Bresenham's Line Algorithm:

Step1: Start Algorithm

Step2: Declare variable $x_1,x_2,y_1,y_2,d,i_1,i_2,dx,dy$

Step3: Enter value of x_1,y_1,x_2,y_2
Where x_1,y_1 are coordinates of starting point
And x_2,y_2 are coordinates of Ending point

Step4: Calculate dx = x_2-x_1
Calculate dy = y_2-y_1
Calculate $i_1=2*dy$
Calculate $i_2=2*(dy-dx)$
Calculate $d=i_1-dx$

Step5: Consider (x, y) as starting point and x_{end} as maximum possible value of x.
If dx < 0
Then x = x_2
y = y_2
$x_{end}=x_1$
If dx > 0
Then x = x_1
y = y_1
$x_{end}=x_2$

Step6: Generate point at (x,y)coordinates.

Step7: Check if whole line is generated.
If x > = x_{end}
Stop.

Step8: Calculate co-ordinates of the next pixel
If d < 0
Then d = d + i_1
If d ≥ 0
Then d = d + i_2
Increment y = y + 1

Step9: Increment x = x + 1

Step10: Draw a point of latest (x, y) coordinates

Step11: Go to step 7

Step12: End of Algorithm

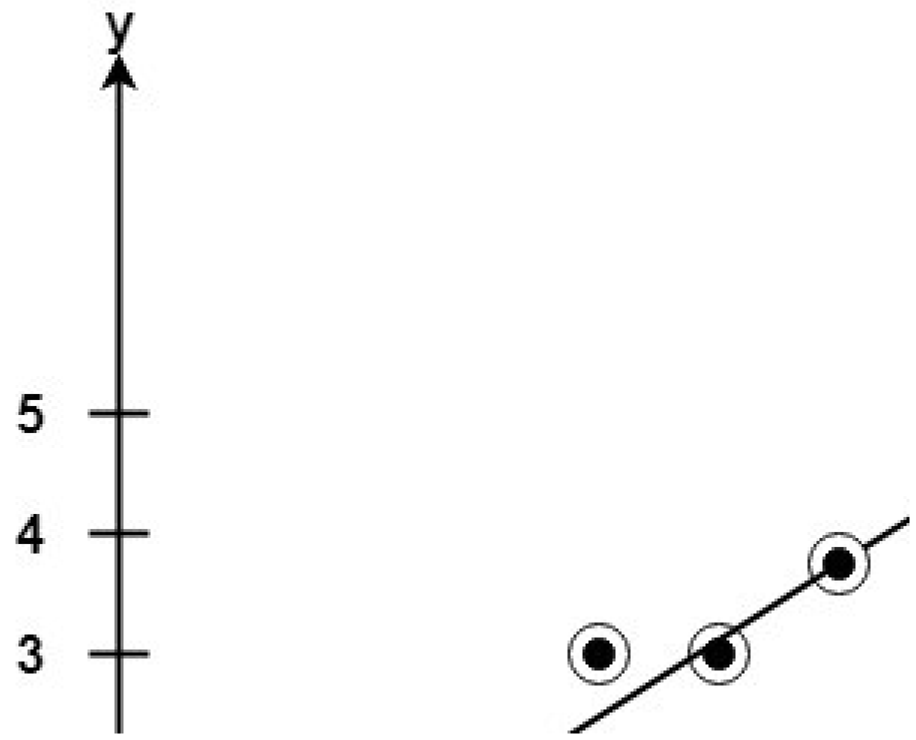
Example: Starting and Ending position of the line are (1, 1) and (8, 5). Find intermediate points.

Solution: $x_1=1$
$y_1=1$
$x_2=8$
$y_2=5$

dx= x_2-x_1=8-1=7
dy=y_2-y_1=5-1=4
I_1=2* Δy=2*4=8
I_2=2*(Δy-Δx)=2*(4-7)=-6

d = I_1-Δx=8-7=1

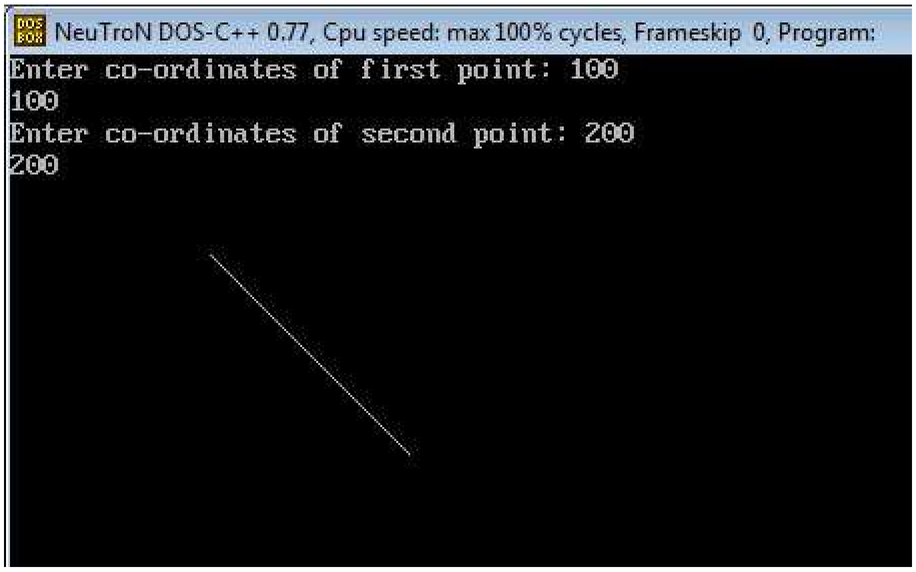
x	y	d=d+I_1 or I_2
1	1	d+I_2=1+(-6)=-5
2	2	d+I_1=-5+8=3
3	2	d+I_2=3+(-6)=-3
4	3	d+I_1=-3+8=5
5	3	d+I_2=5+(-6)=-1
6	4	d+I_1=-1+8=7
7	4	d+I_2=7+(-6)=1
8	5	



Program to implement Bresenham's Line Drawing Algorithm:

```
#include<stdio.h>
#include<graphics.h>
void drawline(int x0, int y0, int x1, int y1)
{
    int dx, dy, p, x, y;
    dx=x1-x0;
    dy=y1-y0;
    x=x0;
    y=y0;
    p=2*dy-dx;
    while(x<x1)
    {
        if(p>=0)
        {
            putpixel(x,y,7);
            y=y+1;
            p=p+2*dy-2*dx;
        }
        else
        {
            putpixel(x,y,7);
            p=p+2*dy;}
            x=x+1;
        }
}
int main()
{
    int gdriver=DETECT, gmode, error, x0, y0, x1, y1;
    initgraph(&gdriver, &gmode, "c:\\turboc3\\bgi");
    printf("Enter co-ordinates of first point: ");
    scanf("%d%d", &x0, &y0);
    printf("Enter co-ordinates of second point: ");
    scanf("%d%d", &x1, &y1);
    drawline(x0, y0, x1, y1);
    return 0;
}
```

Output:


NeuTroN DOS-C++ 0.77, Cpu speed: max 100% cycles, Frameskip 0, Program:

```
Enter co-ordinates of first point: 100
100
Enter co-ordinates of second point: 200
200
```