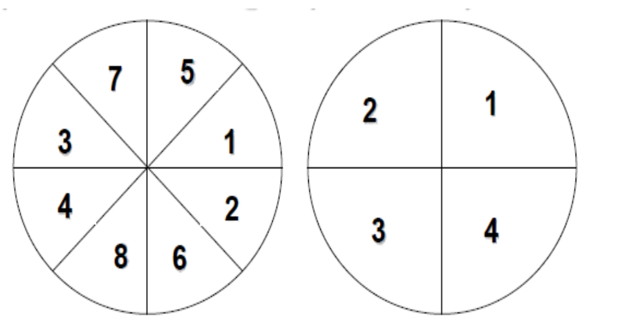
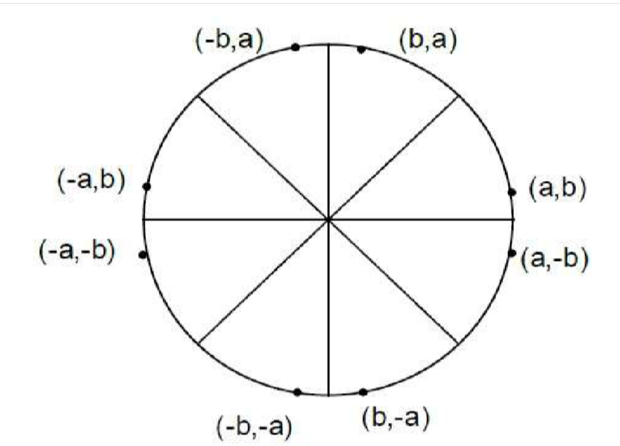
symmetric (incremental) algorithm

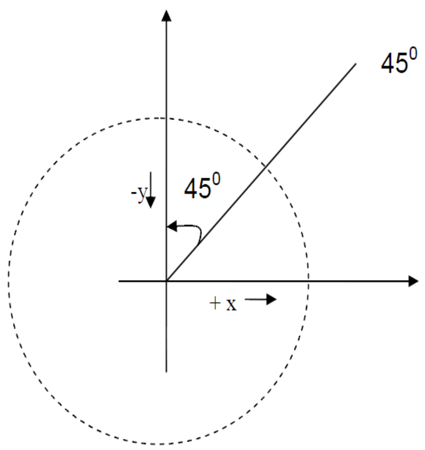
- Computation can be reduced by considering the symmetry of circle.
- The shape of the circle is similar in each quadrant
- We can generate the circle section in the second quadrant by noting that the two circle sections are symmetric with respect to the y axis And circle sections in the third and fourth quadrants can be obtained from sections in the first and second quadrants by considering symmetry about the x axis.



Symmetry in circle

Circle sections in adjacent octants within one quadrant are symmetric with respect to the 45' line dividing the two octants.





Consider a circle center at the origin(0,0),if the point (x,y) is on the circle then we can trivially compute seven other points on the circle.

• This method proposed the center of circle at origin point(0,0),so the first pixel in the circle is (r,0).

• The other pixels are computed depend on polar equation as follow:-

$x = xc + r \cos \theta$

$y = yc + r \sin \theta$ (xc,yc) = (0,0)

$x = r \cos \theta$

$y = r \sin \theta$ **use differential**

$Dx = -r \sin \theta \, d\theta$

$Dy = r \cos \theta \, d\theta$

$Dx = -y \, d\theta$

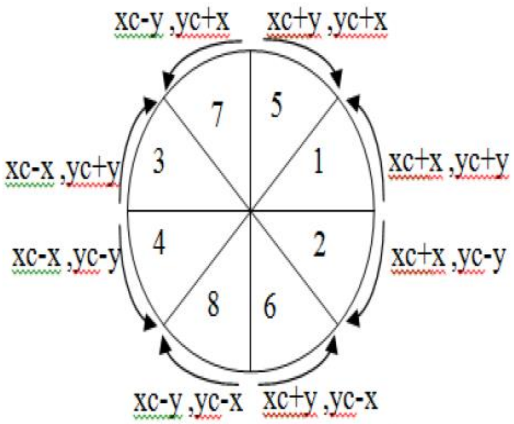
$Dy = x \, d\theta$

as we know

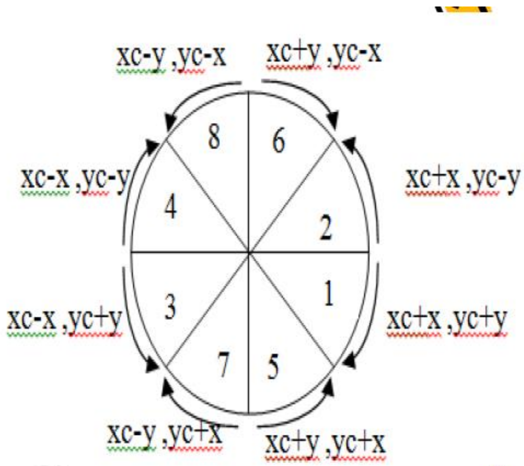
$x = x + Dx$

$y = y + Dy$ **symmetry equation**

if we add center (xc,yc) we obtain the values of 8 pixels of the circle as figure bellow.



Symmetry in mathematical



Symmetry in computer

Symmetric algorithm

```
Start
th=0 , pi=3.141593 , dth=1/r , x=r , y=0;
while th<=pi/4
begin
plot (integer (xc+x) , integer (yc+y) )
plot (integer (xc+x) , integer (yc-y) )
plot (integer (xc-x) , integer (yc+y) )
plot (integer (xc-x) , integer (yc-y) )
plot (integer (xc+y) , integer (yc+x) )
plot (integer (xc+y) , integer (yc-x) )
plot (integer (xc-y) , integer (yc+x) )
plot (integer (xc-y) , integer (yc-x) )
th = th + dth ;
x = x - y * dth ;
y = y + x * dth ;
End while
Finish
```

Midpoint (bresenham) Circle Algorithm

- Bresenham's line algorithm for raster displays is adapted to circle generation by setting up decision parameters (P) for finding the closest pixel to the circumference at each step.

- For a given radius r and screen center position (xc,yc), we can first set up our algorithm to calculate pixel positions around a circle path centered at the coordinate origin (0,0). So first pixel is (0,r)

- each calculated position (x,y) is moved to its proper screen position by adding xc to x and yc to y.

- we compute the first octant pixels from x=0 to x=y.
- Positions for the other seven octants are then obtained by symmetry
- we can take unit steps in the positive x direction over octant and use a decision parameter to determine which of the two possible y positions is closer to the circle path at each step.

$$r^2 = x^2 + y^2$$
$$P = (x+1)^2 + (y-1)^2 - r^2$$
$$p = 2(1-r)$$

- Any point (x , y) on the boundary of the circle with radius r satisfies the equation p = 0.
- If the point is in the interior of the circle, P is negative value.
- if the point is outside the circle, P is positive.

$$P\begin{cases} < 0 & \text{if } (x, y) \text{ is inside the circle b} \\ = 0 & \text{if } (x, y) \text{ is on the circle boun} \\ > 0 & \text{if } (x, y) \text{ is outside the circle} \end{cases}$$

• We need to determine whether the pixel at position (x+ 1, y) or the one at position (x+1,y-1) is closer to the circle. So If p<0,the point is inside the circle and the pixel (x+1,y) is closer to the circle boundary. Otherwise, the point is outside or on the circle boundary and we select the pixel (x+1,y-1).

$$P \begin{cases} < 0 & (x+1\ ,\ y) \\ & p = p + 2x + 1 \\ \geq 0 & (x+1\ ,\ y-1) \\ & p = p + 2(x - y) + 1 \end{cases}$$

midpoint circle algorithm

Start

x = 0 , y = r p = 2*(1 – r)

While x < y

x = x + 1

If p < 0 Then p = p + 2 * x + 1

Else y = y - 1 p = p + 2 * (x - y) + 1

End If

plot (xc+x , yc+y)

plot (xc+x , yc-y)

plot (xc-x , yc+y)

plot (xc-x , yc-y)

plot (xc+y , yc+x)

plot (xc+y , yc-x)

plot (xc-y , yc+x)

plot (xc-y , yc-x)

End while

Finish