**Practical No 2**

**A) Implementation of line generation using DDA line algorithm.**

**Aim: Write a program to implementation of line generation using DDA line algorithm.**

**Theory:**

DDA stands for Digital Differential Analyzer. It is an incremental method of scan conversion of line. In this method the calculation is performed at each step by using the result of previous steps. It is the simplest line drawing algorithm. Given the starting and ending co-ordinates of a line the DDA algorithm attempts to generate the points between the starting and ending coordinates.

**Algorithm:**

In the following algorithm we refer the one point of line as X1, Y1 and second point of line as X2, Y2.

1. Get the input of two end points (X1, Y1) and (X2, Y2).
2. Step 2 − Calculate the difference between two end points.

dx = X2 - X1

dy = Y2 - Y1

1. Based on the calculated difference in step-2, you need to identify the number of steps to put pixel. If dx > dy, then you need more steps in x coordinate; otherwise in y coordinate. if (absolute(dx) > absolute(dy)) Steps = absolute(dx); else Steps = absolute(dy);
2. Calculate the increment in x coordinate and y coordinate. Xincrement = dx / (float) steps; Yincrement = dy / (float) steps;
3. Put the pixel by successfully incrementing x and y coordinates accordingly and complete the drawing of the line. for(int v=0; v < Steps; v++) { x = x + Xincrement; y = y + Yincrement; putpixel(Round(x), Round(y)); }

**Conclusion: We have implemented line generation using DDA line algorithm.**

**Code:**

#include <graphics.h>

#include <iostream.h>

#include <math.h>

#include <conio.h>

void DDA(float x1,float y1,float x2,float y2)

{

float dy,dx,x,y,m;

x=x1;y=y1;

dx=x2-x1;

dy=y2-y1;

if(dy>dx)

{

m=dx/dy;

while(y<y2)

{

y=y+1;

x=x+m;

putpixel(ceil(x),ceil(y),15);

}

}

else

{

m=dy/dx;

while(x<x2)

{

x=x+1;

y=y+m;

putpixel(ceil(x),ceil(y),15);

}

}

}

int main(void)

{

clrscr();

int gdriver = DETECT, gmode, errorcode;

float x1,y1,x2,y2;

cout<<"Enter co-ordinates of first point: ";

cin>>x1>>y1;

cout<<"Enter co-ordinates of second point: ";

cin>>x2>>y2;

initgraph(&gdriver, &gmode, "C:/TC/BGI");

DDA(x1,y1,x2,y2);

getch();

closegraph();

return 0;

}

**Output:**



