/* **Experiment No. 3**

Problem Statement : Implement Bresenham circle drawing algorithm to draw any object. The object should be displayed in all the quadrants with respect to center and radius.

Roll No. : ITSA16

Name : KRUSHNA SUDHIR SHINDE

*/

Program Code :

```
#include <iostream>
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>

using namespace std;

struct Circle {
    int xc, yc;
    int radius;
    float r, g, b;
};

Circle circles[10];
int numCircles;

void plot_point(int xc, int yc, int x, int y) {
    glBegin(GL_POINTS);
    glVertex2i(x + xc, y + yc);
    glVertex2i(-x + xc, y + yc);
    glVertex2i(-x + xc, -y + yc);
    glVertex2i(x + xc, -y + yc);
    glVertex2i(y + xc, x + yc);
    glVertex2i(-y + xc, x + yc);
    glVertex2i(-y + xc, -x + yc);
    glVertex2i(y + xc, -x + yc);
    glEnd();
}

void bresenham_circle(int xc, int yc, int r) {
    int x = 0, y = r;
    float s = 3 - (2 * r);

    while (x <= y) {
        plot_point(xc, yc, x, y);
        if (s < 0) {
            s = s + (4 * x) + 6;
        } else {
            s = s + (4 * x) - (4 * y) + 10;
            y--;
        }
        x++;
    }
}

void concentric_circles() {
    glClear(GL_COLOR_BUFFER_BIT);
    glPointSize(4);
    glLineWidth(3);
    for (int i = 0; i < numCircles; i++) {
        glColor3f(circles[i].r, circles[i].g, circles[i].b);
        bresenham_circle(circles[i].xc, circles[i].yc, circles[i].radius);
    }

    glFlush();
}

void Init() {
    glClearColor(1.0, 1.0, 1.0, 0);
    gluOrtho2D(0, 1280, 0, 1024);  /
}

int main(int argc, char **argv) {
    cout << "Enter number of circles to draw (max 10): ";
    cin >> numCircles;
    if (numCircles > 10) numCircles = 10;

    for (int i = 0; i < numCircles; i++) {
        cout << "\nEnter details for Circle " << i + 1 << ":\n";
        cout << "Enter center (xc, yc): ";
        cin >> circles[i].xc >> circles[i].yc;
        cout << "Enter radius: ";
        cin >> circles[i].radius;
        cout << "Enter color (R G B, values between 0 and 1): ";
        cin >> circles[i].r >> circles[i].g >> circles[i].b;
    }

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowPosition(100, 100);
    glutInitWindowSize(1280, 1024);
    glutCreateWindow("Bresenham Circles");
    Init();
    glutDisplayFunc(concentric_circles);
    glutMainLoop();

    return 0;
}
```