

NAME - SUBHAM

UNIVERSITY ROLL NO - 1121147

SUBJECT - COMPUTER GRAPHICS

SUBJECT CODE - PBC-602

Ans¹ Bresenham's line Algorithm

Assume a fixed $P_1(x_1, y_1)$ then select subsequent pixels as we work our way to the right, one fixed position at a time in the horizontal direction towards $P_2(x_2, y_2)$.

Once a pixel is chosen at any step

The next pixel is

1. Either the one to its right (lower-bound for the line.)
2. One to its right and up (upper-bound for the line)

Algorithm.

Step 1. Start Algorithm

Step 2. Declare variable $x_1, x_2, y_1, y_2, d, i_1, i_2, dx, dy$

Step 3. 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

Step 4. calculate $dx = x_2 - x_1$

calculate $dy = y_2 - y_1$

calculate $i_1 = 2 * dy$

calculate $i_2 = 2 * (dy - dx)$

calculate $d = i_1 - dx$

step 5. 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 $x = x_1$

$y = y_1$

$x_{end} = x_2$

Step 6. Generate point at (x, y) coordinates

Step 7. Check if whole line is generated.

If $x \geq x_{end}$

stop.

Step 8. 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 $y = y + 1$

Step 9. Increment $x = x + 1$

Step 10. Draw a point of least (x, y) coordinates

Step 11. Go to step 7

Step 12. End of 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;  
int initgraph (&gdriver, &gmode, "c:\\turbo3\\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;
```

}

Enter co-ordinates of first point: 100

100

Enter co-ordinates of second point: 200

200

