PRACTICAL NO 1

~ ARYA RAUL

COMPS 3 20

Aim: To implement DDA algorithms for drawing a line segment between two given endpoints.

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:

Step 1: Read end points of the line as (x1, y1) & (x2, y2) such that x1 != x2 and y1 != y2

Step 2: Calculate dx = x2 – x1 and dy = y2 – y1

Step 3: if (dx >= dy)

step = dx

else

step = dy

Step 4: xin = dx / step & yin = dy / step

Step 5: x = x1 + 0.5 & y = y1 + 0.5

Step 6: for (i = 0; i < step; i++)

{

x = x + xin

y = y + yin

putpixel (x, y)

}

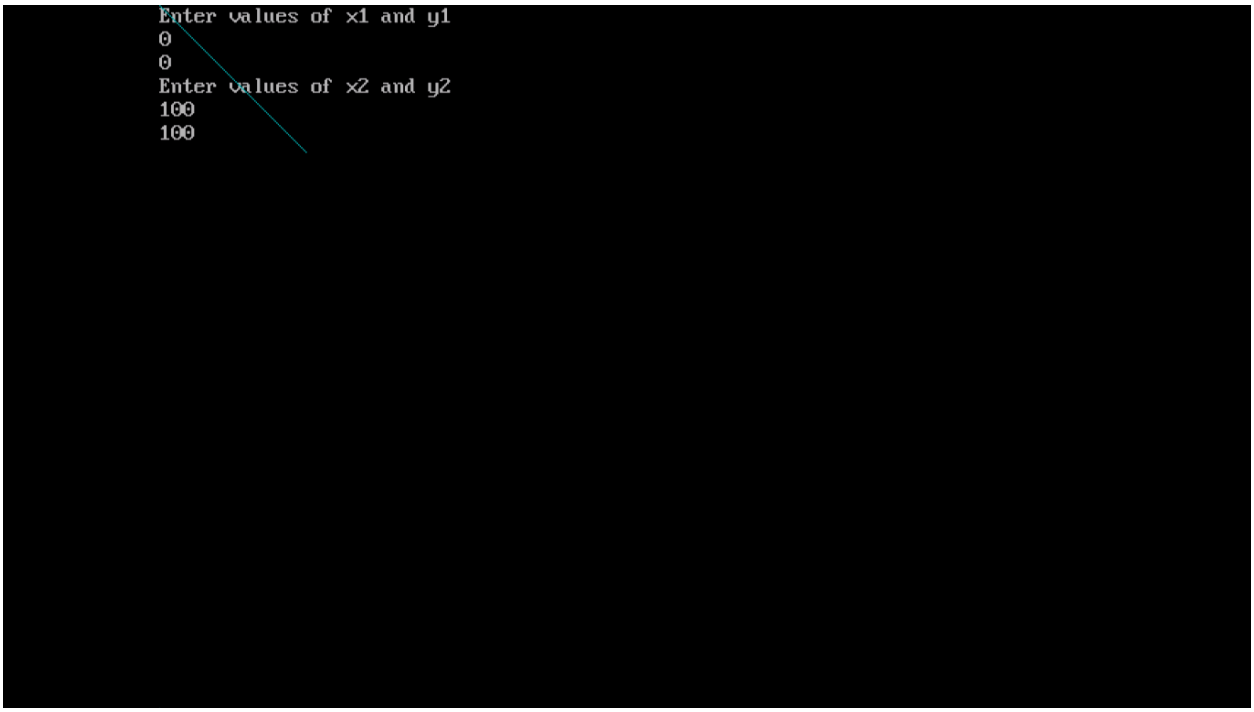
Program:

```
#include<iostream.h>
#include<dos.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>

void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,dx,dy,steps,xinc,yinc;
initgraph(&gd,&gm,"C:\\turboc3\\bgi");
cout<<"Enter values of x1 & y1"<<endl;
cin>>x1>>y1;
cout<<"Enter values of x2 & y2"<<endl;
cin>>x2>>y2;
dx=x2-x1;
dy=y2-y1;
```

```
if(abs(dx)>abs(dy))
steps=abs(dx);
else
steps=abs(dy);
xinc=dx/steps;
yinc=dy/steps;
for(int i=0;i<steps;i++)
{
putpixel(x1,y1,2);
x1=x1+xinc;
y1=y1+yinc;
delay(50);
}
getch();
closegraph();
}
```

Output:



Conclusion:

Comment on -

1. Pixel: working with pixels, the DDA algorithm offers a reliable approach that calculates incremental changes along the axis to generate a smooth and accurate line or shape.

2. Equation for line: y = mx + c

3. Need of line drawing algorithm: In conclusion, line drawing algorithms are essential tools in computer graphics and image processing for generating accurate and efficient representations of lines and curves**.**

4. Slow or fast: fast