**Aim:** *To implement Bresenham’s algorithms for drawing a line segment between two given end points*.

**Objective:**

*Draw a line using Bresenham’s & line algorithm that determines the points of an n-dimensional raster that should be selected to form a close approximation to a straight line between two points*

**Theory:**

*In Bresenham’s line algorithm pixel positions along the line path are obtained by determining the pixels i.e. nearer the line path at each step.*

**Algorithm –**

**x=x1;**

**y=y1;**

**dx=x2-x1;**

**dy=y2-y1;**

**p=2dy-2dx;**

**while(x<=x2)**

**{**

**putpixel(x,y);**

**x++;**

**if(p<0)**

**{**

**p=p+2dy;**

**}**

**else**

**{**

**y=y+1;**

**p=p+2dy-2dx;**

**}**

**}**

**}**

**Program –**

**#include<stdio.h>**

**#include<conio.h>**

**#include<graphics.h>**

**#include<math.h>**

**#include<dos.h>**

**void main()**

**{**

**int gd=DETECT, gm;**

**int x1,y1,x2,y2,dx,dy,p;**

**clrscr();**

**initgraph(&gd,&gm,"c:\\turboc3\\bgi");**

**printf("Enter the first co-ordinates:\n");**

**scanf("%d %d",&x1,&y1);**

**printf("Enter the second co-oedinates:\n");**

**scanf("%d %d",&x2,&y2);**

**dx=x2-x1;**

**dy=y2-y1;**

**p=2\*dy-dx;**

**while(x1<=x2){**

**putpixel(x1,y1,50);**

**x1++;**

**if(p<0){**

**p=p+2\*dy;**

**}**

**else**

**{**

**p=p+2\*dy-2\*dx;**

**y1++;**

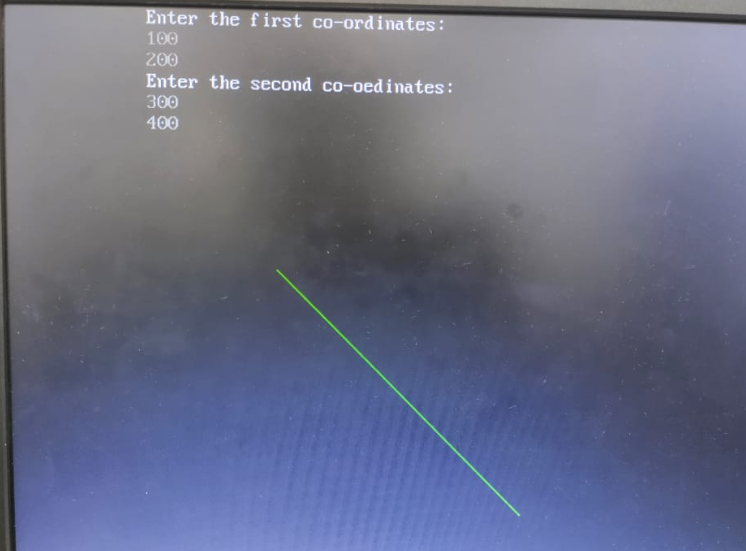
**}**

**}**

**getch();**

**closegraph();**

**Output –**



**Conclusion: Comment on -**

1. *Pixel:*

It is a minute area of [illumination](https://www.google.com/search?sca_esv=554842129&rlz=1C1CHBF_enIN917IN917&q=illumination&si=ACFMAn9IMdf-m8dGI-RtPy6zxE7lk40DkQ_DGyL-HHYIG-m8mAY-ZCukC-VOAEOFCh0Pu8gx8Fy-azrNjbh9n_5qqCU38UczSGVju2P3HC6y3Vv7uIs9yWU%3D&expnd=1) on a display screen, one of many from which an image is composed

1. *Equation for line:*

the equation for a straight line in 2D can be represented as y = mx + b, where 'm' is the slope of the line, 'x' is the x-coordinate of a point on the line, 'y' is the corresponding y-coordinate, and 'b' is the y-intercept (the point where the line crosses the y-axis).

1. *Need of line drawing algorithm:*

The need for a line drawing algorithm arises from the fact that most display devices, such as computer screens, work on a pixel-based raster grid. To display a line on such a grid, the graphics system must determine which pixels to turn on or off to form the desired line segment between the given endpoints.

*4. Slow or fast: Fast*