Name: Shaikh Salif Aminuddin

CLASS: S2 **ROLL NO:** 2201094

EXPERIMENT NO: 2

AIM: Implement Bresenham"s Line algorithm(dotted/dashed/thick)

THEORY: This algorithm is used for scan converting a line. It was developed by Bresenham. It is an efficient method because it involves only integer addition, subtractions, and multiplication operations. These operations can be performed very rapidly so lines can be generated quickly. In this method, next pixel selected is that one who has the least distance from true line.

Advantage:

1. It involves only integer arithmetic, so it is simple.

2. It avoids the generation of duplicate points.

3. It can be implemented using hardware because it does not use multiplication and division.

Disadvantage:

This algorithm is meant for basic line drawing only Initializing is not a part of Bresenham's line algorithm. So to draw smooth lines, you should want to look into a different algorithm.

Bresenham's Line Algorithm:

Step1: Start Algorithm

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

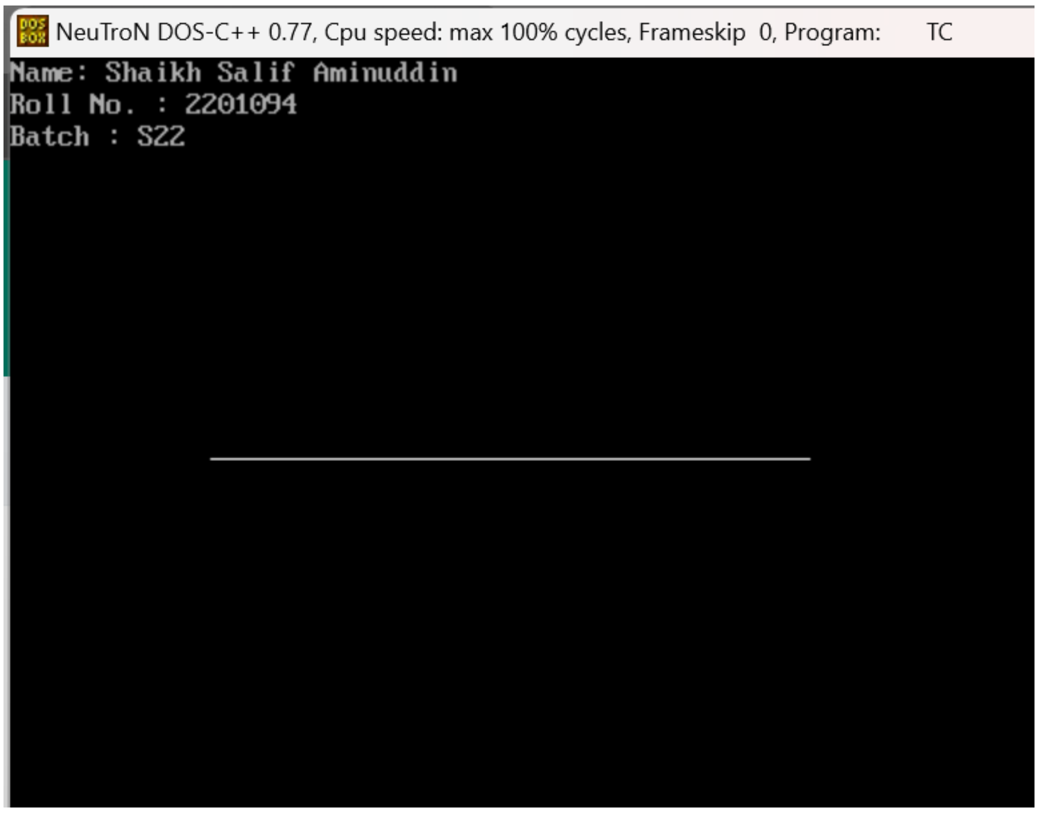
Step3: Enter value of x_1, y_1, x_2, y_2
Where $x_1,y_1$are coordinates of starting point
And x_2, y_2 are coordinates of Ending point

```
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 $d \geq 0$
           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
```

CODE:

```
#include<stdio.h>
#include<graphics.h>
#include<conio.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,WHITE);
                y=y+1;
                p=p+2*dy-2*dx;
                delay (30);
            }
            else {
                putpixel(x,y,WHITE);
                delay (30);
                p=p+2*dy;
                }
                x=x+1;
            }
    }
    int main()
    {
        int gd = DETECT, gm, error, x0, y0, x1, y1;
        initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");
        x0 = 100,y0=200 , x1=400, y1=200;
        printf("Name: Shaikh Salif Aminuddin \nRoll No. : 2201094 
\nBatch : S22 ");
        setbkcolor(BLACK);
        drawline(x0, y0, x1, y1);
        getch();
        return 0;
    }
```

OUTPUT:



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

```
Name: Shaikh Salif Aminuddin
Roll No. : 2201094
Batch : S22
```

CONCLUSION: Bresenham's line drawing algorithm remains a powerful tool in the field of computer graphics. Its simplicity, efficiency, and accuracy have made it a staple in various applications, enabling the creation of stunning visual experiences.