**CSE 4200 Lab 8 – Spencer Wallace**

**Summary:** Lab completed successfully. I was able to use the program from the notes as a reference and animate a bouncing ball. Because I was able to complete this lab I am giving myself full points.

**Outputs**

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated

**Code**

#include <GL/glut.h>

#include <GL/gl.h>

#include <GL/glu.h>

#include <stdio.h>

#define drawOneLine(x1,y1,x2,y2) glBegin(GL\_LINES); glVertex2f ((x1),(y1)); glVertex2f ((x2),(y2)); glEnd();

void init(void)

{

glClearColor(0, 0, 0, 0);

glPointSize(4.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-500, 500, -500, 500);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void Circle(int Move, int Radius)

{

int r = Radius;

int x = 0;int y = r;

int d = 3 / 2 - r; // = 1 - r

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(1.0, 1.0, 1.0); while (x <= y) {

glVertex2i( +x, y + Move);

glVertex2i( +y, x + Move);

glVertex2i( -x, y + Move);

glVertex2i( -y, x + Move);

glVertex2i( -x, -y + Move);

glVertex2i( -y, -x + Move);

glVertex2i( +y, -x + Move);

glVertex2i( +x, -y + Move);

if (d < 0)

d += (2 \* x) + 3;

else {

d += (2 \* (x - y)) + 5;

y -= 1;

}

x++;

}

glFlush();

}

int move = 20;

int radius = 60;

bool bounce = false;

void animate() {

if (bounce == true) { // bounce up

if ((move + radius) <= 300) {

move += 15;

}

if ((move + radius) >= 300) {

bounce = false;

}

}

if (bounce == false) {

move -= 10;

if ((move - radius) <= -200) {

bounce = true;

}

}

glutPostRedisplay();

}

void timerHandle(int value)

{

animate();

glutPostRedisplay();

glutTimerFunc(25, timerHandle, 0);

}

void visHandle(int visible)

{

if (visible == GLUT\_VISIBLE)

timerHandle(0);

else

;

}

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glBegin(GL\_POINTS);

Circle(move, radius);

glEnd();

glColor3f(0.0, 0.3, 0.2);

glLineWidth(5.0);

glBegin(GL\_LINE);

drawOneLine(-500, -210, 500, -210);

glEnd();

glutSwapBuffers();

}

int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowPosition(150, 150);

glutInitWindowSize(500, 500);

glutCreateWindow(argv[0]);

init();

glutDisplayFunc(display);

//glutVisibilityFunc(visHandle);

glutTimerFunc(25, timerHandle, 0);

glutMainLoop();

return(1);

}