EX 4

205001085
SABARIVASAN V

AIM:

a) To plot points that make up the circle with center (xc,yc) and radius r using the 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.

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: Ifpk <0,
plot(xk +1,yk)andpk+1 =pk +2xk+1 +1,
Else,
where2xk+1 =2xk +2and2yk+1 =2yk –2.
plot (xk+1,yk –1)andpk+1 =pk +2xk+1 +1–2yk+1,
4. Determine symmetry points on the other seven octants. 5. 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
6. Repeat steps 3 through 5 until x y.
7. For all points, add the center point (xc, yc)

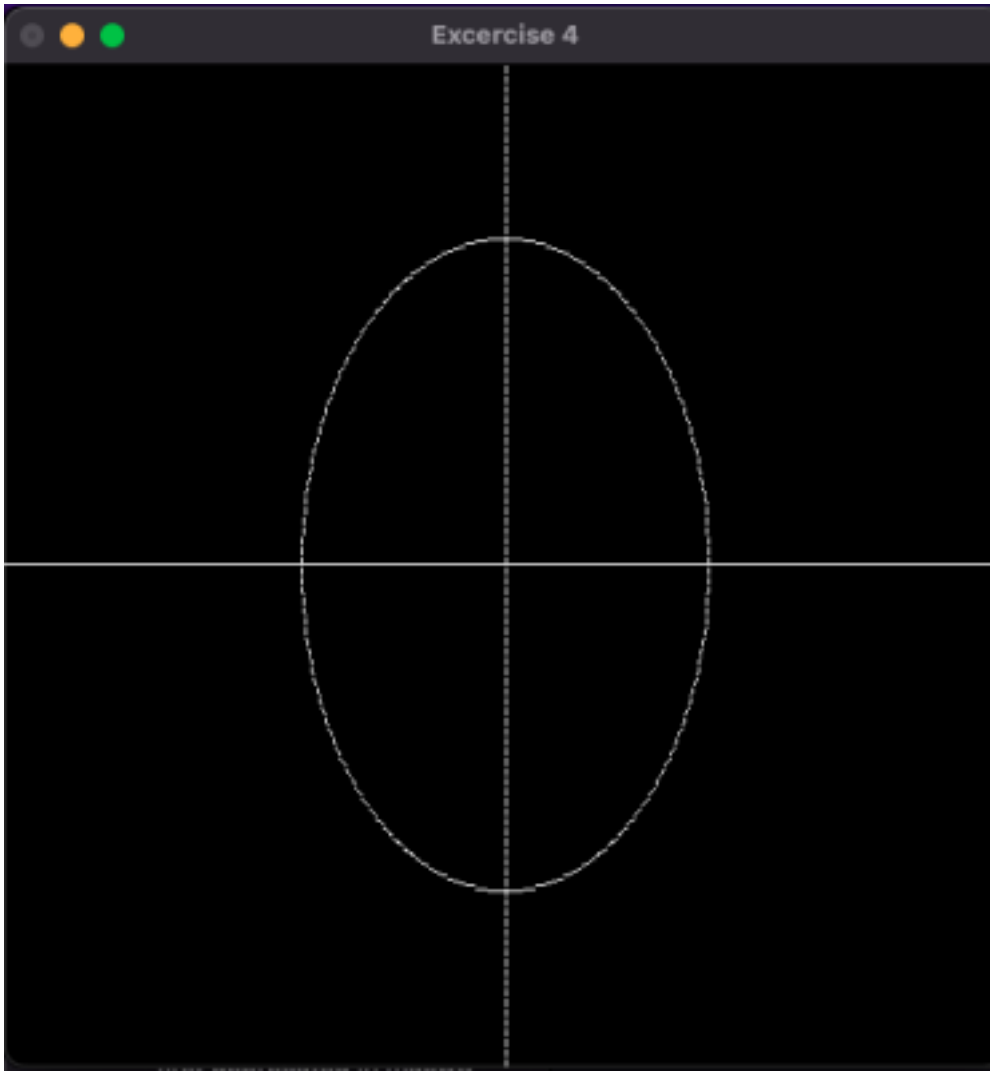
CODE:

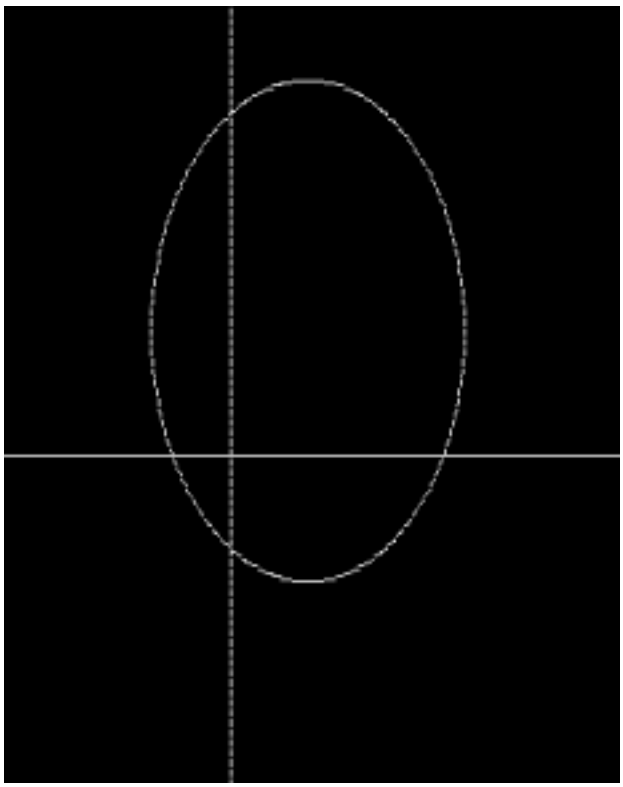
```
#include <GLUT/glut.h>
#include <stdio.h>
#include<math.h>
int xstart, ystart, xend, yend, xc, yc, radius;
void myInit() {
glClear(GL_COLOR_BUFFER_BIT);
glClearColor(0.0, 0.0, 0.0, 1.0);
glMatrixMode(GL_PROJECTION);
gluOrtho2D(-320, 320, -200, 200);
}
void draw_pixel(int x, int y) {
glBegin(GL_POINTS);
glVertex2i(x, y);
glEnd();
}
void draw_line(int xstart, int xend, int ystart, int yend) {
int dx, dy, i, e;
int incx, incy, inc1, inc2;
int x,y;
dx = abs(xend-xstart);
dy = abs(yend-ystart);
incx = 1;
if (xend < xstart) incx = -1;
incy = 1;
if (yend < ystart) incy = -1;
x = xstart;
y = ystart;
if (dx > dy) {
draw_pixel(x, y);
e = 2 * dy-dx;
inc1 = 2*(dy-dx);
inc2 = 2*dy;
for (i=0; i<dx; i++) {
if (e >= 0) {
y += incy;
e += inc1;
}
else
e += inc2;
x += incx;
draw_pixel(x, y);
}
} else {
draw_pixel(x, y);
e = 2*dx-dy;
inc1 = 2*(dx-dy);
inc2 = 2*dx;
```

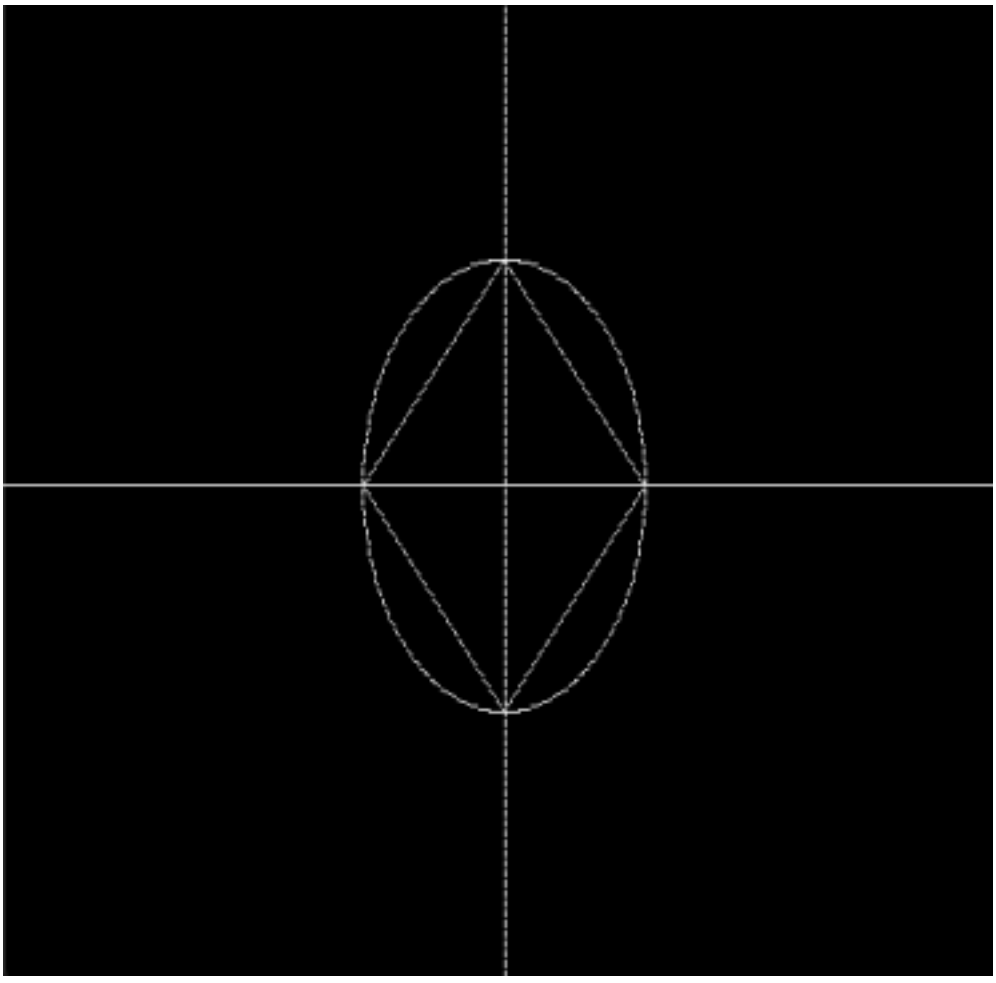
```
for (i=0; i<dy; i++) {
if (e >= 0) {
x += incx;
e += inc1;
}
else
e += inc2;
y += incy;
draw_pixel(x, y);
}
}
}
void comb(int x, int y, int xc, int yc){ // x += xc;
// y += yc;
draw_pixel(x+xc, y+yc);
draw_pixel(y+xc, x+yc);
draw_pixel(-x+xc, y+yc);
draw_pixel(-y+xc, x+yc);
draw_pixel(-x+xc, -y+yc);
draw_pixel(-y+xc, -x+yc);
draw_pixel(x+xc, -y+yc);
draw_pixel(y+xc, -x+yc);
}
void draw_circle(int xc, int yc, int radius){ int p0 = 1 - radius;
int xstart = 0;
int ystart = radius;
comb(xstart, ystart, xc, yc);
while(xstart < ystart){
xstart += 1;
int pnew = p0 + 2*xstart + 1;
if(p0 > 0){
ystart -= 1;
pnew -= 2*ystart;
}
comb(xstart, ystart, xc, yc);
p0 = pnew;
}
}
void custom_shape(){
}
void myDisplay() {
//draw_line(xstart, xend, ystart, yend); draw_line(-1000, 1000,0, 0);
draw_line(0, 0,-1000, 1000);
draw_circle(xc, yc, radius);
draw_line(radius, 0, 0, radius);
draw_line(-radius, 0, 0, radius);
draw_line(radius, 0, 0, -radius);
draw_line(-radius, 0, 0, -radius);
glFlush();
}
int main(int argc, char **argv) {
// printf( "Enter (xstart, ystart, xend, yend)\n"); // scanf("%d %d %d %d", &xstart, &ystart, &xend, &yend);
printf("Enter xc, yc and radius\n");
```

```
scanf("%d %d %d", &xc, &yc, &radius);
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB); glutInitWindowSize(500, 500);
glutInitWindowPosition(0, 0);
glutCreateWindow("Excercise 4");
myInit();
glutDisplayFunc(myDisplay);
glutMainLoop();
                }
```

SAMPLE I/O:







LEARNING OUTCOME:

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