Assignment 2b: Programming Interactive 2D Graphics with OpenGL and GLFW

The goal of this assignment is to give you experience with using simple 2D geometric transformations in a vertex shader to allow a user to interactively reposition and resize a single object in a window via the mouse and keyboard. You will use GLFW to read the user input.

Specifically you are asked to:

1) Define a simple 2D geometric object, such as a triangle

2) Associate distinct color information with each vertex, to avoid symmetry that could potentially confound the appreciation of a rotation operation.

You may either hard-code your object description into your program, or have your program read an object description from a file, leveraging the input code you wrote for assignment 1. Please be sure to define your model so that it is asymmetric, so that you will be able to easily differentiate changes in its height, width, and/or orientation after it has been resized and/or rotated.

3) Write a program, using OpenGL and GLFW that:

    a. Allows the user to scale the model about its centroid and along its intrinsic width or height axes using the arrow keys on the keyboard, such that the left arrow key causes the object to become thinner, the right arrow key causes the object to become fatter, the up arrow causes the object to become taller, and the down arrow causes the object to become more squat, regardless of the object's orientation in the image plane.

    b. Allows the user to rotate the model about its centroid by moving the mouse while the left button is pressed. The object should rotate in a clockwise direction when the mouse is moved to the right, and in a counterclockwise direction when the mouse is moved to the left. To facilitate rotating the object about its centroid, the object model can be defined to have (0, 0) at its center.

    c. Allows the user to translate the model within the display window by moving the mouse while the left button is pressed and the 'alt' key is down. The direction of motion of the object should be defined by the direction of motion of the mouse and the amount of object motion should be matched as closely as reasonably possible to the

amount of mouse movement. This means that when the user moves the mouse upwards by 10 pixels, the object should move upwards by about 10 pixels, regardless of its rotational orientation. Note that the mouse coordinates will be 'flipped' along the *y* axis with respect to the world coordinates of your scene because the origin of the window is defined to be at its upper left corner.

What you should turn in

- all of your source code
- one image showing the results of running your program
- a readme in which you provide detailed instructions to the TA on how to compile your program