BANGLADESH UNIVERSITY OF BUSINESS AND TECHNOLOGY

Committed to Academic Excellence

Lab Report

Title	Marks
Course Code: CSE-342 **Course Title**: Computer Graphics **Lab no**: 03	

Submitted By	Submitted To
Name: Saba-E-Zannat **ID**: 21225103304 **Intake**: 49 **Section**: 05 Department of CSE Bangladesh University of Business & Technology	**Name**: Md. Khairul Islam **Designation**: Lecturer Department of CSE Bangladesh University of Business & Technology

Submission Date *Teacher's Signature*

1. **Title**: Midpoint Circle Drawing Algorithm

2. **Objective**:
To implement and understand the Midpoint Circle Drawing Algorithm for plotting a circle in a computer graphics environment, focusing on efficiency and accuracy while reducing computational cost.

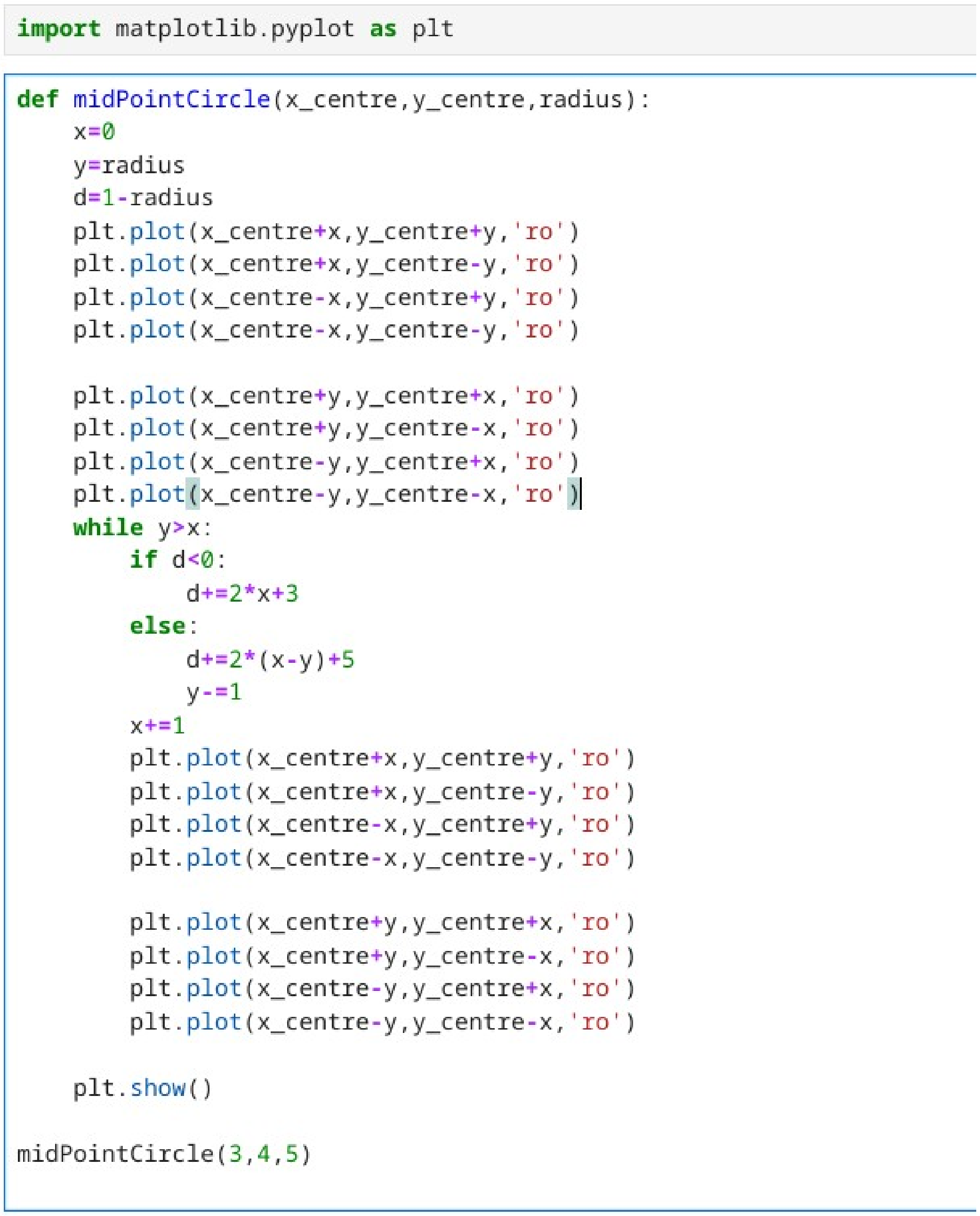
3. **Environment**:
- Operating System: Windows
- Programming Language: Python
- Editor: Jupyter Notebook
- Graphics Library: matplotlib
- Hardware: Standard computer system with basic graphics capabilities

4. **Introduction**:
The Midpoint Circle Drawing Algorithm is an efficient way to draw a circle by determining which pixel to turn on at each step using integer arithmetic. It was proposed by **Jack E. Bresenham** and is a variation of Bresenham's line algorithm. This algorithm leverages **symmetry** to plot only one-eighth of a circle and mirrors the points to complete the full circle. It avoids floating-point calculations, making it faster and suitable for raster displays.

5. **Algorithm**:
Initialization:
Define the circle's radius r and the center (xc, yc).
Set the initial point as (x, y) = (0, r).
Calculate the initial decision parameter:
$p_0=1-r$

```
• Iterate over the first octant:
  While $x \le y$:
• Plot the current point $(xc+x, yc+y)$ and its symmetrical points in other octants.
• Update the decision parameter:
    • If $p<0$: $p=p+2x+3$
    • Else: $p=p+2x-2y+5$ and $y=y-1$
• Increment $x=x+1$.
```

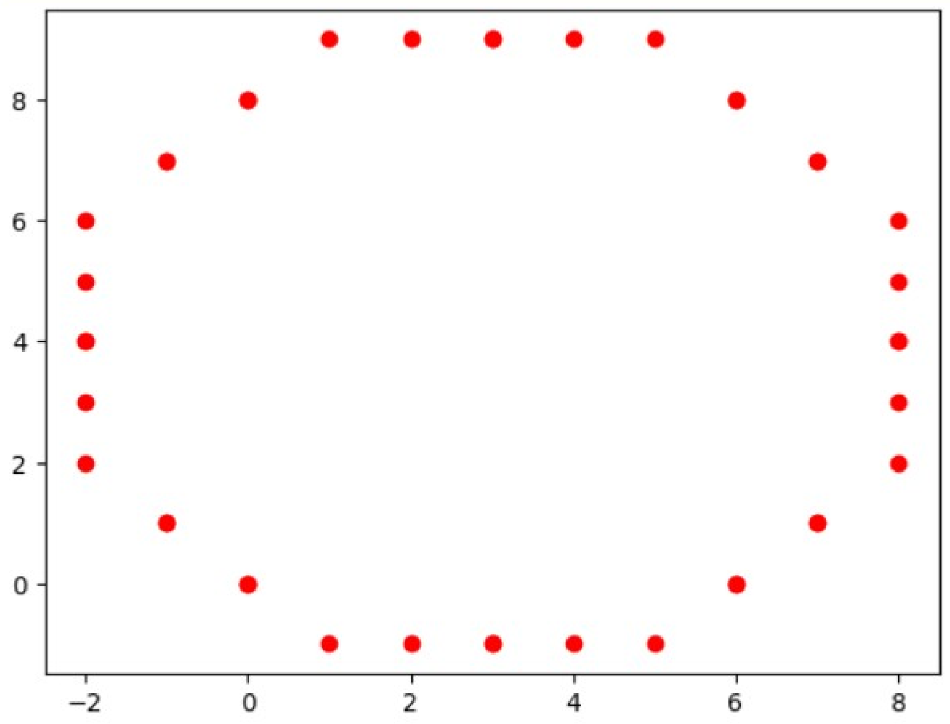
6. **Code**:



```
import matplotlib.pyplot as plt
```

```
def midPointCircle(x_centre,y_centre,radius):
    x=0
    y=radius
    d=1-radius
    plt.plot(x_centre+x,y_centre+y,'ro')
    plt.plot(x_centre+x,y_centre-y,'ro')
    plt.plot(x_centre-x,y_centre+y,'ro')
    plt.plot(x_centre-x,y_centre-y,'ro')

    plt.plot(x_centre+y,y_centre+x,'ro')
    plt.plot(x_centre+y,y_centre-x,'ro')
    plt.plot(x_centre-y,y_centre+x,'ro')
    plt.plot(x_centre-y,y_centre-x,'ro')
    while y>x:
        if d<0:
            d+=2*x+3
        else:
            d+=2*(x-y)+5
            y-=1
        x+=1
        plt.plot(x_centre+x,y_centre+y,'ro')
        plt.plot(x_centre+x,y_centre-y,'ro')
        plt.plot(x_centre-x,y_centre+y,'ro')
        plt.plot(x_centre-x,y_centre-y,'ro')

        plt.plot(x_centre+y,y_centre+x,'ro')
        plt.plot(x_centre+y,y_centre-x,'ro')
        plt.plot(x_centre-y,y_centre+x,'ro')
        plt.plot(x_centre-y,y_centre-x,'ro')

    plt.show()

midPointCircle(3,4,5)
```

7.**Snapshot(Input & Output)**:



8. **Discussion & conclusion**:

The Midpoint Circle Drawing Algorithm is a fundamental method in computer graphics, providing an efficient way to draw circles by using simple integer operations and symmetry. Understanding this algorithm helps in learning more advanced raster graphics techniques. Its practical significance lies in its speed and accuracy, making it a standard choice in low-level graphics programming.