# COSC 4370 - Homework 1

Rizwan Lokhandwala PSID: 1887820

October 2022

## 1 Problem

The assignment was a introduction to openGL. For problem 1, we needed to make a circle of teapots (each rotated a different amount)

For Problem 2, we needed to make a staircase of cubes.

For Problem 3, we needed to make a pyramid of teapots.

For Problem 4, I created some poker chips and organized them as if they were on a tabletop being played with. (no tabletop included)

## 2 Method: Problem 1

Only main.cpp is to be modified. Problem 1 was hardcoded.

Each object was created in its own "glPushMatrix()" which allows for translations and rotations to be automatically cleared after "glPopMatrix()" is called.

I translated and rotated the matrix before calling "glutSolidTeapot()" to create the teapot in the desired location.

I did this for each teapot in order to recreate the image.

## 3 Method: Problem 2

The thickness of the staircase was wider than the indentation on each step, because of this, we had to create many small cubes to form larger rectangular prisms.

In order to do this, I used a universal cubeSize of y-indentation.

This way, with each run through my nested forloops, the y coordinate could be incremented by: y-start -= cubeSize;

I made use of 4 nested for loops. The first incremented through x-coordinates (ONLY ON THE SAME "Y" HEIGHT)

The second loops incremented through y-coordinate, beginning at variable "y-start" and ending at "y-end". Y start continously decremented with each step of x-coordinate.

The third loop incremented through z-coordinate to account for the thickness of the steps.

The fourth loop would loop through x in increments of cubeSize. Since the cubeSize was smaller than the horrizontal distance between each step, This fourth loop allows us to increment between each "X" that shares the same max height "Y".

With this algorithm ,the staircase was created.

## 4 Method: Problem 3

The pyramid algorithm was a simple nested for loop, with the first incrementing through the y coordinate and the second incrementing though x.

it follows the basic format of:

For each y in range -3:2, For each x in range -(y+4)/2 : (y+4)/2 place a teapot at (x,y,0)

I then added some extra transformation to better fit in the window.

# 5 Method: Problem 4

Problem 4 made use of some structs/functions.

Structs: Color - just stored 3 floats (R,G,B)

functions: void setColor(Color) - simply set the color using glColor3f(r,g,b). This just made setting colors easier by allowing me to pass 1 parameter rather than 3.

bool toggle(bool) - toggled a boolean between true and false.

void drawPokerChip(float diameter, Color chipColor) - This is the main function and will be explained in implementation.

These functions were used to create Problem 4.

The main function "Problem4()" simply hardcoded some poker chips using for loops to look nice. Same method for the most part as the other three problems.

# 6 Implementation

drawPokerChip(float diameter, Color chipColor)

the only parameters is the diameter and chipColor. With the diameter, I create some constant floats:

RADIUS = diameter/2 HEIGHT = diameter/10

and cubeSize is set to 0.01

I use Color "curOutlineCcolor" and "boolean "is-grey" to keep track of which color I will be using (the edge/outline of the chip has an alternating pattern)

I begin by traversing through X in increments of cubeSize. I only traverse the region bounded by

[x — -(RADIUS * cos(1.309 rad)) ¡ x ¡ RADIUS * cos(1.309 rad)]

This provides a 30 degree arc from [75, 105].

at each x, I determine the z coordinate of the next cube and toggle through outline colors.

I then traverse through the y-coordinates (the height of the chip)

At each coordinate (x,y,z) i place a cube. I then rotate the matrix by 30 degrees and repeat 6 times. This forms a complete and round circle which acts as the outline of the poker chip.

Then I draw the top and bottom surface of the poker chip. This is where Color "chipColor" comes in. That color the color of the front and back face of the chip.

To draw the surfaces, I increment through Z's from the outline towards the center, stopping before I exit the circle.

I make use of some booleans to know if I am at the border between the face and outline. If I am, I color it black. This surface is also indented by 1 cubeSize to add some a more dramatic 3d viewing experience.

This matrix is also rotated by 30 degrees and completed again completely filling the poker chip.

Thats the basis of my algorithm to create the poker chip.