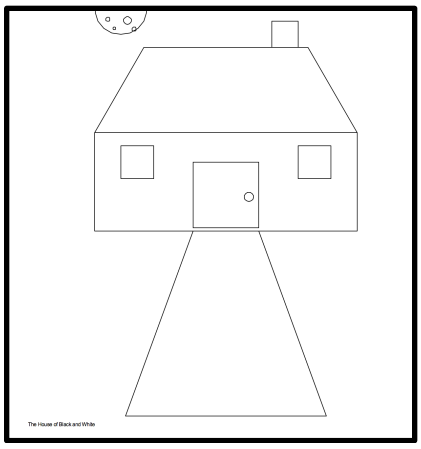
CS8
Lab 3
Functions

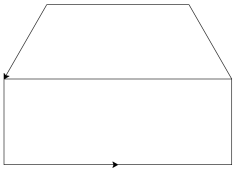
In Python, a function may be called inside another function. This is how complex tasks are solved and programmed. A complex task is subdivided into simpler tasks, then, the simpler tasks are converted into functions (they are easy to program). The complex task, a function, is now programmed as a series of function calls to the simpler tasks. In this lab, you will create several functions that help draw the scene below. I suggest you work in pairs to solve this problem.



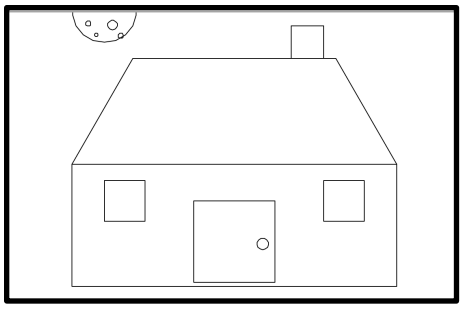
The TA will provide a Python program to help get you started (studenthouse.py). You will begin adding lines of code to complete the functions in the order listed below (these are the simple tasks). These functions are then called from the drawhouse() function (the complex task) to produce the image above.

- drawoutline()
- drawwindow()
- drawdoorknob() – used to draw all circular objects
- drawwalk()

After completing drawoutline(), a call to this function produces the image:

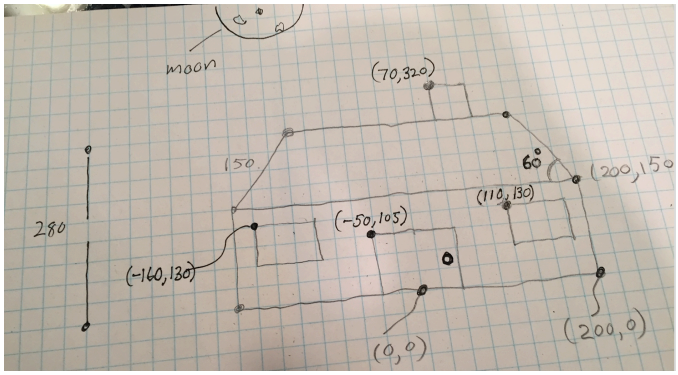


After completing drawwindow(), if you call this function four times inside the drawhouse() function, the first call draws the right-window (be careful, you must provide the x, y coordinates and the side length), the second call the left-window, the third call draws the door, and the last call draws the chimney. These four calls made after the outline of the house produces this image:



After completing the drawdoorknob() function, by placing six calls to drawdoorknob() inside drawhouse() allows you to draw a doorknob, a moon, and four craters on the moon to get the final image (after all this function simply displays a circle of radius r centered at the point (x,y)). Finally, complete the function drawwalk() that displays the walkway leading from the house to the street.

I'm providing a photo of my "design work" before programming this lab. I hope it helps you decide on the x, y coordinates, distances and radii needed as arguments to the function calls.



Getting Help
Inside the Python shell, type

>>> import turtle

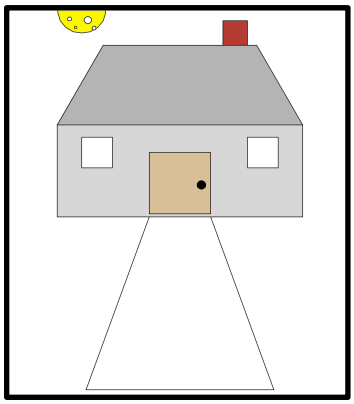
then type

>>> help(turtle)

A complete listing of the turtle module is displayed. By scrolling up and down you can see the functions provided in the turtle module available to you and how they are called. In addition, in lecture 3 there is a link to a Turtle document with additional examples of using turtle graphics.

Additional stimulation
Last but not least, add color to the scene. Did someone mention a silvery moon or was it a golden one! To do this, make a copy of your program and rename it ***colorhouse.py***. In each function where a color is to be used, add an additional

parameter at the end of the parameter list, say **c,** for the color. Change each function call by adding one additional argument to the end for specifying the color, say 'red', or, 'tan', or 'darkgray' or your choice. Note that a color is simply a string!



Program Submission

Your TA may choose to have each of you execute your final solution, so he can see that you've successfully built this house under a silvery moon. Or, he may request a formal submission to CourseWeb. Have fun!