Name: Shaikh Salif Aminuddin

CLASS: S2 **ROLL NO:** 2201094

EXPERIMENT NO: 4

AIM: Implementing Midpoint Ellipse Algorithm

THEORY: This is an incremental method for scan converting an ellipse that is centered at the origin in standard position i.e., with the major and minor axis parallel to coordinate system axis. It is very similar to the midpoint circle algorithm.

Mid-Point Ellipse Algorithm :

1. Take input radius along x axis and y axis and obtain center of ellipse.

2. Initially, we assume ellipse to be centered at origin and the first point as : (x, y0)= (0, ry).

3. Obtain the initial decision parameter for region 1 as: p10=ry2+1/4rx2-rx 2ry

4. For every xk position in region 1 :
   If p1k<0 then the next point along the is (xk+1 , yk) and
   p1k+1=p1k+2ry2xk+1+ry2
   Else, the next point is (xk+1, yk-1)
   And p1k+1=p1k+2ry2xk+1 – 2rx2yk+1+ry2

5. Obtain the initial value in region 2 using the last point (x0, y0) of region 1 as:
   p20=ry2(x0+1/2)2+rx2 (y0-1)2-rx2ry2

6. At each yk in region 2 starting at k =0 perform the following task. If p2k>0 the next point is (xk, yk-1) and p2k+1=p2k-2rx2yk+1+rx2

7. Else, the next point is (xk+1, yk -1) and p2k+1=p2k+2ry2xk+1 - 2rx2yk+1+rx2.

8. Now obtain the symmetric points in the three quadrants and plot the coordinate value as: x=x+xc, y=y+yc.

9. Repeat the steps for region 1 until 2ry2x>=2rx2y.

CODE:

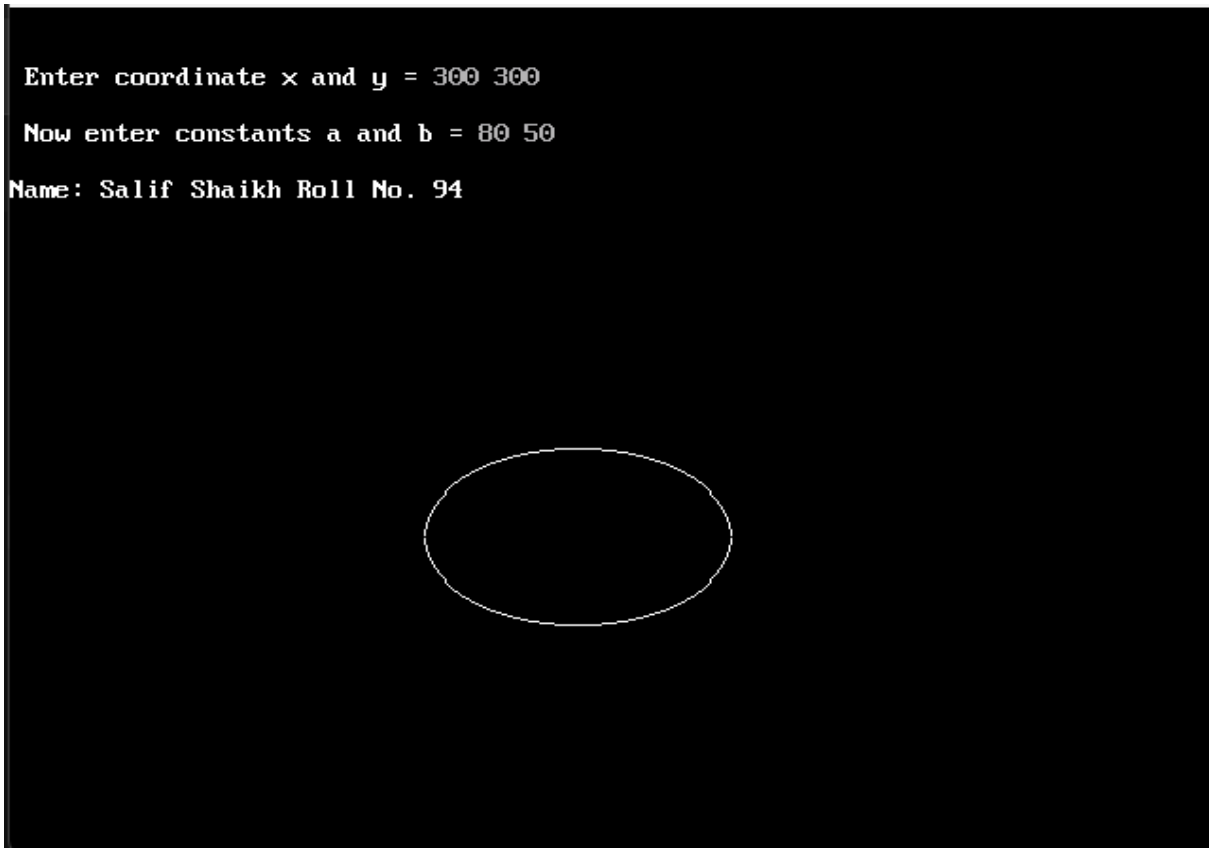
```
#include <stdio.h>
#include <graphics.h>
void main()
{
   long x, y, x_center, y_center;
   long a_sqr, b_sqr, fx, fy, d, a, b, tmp1, tmp2;
   int g_driver = DETECT, g_mode;
   clrscr();
   initgraph(&g_driver, &g_mode, "c:\\tc\\bgi");
   printf("\n\n Enter coordinate x and y = ");
   scanf("%ld%ld", &x_center, &y_center);
   printf("\n Now enter constants a and b = ");
   scanf("%ld%ld", &a, &b);
  printf(“Salif Shaikh Roll No. 94 ”);
   x = 0;
   y = b;
   a_sqr = a * a;
   b_sqr = b * b;
   fx = 2 * b_sqr * x;
   fy = 2 * a_sqr * y;
```

```
d = b_sqr - (a_sqr * b) + (a_sqr * 0.25);
do
{
   putpixel(x_center + x, y_center + y, 7);
   putpixel(x_center - x, y_center - y, 7);
   putpixel(x_center + x, y_center - y, 7);
   putpixel(x_center - x, y_center + y, 7);
   if (d < 0)
   {
      d = d + fx + b_sqr;
   }
   else
   {
      y = y - 1;
      d = d + fx + -fy + b_sqr;
      fy = fy - (2 * a_sqr);
   }
   x = x + 1;
   fx = fx + (2 * b_sqr);
   delay(10);
} while (fx < fy);
tmp1 = (x + 0.5) * (x + 0.5);
tmp2 = (y - 1) * (y - 1);
d = b_sqr * tmp1 + a_sqr * tmp2 - (a_sqr * b_sqr);
do
{
   putpixel(x_center + x, y_center + y, 7);
```

```
        putpixel(x_center - x, y_center - y, 7);
        putpixel(x_center + x, y_center - y, 7);
        putpixel(x_center - x, y_center + y, 7);

        if (d >= 0)
            d = d - fy + a_sqr;
        else
        {
            x = x + 1;
            d = d + fx - fy + a_sqr;
            fx = fx + (2 * b_sqr);
        }
        y = y - 1;
        fy = fy - (2 * a_sqr);
    } while (y > 0);
    getch();
    closegraph();
}
```

OUTPUT:



CONCLUSION:

The Midpoint Ellipse Algorithm is a widely used technique for drawing ellipses on computer screens due to its efficiency and ability to produce reasonably smooth curves. It's a fundamental concept in computer graphics, especially when drawing shapes that resemble ellipses. While it may not be suitable for all ellipse-drawing scenarios (especially rotated ellipses), it serves as an excellent starting point and foundation for more advanced techniques in the field of computer graphics and image rendering.