**LAB-2**

**Bresenham's line drawing algorithm**

**Objective: -**

* To determine the points to select in a raster to form a close approximation of a straight line between two points
* Minimizes the error in pixel approximation to produce visually appealing lines.

**Theory: -**

Bresenham's line algorithm uses integer operations to efficiently draw lines on a raster grid. It selects the next pixel by comparing a decision parameter to zero, ensuring accuracy. This incremental method reduces computations, enabling hardware implementation.

**Algorithm: -**

**Input**

* Start point and end point .

**Steps**

1. **Calculate Differences:**
2. **Determine the Dominant Axis:**
   * If Δx≥Δy, the line has a shallow slope (∣m∣ ≤ 1|).
   * If Δx<Δy, the line has a steep slope (∣m∣ > 1).
3. **Set Step Directions:**
   * , otherwise
   * , otherwise
4. **Initialize Decision Parameter:**
   * For **shallow slopes** (∣m∣≤1|)
   * For **steep slopes** (∣m∣>1|)  
     Swap roles of x and y, and set:
5. **Plot the Initial Point:**
   * Plot .
6. **Iterate Until the Endpoint is Reached:**

* For k = 0 to the dominant axis length :
  + **For shallow slopes** (∣m∣≤1|)
    - Increment .
    - If
      * Increment .
      * Update
    - Else:
      * Update .
  + **For steep slopes (∣m∣>1|m| > 1):**
    - Increment .
    - If
      * Increment .
      * Update .
    - Else:
      * Update .
    - Plot the new point .

1. **Terminate:**

* Stop when the endpoint is plotted.

**Code: -**

import matplotlib.pyplot as plt

def BA():

    x1=int(input('Enter the value of x1: '))

    y1=int(input('Enter the value of y1: '))

    x2=int(input('Enter the value of x2: '))

    y2=int(input('Enter the value of y2: '))

    dx=abs(x2-x1)

    dy=abs(y2-y1)

    xes=[]

    yes=[]

    x,y = x1,y1

    if(dx>=dy):

        if(x2>x1):

            sx=1

        else:

            sx=-1

        if(y2>y1):

            sy=1

        else:

            sy=-1

        k=0

        Po=(2\*dy)-dx

        Pk=Po

        while(x<x2):

            xes.append(x)

            yes.append(y)

            x=x+sx

            if(Pk>=0):

                y=y+sy

                pkk=Pk+2\*dy-2\*dx

            else:

                pkk=Pk+2\*dy

    if(dy>=dx):

        if(x2>x1):

            sx=1

        else:

            sx=-1

        if(y2>y1):

            sy=1

        else:

            sy=-1

        k=0

        Po=(2\*dx)-dy

        Pk=Po

        while(x<x2):

            xes.append(x)

            yes.append(y)

            y=y+sy

            if(Pk>=0):

                x=x+sx

                pkk=Pk+2\*dx-2\*dy

            else:

                pkk=Pk+2\*dx

    plt.plot(xes, yes, marker='x')

    plt.show()

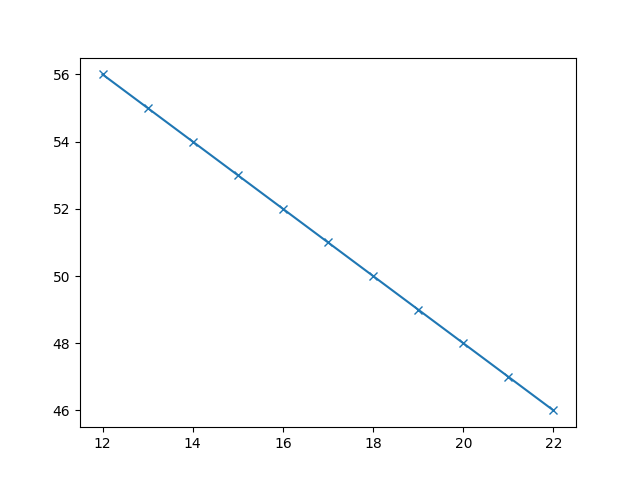
BA()

**Output: -**

Enter the value of x1: 12

Enter the value of y1: 56

Enter the value of x2: 23

Enter the value of y2: 41

**Discussion**

The Bresenham's Line Drawing Algorithm efficiently plots lines on a raster grid using integer arithmetic, avoiding floating-point operations. It handles shallow (∣m∣≤1) and steep (∣m∣>1) slopes by incrementally updating the decision parameter. Implementation challenges, such as managing negative slopes and switching dominant axes for steep lines, were resolved effectively. The experiment demonstrated the algorithm's accuracy and efficiency, even for vertical and horizontal lines.

**Conclusion**

Bresenham's Algorithm is a reliable and efficient method for rasterizing lines, ideal for computer graphics. Its simplicity, accuracy, and efficiency ensure its lasting importance in practical applications.