EXPERIMENT NO. 03

AIM: To implement midpoint circle generation algorithm circle algorithm for drawing a circle of given center (x, y) and radius r.

SOFTWARE REQUIRED: TurboC.

THEORY:

The Midpoint circle drawing algorithm considers the eight way symmetry of the circle to generate it. To achieve best approximation to the true circle we have to select these pixels in the Raster that fall the least distance from the true circle. Let us assume point P as a last Scan converted pixel. Now, we have two options either to choose pixel A or pixel B.

The distance of pixels A and B from the origin are, $DA = \sqrt{[(Xi+1)^2+(Yi)^2]}$

$DB = \sqrt{[(Xi+1)^2+(Yi-1)^2]}$

$\delta A = DA - r$ $\delta B = DB - r$.

For more efficient δA and δB are defined as δA = DA^2 - r^2

δB = DB^2 - r^2

From fig(b) we notice that δA is always the positive &δB is always negative.

Thus δi can be defined as following δi = δA + δB

we can say that if δi<0 , i.e. δA<δB then only x is incremented , otherwise x is incremented is positive direction & y is incremented in negative direction i.e.

For di<0 , Xi+1 = Xi + 1 and

For di >=0 , Xi+1 = Xi + 1 and Yi+1 = Yi – 1

The equation for di at starting point can be simplified as

di = δA + δB = (Xi + 1)^2 +(Yi)^2 - r^2 +(Xi + 1)^2 (Yi – 1)^2 - r^2

= (0 + 1)^2 +(r)^2 - r^2 +(0 + 1)^2 (r – 1)^2 - r^2 = 3-2r

Similarly, the equation for di+1 for both the options are given as

For d<0 , d = d+4x+6 and

For d>=0 , d=d+4(x+y)+10

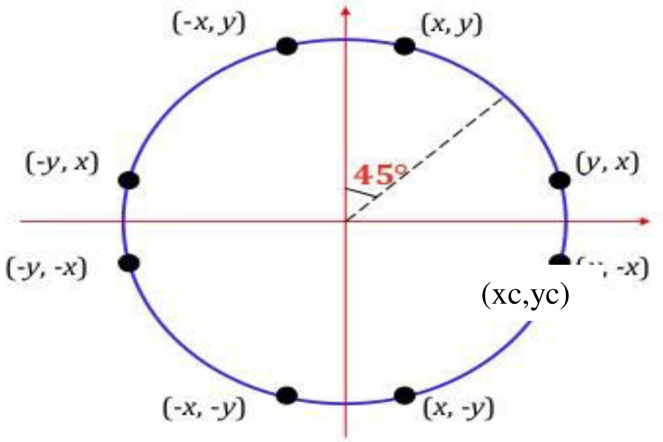
ALGORITHM:

1. Read radius and centre of circle
2. Initially current x=0 current y=radius
3. Initialize decision variable

```
d = 3 – 2r
4. Draw circle.

5. If d<0
        Then d = d + 4(current x) + 6
        Else
        d = d + 4(current x-y) + 10
        (current y) = (current y) – 1
        end if

6. Calculate (current x) < (current y)
        Go to step 4
        Else
        Go to step 8
7. Draw circle
8. Stop.
        Draw cir ()
        Putpixel (x + xc , y + yc)
        Putpixel (y + xc , x + yc)
        Putpixel (-x + xc , y + yc)
        Putpixel (x + xc , -y + yc)
        Putpixel (x + xc , -y + yc)
        Putpixel (x + xc , -x + yc)
        Putpixel (-x + xc , -y + yc)
        Putpixel (-y + xc , -x + yc)
        Return
```



CONCLUSION: Thus successfully Implemented the Midpoint Circle drawing algorithm which is able to find a path through the pixel grid using pixels which are as close as possible to solutions of $x^2+y^2=r^2$

SIGN **GRADE**

DATE