```
#include<gl/glut.h>
#include<stdio.h>
#include<math.h>
int xc, yc, r;
int rx, ry, xce, yce;
void draw_circle(int xc, int yc, int x, int y)
{
        glBegin(GL_POINTS);
        glVertex2i(xc + x, yc + y);
        glVertex2i(xc - x, yc + y);
        glVertex2i(xc + x, yc - y);
        glVertex2i(xc - x, yc - y);
        glVertex2i(xc + y, yc + x);
        glVertex2i(xc - y, yc + x);
        glVertex2i(xc + y, yc - x);
        glVertex2i(xc - y, yc - x);
        glEnd();
}
void circlebres()
{
        glClear(GL_COLOR_BUFFER_BIT);
        int x = 0, y = r;
        int d = 3 - 2 * r;
        while (x <= y)
        {
                draw_circle(xc, yc, x, y);
                x++;
                if (d < 0)
                        d = d + 4 * x + 6;
                else
                {
                        y--;
                        d = d + 4 * (x - y) + 10;
                }
                draw_circle(xc, yc, x, y);
        }
        glFlush();
}
int p1_x, p2_x, p1_y, p2_y;
int point1_done = 0;
void myMouseFunccircle(int button, int state, int x, int y)
{
        if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN &&
point1_done == 0)
        {
                p1_x = x - 250;
                p1_y = 250 - y;
                point1_done = 1;
        }
        else if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
        {
                p2_x = x - 250;
                p2_y = 250 - y;
                xc = p1_x;
                yc = p1_y;
                float exp = (p2_x - p1_x) * (p2_x - p1_x) + (p2_y -
```

```
p1_y) * (p2_y - p1_y);
                r = (int)(sqrt(exp));
                circlebres();
                point1_done = 0;
        }
}


/////ELLIPSE////////////
void draw_ellipse(int xce, int yce, int x, int y)
{
        glBegin(GL_POINTS);
        glVertex2i(x + xce, y + yce);
        glVertex2i(-x + xce, y + yce);
        glVertex2i(x + xce, -y + yce);
        glVertex2i(-x + xce, -y + yce);
        glEnd();
}
void midptellipse()
{
        glClear(GL_COLOR_BUFFER_BIT);
        float dx, dy, d1, d2, x, y;
        x = 0;
        y = ry;

        // Initial decision parameter of region 1
        d1 = (ry * ry) - (rx * rx * ry) +
                (0.25 * rx * rx);
        dx = 2 * ry * ry * x;
        dy = 2 * rx * rx * y;

        // For region 1
        while (dx < dy)
        {

                // Print points based on 4-way symmetry
                draw_ellipse(xce, yce, x, y);



                // Checking and updating value of
                // decision parameter based on algorithm
                if (d1 < 0)
                {
                        x++;
                        dx = dx + (2 * ry * ry);
                        d1 = d1 + dx + (ry * ry);
                }
                else
                {
                        x++;
                        y--;
                        dx = dx + (2 * ry * ry);
                        dy = dy - (2 * rx * rx);
                        d1 = d1 + dx - dy + (ry * ry);
                }
        }
```

```
        // Decision parameter of region 2
        d2 = ((ry * ry) * ((x + 0.5) * (x + 0.5))) +
                ((rx * rx) * ((y - 1) * (y - 1))) -
                (rx * rx * ry * ry);

        // Plotting points of region 2
        while (y >= 0)
        {

                // Print points based on 4-way symmetry
                draw_ellipse(xce, yce, x, y);

                // Checking and updating parameter
                // value based on algorithm
                if (d2 > 0)
                {
                        y--;
                        dy = dy - (2 * rx * rx);
                        d2 = d2 + (rx * rx) - dy;
                }
                else
                {
                        y--;
                        x++;
                        dx = dx + (2 * ry * ry);
                        dy = dy - (2 * rx * rx);
                        d2 = d2 + dx - dy + (rx * rx);
                }
        }
        glFlush();
}
int p1e_x, p2e_x, p1e_y, p2e_y, p3e_x, p3e_y;
int point1e_done = 0;
void myMouseFunc(int button, int state, int x, int y)
{
        if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN &&
point1e_done == 0)
        {
                p1e_x = x - 250;
                p1e_y = 250 - y;
                xce = p1e_x;
                yce = p1e_y;
                point1e_done = 1;
        }
        else if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN &&
point1e_done == 1)
        {
                p2e_x = x - 250;
                p2e_y = 250 - y;
                float exp = (p2e_x - p1e_x) * (p2e_x - p1e_x) + (p2e_y
- p1e_y) * (p2e_y - p1e_y);
                rx = (int)(sqrt(exp));
                //midptellipse();
                point1e_done = 2;
        }
        else if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN &&
```

```
point1e_done == 2)
        {
                p3e_x = x - 250;
                p3e_y = 250 - y;
                float exp = (p3e_x - p1e_x) * (p3e_x - p1e_x) + (p3e_y
- p1e_y) * (p3e_y - p1e_y);
                ry = (int)(sqrt(exp));
                midptellipse();
                point1e_done = 0;
        }
}
void myDrawing()
{ }
void myDrawingc()
{ }

void minit()
{
        glClearColor(1, 1, 1, 1);
        glColor3f(1.0, 0.0, 0.0);
        glPointSize(3.0);
        gluOrtho2D(-250, 250, -250, 250);
}

void main(int argc, char* argv[])
{

        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
        glutInitWindowSize(500, 500);
        glutInitWindowPosition(0, 0);
        /*
        //FOR MOUSE
        int id1 = glutCreateWindow("Circle");
        glutSetWindow(id1);
        glutMouseFunc(myMouseFunccircle);
        glutDisplayFunc(myDrawingc);
        minit();
        glutInitWindowSize(500, 500);
        glutInitWindowPosition(600, 100);
        int id2 = glutCreateWindow("Ellipse");
        glutSetWindow(id2);
        glutMouseFunc(myMouseFunc);
        glutDisplayFunc(myDrawing);
        //END MOUSE
        */
        //FOR KEYBOARD
        printf("Enter 1 to draw circle , 2 to draw ellipse\n");
        int ch;
        scanf("%d",&ch);
        switch(ch){
        case 1:
        printf("Enter coordinates of centre of circle and radius\n");
        scanf("%d%d%d",&xc,&yc,&r);
        glutCreateWindow("Circle");
        glutDisplayFunc(circlebres);
        break;
```

```
        case 2:
        printf("Enter coordinates of centre of ellipse and major and
minor radius\n");
        scanf("%d%d%d%d",&xce,&yce,&rx,&ry);
        glutCreateWindow("Ellipse");
        glutDisplayFunc(midptellipse);
        break;
        }
        //END KEYBOARD
        minit();
        glutMainLoop();
        }
```