CSC 471: Program 2 – Due Monday, April 29, 2013 at 11:59pm. The late policy on the syllabus applies. This programming assignment should be done **individually**! You may talk to one another about the program, but you may not look at someone's working code.

Objectives: Learn and apply 3D transformations.

Write a program that allows you to "build" a house by manipulating two different meshes to "compose" them in the same space. In general, your code will display a 3D cube and a 3D "peaked roof" (pyramid shape) next to one another on a ground plane. Your program must allow the user to manipulate the objects directly using the mouse. The goal is to allow the user to transform the cube and roof in order to scale, translate and rotate them in order to compose a "house". Transforms must be performed in a very specific way described below – in general rotations will be the largest technical challenge and you need to implement them via the 'virtual trackball' algorithm.

I encourage you to do this task in stages. To start, consider using the base code from one of your labs. Please be sure to use a VBO and IBO to represent the cube and roof. I also encourage you to use glm to control the matrix transforms to the objects. Note we will use a perspective projection (similar to Lab) – For this program, we will not be changing the view or projection matrix, only the model. Then consider adding transforms one at a time.

Tasks:

- Allow the user to toggle between the two objects via selecting the "1" key to transform the cube and the "2" key to transform the roof. Use a change of color to "highlight" the current object.
- Add the ability to transform the objects using the mouse and a keyboard press. The user will select which type of transform using the keyboard from: translate, rotate, scale. The "t" key will map to a translation, the "r" key will map to a rotation and the "s" key will map to scale. The user should be able to apply multiple transforms. **All the transformations are relative to the object's center!** Work on translations and scaling first and get these two transforms to work together.
    - Use the mouse to determine the amount to translate or scale the object. Click down anywhere in the OpenGL window, move the cube in the direction that the mouse is moving. Translations need only be done for X and Y. You can either convert viewport (pixel) coordinates from the amount of mouse movement to world coordinates to determine how far to move the cube or use a reasonable scaling term. Set bounds on the possible translation and scaling amounts so that the object(s) do not fly off the screen. If the cube has previously been translated, and is then rotated, it should be rotated about its new position. You are not required to implement non-uniform scaling.
    - Implement a virtual trackball to rotate the current object. The rotation is again with respect to the cube's center. We will talk more about how to do this in

class. You must implement this on your own! Only use class notes as a resource.

- Add a reset key "h" to bring the current object back to the origin of the world with the original size and orientation.

Your program should work (as expected) when the window is resized or moved. You will need to hand in all your code and an example image of a house you built using your own tool.

Point breakdown (A completely functional program is worth 100 points total.):

20 points for working translate
20 points for working scale
30 points for working rotate
20 points for transforms working together
10 points for general program behavior