UCS1712 – GRAPHICS AND MULTIMEDIA LAB

Ex. No. 4 Midpoint Circle Drawing Algorithm in C++ using OpenGL

Date: 9/8/21 **Name:** Srinath S

Class: CSE-C **Roll:** 185001205

Question:

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

Code:

a)

```
#include <stdlib.h>
#include <GL/glut.h>
#include <iostream>
using namespace std;

int xc, yc, r;

void myInit()
{
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glColor3f(0.4, 0.4, 0.9);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glPointSize(2);
    gluOrtho2D(-250.0, 250.0, -250.0, 250.0);
}

void plotAll(int x, int y, int xc, int yc)
{
    glVertex2d(x + xc, y + yc);
    glVertex2d(x + xc, -y + yc);
    glVertex2d(-x + xc, y + yc);
    glVertex2d(-x + xc, -y + yc);

    glVertex2d(y + xc, x + yc);
    glVertex2d(y + xc, -x + yc);
    glVertex2d(-y + xc, x + yc);
    glVertex2d(-y + xc, -x + yc);
}

void circle()
{
    int x = r, y = 0, pk = 1 - r;
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);
    plotAll(x, y, xc, yc);
    while (x > y)
    {
        y++;
        if (pk < 0)
        {
            pk += (2 * y) + 1;
        }
        else
        {
            x--;
            pk += (2 * y) - (2 * x) + 1;
        }
        plotAll(x, y, xc, yc);
    }
    glEnd();
    glFlush();
}

int main(int argc, char* argv[])
{
    cout << "Enter coordinates of line center of circle and radius xc,yc,r: ";
    cin >> xc >> yc >> r;

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Expt 04a - Mid Point Circle Algorithm");
    glutDisplayFunc(circle);
    myInit();
    glutMainLoop();
    return 1;
}
```

b)

```
#include <stdlib.h>
#include <GL/glut.h>
#include <iostream>
using namespace std;

int xc, yc, r;

void myInit()
{
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glColor3f(0.4, 0.4, 0.9);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glPointSize(2);
    gluOrtho2D(-250.0, 250.0, -250.0, 250.0);
}

void plotAll(int x, int y, int xc, int yc)
{
    glVertex2d(x + xc, y + yc);
    glVertex2d(x + xc, -y + yc);
    glVertex2d(-x + xc, y + yc);
    glVertex2d(-x + xc, -y + yc);

    glVertex2d(y + xc, x + yc);
    glVertex2d(y + xc, -x + yc);
    glVertex2d(-y + xc, x + yc);
    glVertex2d(-y + xc, -x + yc);
}
```

```
void circle()
{
    int x = r, y = 0, pk = 1 - r;
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POINTS);
    plotAll(x, y, xc, yc);
    while (x > y)
    {
        y++;

        if (pk < 0)
        {
            pk += (2 * y) + 1;
        }
        else
        {
            x--;
            pk += (2 * y) - (2 * x) + 1;
        }
        plotAll(x, y, xc, yc);
    }

    glVertex2d(100, 200);
    glVertex2d(50, 200);
    glVertex2d(20, 100);
    glVertex2d(40, 100);

    glEnd();
    glFlush();
}

int main(int argc, char* argv[])
{
    cout << "Enter coordinates of line center of circle and radius xc,yc,r: ";
    cin >> xc >> yc >> r;

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Expt 04a - Mid Point Circle Algorithm");
    glutDisplayFunc(circle);
    myInit();
    glutMainLoop();
    return 1;
}
```

Output: