Ex. No. : 1 Date:

Register No.: 231701055 Name: Suga V

Bresenham’s Line Algorithm, Midpoint Circle Algorithm, and Midpoint Ellipse Algorithm AIM:

To implement:

- Bresenham’s Line Drawing Algorithm
- Midpoint Circle Drawing Algorithm
- Midpoint Ellipse Drawing Algorithm
  And draw geometric shapes on the screen.

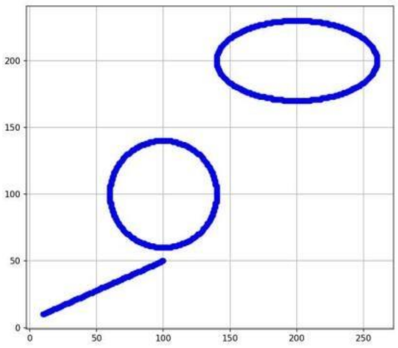
Procedure:

1. Initialize graphics mode using suitable graphics libraries (e.g., OpenGL in C++, turtle or matplotlib in Python).
2. For each algorithm:
   o Accept user input for coordinates or radius. o Implement the plotting logic using the respective algorithm. o Plot the pixels on the screen.

Program:

```
import matplotlib.pyplot as plt

def plot_point(x, y):     plt.plot(x,
y, 'bo')


def bresenham_line(x1, y1, x2, y2):
    dx = abs(x2 - x1)     dy =
abs(y2 - y1)     x, y = x1, y1
sx = 1 if x2 > x1 else -1     sy
= 1 if y2 > y1 else -1     if dx
> dy:
        err = dx / 2.0
while x != x2:
```

```
            plot_point(x, y)
err -= dy
if err < 0:                 y +=
sy                  err += dx
x += sx     else:
        err = dy / 2.0
while y != y2:
plot_point(x, y)             err
-= dx             if err <
0:                  x += sx                  err
+= dy              y += sy
    plot_point(x, y)

def midpoint_circle(xc, yc, r):
      x    = 0     y = r     p
      = 1 - r     while x <=
      y:
        for a, b in [(x, y), (y, x), (-x, y), (-y,
x),                       (-x, -y), (-y, -x), (x, -y), (y,
-x)]:               plot_point(xc + a, yc + b)
x += 1
if p < 0:               p
+= 2*x + 1
else:             y -= 1
            p += 2*(x - y) + 1

def midpoint_ellipse(rx, ry, xc, yc):
x, y = 0, ry     rx2, ry2 = rx**2, ry**2
p1 = ry2 - (rx2 * ry) + (0.25 * rx2)     dx
= 2 * ry2 * x     dy = 2 * rx2 * y

    while dx < dy:         for a, b in [(x, y), (-x, y),
(x, -y), (-x, -y)]:
            plot_point(xc + a, yc + b)  x
                        += 1
            dx = 2 * ry2 * x
            if p1 < 0:
            p1 += dx + ry2
else:
      y    -= 1             dy =
      2 * rx2 * y
          p1 += dx - dy + ry2

   p2 = (ry2 * (x + 0.5)**2) + (rx2 * (y - 1)**2) - (rx2 * ry2)
while y >= 0:        for a, b in [(x, y), (-x, y), (x, -y), (-x, -y)]:
          plot_point(xc + a, yc + b)
y          -= 1        dy
= 2 * rx2 * y        if
p2 > 0:             p2 -=
dy + rx2
else:
          x += 1            dx
= 2 * ry2 * x             p2
+= dx - dy + rx2

plt.figure(figsize=(8, 8)) bresenham_line(10, 10,
100, 50) midpoint_circle(100, 100, 40)
midpoint_ellipse(60, 30, 200, 200)
plt.gca().set_aspect('equal', adjustable='box')
plt.grid(True) plt.show()
```



Result:

Thus, the line, circle, and ellipse were successfully drawn using Bresenham's and Midpoint algorithms.