IITM-CS2710 : Programming & Data Structures Lab (Jul-Nov 2018) Lab Session 2

Aug 6, 2018 **From C to C++ : Introduction to Classes in C++**[1] Mode : On Hacker Rank

Objectives:

- Implement a simple class with public and private members and multiple constructors.
- Gain a better understanding of the building and using of classes and objects.
- Practice problem solving using Object Oriented Programming.

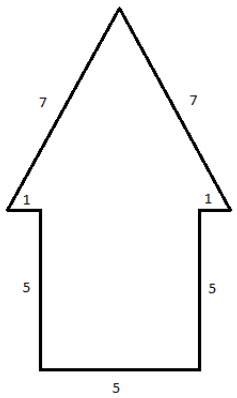
Overview:

In this assignment, you are expected to *write a class called House* that will allow creation and handling of House objects, as described below.

House:

Each house object should be pictured as follows:

- The base of the house is a square with a given side length. Minimum length is 3, maximum is 37. This will be referred to as the base size of the house.
- The roof of the house is an equilateral triangle.
- The roof will always overhang the base by one length unit on each side, so the length of the triangle's side will be 2 more than the length of the square's side
- Example house image, with base size 5:



Detailed Structure of Assignment Task:

This assignment has two components. The first component is to implement a `House` class as specified below. The second component is a driver program that uses the `House` class to draw this house.

[1]This assignment has been reproduced from http://www.cs.fsu.edu/ myers/cop3330/hw/hw01.html

Part I [InLab] : Designing the Class "House": We first discuss the design of the structure:

- The `House` class should have one private data member of the `int` type, *size* and two private data members of the `char` type, *border* and $fill$.
- The `House` class should have the following public functions:
  - `House(int s,char c1,char c2);` : For this three parameter constructor, you just need to make sure that size (s),border ($c1$) and fill($c2$) are legitimate and set $size = s$, $border = c1$ and $fill = c2$. If the size provided is less than 3, set the size to 3. If the size provided is greater than 37, set the size to 37.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 border default is 'X' and the fill default is '*').
  - There should be member functions `GetSize, Perimeter, and Area`, which will return the house's base size, the perimeter of the house, and the area of the house, 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 base size of the house by 1, unless this would cause the size to go out of bounds (out of the 3-37 range); in the latter case, Grow and Shrink should make no change to the size.
  - 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. 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 border default is 'X' and the fill default is '*').

Part II[Outside Lab] : Draw function and Driver Program:

- There should be a member function called `Draw` that will display a picture of the house on the screen. Use the border character to draw the border of the house, 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 house look more proportional (to approximate the look of an equilateral triangle and a square). You may not use formatting functions like setw to draw the figure. 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). Note that the bottom of the roof and the top of the base will be on a shared line – the overhang area will be border characters, but everything internal will be fills. See the sample output in moodle to see exactly how this should look.

After defining the class, you should write a driver program that reads in the line of six inputs as specified below and generates the corresponding output as specified. The driver performs the basic IO in C++.

Input-Ouput:

The input consists of 6 tuple (a b c d e f)

- a is of type int and is the base size of the house
- b is of type char and is the border character
- c is of type char and is the fill character
- d is of type int and can be 1 or 2 which denotes grow or shrink respectively
- e is of type int and can be 1 or 2 which denotes border or fill respectively
- f is of type char and is the new character for either border or fill as determined from above input.

The output should draw the house twice. The original with a,b,c. The modified according to d,e,f.

Sample Output for the input [5 $X * 1$ 1 ^] is available in moodle.