HW #3 (Diamond)

- 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.
- An equilateral triangle等邊三角形 is a triangle whose 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 C++, or Java if you prefer, that will allow the creation and handling of diamonds based on the above description, whose sides are integers in the ranger 1-39.

HW #3 (2)

- 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 ‘#’); 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 must be kept track of.

HW #3 (3)

- 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.

- There should be member functions **Grow** and **Shrink**, which will increase or decrease (respectively) the size of the Diamond's side 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.

HW #3 (4)

- There should be member functions **SetBorder** and **SetFill**, which each allows a new border or fill character (respectively) to be passed in as a parameter. The characters that should be allowed for the border or fill characters are 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 ‘*’).

HW #3 (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 new line 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.

HW #3 (6)

- 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 function like **setw** to draw the diamond. 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.)
- 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. You output should be in the exact same format as sample run below.

HW #3 (7)

- Following is a sample driver program that uses objects of type Diamond and illustrates the usage of the member functions.

```
#include <iostream>
#include <iomanip>
#include "diamond.h“

using namespace std;

int main()
{
  // set decimal outputs to 2 decimal places
  cout << setiosflags( ios::fixed | ios::showpoint ) << setprecision( 2 );
```

```
// create some Diamonds
Diamond d1( -5 ), d2( 7, '^' ), d3( 12, 'X', 'O' ), d4( 50 , '$' , 'o');
// display original Diamonds
cout << "d1 has size = " << d1.GetSize() << " units.\n";
d1.Draw();
cout << "\nd2 has size = " << d2.GetSize() << " units.\n";
d2.Draw();
cout << "\nd3 has size = " << d3.GetSize() << " units.\n";
d3.Draw();
cout << "\nd4 has size = " << d4.GetSize() << " units.\n";
d4.Draw(); cout << '\n';

d1.Shrink(); // demonstrate shrink
d2.Shrink();
d3.Grow(); // and grow
d4.Grow();
cout << "d1 now has size = " << d1.GetSize() << " units.\n";
cout << "d2 now has size = " << d2.GetSize() << " units.\n";
cout << "d3 now has size = " << d3.GetSize() << " units.\n";
cout << "d4 now has size = " << d4.GetSize() << " units.\n";
```

```
// demonstrate perimeter
cout << "d2 has perimeter = " << d2.Perimeter() << " units.\n";
cout << "d3 has perimeter = " << d3.Perimeter() << " units.\n";
// and area
cout << "d2 has area = " << d2.Area() << " square units.\n\n";
cout << "d3 has area = " << d3.Area() << " square units.\n\n";

d1.Draw();
d1.Grow(); // show that fill character
cout << "d1 grows:\n"; // appears only when size
d1.Draw(); // is at least 3
d1.Grow();
cout << "... and grows:\n";
d1.Draw();
cout << '\n';
d1 = d2; // demonstrate the default overload of the
// assignment operator
cout << "d1 now has size = " << d1.GetSize() << " units.\n";
d1.Draw();
```

```
  // demonstrate the changing of border and fill characters
  d2.SetBorder('@');
  d2.SetFill('-');
  cout << "d2 now looks like:\n";
  d2.Draw();
  cout << '\n';
  d2.SetBorder('\n'); // illegal border
  d2.SetFill('\a'); // illegal fill
  cout << "d2 now looks like:\n";
  d2.Draw();
  cout << '\n';

  cout << "\nHere is a summary on d3:\n"; // demonstrate summary
  d3.Summary();

  return 0;
}
```

HW #3 (8)

- Use the **main function** to test your class. Your class must work with this code, without alteration. Your class will be tested with a large set of calls than this driver program represents.
- We have also provided the output from the sample execution of the driver program.
- 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 the driver program and the sample execution.

```
d1 has size = 1 units.
#

d2 has size = 7 units.
      ^
     ^ ^
    ^ * ^
   ^ * * ^
  ^ * * * ^
 ^ * * * * ^
^ * * * * * ^
 ^ * * * * ^
  ^ * * * ^
   ^ * * ^
    ^ * ^
     ^ ^
      ^

d3 has size = 12 units.
           X
          X X
         X O X
        X O O X
       X O O O X
      X O O O O X
     X O O O O O X
    X O O O O O O X
   X O O O O O O O X
  X O O O O O O O O X
 X O O O O O O O O O X
X O O O O O O O O O O X
 X O O O O O O O O O X
  X O O O O O O O O X
   X O O O O O O O X
    X O O O O O O X
     X O O O O O X
      X O O O O X
       X O O O X
        X O O X
         X O X
          X X
           X

d4 has size = 39 units.
                                      $
                                     $ $
                                    $ o $
                                   $ o o $
                                  $ o o o $
                                 $ o o o o $
                                $ o o o o o $
                               $ o o o o o o $
                              $ o o o o o o o $
                             $ o o o o o o o o $
                            $ o o o o o o o o o $
                           $ o o o o o o o o o o $
                          $ o o o o o o o o o o o $
                         $ o o o o o o o o o o o o $
                        $ o o o o o o o o o o o o o $
                       $ o o o o o o o o o o o o o o $
                      $ o o o o o o o o o o o o o o o $
                     $ o o o o o o o o o o o o o o o o $
                    $ o o o o o o o o o o o o o o o o o $
                   $ o o o o o o o o o o o o o o o o o o $
                  $ o o o o o o o o o o o o o o o o o o o $
                 $ o o o o o o o o o o o o o o o o o o o o $
                $ o o o o o o o o o o o o o o o o o o o o o $
               $ o o o o o o o o o o o o o o o o o o o o o o $
              $ o o o o o o o o o o o o o o o o o o o o o o o $
             $ o o o o o o o o o o o o o o o o o o o o o o o o $
            $ o o o o o o o o o o o o o o o o o o o o o o o o o $
           $ o o o o o o o o o o o o o o o o o o o o o o o o o o $
          $ o o o o o o o o o o o o o o o o o o o o o o o o o o o $
         $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
        $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
       $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
      $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
     $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
    $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
   $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
  $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
 $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
$ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
 $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
  $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
   $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
    $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
     $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
      $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
       $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
        $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
         $ o o o o o o o o o o o o o o o o o o o o o o o o o o o o $
          $ o o o o o o o o o o o o o o o o o o o o o o o o o o o $
           $ o o o o o o o o o o o o o o o o o o o o o o o o o o $
            $ o o o o o o o o o o o o o o o o o o o o o o o o o $
             $ o o o o o o o o o o o o o o o o o o o o o o o o $
              $ o o o o o o o o o o o o o o o o o o o o o o o $
               $ o o o o o o o o o o o o o o o o o o o o o o $
                $ o o o o o o o o o o o o o o o o o o o o o $
                 $ o o o o o o o o o o o o o o o o o o o o $
                  $ o o o o o o o o o o o o o o o o o o o $
                   $ o o o o o o o o o o o o o o o o o o $
                    $ o o o o o o o o o o o o o o o o o $
                     $ o o o o o o o o o o o o o o o o $
                      $ o o o o o o o o o o o o o o o $
                       $ o o o o o o o o o o o o o o $
                        $ o o o o o o o o o o o o o $
                         $ o o o o o o o o o o o o $
                          $ o o o o o o o o o o o $
                           $ o o o o o o o o o o $
                            $ o o o o o o o o o $
                             $ o o o o o o o o $
                              $ o o o o o o o $
                               $ o o o o o o $
$ o o o o o $
 $ o o o o $
  $ o o o $
   $ o o $
    $ o $
     $ $
      $


d1 now has size = 1 units.
d2 now has size = 6 units.
d3 now has size = 13 units.
d4 now has size = 39 units.
d2 has perimeter = 24 units.
d3 has perimeter = 52 units.
d2 has area = 31.18 square units.

d3 has area = 146.36 square units.


#
d1 grows:
 #
# #
 #
... and grows:
  #
 # #
# * #
 # #
  #


d1 now has size = 6 units.
     ^
    ^ ^
   ^ * ^
  ^ * * ^
 ^ * * * ^
^ * * * * ^
 ^ * * * ^
  ^ * * ^
   ^ * ^
    ^ ^
     ^
d2 now looks like:
     @
    @ @
   @ - @
  @ - - @
 @ - - - @
@ - - - - @
 @ - - - @
  @ - - @
   @ - @
    @ @
     @


d2 now looks like:
     #
    # #
   # * #
  # * * #
 # * * * #
# * * * * #
 # * * * #
  # * * #
   # * #
    # #
     #



Here is a summary on d3:
Size of diamond's side = 13 units.
Perimeter of diamond = 52 units.
Area of diamond = 146.36 units.
Diamond looks like:
            X
           X X
```

```
          X O X
         X O O X
        X O O O X
       X O O O O X
      X O O O O O X
     X O O O O O O X
    X O O O O O O O X
   X O O O O O O O O X
  X O O O O O O O O O X
 X O O O O O O O O O O X
X O O O O O O O O O O O X
 X O O O O O O O O O O X
  X O O O O O O O O O X
   X O O O O O O O O X
    X O O O O O O O X
     X O O O O O O X
      X O O O O O X
       X O O O O X
        X O O O X
         X O O X
          X O X
           X X
            X
```

```
import lib.diamond;

public class test {
  public static void main(String [] args) {
    diamond d1 = new diamond (-5);
    diamond d2 = new diamond (7,'^');
    diamond d3 = new diamond (12,'X','O');
    diamond d4 = new diamond (50,'$','o');

    System.out.println("d1 has size = " + d1.GetSize() + " utils");
    d1.Draw();
    System.out.println("d2 has size = " + d2.GetSize() + " utils");
    d2.Draw();
    System.out.println("d3 has size = " + d3.GetSize() + " utils");
    d3.Draw();
    System.out.println("d4 has size = " + d4.GetSize() + " utils");
    d4.Draw();

    d1.Shrink(); d2.Shrink();
    d3.Grow(); d4.Grow();
    System.out.println("d1 now has size = " + d1.GetSize() + " utils");
    System.out.println("d2 now has size = " + d2.GetSize() + " utils");
    System.out.println("d3 now has size = " + d3.GetSize() + " utils");
    System.out.println("d4 now has size = " + d4.GetSize() + " utils");

    System.out.println("d2 has perimeter = " + d2.Perimeter() + " utils");
    System.out.println("d3 has perimeter = " + d3.Perimeter() + " utils");
    System.out.println("d2 has area = " + d2.Area() + " utils");
    System.out.println("d3 has area = " + d3.Area() + " utils");
d1.Draw();
d1.Grow();
System.out.println("... and grows:");
d1.Draw();
System.out.println();

d1=d2;
System.out.println("d1 now has size = " + d1.GetSize() + " utils");
d1.Draw();

d2.SetBorder('@');
d2.SetFill('-');
System.out.println("d2 now looks like:");
d2.Draw();
System.out.println();

d2.SetBorder('\n');
d2.SetFill('\n');
System.out.println("d2 now looks like:");
d2.Draw();
System.out.println();

System.out.println("Here is a summary on d3:");
d3.Summary();

} }
```