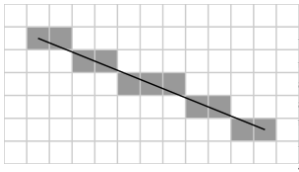
**Practical No 2**

**B) Implementation of line generation using Bresenhams line algorithm.**

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

**Theory:**

Bresenham's line algorithm is an algorithm that determines the points of an n-dimensional raster that should be selected in order to form a close approximation to a straight line between two points. It is commonly used to draw lines on a computer screen, as it uses only integer addition, subtraction and bit shifting, all of which are very cheap operations in standard computer architectures. It is one of the earliest algorithms developed in the field of computer graphics. An extension to the original algorithm may be used for drawing circles.



While algorithms such as Wu's algorithm are also frequently used in modern computer graphics because they can support antialiasing, the speed and simplicity of Bresenham's line algorithm means that it is still important. The algorithm is used in hardware such as plotters and in the graphics chips of modern graphics cards. It can also be found in many software graphics libraries. Because the algorithm is very simple, it is often implemented in either the firmware or the graphics hardware of modern graphics cards.

**Algorithm:**

1. Input the two-line endpoints and store left endpoint as (x0, y0).
2. Pre-calculate the value of dx, dy,2dy and 2dy-2dx.
3. Color the initial pixels (x0, y0).
4. Let P0=2dy-dx.
5. At each xk along the line starting with k = 0. - If pk<0 then the next point to plot is (xk+1, yk) and pk+1=pk+2dy. - Else the next point to be plotted is (xk+1,yk+1) and pk+1=pk+2dy-dx.
6. Repeat Step 5 dx times.

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

**Code:**

#include<stdio.h>

#include<conio.h>

#include<iostream.h>

#include<graphics.h>

#include<math.h>

#include<dos.h>

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

{

int dx, dy, p, end;

float x, y;

dx = abs(x1 - x2);

dy = abs(y1 - y2);

p = 2\*dy-dx;

if(x1>x2)

{

x =x2;

y =y2;

end =x1;

}

else

{

x =x1;

y =y1;

end =x2;

}

putpixel(x,y,10);

while(x<end)

{

x = x+1;

if(p<0)

{

p = p+2\*dy;

}

else

{

y = y+1;

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

}

putpixel(x, y,WHITE);

delay(10);

}

}

void main()

{

clrscr();

int gd = DETECT, gm;

float x1, x2, y1, y2, i;

initgraph(&gd, &gm, "C:/TC/BGI");

cout<<"Enter co-ordinates of X1: ";

cin>>x1;

cout<<"Enter co-ordinates of Y1: ";

cin>>y1;

cout<<"Enter co-ordinates of X2: ";

cin>>x2;

cout<<"Enter co-ordinates of Y2: ";

cin>>y2;

draw(x1,y1,x2,y1);

draw(x1+10,y1,x2+10,y2);

for(i=20;i<90;i+=10)

{

draw(x1+i,y1,x2+i,y2);

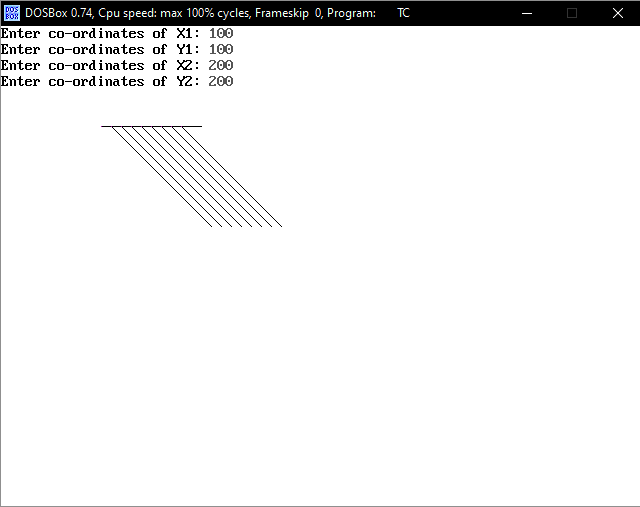
}

getch();

closegraph();

}

**Output:**

****