Name – Sonal
University Roll No – 1121145
Course – BCA
Section – C
Semester – 6th
Paper Name – Computer Graphics and Animation Practical
Paper Code – PBC – 602

Answer – 1

Program to implement Bresenham Line Drawing Algorithm.

Algorithm:

Step : 1 Start

Step : 2 Enter value of $x1$, $y1$, $x2$, $y2$, d, i1, i2, dx, dy

Step : 3 Enter value of $x1$, $y1$, $x2$, $y2$ where $x1$, $y1$ are coordinates of starting point and $x2$, $y2$ are coordinates of ending point

Step : 4 Calculate $dx = x2 - x1$
calculate $dy = y2 - y1$
Calculate $i1 = 2 \times dy$

Sonal

Calculate $i2 = 2 * (dy - dx)$

Calculate $d = i1 - dx$

Step : 5 Consider (x, y) as starting point and xend maximum possible value of x

if $dx < 0$

Then $x = x2$

$y = y2$

$xend = x1$

if $dx > 0$

Then $x = x1$

$y = y1$

$xend = x2$

step : 6 Generate point at (x, y) coordinates

step : 7 Check if whole line is generated.

if $x >= xend$

stop.

step : 8 Calculate coordinates of the next pixel

if $d < 0$

Then $d = d + i1$

if $d \geq 0$

Then $d = d + i2$

Increment $y = y + 1$

Final

step : 9 Increment $x = x+1$

step : 10 Draw a point of latest (x, y) coordinates

step : 11 go to step 7

step : 12 End

Code :

```
# 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 ;
        }
```

Final

```
else
{   putpixel (x, y, 7);
    p = p + 2 * dy;
  }
  x = x + 1;
 }
}
int main ()
{
  int gdrive = DETECT, gmode, error, x0, x1, y0, 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;
}
```

Sonal

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

Name – Sonal
Roll No – 1121145

Answer – 2

Algorithm and Program to implement Mid Point Circle Drawing Algorithm:

Algorithm :

step :1 put $x =$ radius, $y = 0$, $err = 0$
we have $p = 1 - r$

step: 2 Repeat steps while $(x <= y)$

Plot (x, y)

if $(p < 0)$

Then set $p = p + 2x + 3$

else

$p = p + 2(x - y) + 5$

$y = y - 1$

$x = x + 1$

step :3 End

Sonal

Code :

```
#include <stdio.h>
#include <graphics.h>
void drawcircle (int xo, int yo, int radius)
{
    int x = radius;
    int y = 0;
    int err = 0;
    while (x >= y)
{
    putpixel (xo + x, yo + y, 7);
    putpixel (xo + y, yo + x, 7);
    putpixel (xo - y, yo + x, 7);
    putpixel (xo - x, yo + y, 7);
    putpixel (xo - x, yo - y, 7);
    putpixel (xo - y, yo - x, 7);
    putpixel (xo + y, yo - x, 7);
    putpixel (xo + x, yo - y, 7);
    if (err <= 0)
    {   y += 1;
        err += 2*y + 1;
    }
```

Final

```
if (err >0)
{  x -=1 ;
 err + = 2 *x + 1 ;
 }
 }
 }
int main ( )
{  int gdriver = DETECT , gmode , error , x , y , r ;
 initgraph (&gdriver , &gmode , "C: \\turboc3\\bgi) ;
 Printf (" Enter radius of circle : ");
 Scanf (" %d" , & r ) ;
 Printf ("Enter co-ordinates of center(x and y): ");
 Scanf(" %d %d" ; & x , & y ) ;
 Drawcircle (xc , y , r ) ;
 return 0 ;
 }
```

Sonal

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

```
Enter radius of circle: 100
Enter co-ordinates of center(x and y): 150
150
```