Shivcharan T
205001100
CSE - B

SSN COLLEGE OF ENGINEERING, KALAVAKKAM
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UCS1712 – GRAPHICS AND MULTIMEDIA LAB

Lab Exercise 4: Midpoint Circle Drawing Algorithm in C++ using OpenGL

a) To plot points that make up the circle with center (xc,yc) and radius r using Midpoint circle drawing
algorithm. Give atleast 2 test cases.
Case 1: With center (0,0)
Case 2: With center (xc,yc)
b) To draw any object using line and circle drawing algorithms.

Aim:
To implement circle drawing mid point algorithm.

Algorithm:

1. Input radius r and circle center (xc, yc). set the first point (x0 , y0) = (0, r).
2. Calculate the initial value of the decision parameter as p0 = 1 – r.
3. At each xk position, starting at k = 0, perform the following test:
4. If pk <0,
5. plot(xk +1,yk)andpk+1 =pk +2xk+1 +1,
6. Else,
7. where2xk+1 =2xk +2and2yk+1 =2yk –2.
8. plot (xk+1,yk –1)andpk+1 =pk +2xk+1 +1–2yk+1,
9. Determine symmetry points on the other seven octants.
10. Move each calculated pixel position (x, y) onto the circular path centered on (xc, yc) and plot the coordinate values: x = x +xc ,y=y+yc
11. Repeat steps 3 though 5 until x y.
12. For all points, add the center point (xc, yc)

Code:

```
#include <stdio.h>
#include <GL/glut.h>
#include <math.h>
#include <cstring>
#include <iostream>
#define pi 3.142857
using namespace std;

int windowWidth = 1000;
int windowHeight = 1000;

void myInit(void)
{
    glClearColor(0.0, 0.0, 0.0, 1.0);
```

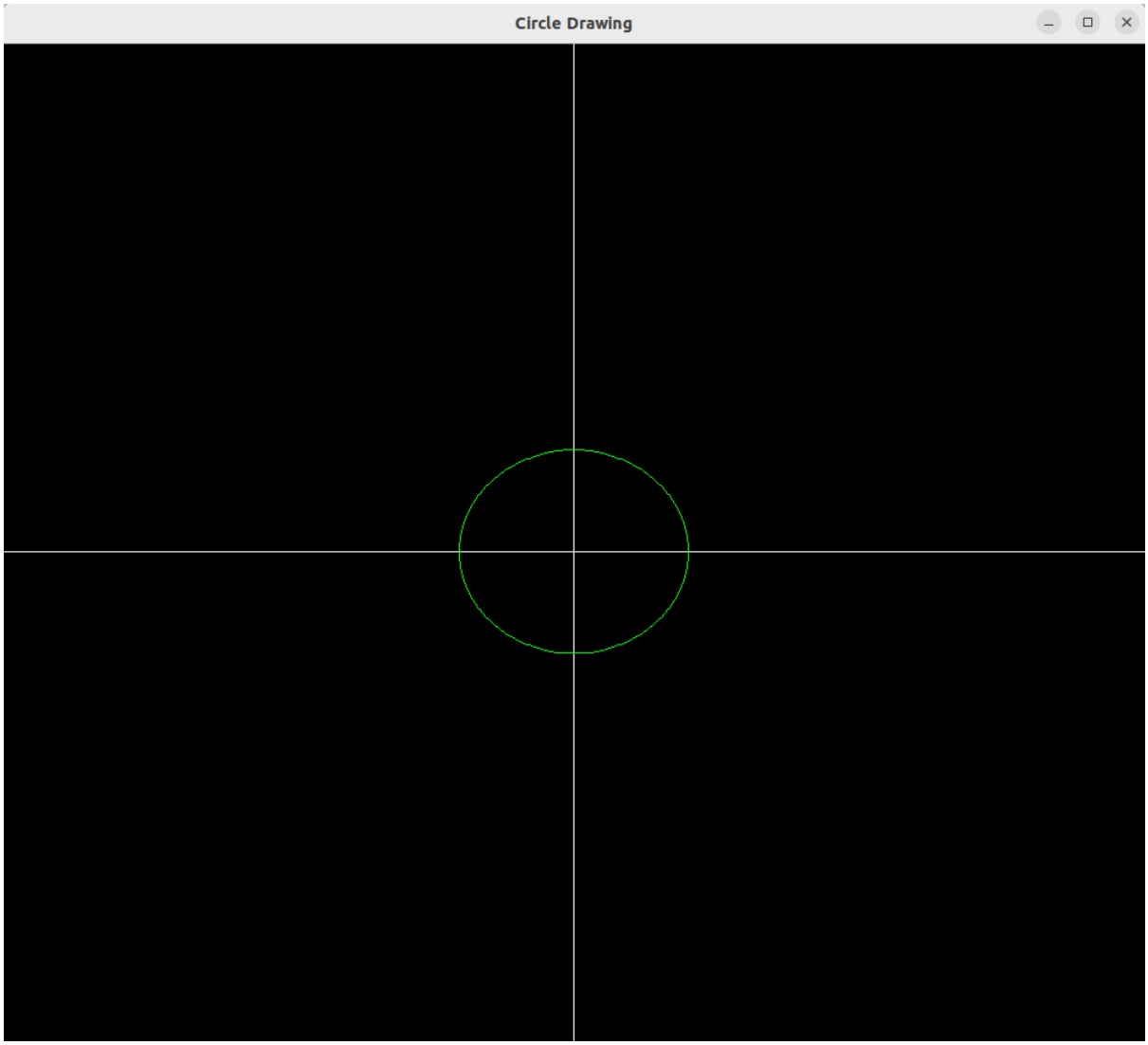
```
    glColor3f(0.0, 1.0, 0.0); // making picture color green (in RGB mode), as middle
argument is 1.0
    glPointSize(1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-windowHeight / 2, windowHeight / 2, -windowWidth / 2, windowWidth
/ 2);
}
void draw_pixel(int x, int y)
{
    glBegin(GL_POINTS);
    glVertex2i(x, y);
    glEnd();
    glFlush();
}
void drawLineDDA(float x0, float y0, float xn, float yn)
{
    // glClear(GL_COLOR_BUFFER_BIT);
    float dx = xn - x0;
    float dy = yn - y0;
    float steps = fabs(dx) > fabs(dy) ? fabs(dx) : fabs(dy);
    float xIncrement = dx / steps;
    float yIncrement = dy / steps;
    float x = x0;
    float y = y0;
    for (int i = 0; i <= steps; ++i)
    {
        draw_pixel(static_cast<int>(x + 0.5), static_cast<int>(y + 0.5));
        x += xIncrement;
        y += yIncrement;
    }
}

void draw_axis()
{
    glColor3f(1.0, 1.0, 1.0);
    glBegin(GL_LINES);
    glVertex2d(-2000, 0);
    glVertex2d(2000, 0);
    glEnd();

    glBegin(GL_LINES);
    glVertex2d(0, 2000);
    glVertex2d(0, -2000);
    glEnd();
    glFlush();
}
void draw_in_each_oct(GLint xk, GLint yk, GLint xc, GLint yc)
{
    draw_pixel(xc + xk, yc + yk);
    draw_pixel(xc + yk, yc + xk);
    draw_pixel(xc - yk, yc + xk);
    draw_pixel(xc - xk, yc + yk);
    draw_pixel(xc - xk, yc - yk);
    draw_pixel(xc - yk, yc - xk);
```

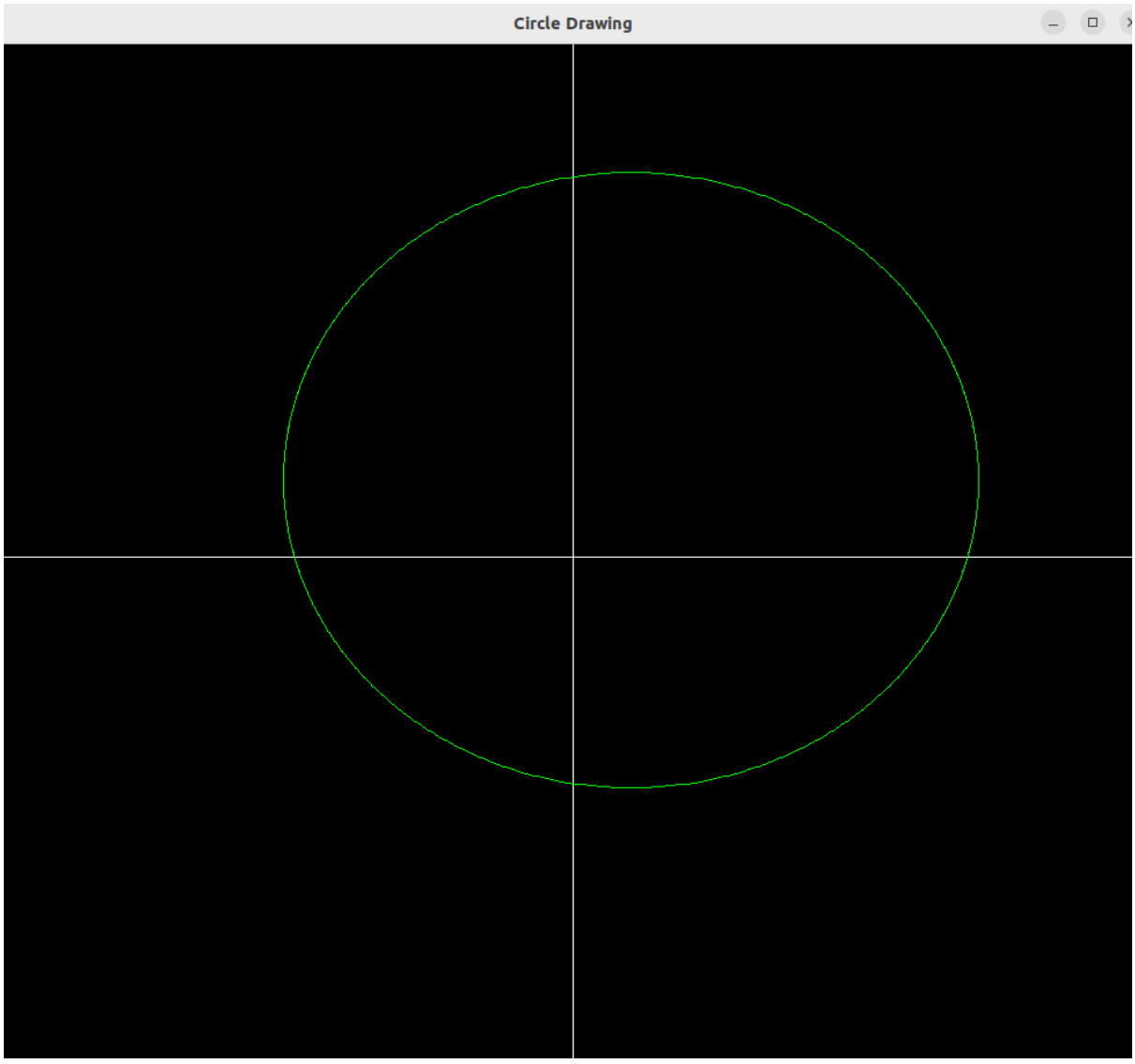
```
    draw_pixel(xc + yk, yc - xk);
    draw_pixel(xc + xk, yc - yk);
}

void midPtCircle(GLint xc, GLint yc, GLint r)
{
    GLint pk, xk, yk;
    pk = 1 - r;
    xk = 0;
    yk = r;
    draw_in_each_oct(xk, yk, xc, yc);
    while (xk <= yk)
    {
        if (pk < 0)
        {
            xk = xk + 1;
            pk = pk + (2 * xk) + 1;
        }
        else
        {
            xk = xk + 1;
            yk = yk - 1;
            pk = pk + (2 * xk) + 1 - (2 * yk);
        }
        draw_in_each_oct(xk, yk, xc, yc);
    }
}
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    draw_axis();
    glColor3f(0.0, 1.0, 0.0);

    GLint xc, yc, r;
    cout << "Enter xc, yc, radius(resp): ";
    cin >> xc >> yc >> r;
    midPtCircle(xc, yc, r);
    drawLineDDA(0, r, r, 0);
    drawLineDDA(r, 0, 0, -r);
    drawLineDDA(0, -r, -r, 0);
    drawLineDDA(-r, 0, 0, r);
}
int main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(windowHeight, windowWidth);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Circle Drawing");
    myInit();
    glutDisplayFunc(display);
    glutMainLoop();
    return 1;
}
```

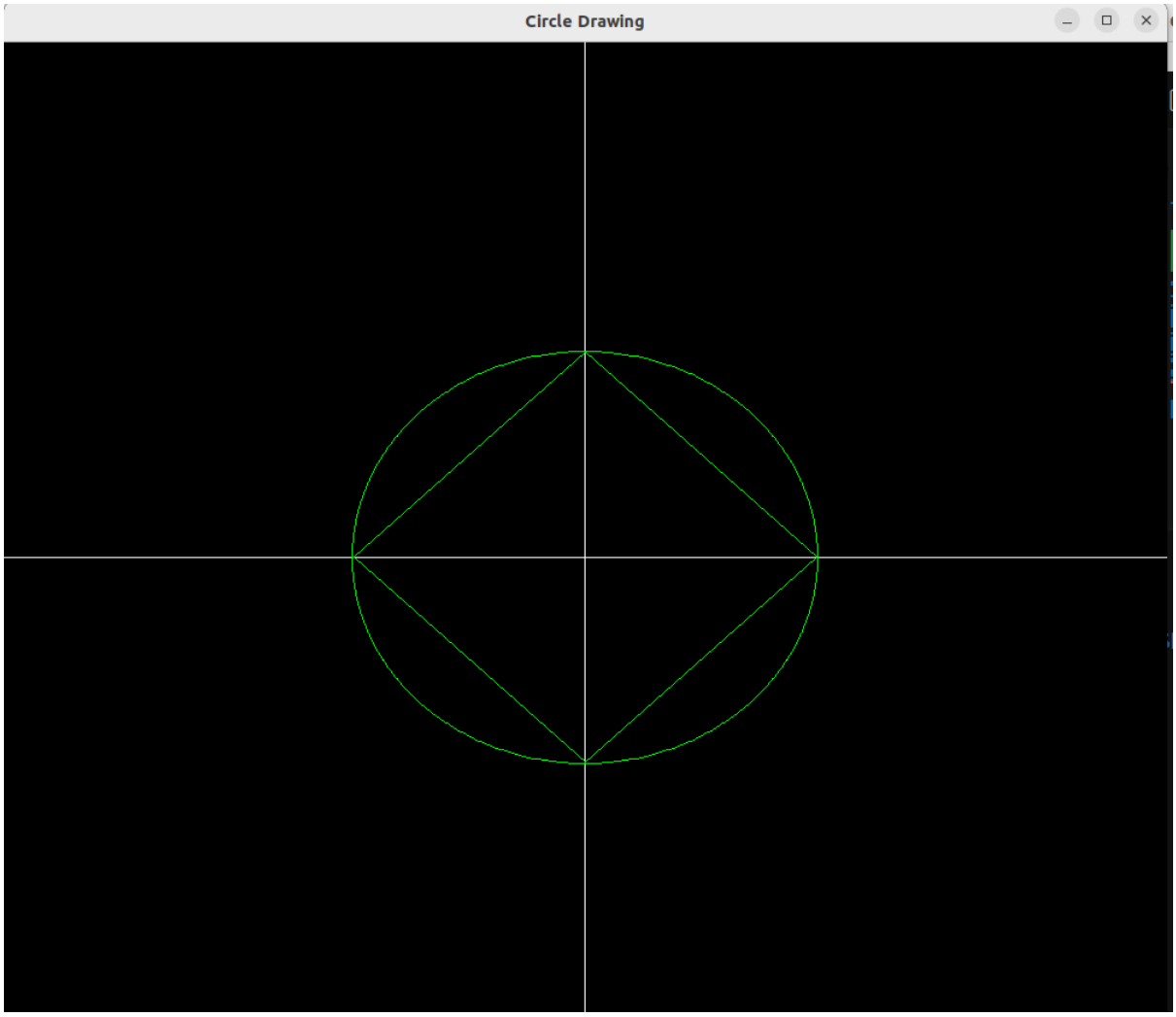
run.sh:

```
g++ 4.cpp -lGL -lglut -lGLU
./a.out
```

Sample I/O:




Circle Drawing



Learning Outcomes:

I learned how to use the midpoint circle drawing algorithm in c++ using the openGL library to draw circles.