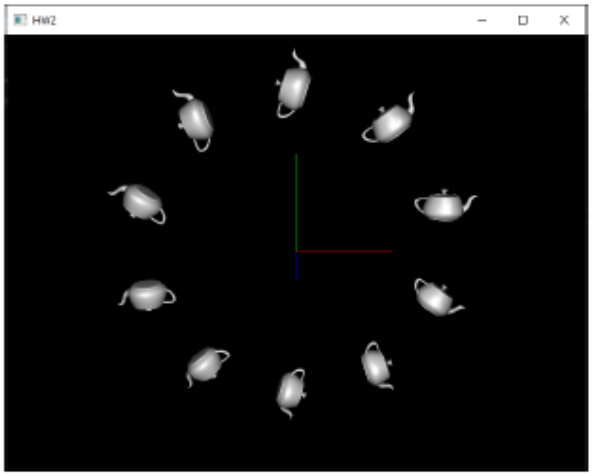
COSC 4370 - Homework 2

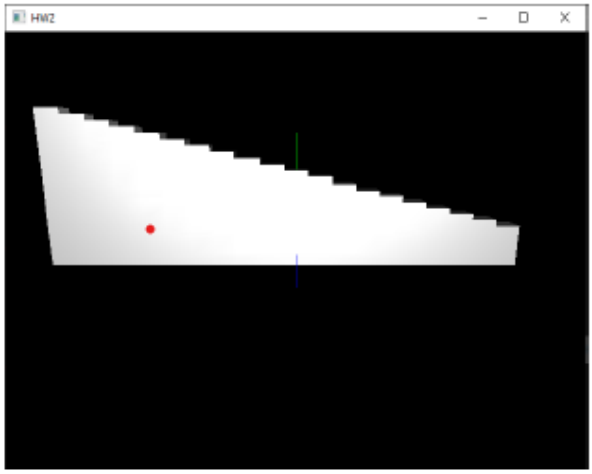
Nick Hiller - 1896327

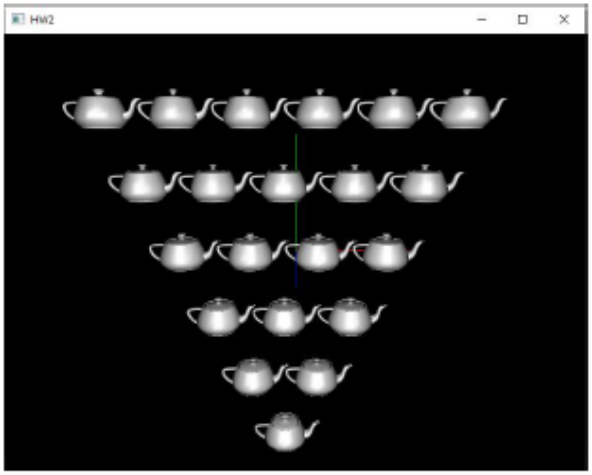
October 4

1 Problem

For this homework, you will be using your newfound OpenGL skills from class, as as well as your artistic creativity, to create several 3D scenes with OpenGL. The first part of the assignment is to reproduce each of the following three images:







The second part of the assignment is to create, using similar techniques, a scene of your own imagination. All the code for this homework lives in main.cpp; you need to fill in the functions problem1, problem2, problem3, and problem4. You can switch between the different examples while the program is running by pressing the 1,2, . . .

keys. Hence you don't need to recompile in order to run different examples. Additionally, you can quit the program at any time by pressing 'q', 'Q', or the Escape key.

2 Method

From the given source code I altered the 4 functions that were provided to solve each problem. I used the functions given to us with the OpenGL Library to recreate the images that were provided in the original instruction document. I used glutsolidteapot(), glutsolidcube(), glutsolidsphere, glpushmatrix(), glrotatef(), gltranslatef(), and glpopmatrix as well. I also added two functions of my own that would assist me in some repetitive code to reduce excessive lines in the code.

3 Implementation

Given the 7 functions from the OpenGL library I began designing the first problem, rendering a circle of teapots. I started with turning the initial teapot 90 degrees, using glrotatef(), and went from there. I would move each teapot the same distance with gltranslatef(), and then rotate each by 36 degrees because the problem had 10 teapots so I divided 360/10 to get 36 degrees each rotation. Following problem 2 I created my own function so that it would make a set of cubes along the x axis with the starting position already set in the main function. The purpose of making the secondary function was to reduce repetitiveness because this was the same process for each layer of steps. I would make the initial level in the main function and have a for loop calling the secondary function passing it the level it was on and before calling the next level in the secondary function I would translate the position up and to the left to design the first step. Problem 3 had the same setup as problem 2 because it also had the same repetitiveness as problem 2 because each level was just adding an additional teapot as it grew taller. The main function would place the first teapot and then move up to the left and call the secondary function to make a layer of teapots. Spacing them all out equally then at the end would go up and right to get the next edge and call the secondary function again, this went through until all levels were made and was comparable to the picture shown in the instruction document. For the final problem where I got to pick my own rendering I chose to make a diamond using the glutsolidsphere() function and I also used glpushmatrix() and glpopmatrix(). I started by making one side of the diamond and to keep track of my positions I used glpushmatrix to store my starting position before I went up the sides. Once I did the initial side I would use glpopmatrix() to return to my beginning

position, this helped me with ease of designing because I didn't have to back track my position to the beginning and instead was able to use the function to return back to the start. I repeated the same process for the other half of the diamond thus creating the final rendering you see below.

4 Results

The output of the program was the following four renderings, a circle of 10 teapots, a staircase, an upside down pyramid of teapots consisting of 6 rows each adding a teapot onto the next row, and finally I designed a symmetrical diamond of connecting spheres.