## Source Code

```
#include<stdlib.h>

#include<stdio.h>

#include <GL/gl.h>

#include <GL/glut.h>


float a, b, xc, yc;


void DrawPoint(GLint x,GLint y)

{

glBegin(GL_POINTS);

glVertex2f(x,y);

glEnd();

}


void TakeInput()

{

printf("Value of x co-ordinate of center : ");

scanf("%f", & xc);


printf("Value of y co-ordinate of center : ");

scanf("%f", & yc);
```

```
printf("Enter the length of semi-major axis: ");
scanf("%f", & a);

printf("Enter the length of semi-minor axis: ");
scanf("%f", & b);
}
void display()
{
float x = 0;
float y = b;//(0,b) ---
float p1 = b * b - (a * a)* b + (a * a) * (0.25) ;
float dx = 2 * (b * b) * x;
float dy = 2 * (a * a) * y;
while(dx < dy)
{
DrawPoint(xc + x, yc + y);
DrawPoint(xc - x, yc + y);
DrawPoint(xc + x , yc - y);
DrawPoint(xc - x, yc - y);
if(p1 < 0)
{
```

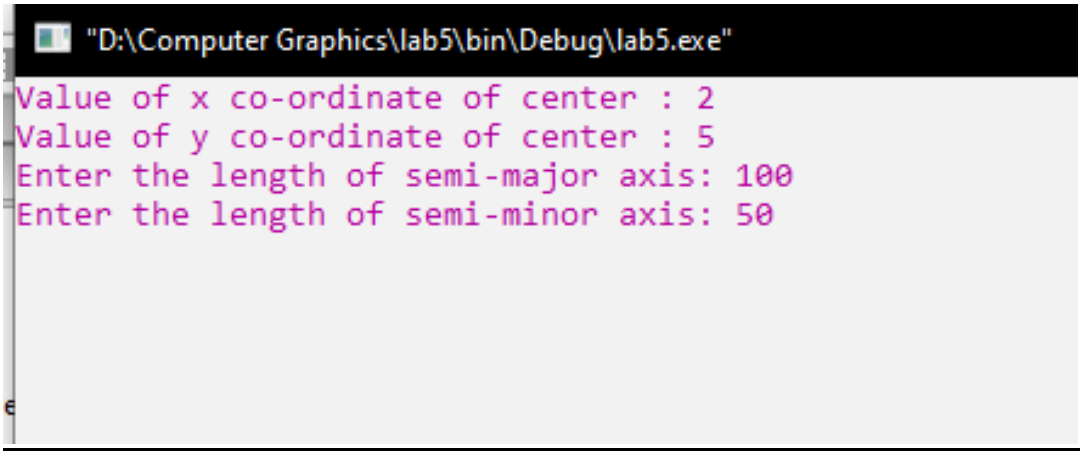
```
x = x + 1;

dx = 2 * (b * b) * x;

p1 = p1 + 2 * (b * b) * x + (b * b);

}

else

{

x = x + 1;

y = y - 1;

dx = 2 * (b * b) * x;

dy = 2 * (a * a) * y;

p1 = p1 + 2 * (b * b) * x - 2 * (a * a) * y +(b * b);

}

}


float p2 = (b * b )* ( x + 0.5) * ( x + 0.5) + ( a * a) * ( y
- 1) * ( y - 1) - (a * a )* (b * b);


while(y > 0)

{

DrawPoint(xc + x , yc+y);

DrawPoint(xc - x, yc + y);

DrawPoint(xc + x , yc - y );

DrawPoint(xc - x , yc - y); //glEnd();
```

```
if(p2 > 0)

{

x = x;

y = y - 1;

dy = 2 * (a * a) * y;

p2 = p2 - 2 * (a * a) * y + (a * a);

}

else

{

x = x + 1;

y = y - 1;

dy = 2 * (a * a) * y;

dx = 2 * (b * b) * x;

p2 = p2 + dx - dy + (a * a);

}

}

glFlush();

}

int main(int argc, char ** argv)

{

TakeInput();
```

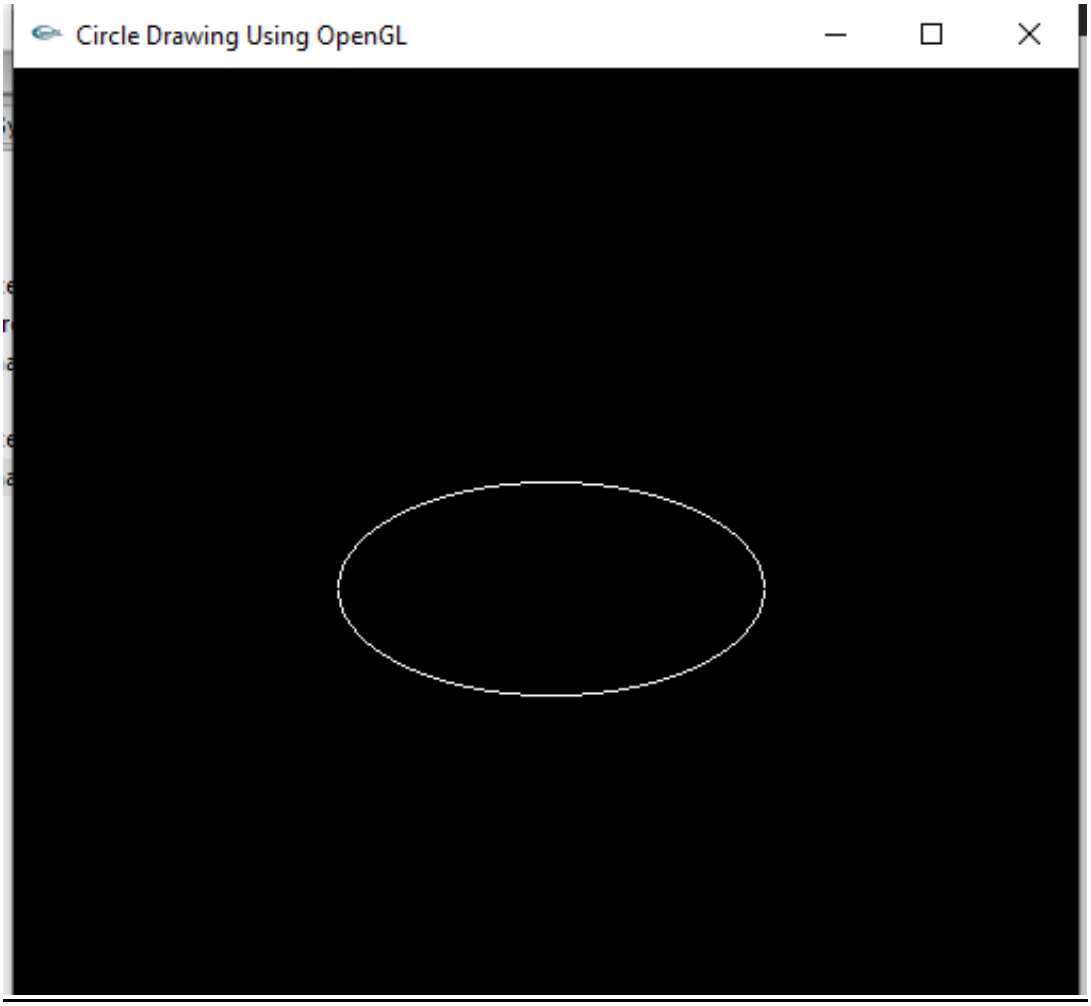
```
glutInit(&argc, argv);

glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);

glutInitWindowSize(500, 500);

glutInitWindowPosition(100, 150);

glutCreateWindow("Circle Drawing Using OpenGL");


glClearColor(0, 0, 0, 0);

glClear(GL_COLOR_BUFFER_BIT);

gluOrtho2D(-250, 250, -250, 250);

glMatrixMode(GL_PROJECTION);

glViewport(0, 0, 500, 500);


glutDisplayFunc(display);

glutMainLoop();

return 0;

}
```

**Results when a > b : major axis is parallel to x axis or x axis itself is major axis**

**Input:**

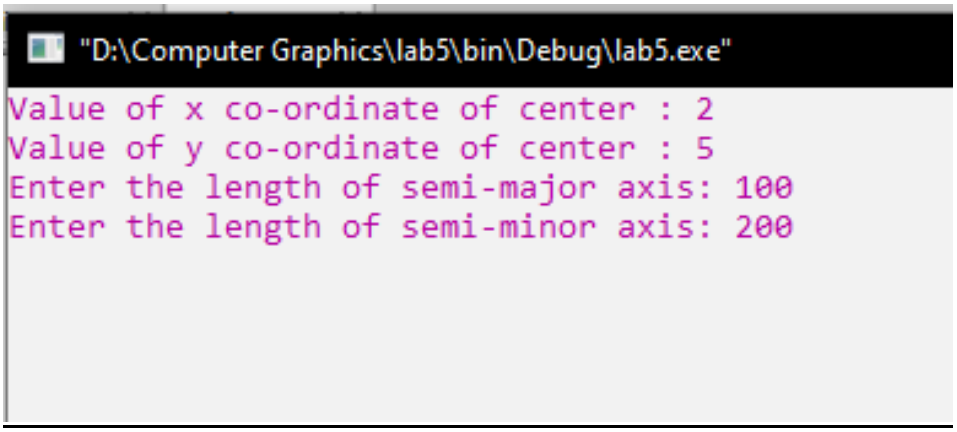



**Output:**



**Results when a < b : major axis is parallel to y axis or y axis itself is major axis**

**Input:**




**Output:**