**Experiment No. 3**

Aim: To implement midpoint circle algorithm.

Objective:

Draw a circle using mid-point circle drawing algorithm by determining the points needed for

rasterizing a circle. The mid-point algorithm to calculate all the perimeter points of the circle

in the first octant and then print them along with their mirror points in the other octants.

Theory:

The shape of the circle is similar in each quadrant. We can generate the points in one section

and the points in other sections can be obtained by considering the symmetry about x-axis

and y-axis.
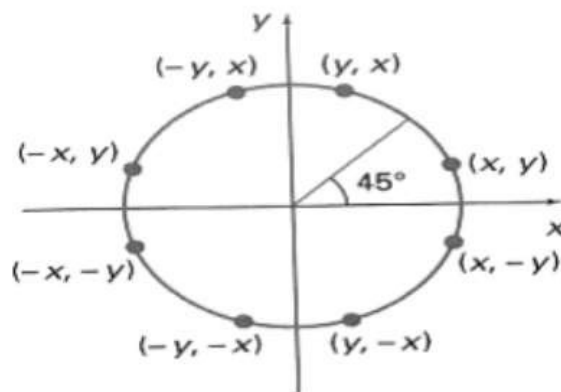
The equation of circle with center at origin is $x^2 + y^2 = r^2$
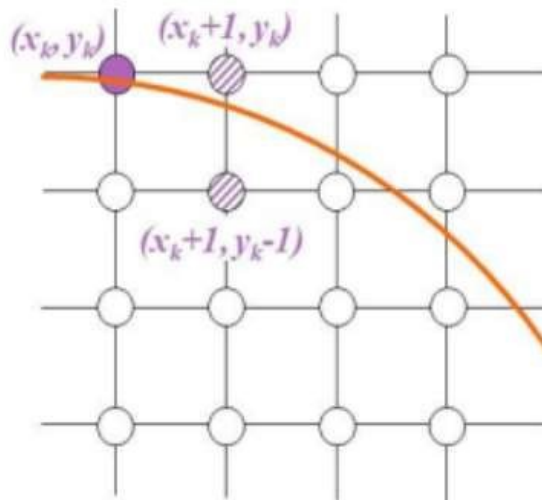
Let the circle function is f circle (x, y) -

□ is < 0, if (x, y) is inside circle

boundary,

• is = 0, if (x, y) is on circle boundary,

• is > 0, if (x, y) is outside circle boundary.

Consider the pixel at (x_k, y_k) is plotted,

Now the next pixel along the circumference of the circle will be either (xk + 1, yk) or (xk + 1, yk – 1) whichever is closer the circle boundary.

Let the decision parameter pk is equal to the circle function evaluate at the mid-point between two pixels.

If pk < 0, the midpoint is inside the circle and the pixel at yk is closer to the circle boundary. Otherwise, the midpoint is outside or on the circle boundary and the pixel at yk – 1 is closer to the circle boundary.

Algorithm –

Step1: Put x =0, y =r in equation 2

　　　We have p=1-r

Step2: 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 (end if)   x

=x+1 (end loop)

Step3: End

Program –

```c
#include<stdio.h>

#include<conio.h>

#include<graphics.h> void

pixel(int x,int y,int xc,int yc)

{

        putpixel(x+xc,y+yc,BLUE);

putpixel(x+xc,-y+yc,BLUE);

putpixel(-x+xc,y+yc,BLUE);

putpixel(-x+xc,-y+yc,BLUE);

putpixel(y+xc,x+yc,BLUE);

putpixel(y+xc,-x+yc,BLUE);

putpixel(-y+xc,x+yc,BLUE);

putpixel(-y+xc,-x+yc,BLUE);

} main()

{

 int gd=DETECT,gm=0,r,xc,yc,x,y;

float p;

        //detectgraph(&gd,&gm);

initgraph(&gd,&gm," ");
```

```
printf("\n Enter the radius of the circle:");

        scanf("%d",&r);

        printf("\n Enter the center of the circle:");

        scanf("%d %d",&xc,&yc);

         y=r;

x=0;

        p=(5/4)-r;

while(x<y)

        {

                if(p<0)

                {

                        x=x+1;

                 y=y;

                        p=p+2*x+3;

                }

                else

                {

                        x=x+1;

                 y=y-1;

                        p=p+2*x-2*y+5;

                }

                pixel(x,y,xc,yc);

        }
```
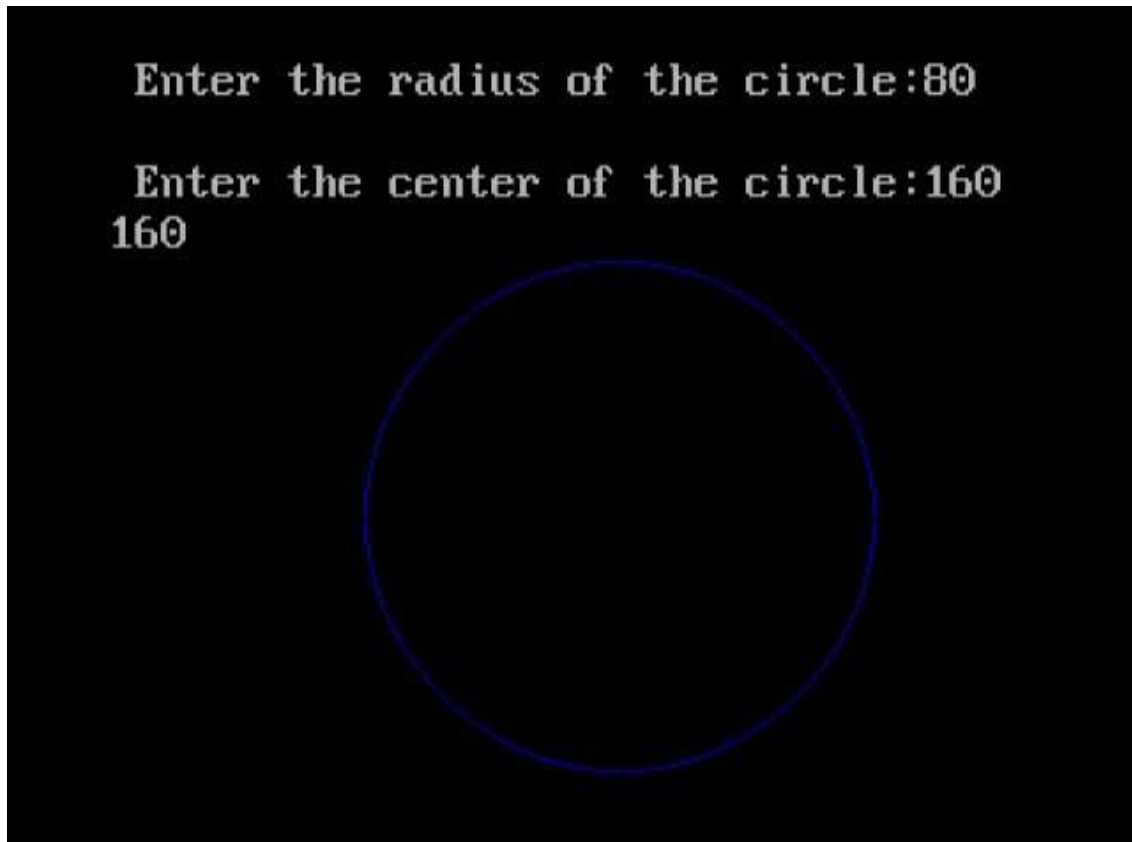
```
        getch();

closegraph();

}
```

output –

Conclusion: Comment on

1. Fast or slow

2. Draw one arc only and repeat the process in 8 quadrants

3. Difference with line drawing method