Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

Computer Graphics Lab Report 02

on

‘**Line Drawing Algorithms - Lab 02 Task**’

Submitted By:

Reewaj Khanal (61)

Submitted to:

Mr. Dhiraj Shrestha

Assistant Professor

Department of Computer Science and Engineering

School of Engineering

Kathmandu University

Dhulikhel, Kavre

Submission Date: Thursday 16 May 2024

Question No. 1 Implement Digital Differential Analyzer Line drawing algorithm.

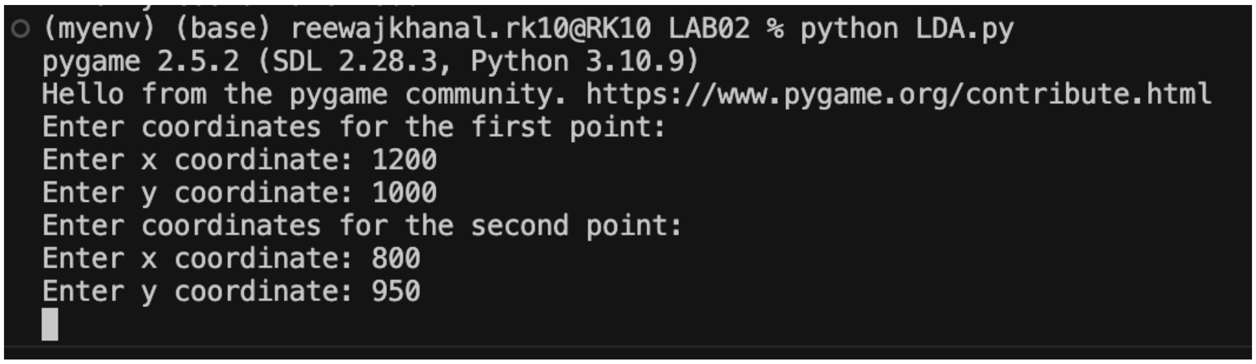
Answer:

```
import pygame

from pygame.locals import *

from OpenGL.GL import *


def DDA_Line(x1, y1, x2, y2):

   dx = x2 - x1

   dy = y2 - y1


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


   x_increment = dx / steps

   y_increment = dy / steps


   x = x1

   y = y1


   glBegin(GL_POINTS)

   glVertex2f(x, y)

   glEnd()


   for _ in range(steps):

       x += x_increment

       y += y_increment

       glBegin(GL_POINTS)

       glVertex2f(round(x), round(y))

       glEnd()
```

```
def get_point():

   x = int(input("Enter x coordinate: "))

   y = int(input("Enter y coordinate: "))

   return x, y


def main():

   # Prompt user for coordinates of two points

   print("Enter coordinates for the first point:")

   point1 = get_point()

   print("Enter coordinates for the second point:")

   point2 = get_point()


   # Determine screen dimensions based on input points

   max_x = max(point1[0], point2[0])

   max_y = max(point1[1], point2[1])

   screen_width = max_x + 100  # Add padding

   screen_height = max_y + 100  # Add padding


   pygame.init()

   display = (screen_width, screen_height)

   pygame.display.set_mode(display, DOUBLEBUF | OPENGL)


   glOrtho(0, screen_width, 0, screen_height, -1, 1)


   while True:

       for event in pygame.event.get():

           if event.type == pygame.QUIT:

               pygame.quit()

               quit()
```

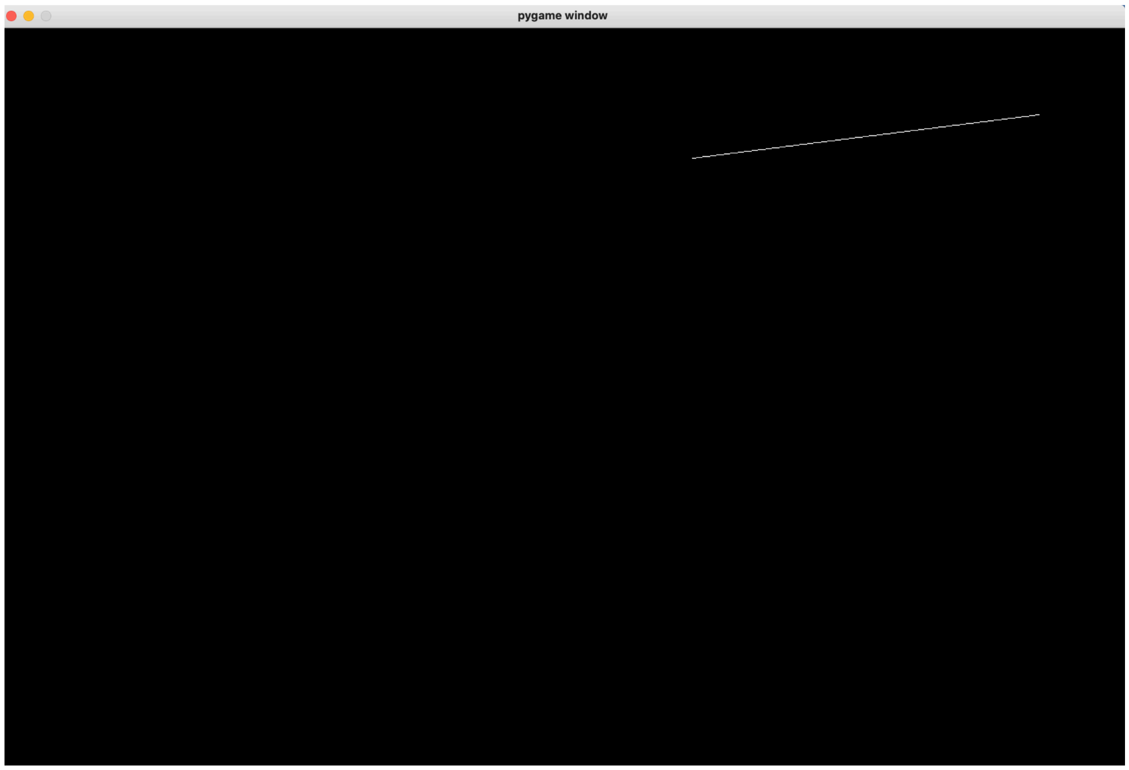
```python
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

        glColor3f(1, 1, 1)  # Set line color to white

        # Drawing the line using DDA algorithm
        DDA_Line(*point1, *point2)

        pygame.display.flip()

if __name__ == "__main__":
    main()
```

Input:



```
(myenv) (base) reewajkhanal.rk10@RK10 LAB02 % python LDA.py
pygame 2.5.2 (SDL 2.28.3, Python 3.10.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
Enter coordinates for the first point:
Enter x coordinate: 1200
Enter y coordinate: 1000
Enter coordinates for the second point:
Enter x coordinate: 800
Enter y coordinate: 950
```

Output Generated:



Question No. 2 Implement Bresenham Line Drawing algorithm for both slopes(|m|<1 and |m|>=1).

Answer:

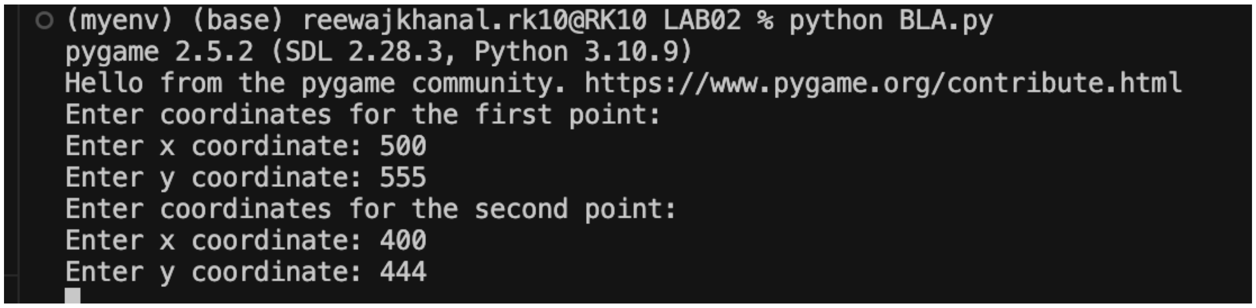
```
import pygame

from pygame.locals import *

from OpenGL.GL import *


def Bresenham_Line(x1, y1, x2, y2):

   dx = abs(x2 - x1)

   dy = abs(y2 - y1)
```

```
slope_error = dx - dy


x, y = x1, y1


x_increment = 1 if x2 > x1 else -1

y_increment = 1 if y2 > y1 else -1


glBegin(GL_POINTS)

glVertex2f(x, y)


if dx > dy:  # |m| < 1

    slope_double_error = slope_error * 2

    while x != x2:

        x += x_increment

        if slope_error >= 0:

            y += y_increment

            slope_error -= slope_double_error

        slope_error += dx * 2

        glVertex2f(x, y)

else:  # |m| >= 1

    slope_double_error = slope_error * 2

    while y != y2:

        y += y_increment

        if slope_error >= 0:

            x += x_increment

            slope_error -= slope_double_error

        slope_error += dy * 2

        glVertex2f(x, y)


glEnd()
```

```
def get_point():
   x = int(input("Enter x coordinate: "))
   y = int(input("Enter y coordinate: "))
   return x, y

def main():
   # Prompt user for coordinates of two points
   print("Enter coordinates for the first point:")
   point1 = get_point()
   print("Enter coordinates for the second point:")
   point2 = get_point()

   # Determine screen dimensions based on input points
   max_x = max(point1[0], point2[0])
   max_y = max(point1[1], point2[1])
   screen_width = max_x + 100  # Add padding
   screen_height = max_y + 100  # Add padding

   pygame.init()
   display = (screen_width, screen_height)
   pygame.display.set_mode(display, DOUBLEBUF | OPENGL)

   glOrtho(0, screen_width, 0, screen_height, -1, 1)

   while True:
       for event in pygame.event.get():
           if event.type == pygame.QUIT:
               pygame.quit()
               quit()
```

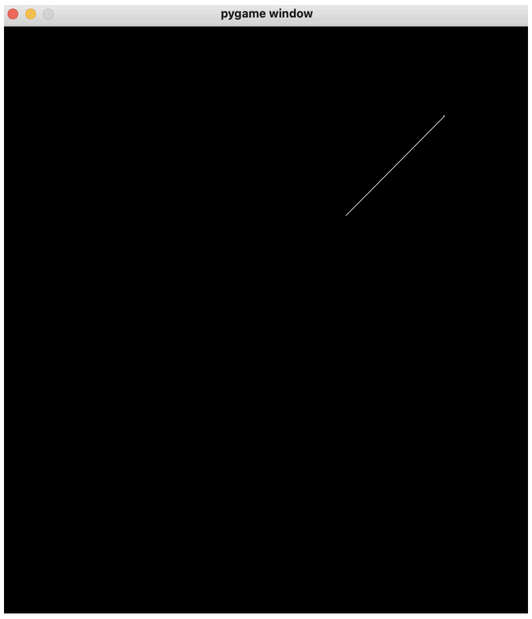
```
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)

        glColor3f(1, 1, 1)  # Set line color to white

        # Drawing the line using Bresenham algorithm
        Bresenham_Line(*point1, *point2)

        pygame.display.flip()

if __name__ == "__main__":
    main()
```

Input:



```
(myenv) (base) reewajkhanal.rk10@RK10 LAB02 % python BLA.py
pygame 2.5.2 (SDL 2.28.3, Python 3.10.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
Enter coordinates for the first point:
Enter x coordinate: 500
Enter y coordinate: 555
Enter coordinates for the second point:
Enter x coordinate: 400
Enter y coordinate: 444
```

Output:



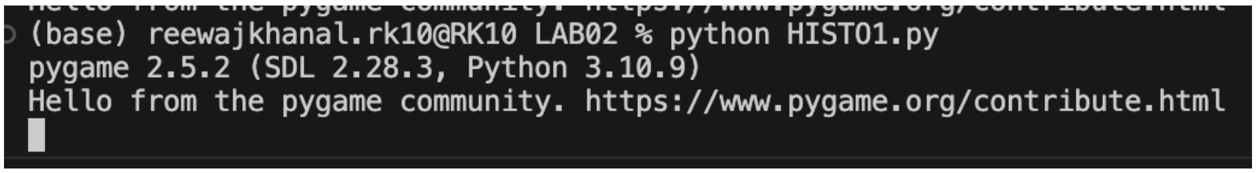
Question No. 3 Implement the given line drawing algorithm to draw a line histogram for any given frequency inputs.

Answer [From BLA Approach]:

```python
import pygame
import sys

# Initialize Pygame
pygame.init()

# Constants
WINDOW_SIZE = (800, 600)
BG_COLOR = (255, 255, 255)
BAR_WIDTH = 50
BAR_GAP = 0  # Reduced gap between bars to zero

# Data for histogram
frequencies = [30, 50, 20, 60, 40, 70, 10, 35, 45, 55]

# Colors for histogram lines
LINE_COLORS = [(140, 19, 185), (52, 60, 147), (0, 128, 0), (255, 69, 0), (255, 215, 0),
               (255, 20, 147), (0, 191, 255), (255, 105, 180), (128, 128, 128), (0, 0, 0)]

# Initialize the screen
screen = pygame.display.set_mode(WINDOW_SIZE)
pygame.display.set_caption("Histogram using BLA")

def draw_line(x1, y1, x2, y2, color):
   dx = x2 - x1

   dy = y2 - y1

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

   if steps == 0:

       return

   x_inc = dx / steps

   y_inc = dy / steps

   x = x1

   y = y1

   for _ in range(int(steps)):

       pygame.draw.rect(screen, color, (int(x), int(y), BAR_WIDTH, 1))

       x += x_inc

       y += y_inc


def draw_histogram(frequencies):

   x = 14

   max_freq = max(frequencies)

   for freq, color in zip(frequencies, LINE_COLORS):

       scaled_freq = freq * (WINDOW_SIZE[1] - 100) / max_freq

         draw_line(x, WINDOW_SIZE[1], x, WINDOW_SIZE[1] - scaled_freq, color)  # Draw from bottom to top

       x += BAR_WIDTH - 10  # Adjusted x-coordinate for the next bar with overlap


def main():

   running = True

   while running:

       for event in pygame.event.get():

           if event.type == pygame.QUIT:

               running = False


       screen.fill(BG_COLOR)
```

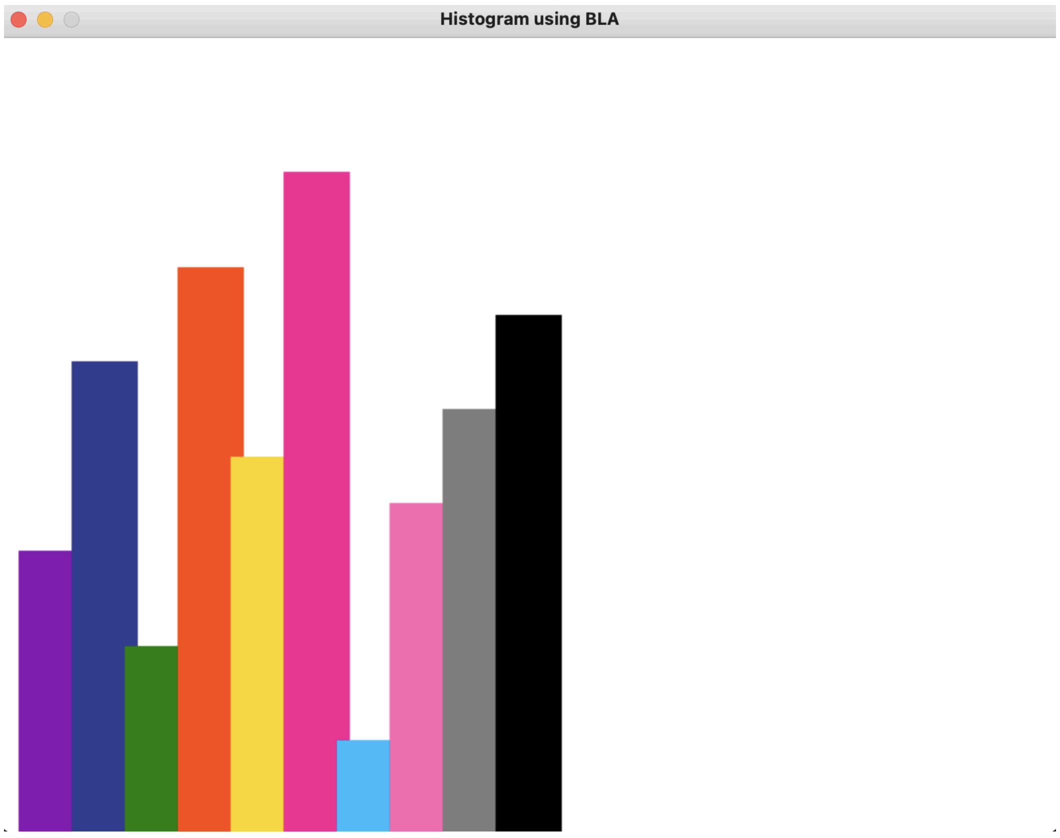
```python
        draw_histogram(frequencies)

        pygame.display.flip()


    pygame.quit()

    sys.exit()


if __name__ == "__main__":

    main()
```

Input:



```
(base) reewajkhanal.rk10@RK10 LAB02 % python HISTO1.py
pygame 2.5.2 (SDL 2.28.3, Python 3.10.9)
Hello from the pygame community. https://www.pygame.org/contribute.html
```

Output:

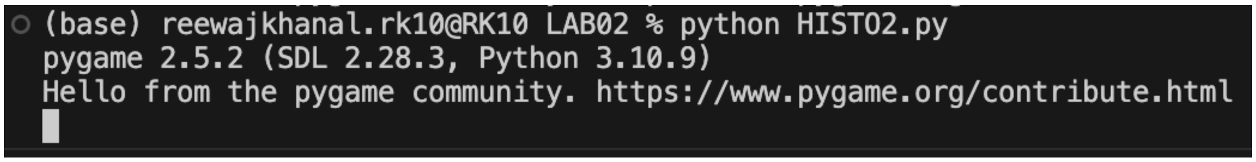


Answer [From DDA Approach]:

```python
import pygame

import sys


# Initialize Pygame

pygame.init()


# Constants

WINDOW_SIZE = (800, 600)

BG_COLOR = (255, 255, 255)

BAR_WIDTH = 60

BAR_GAP = 0  # Reduced gap between bars to zero

BAR_THICKNESS = 20


# Data for histogram

frequencies = [30, 50, 20, 60, 40, 70, 10, 35, 45, 55]


# Colors for histogram lines

LINE_COLORS = [(140, 19, 185), (52, 60, 147), (0, 128, 0), (255, 69, 0), (255, 215,
0), (255, 20, 147), (0, 191, 255), (255, 105, 180), (128, 128, 128), (0, 0, 0)]


# Initialize the screen

screen = pygame.display.set_mode(WINDOW_SIZE)

pygame.display.set_caption("Histogram using DDA")


def draw_line(x1, y1, x2, y2, color):

   dx = x2 - x1

   dy = y2 - y1

   steps = max(abs(dx), abs(dy))
   if steps == 0:
       return
   x_inc = dx / steps
   y_inc = dy / steps
   x = x1
   y = y1
   for _ in range(int(steps)):
       pygame.draw.rect(screen, color, (int(x), int(y), BAR_THICKNESS, 1))
       x += x_inc
       y += y_inc

def draw_histogram(frequencies):
   x = 14
   max_freq = max(frequencies)
   for freq, color in zip(frequencies, LINE_COLORS):
        scaled_freq = freq * (WINDOW_SIZE[1] - 100) / max_freq  # Scale the frequency to fit the window height
       draw_line(x, WINDOW_SIZE[1] - 50, x, WINDOW_SIZE[1] - 50 - scaled_freq, color)
       x += BAR_THICKNESS  # Adjusted x-coordinate for the next bar

def main():
   running = True
   while running:
       for event in pygame.event.get():
           if event.type == pygame.QUIT:
               running = False

       screen.fill(BG_COLOR)
       draw_histogram(frequencies)
       pygame.display.flip()
```

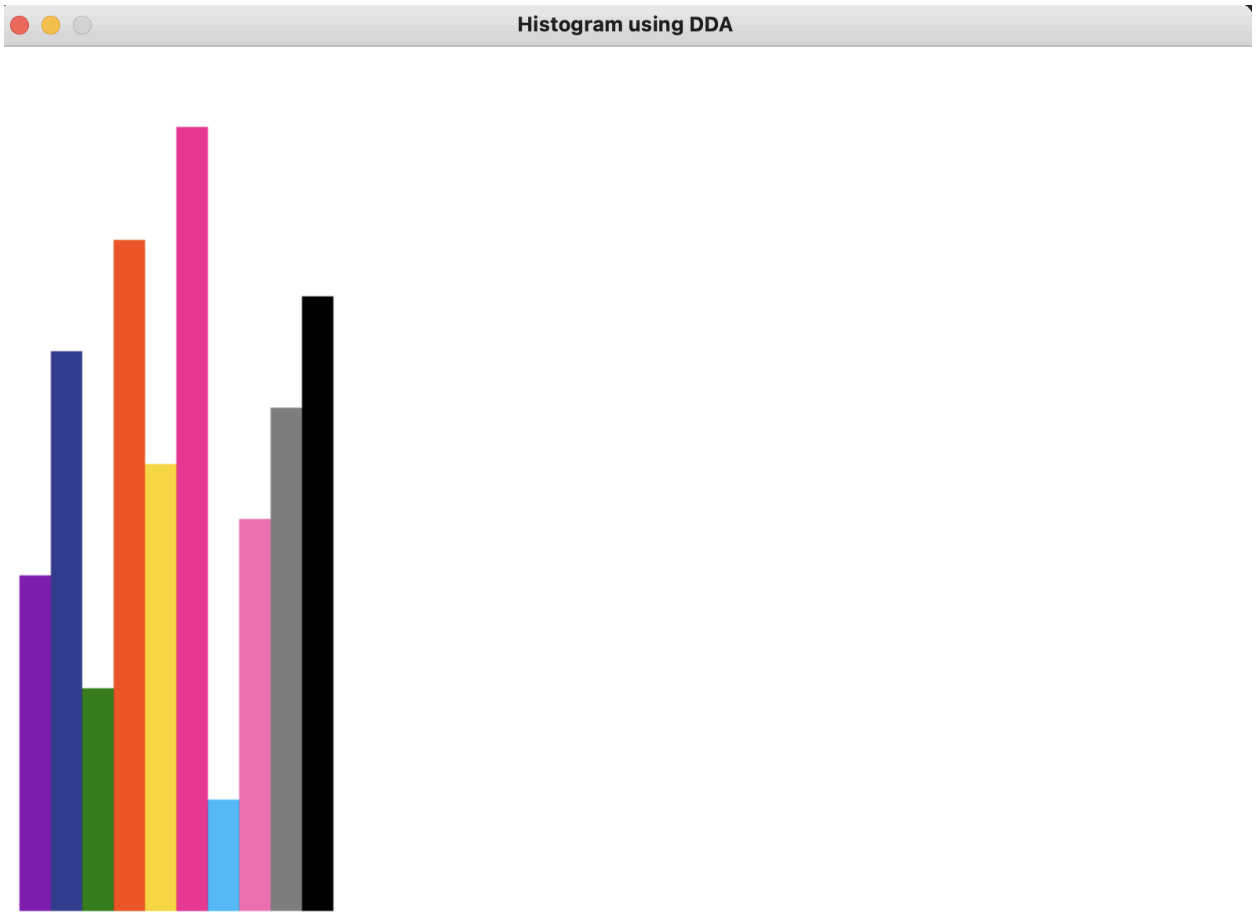
```
    pygame.quit()

    sys.exit()


if __name__ == "__main__":

    main()
```

Input:



```
○ (base) reewajkhanal.rk10@RK10 LAB02 % python HISTO2.py
  pygame 2.5.2 (SDL 2.28.3, Python 3.10.9)
  Hello from the pygame community. https://www.pygame.org/contribute.html
```

Output:


Histogram using DDA