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

**Objective:** Draw the line using (vector) generation algorithms which determine the pixels that should be turned ON are called as digital differential analyzer (DDA).It is one of the techniques for obtaining a rasterized straight line. This algorithm can be used to draw the line in all the quadrants.

**Theory:**

DDA algorithm is an incremental scan conversion method. Here we perform calculations at each step using the results from the preceding step. The characteristic of the DDA algorithm is to take unit steps along one coordinate and compute the corresponding values along the other coordinate. Digital Differential Analyzer (DDA) algorithm is the simple line generation algorithm which is explained step by step here.

**Algorithm:**

**{**

**dx = x2-x1;**

**dy = y2-y1;**

**if (abs(dx)>abs(dy))**

**{ step = dx;}**

**else { step = dy;}**

**xn = dx/step;**

**yn = dy/step;**

**for (int i = 0;i<= step;i++)**

**{ putpixel (x1,y1,RED);**

**x1 = x1+xn;**

**y1= y1+yn;**

**}**

**}**

**Program:**

#include<conio.h>

#include<math.h>

#include<stdio.h>

#include<graphics.h>

void main()

{

int x1,y1, x2, y2,xn,yn, dx, dy, step, i;

int gd= DETECT, gm;

initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");

printf("enter the value of x1,y1:");

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

printf("enter the value for x2,y2");

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

dx=x2-x1;

dy=y2-y1;

if (abs(dx)>abs(dy))

{ step=dx;}

else {

step=dy;

}

xn=dx/step;

yn=dy/step;

for(i=0;i<=step;i++)

{

putpixel(x1,y1,RED);

x1=x1+xn;

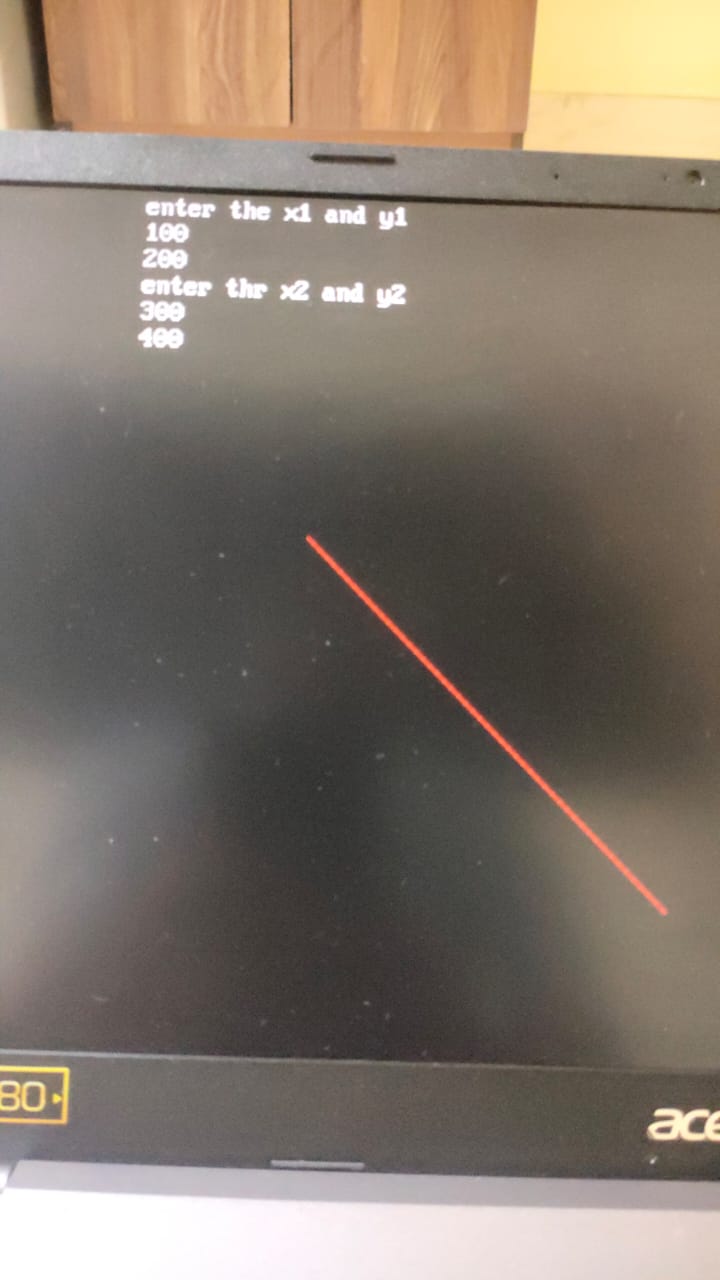
y1=y1+yn;

}

getch();

closegraph();

}

**Output:**

**Conclusion:**

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*