



**Pseudo Code:**

def midpoint\_ellipse(rx, ry, xc, yc): # Step 1: Initialize variables

x = 0 y = ry

# Step 2: Region 1 Initial Decision Parameter p1 = ry2 - (rx2 \* ry) + (0.25 \* rx2)

# Step 3: Iterate through Region 1 While 2\*ry2\*x= 2\*rx2\*y:

# Plot points for all four quadrants plot(x + xc, y + yc)

plot(-x + xc, y + yc) plot(x + xc, -y + yc) plot(-x + xc, -y + yc)

# Update x and decision parameter if p1 < 0:

x += 1

p1 += 2\*ry2 \* x + ry2 else:

x += 1

y -= 1

p1 += 2\*ry2 \* x - 2\*rx2 \* y + ry2

# Step 4: Region 2 Initial Decision Parameter

p2 = (ry2 \* (x + 0.5) \*\* 2) + (rx2 \* (y - 1) \*\* 2) - (rx2 \* ry2)

# Step 5: Iterate through Region 2 while y >= 0:

# Plot points for all four quadrants plot(x + xc, y + yc)

plot(-x + xc, y + yc) plot(x + xc, -y + yc) plot(-x + xc, -y + yc)

# Update y and decision parameter if p2 > 0:

y -= 1

p2 -= 2\*rx2 \* y + rx2 else:

x += 1

y -= 1

p2 += 2\*ry2 \* x - 2\*rx2 \* y + rx2

def plot(x, y):

//implement like midpoint circle algorithm