member function called **`Draw`** that will display a picture of the Diamond 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 diamond). Use the border character to draw the border of the diamond, 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 Diamond look more proportional (so that the halves look more like equilateral triangles). You may not use formatting functions like `setw` to draw the diamond. Instead, 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);

6. Provide a member function called **`Summary`** that displays all information about a diamond: 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 ours, which will be provided in a separate sample file;

7. We will also provide a sample program (called `program.cpp`) that uses objects of type Diamond and illustrates the usage of the member functions. We also provide the output from the execution of the `program.cpp` program. Your class declaration and definition files must work with our program, as-is (do not change our program to make your code work!). You are encouraged to write your own test routines to further test the functionality of your class, as well. **Please keep in mind**, `program.cpp` is just a **sample**. Your class must meet the specified requirements listed above in the specification -- not **just** satisfy **this** sample program. (For instance, we haven't tested **every** illegal fill character in this program -- we just show a sample). Your class will be tested with a larger set of calls than this sample program represents.

General Requirements

- No global variables, other than constants!
- All member data of your class must be private;
- You 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 diamond. You may use the `<cmath>` library;
- When you write source code, it should be readable and well-documented;
- Your diamond.h file should contain the class declaration only. The diamond.cpp file should contain the member function definitions.

Submitting:

Program submissions should be done through Canvas. Do **not** send program submissions through e-mail -- e-mail attachments will **not** be accepted as valid submissions.

General Advice - e-mail a copy of your finished homework files to your own FSU account. This e-mail will have a time stamp that shows when they were sent (i.e. before the due date would be the best idea), and they will also serve as a backup. It's not a bad idea to keep a copy on your CS account (as well as on a personal computer) -- backing up your work is a GOOD thing!

For HW #1, submit a zipped package (in `.tar.gz` format) including the following files

```
diamond.h
diamond.cpp
makefile (the final executable is named prog1)
readme (optional)
```

Make sure your filenames are these exact names, and do not submit the `program.cpp` file (or any other main program you might create for testing purposes).