PRACTICAL – 1

PROGRAM -1

AIM- WAPP for DDA Algorithem

CODE-

from matplotlib import pyplot as plt

plt.rcParams["figure.figsize"] = [7.00, 3.50]

plt.rcParams["figure.autolayout"] = True

try:

    xo, yo = map(int, input("Enter the starting points: ").split(","))

    xn, yn = map(int, input("Enter the ending points: ").split(","))

except ValueError:

    print("Invalid input. Please enter the points in the format (x, y).")

    exit()

dx = xn - xo

dy = yn - yo

m=dy/dx

steps = max(abs(dx),abs(dy))

x\_Codes=[xo]

y\_Codes=[yo]

if m>1:

    for i in range(steps):

        x\_Codes.append((1/m)+x\_Codes[i])

        y\_Codes.append(1+y\_Codes[i])

if m<1:

    for i in range(steps):

        x\_Codes.append(1+x\_Codes[i])

        y\_Codes.append(m+y\_Codes[i])

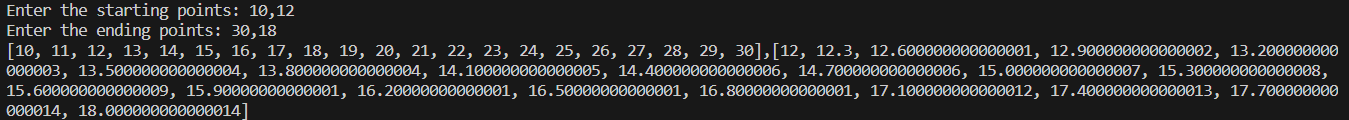
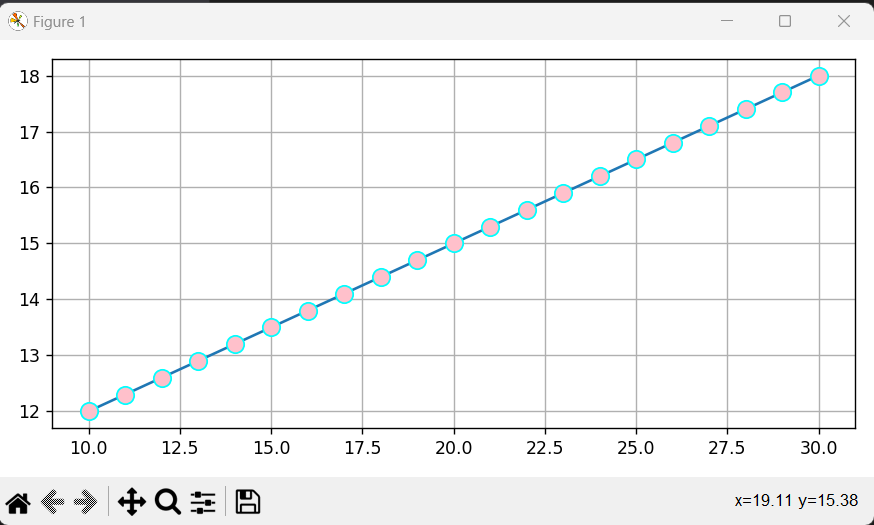
plt.grid()

print(f"{x\_Codes},{y\_Codes}")

plt.plot(x\_Codes,y\_Codes,marker="o", markersize=10, markeredgecolor="cyan", markerfacecolor="pink")

plt.show()

OUTPUT-



PROGRAM -2

AIM- WAPP for Bresenham Algorithm

CODE-

import matplotlib.pyplot as plt

def draw\_line\_Bresenham(x1, y1, x2, y2):

dx = abs(x2 - x1)

dy = abs(y2 - y1)

xincr = 1 if x2 > x1 else -1

yincr = 1 if y2 > y1 else -1

if dx > dy:

p = 2 \* dy - dx

x = x1

y = y1

end = dx

else:

p = 2 \* dx - dy

x = x1

y = y1

end = dy

xcoordinates = []

ycoordinates = []

for i in range(end):

xcoordinates.append(x)

ycoordinates.append(y)

if dx > dy:

x += xincr

if p < 0:

p += 2 \* dy

else:

y += yincr

p += 2 \* (dy - dx)

else:

y += yincr

if p < 0:

p += 2 \* dx

else:

x += xincr

p += 2 \* (dx - dy)

plt.plot(xcoordinates, ycoordinates)

plt.xlabel("X-Axis")

plt.ylabel("Y-Axis")

plt.title("Bresenham Algorithm")

plt.show()

# Example usage

x1 = int(input("Enter x1: "))

y1 = int(input("Enter y1: "))

x2 = int(input("Enter x2: "))

y2 = int(input("Enter y2: "))

draw\_line\_Bresenham(x1, y1, x2, y2)

OUTPUT-

