HIMALAYA COLLEGE OF ENGINEERING

Computer Graphics Lab Report

Lab 1: Rectangle Plotting Using Python and Matplotlib

Prepared By: Mukesh Pandeya

Subject: Computer Graphics

Program: Bachelors of Electronics and Computer Engineering

Institution: Himalaya College of Engineering

Date: Dec 12, 2025

Theory:

Line Drawing Using the DDA Algorithm

In computer graphics, drawing a straight line between two points is one of the most fundamental operations. The Digital Differential Analyzer (DDA) algorithm is a simple and efficient method used for rasterizing a line by calculating intermediate points between two given coordinates.

Given two endpoints:

- (x1, y1)
- (x2, y2)

The DDA algorithm works by:

1. Calculating the differences
    - dx = x2 – x1
    - dy = y2 – y1
2. Determining the number of steps based on the larger of |dx| or |dy|.
3. Computing the increments for each step:
    - xinc = dx / steps
    - yinc = dy / steps
4. Starting from the initial point and repeatedly adding the increments to generate all intermediate points along the line.

These points are rounded and stored in lists, which are then plotted using Matplotlib. The algorithm ensures smooth plotting of the line regardless of its slope.

This program demonstrates the use of:

- Basic arithmetic computations
- Incremental plotting techniques
- Storing and visualizing line coordinates using Python lists and Matplotlib

Rectangle Drawing Using Coordinate Geometry

A rectangle can be drawn on a 2D plane using the coordinates of two opposite corners. Once the diagonal points are known, the other two corners can be determined by combining the respective x- and y-values.

Given:

- First corner: (x1, y1)
- Opposite corner: (x2, y2)

The remaining corners become:

- (x2, y1)
- (x1, y2)

To plot the rectangle, the coordinates are arranged in a sequence that ensures the edges connect properly. Typically, the first point is repeated at the end to close the shape. These x- and y-coordinate lists are plotted using Matplotlib with markers and dashed lines.

The rectangle program demonstrates:

- Use of coordinate geometry to construct shapes
- Managing points using Python lists
- Visualizing connected shapes with plotting functions such as plot(), grid(), and axis('equal')

Both programs make use of Python's graphical plotting capabilities to illustrate fundamental graphic algorithms and geometric constructions.

Lab Questions:

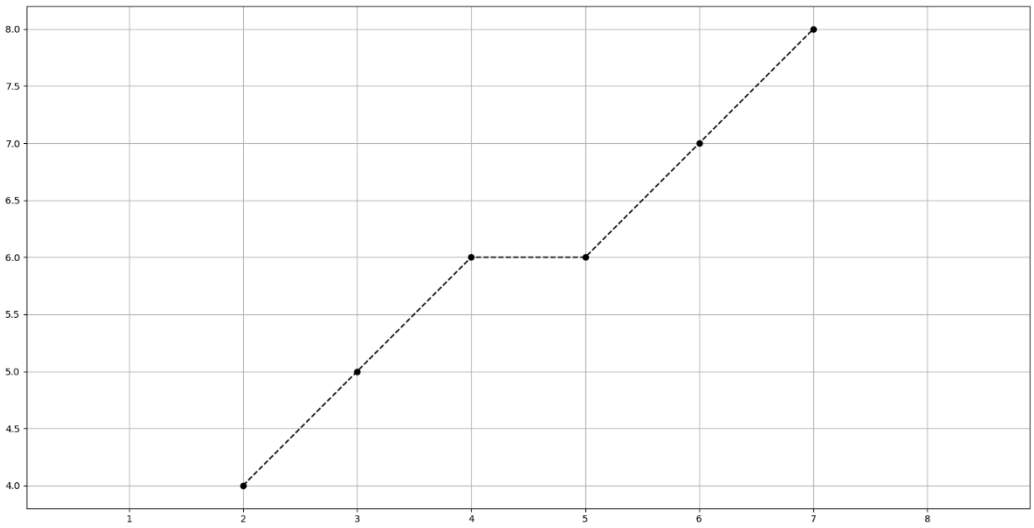
Q no 1: Write a Python program to draw a straight line between two points using the DDA algorithm.

Code:

```
import matplotlib.pyplot as plt
x1=int(input("Enter x1: "))
y1=int(input("Enter y1: "))
x2=int(input("Enter x2: "))
y2=int(input("Enter y2: "))
dx=x2-x1
dy=y2-y1
if(abs(dx)>abs(dy)):
    steps=abs(dx)
else:
    steps=abs(dy)
xinc=dx/steps
yinc=dy/steps
x=x1
y=y1
xcord=[]
ycord=[]
for i in range (0,steps):
    xcord.append(round(x))
    ycord.append(round(y))
    x=x+xinc
    y=y+yinc
xcord.append(round(x2))
ycord.append(round(y2))
plt.plot(xcord,ycord,marker="o",linestyle="--",color="black")
plt.grid(True)
plt.axis('equal')
plt.show()
```

Output:

```
Enter x1: 2
Enter y1: 4
Enter x2: 7
Enter y2: 8
```



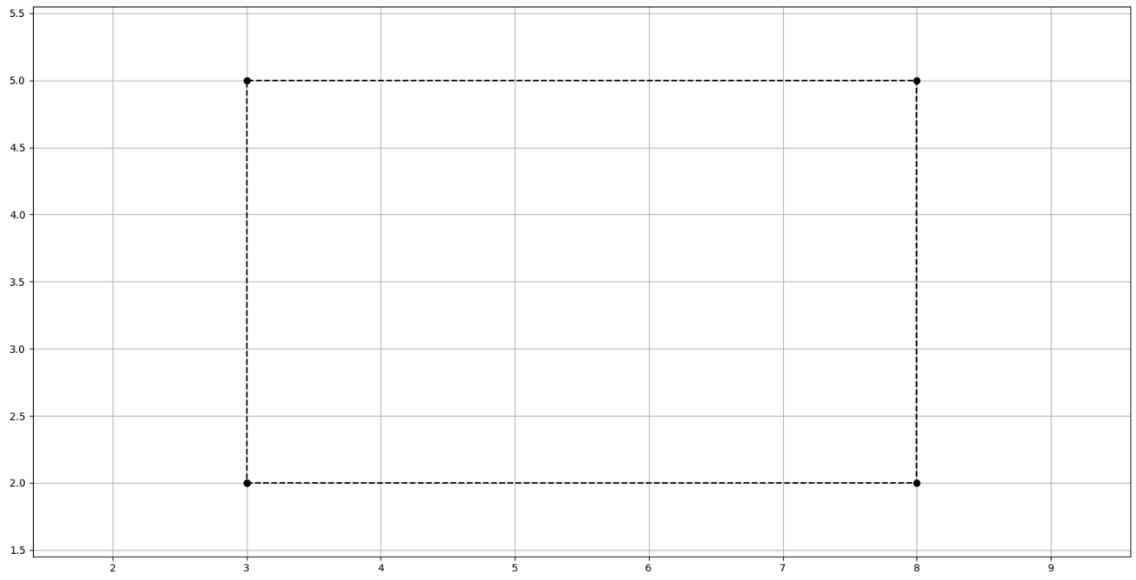
Q no 2: Write a Python program to draw a rectangle on a 2D plane using the coordinates of two diagonally opposite corners.

Code:

```
1. import matplotlib.pyplot as plt
2. x1=int(input("Enter x1: "))
3. y1=int(input("Enter y1: "))
4. x2=int(input("Enter x2: "))
5. y2=int(input("Enter y2: "))
6. xcords=[x1,x2,x2,x1,x1]
7. ycords=[y1,y1,y2,y2,y1]
8. plt.plot(xcords,ycords,marker="o",linestyle="--",color="black")
9. plt.grid(True)
10. plt.axis('equal')
11. plt.show()
```

Output:

```
Enter x1: 3
Enter y1: 2
Enter x2: 8
Enter y2: 5
```



Conclusion:

In this lab, we explored essential concepts of computer graphics using Python by implementing programs for line drawing and rectangle construction. Through the DDA line drawing algorithm and coordinate-based rectangle plotting, we gained practical experience in handling geometric computations and visualizing shapes using Matplotlib. This hands-on work strengthened our understanding of graphical plotting, coordinate systems, and incremental algorithms. The concepts learned here provide a strong foundation for further studies in computer graphics, Python visualization, and more advanced rendering techniques.