CSE4203: Computer Graphics

Lecture – 6 (part - B)

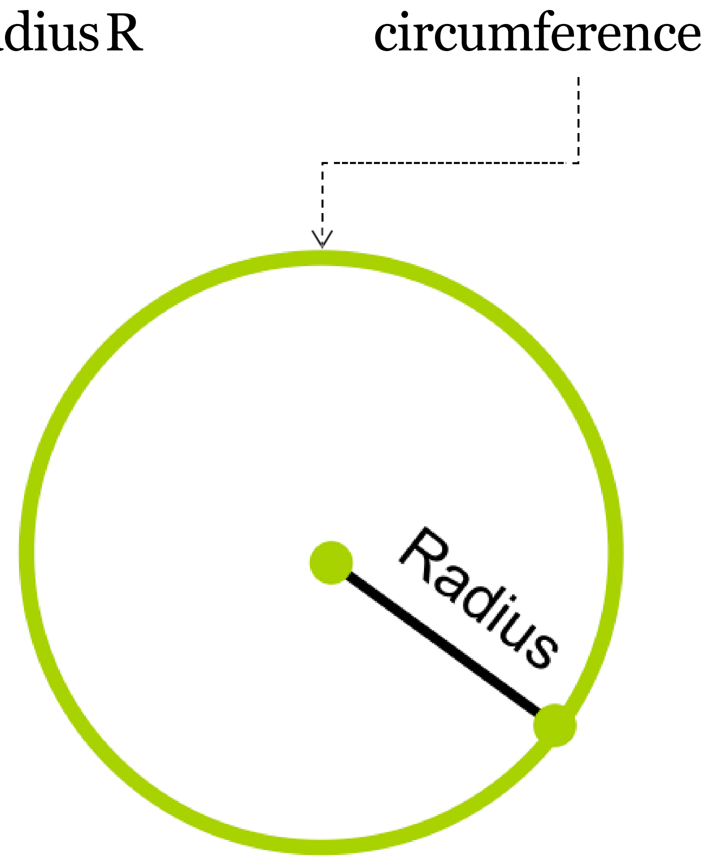
Graphics Pipeline

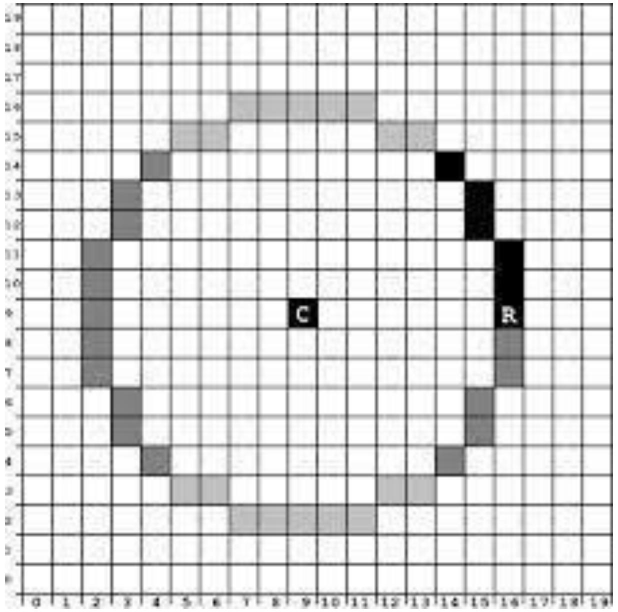
Outline

- Bresenham’s Circle Drawing Algorithm

Assumptions

Given,
Radius R



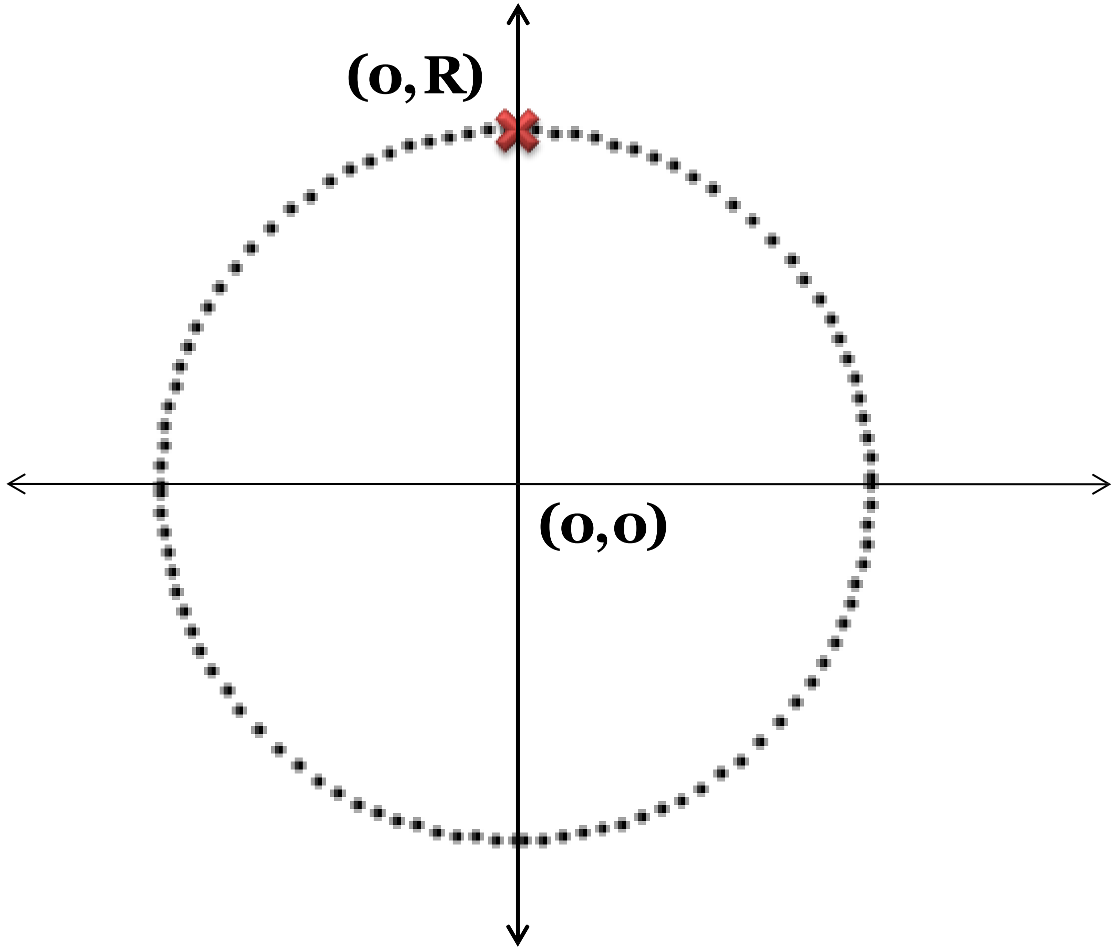


We have to develop an algorithm that generates this circumference

Assumptions

The first pixel of the circumference is plotted on (0, R)

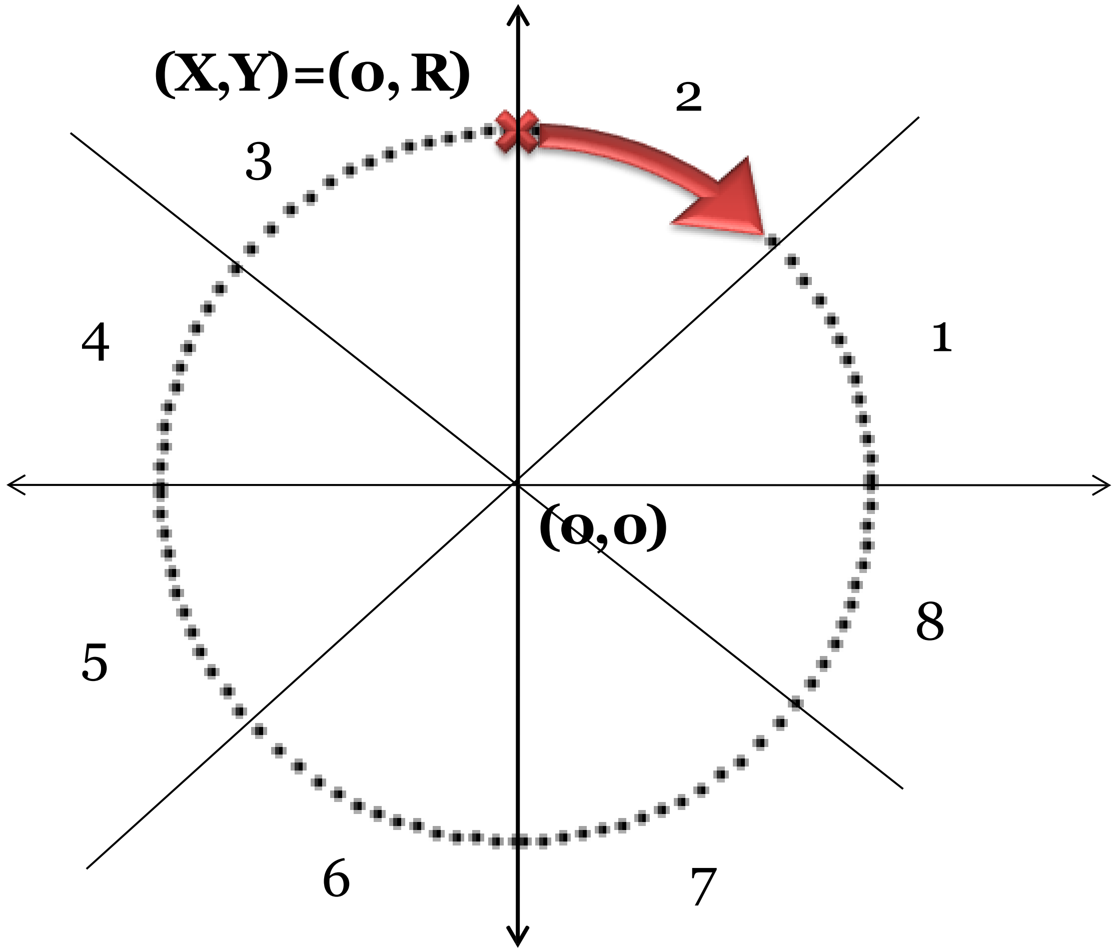
Given,
Radius R



Observation

The first pixel of the circumference is plotted on (0, R)

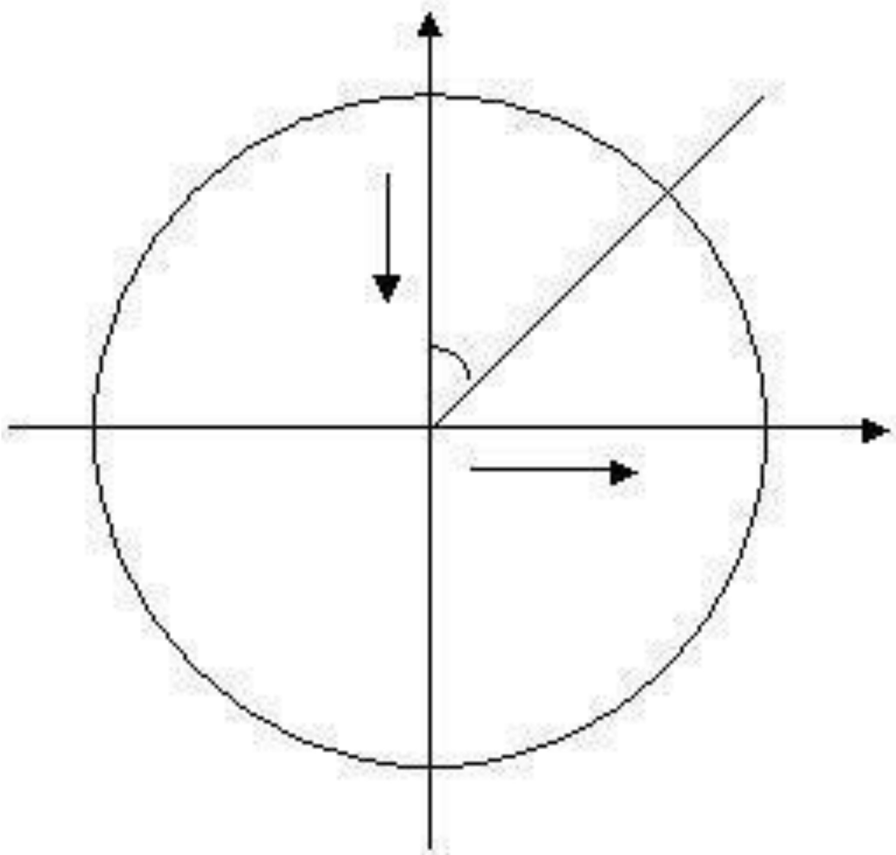
Then the plotting of next pixels starts clock-wise....



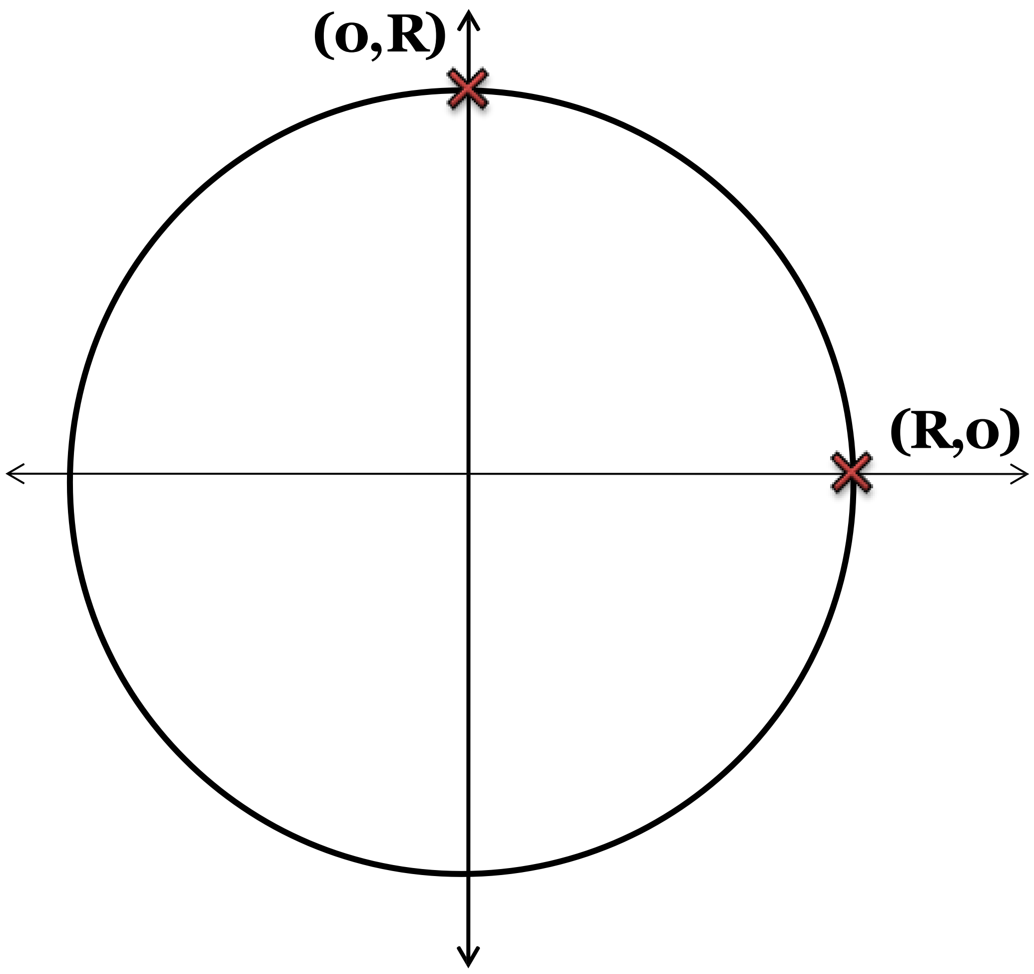
That means the plotting starts from (0,R) and moving into the 2nd Octant

Observation

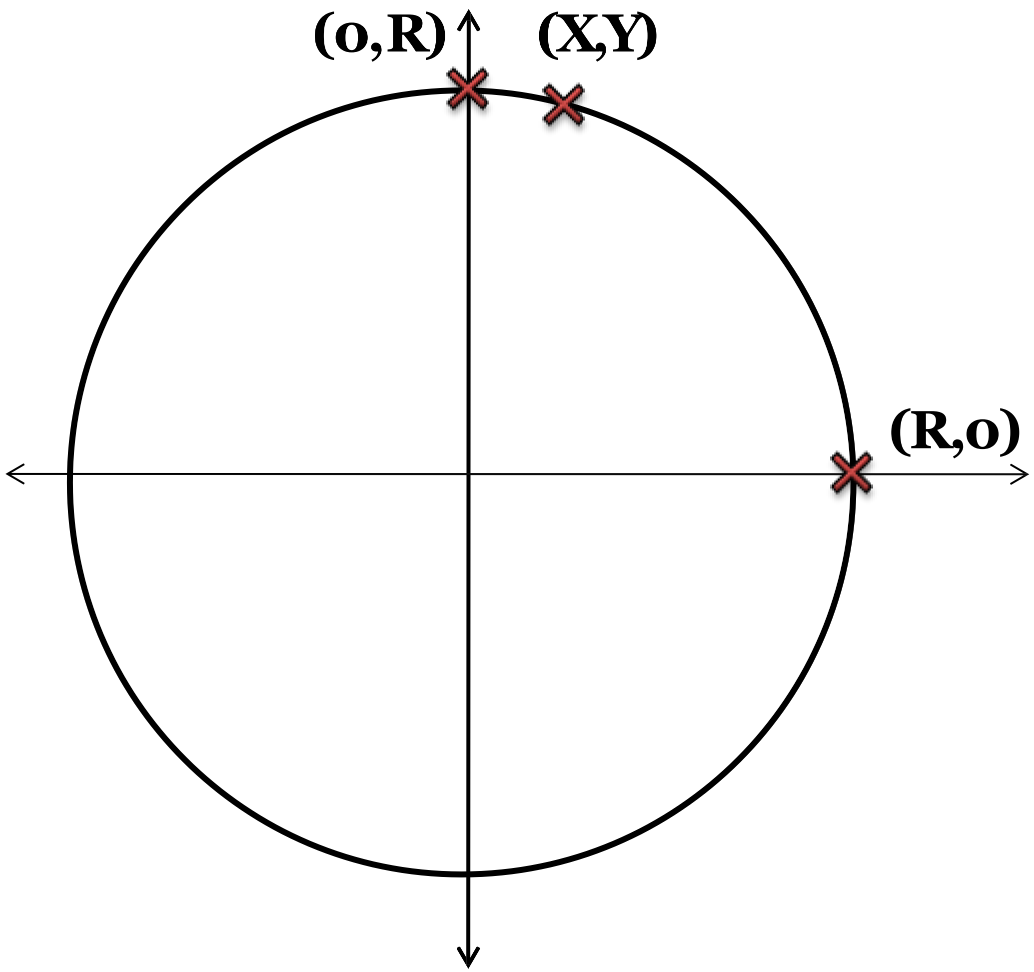
while moving through the 2nd octant, the X value is increasing and Y value is decreasing



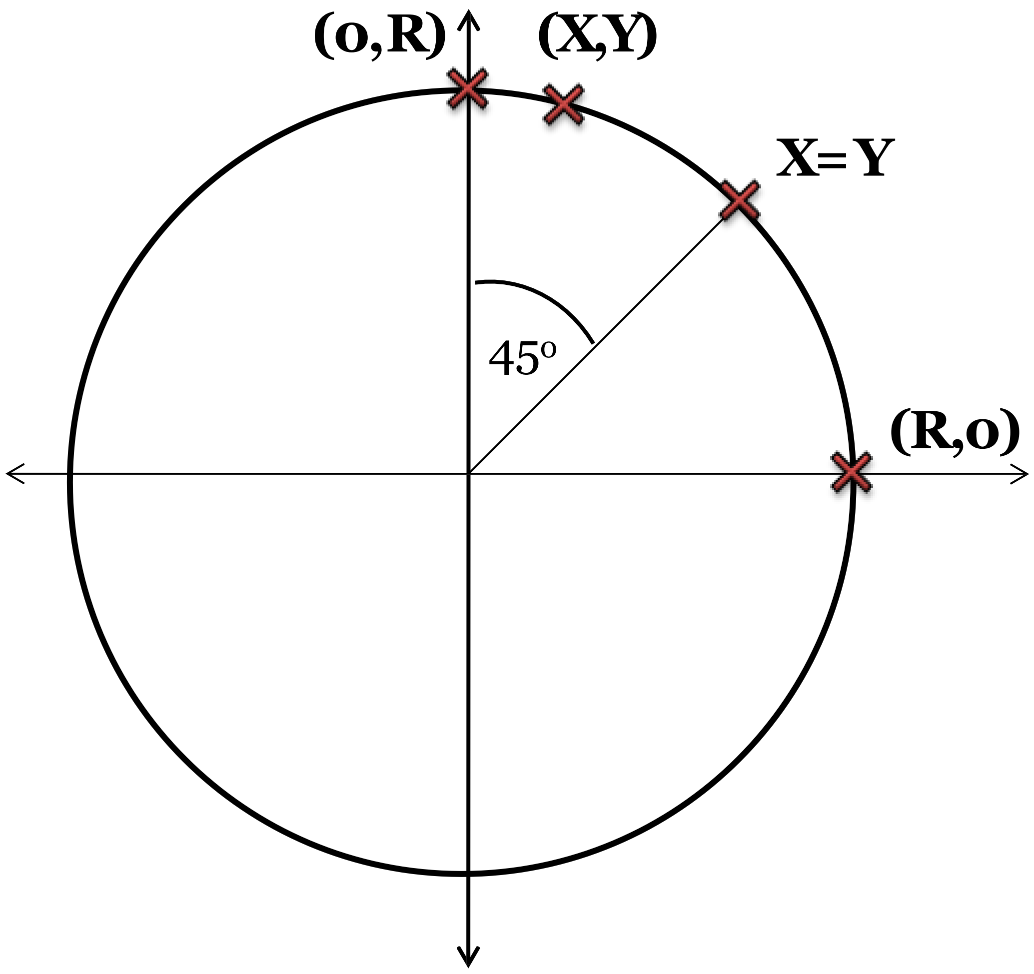
Observation



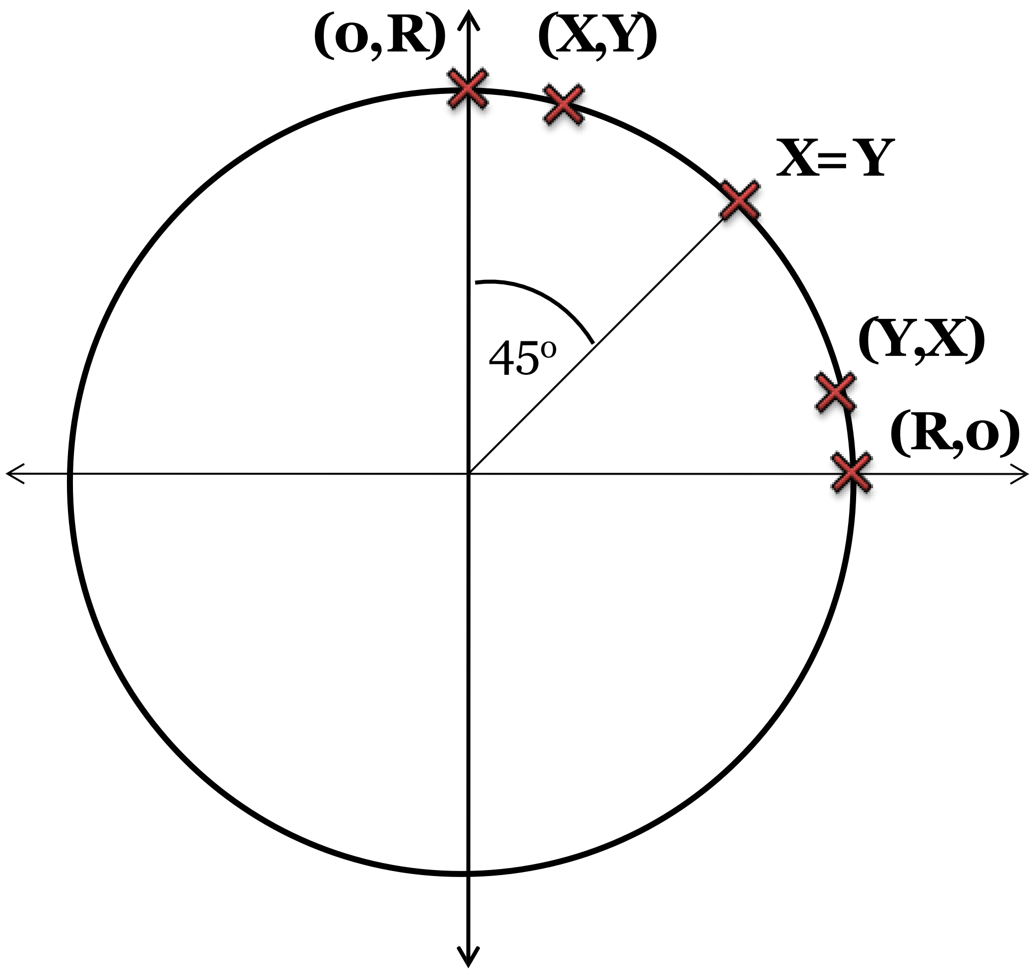
Observation



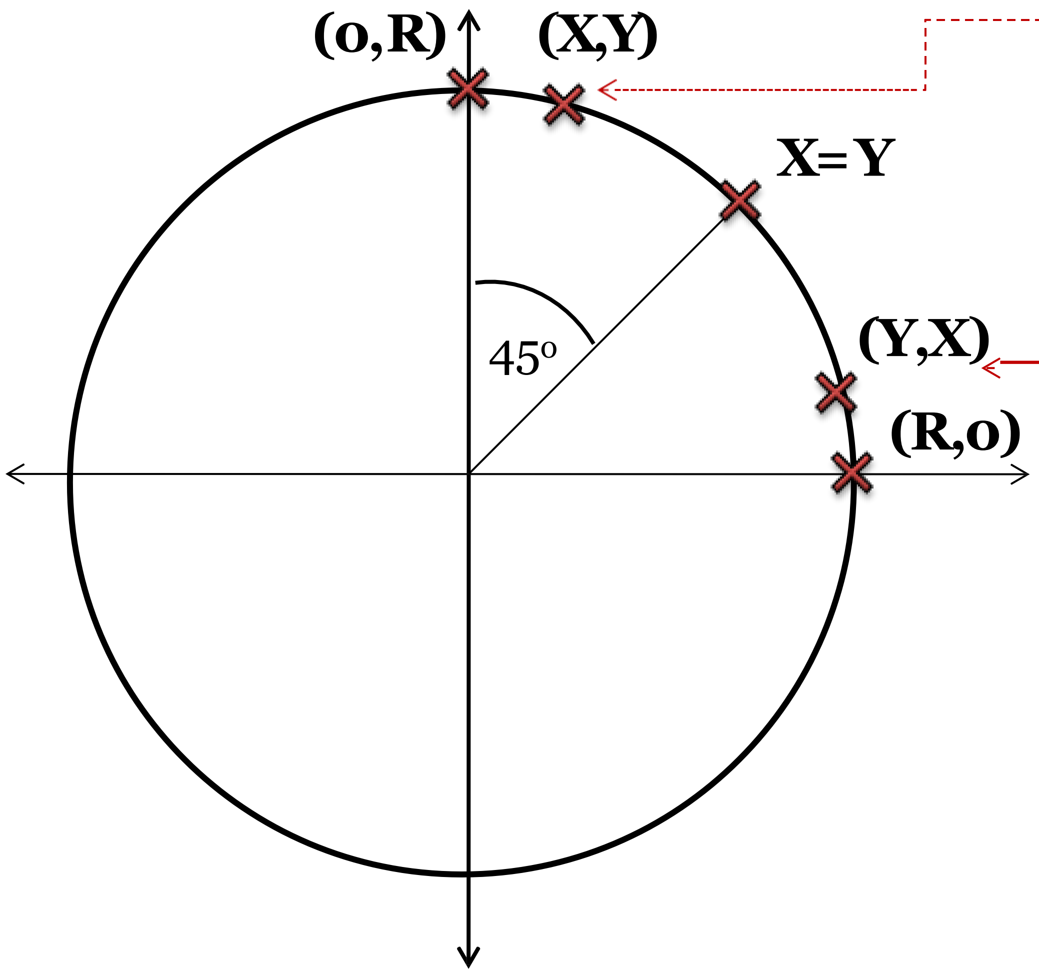
Observation



Observation



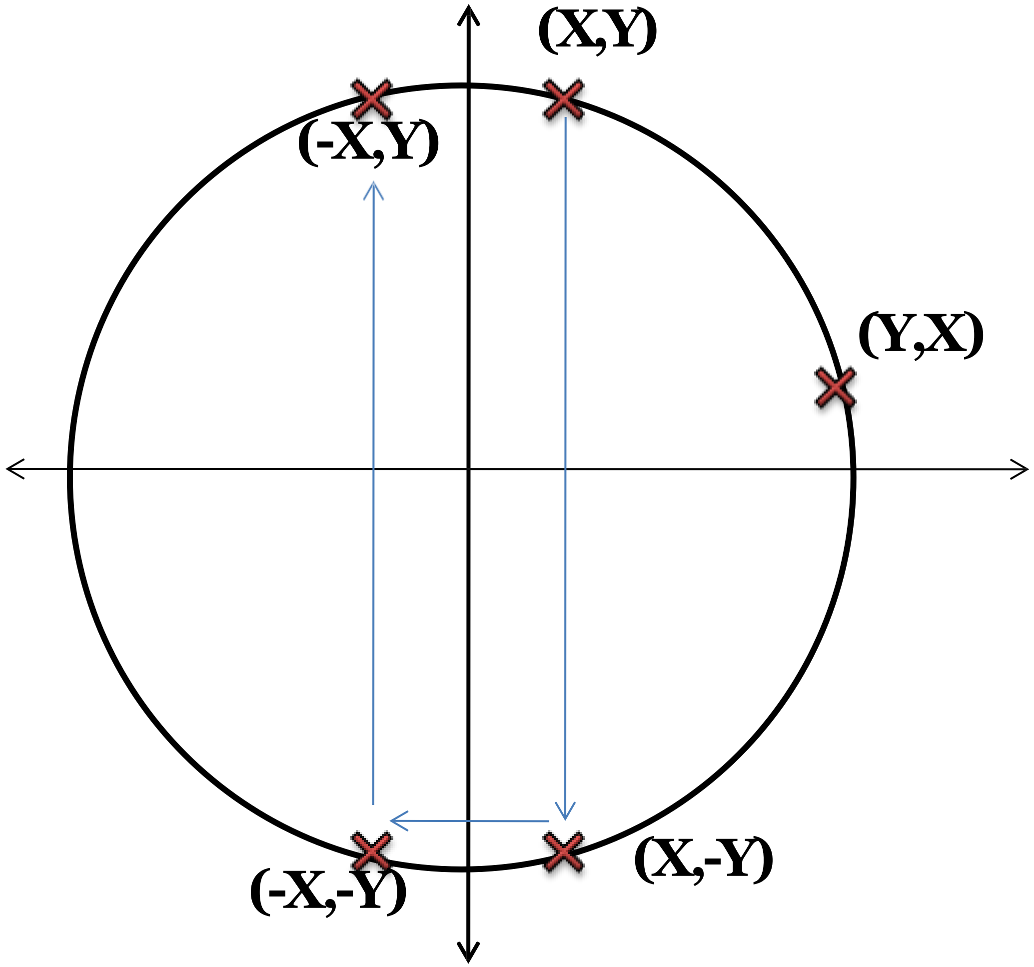
Observation



In 2^{nd} Octant : $\mathbf{X<Y}$

But, in 1^{st} Octant : $\mathbf{X>Y}$

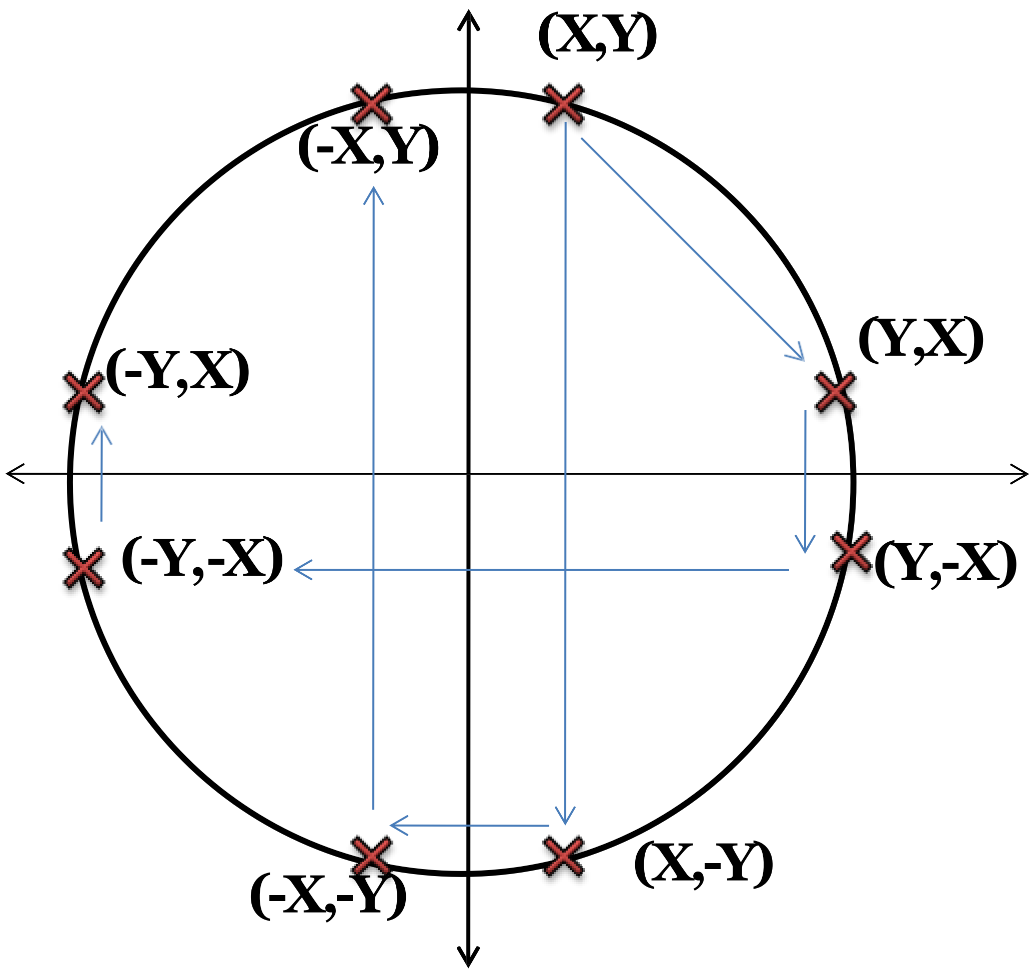
Observation



So, if we can obtain (X,Y) in 2nd octant, we can calculate the points-

- 7th Octant : (X,-Y)
- 6th Octant : (-X, -Y)
- 3rd Octant : (-X, Y)

Observation

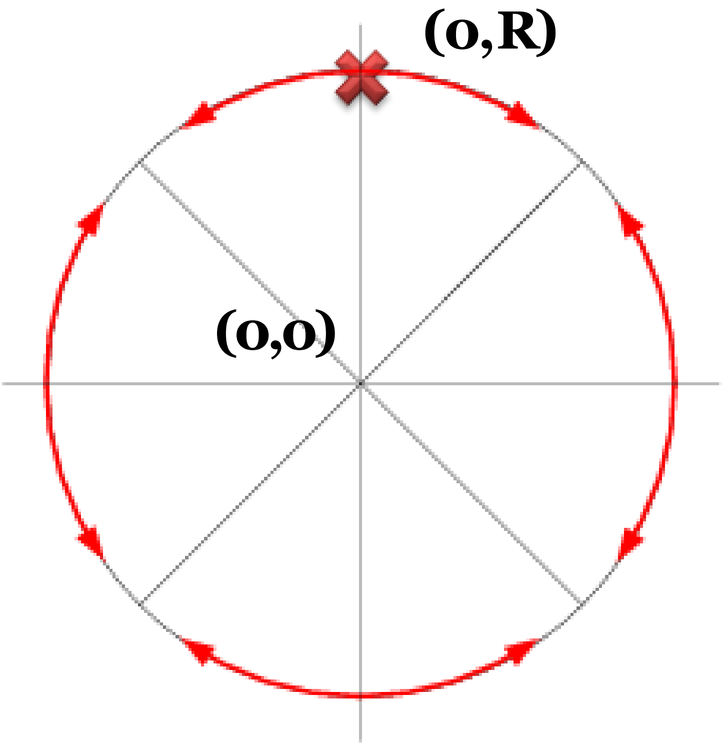


So, if we can obtain (X,Y) in 2nd octant, we can simply swap X and Y to get the points-

- 1st Octant : (Y,X)
- 8th Octant : (Y, -X)
- 5th Octant : (-Y,-X)
- 4th Octant : (-Y, X)

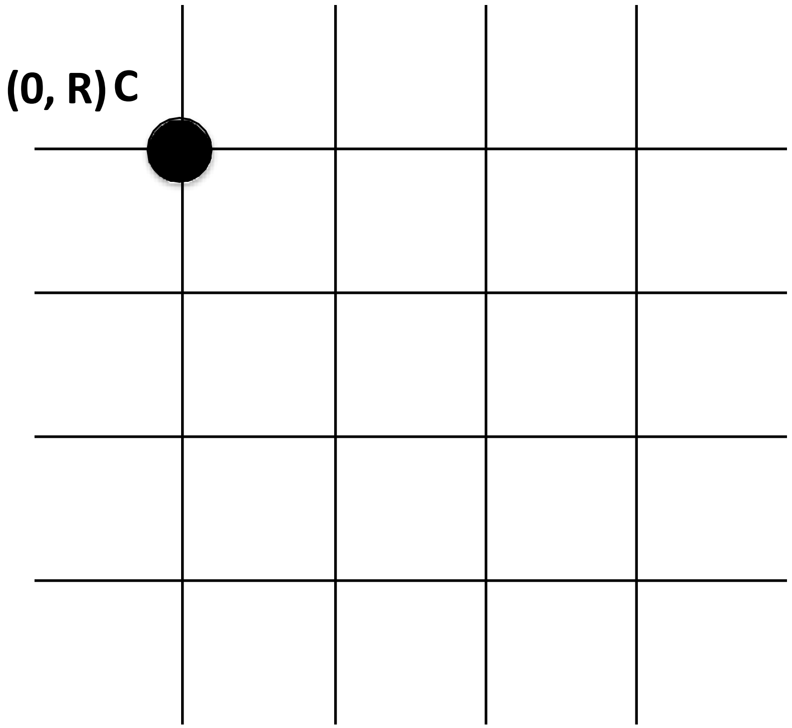
Drawing in all the octants

So, if we can obtain (X,Y) in 2nd octant, we can calculate the points in other 7 octants

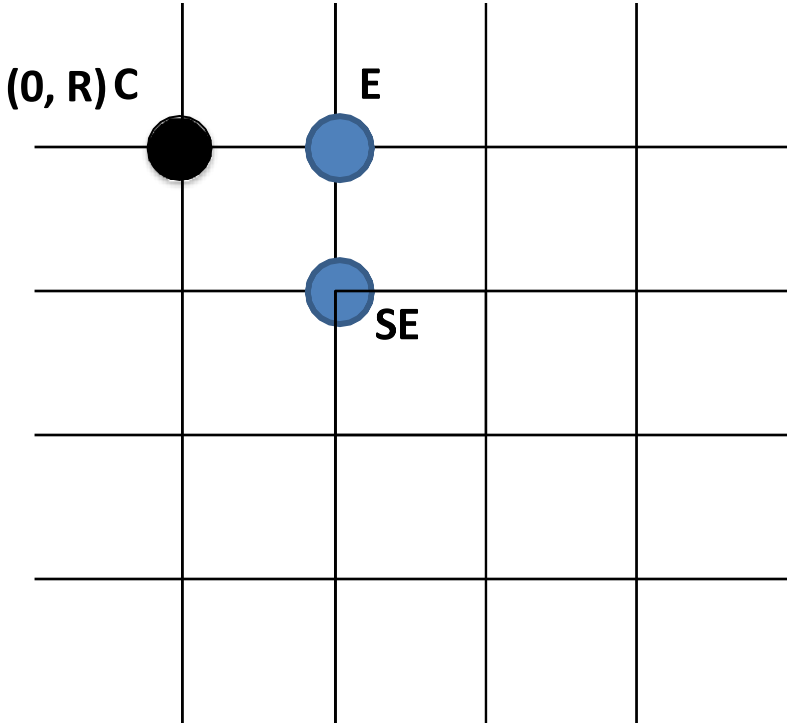


So, our target is to get the pixels of only 2nd octant of the circumference

Bresenham's Circle Drawing Algorithm: How it works

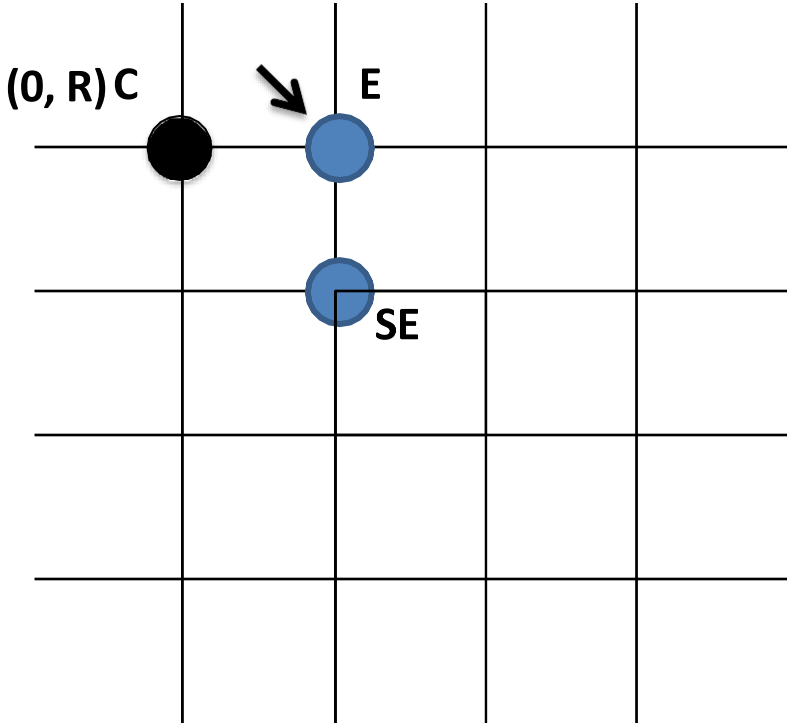


Bresenham's Circle Drawing Algorithm: How it works



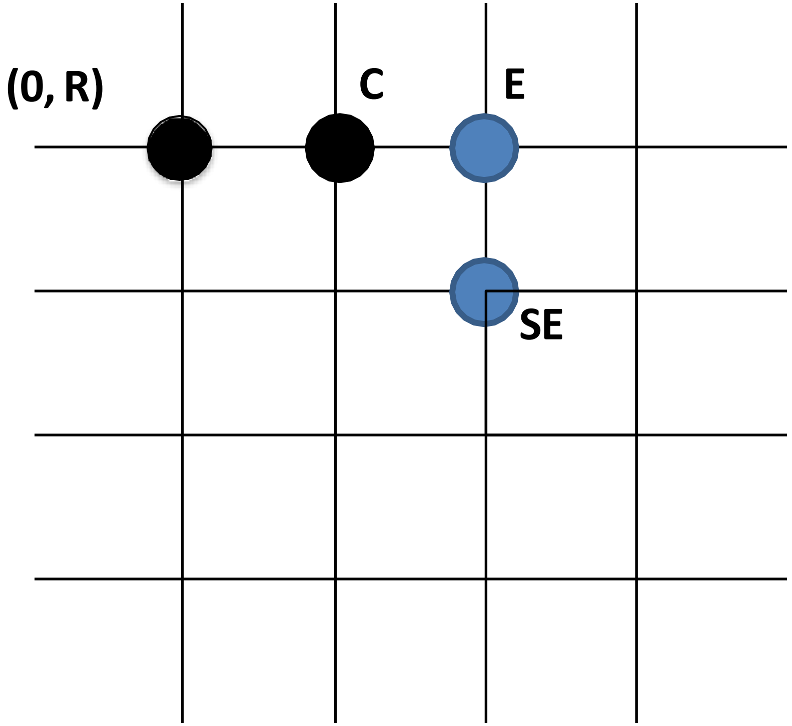
Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



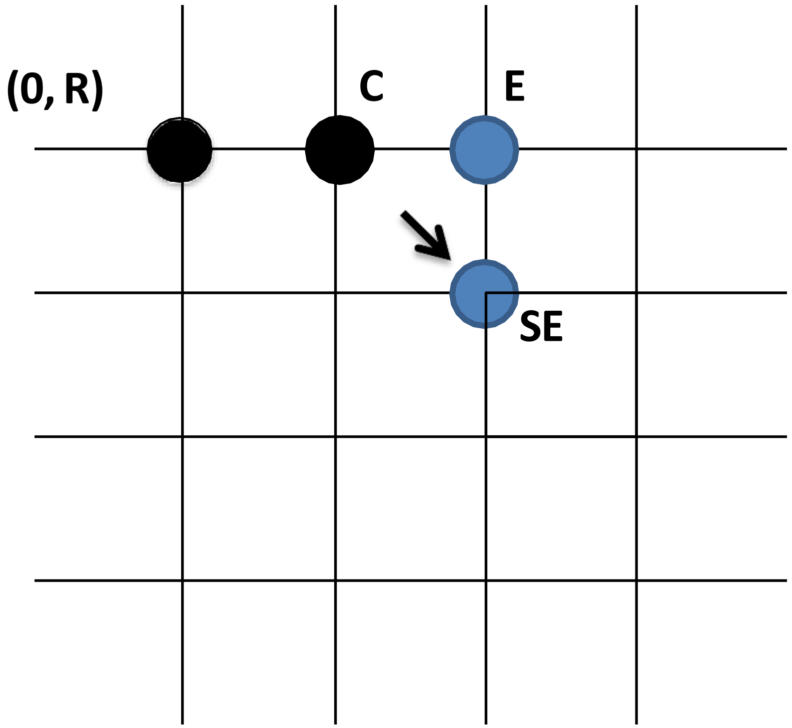
Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



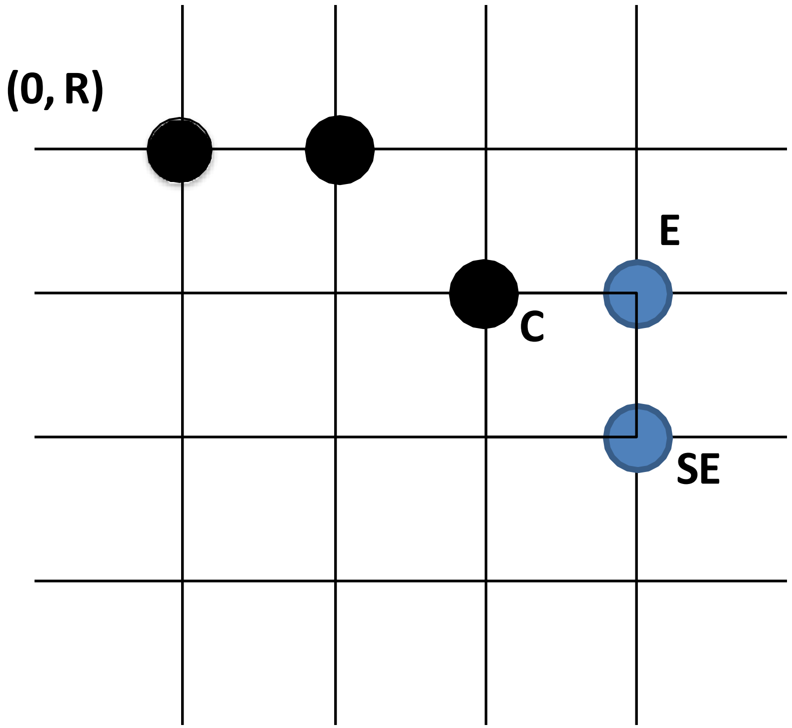
Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



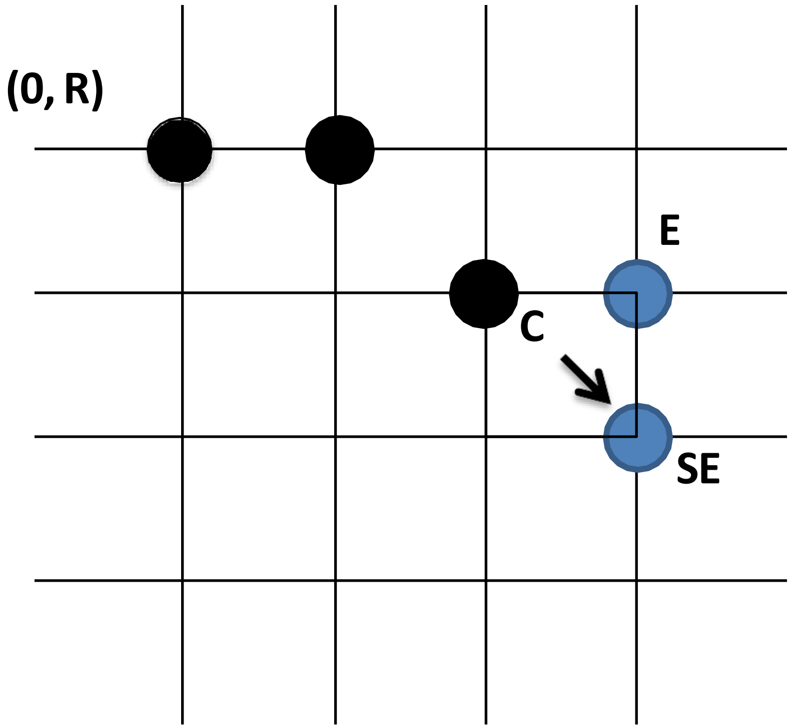
Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



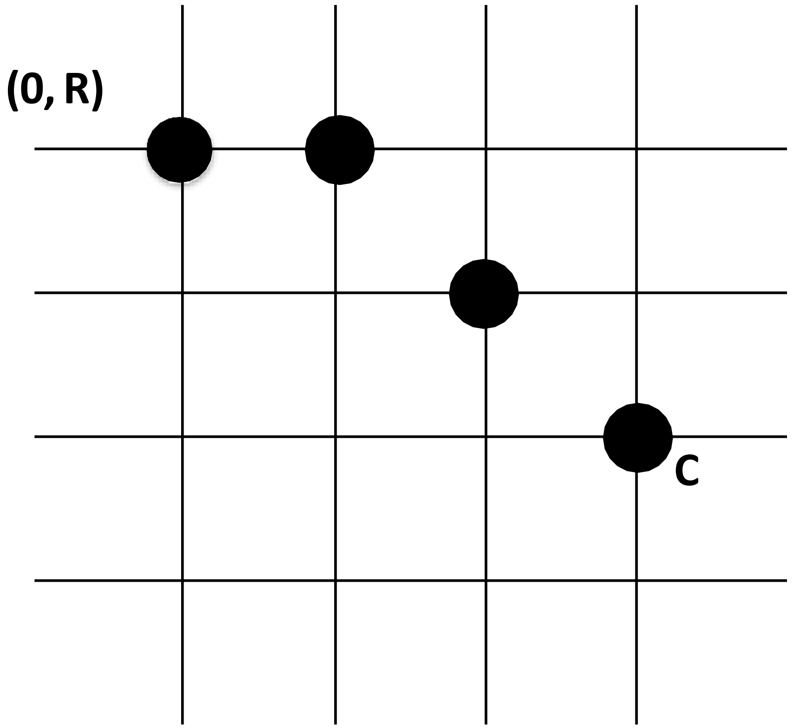
Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



Next pixel is chosen (from E or SE) to build the line successively

Bresenham's Circle Drawing Algorithm: How it works



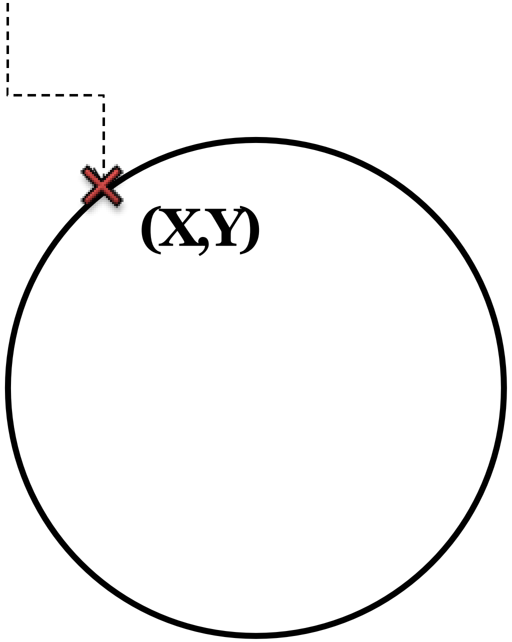
As we know that,
In 2nd Octant : **X < Y**
In 1st Octant : **X > Y**

We will stop when X > Y, that means when 2nd octant is completed

Equation of Circle and its function representation

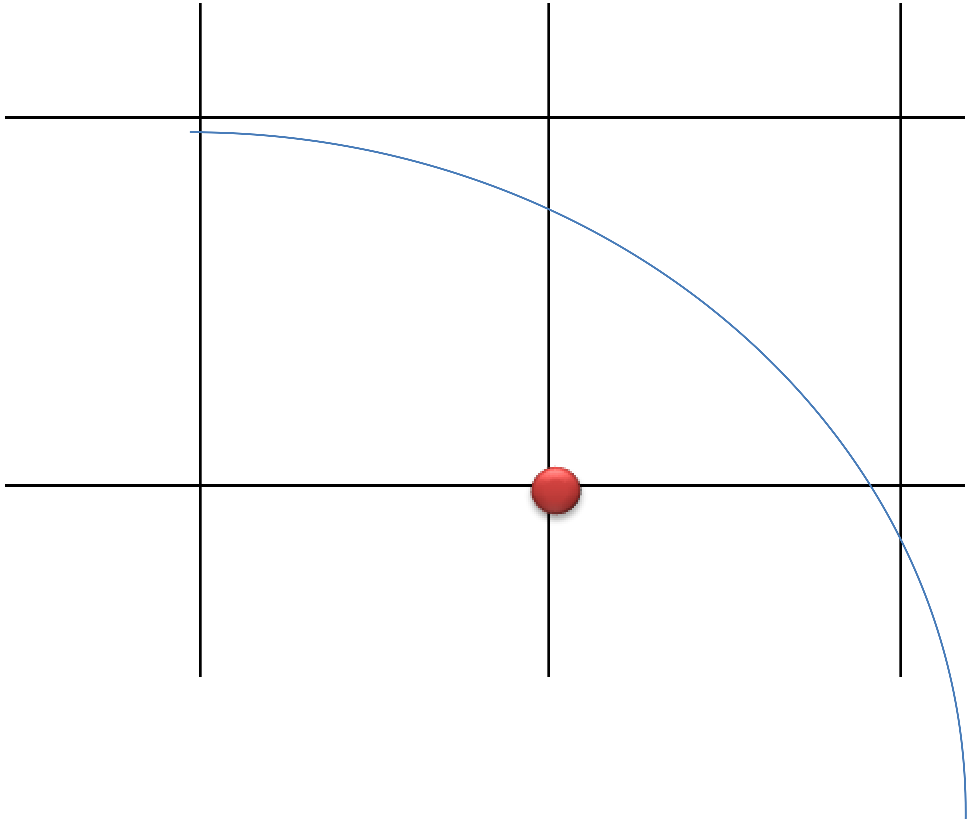
$$x^2 + y^2 = R^2$$

$$\boxed{F(x, y) = x^2 + y^2 - R^2 = 0}$$



Equation of Circle and its function representation

$$F(x, y) = x^2 + y^2 - R^2$$

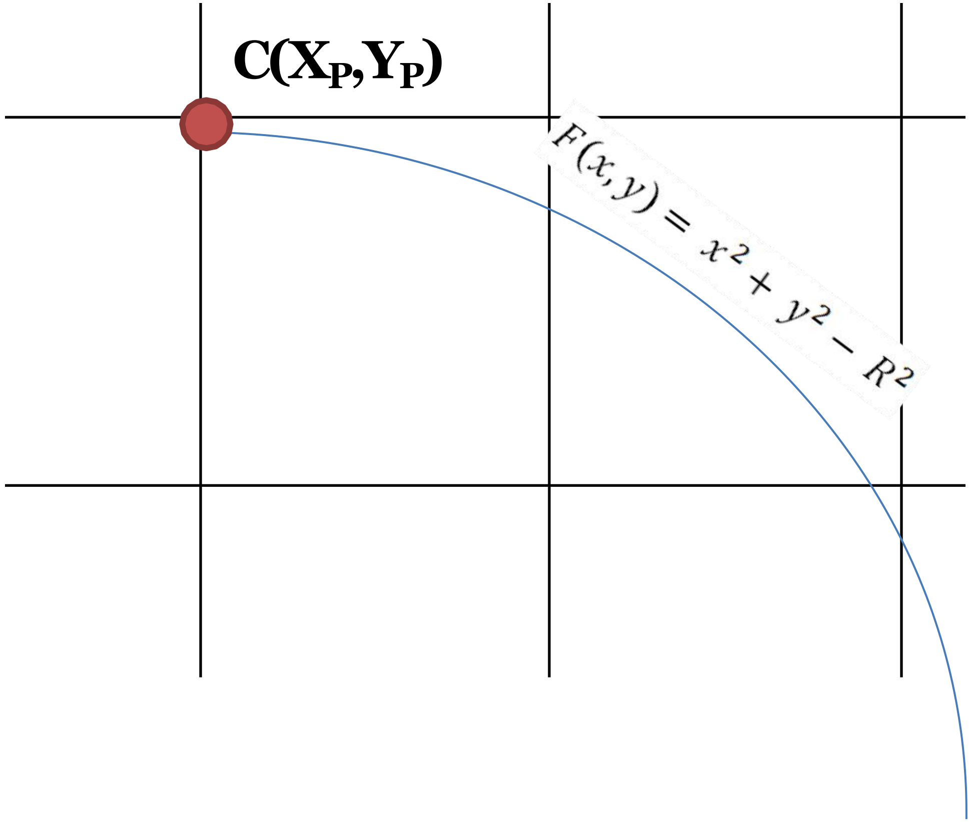


If **F(X,Y) = 0,** the point (X,Y) on the circle

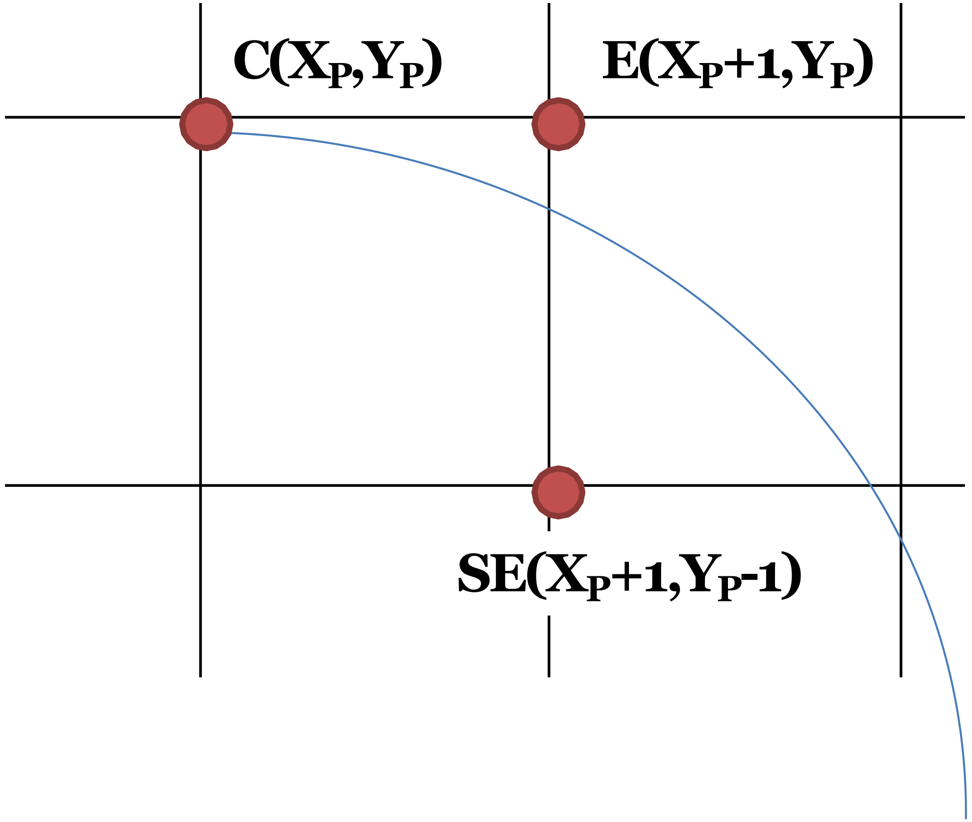
If **F(X, Y) > 0,** the point (X,Y) is outside the circle

If **F(X, Y) < 0,** the point (X,Y) is inside the circle

Selecting E or SE



Selecting E or SE

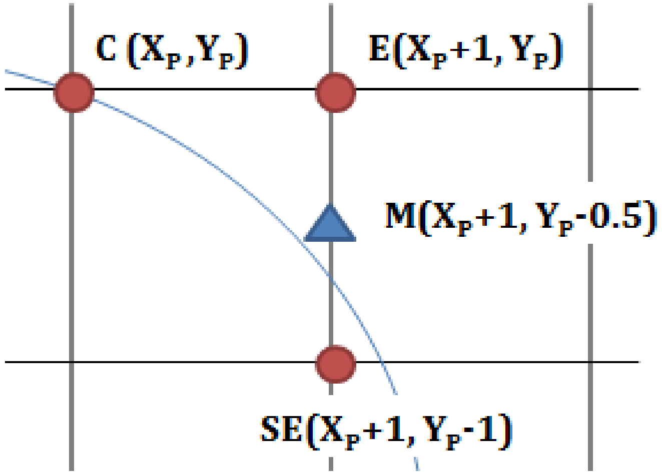


Selecting E or SE depends on closeness to the circumference.

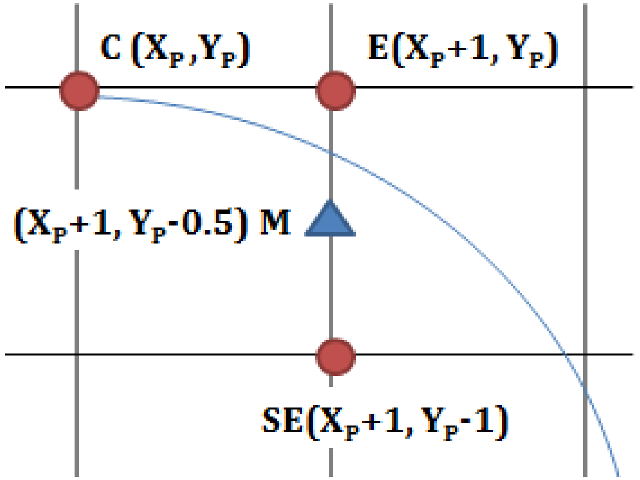
If E is closer to circumference, then E is selected

If SE is closer, then SE is selected

Selecting E or SE using Mid Point Criteria



If midpoint M is outside the circle, SE is closer to the circumference,
So, **SE** is selected



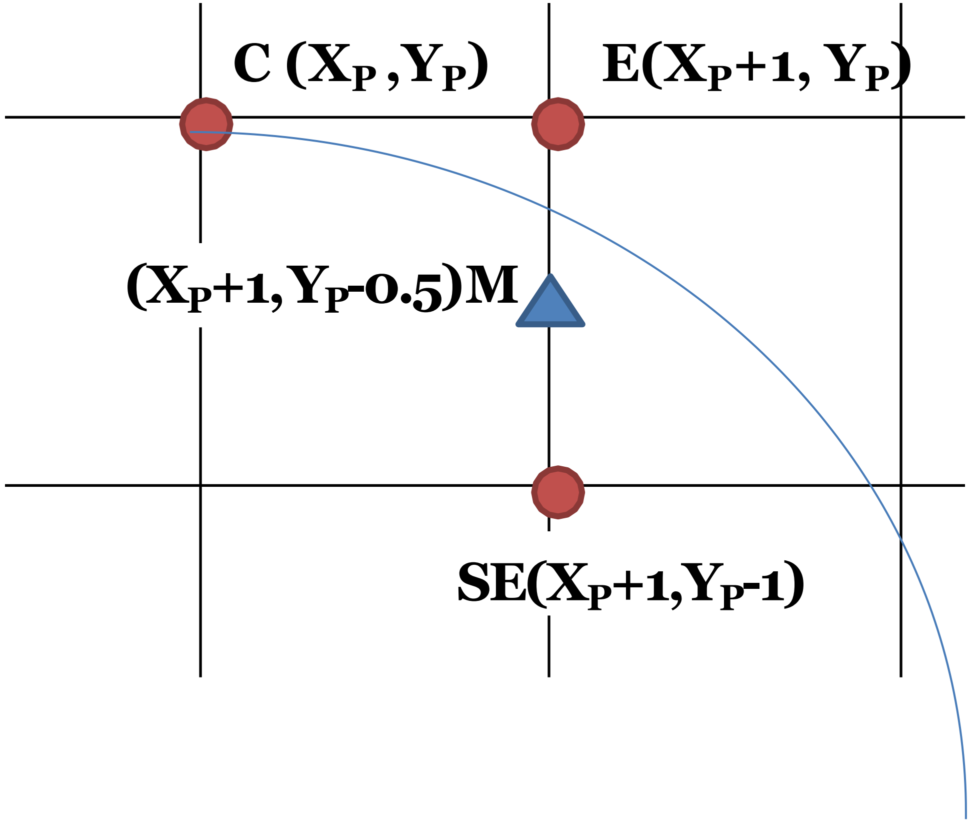
If midpoint M is inside the circle, E is closer to the circumference,
So, **E** is selected

Selecting E or SE using Mid Point Criteria

We know, $F(x,y) = x^2 + y^2 - R^2$

Lets put the mid point **M**'s coordinate in function F(X,Y)

$F(M) = F(X_P + 1, Y_P - 0.5) = (X_P + 1)^2 + (Y_P - 0.5)^2 - R^2$

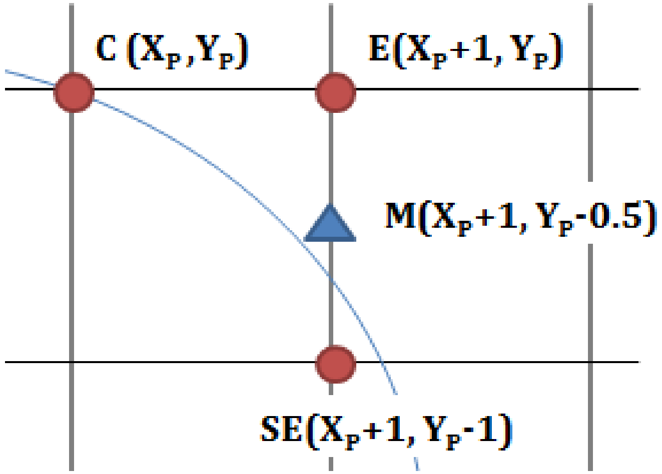


Lets store **F(M)** in a variable **d**

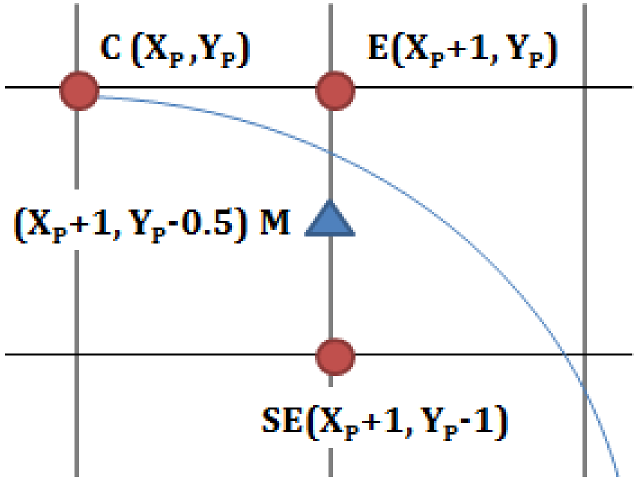
So, **d = F(M)**

d is called 'decision variable'

Selecting E or SE using Mid Point Criteria

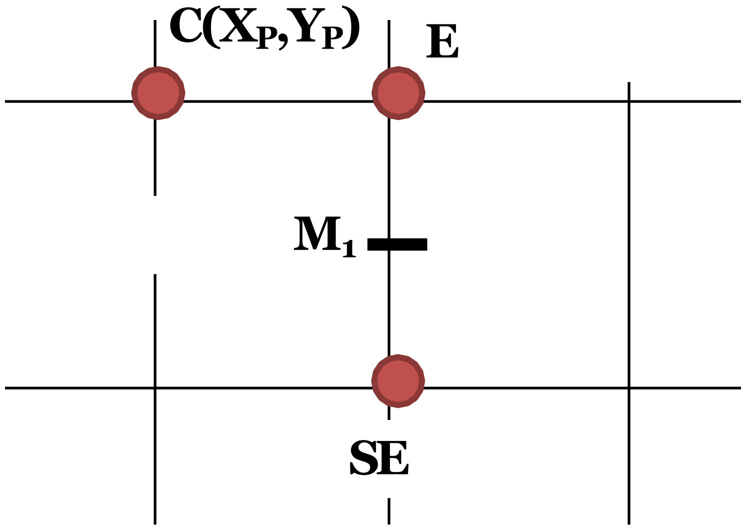


If **d >= o**, then midpoint M is outside the circle, SE is closer to the circumference,
So, **SE** is selected



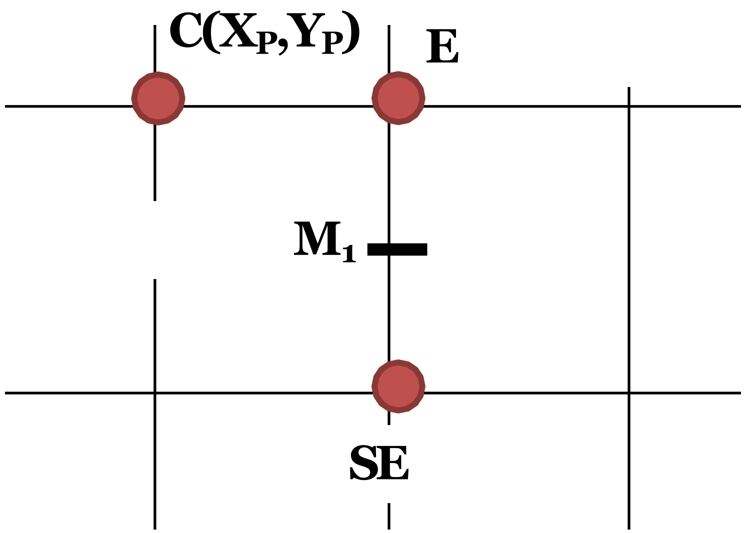
If **d < o**, then midpoint M is inside the circle, E is closer to the circumference,
So, **E** is selected

Bresenham's Mid Point Criteria : Successive Updating (for selecting E)



$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$

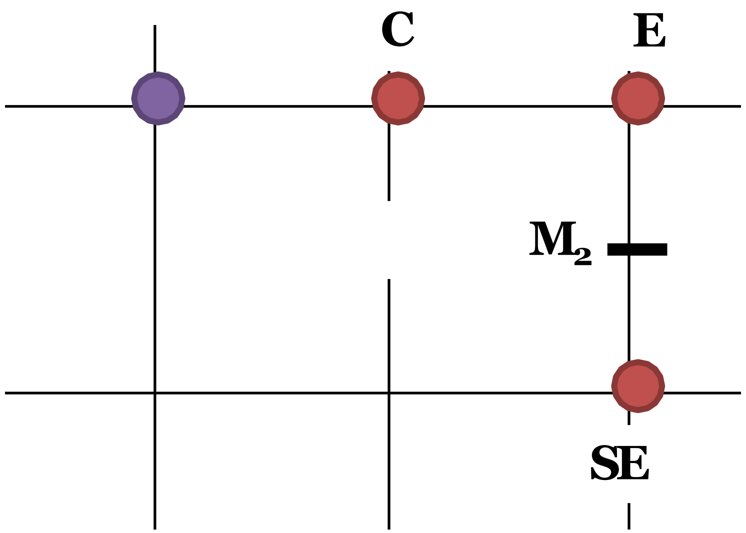
Bresenham's Mid Point Criteria : Successive Updating (for selecting E)



$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$

If $d_1 < 0$, $E(X_P = X_P+1, Y_P)$

Bresenham's Mid Point Criteria : Successive Updating (for selecting E)

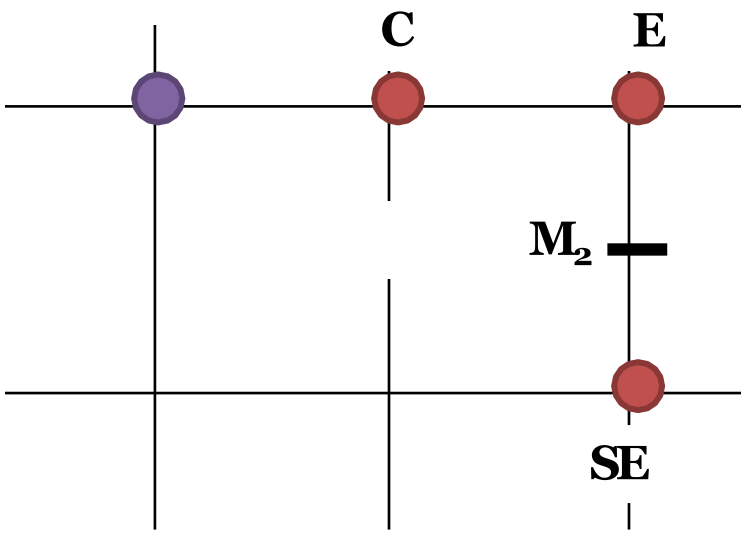


$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$

If $d_1 < 0$, $E(X_P = X_P+1, Y_P)$

$$d_2 = F(M_2)$$
$$= F(X_P+2,\ Y_P-0.5)$$
$$= (X_P+2)^2+(Y_P-0.5)^2-R^2$$
$$= X_P^2+4X_P+4+(Y_P-0.5)^2-R^2$$
$$= X_P^2+2X_P+1+(Y_P-0.5)^2-R^2+2X_P+3$$
$$= d_1+(2X_P+3)$$

Bresenham's Mid Point Criteria : Successive Updating (for selecting E)



$$d_1 = F(M_1)$$
$$= F(X_P+1, Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$

If $d_1 < 0$, $E(X_P = X_P+1, Y_P)$

$$d_2 = F(M_2)$$
$$= F(X_P+2, Y_P-0.5)$$
$$= (X_P+2)^2+(Y_P-0.5)^2-R^2$$
$$= X_P^2+4X_P+4+(Y_P-0.5)^2-R^2$$
$$= X_P^2+2X_P+1+(Y_P-0.5)^2-R^2+2X_P+3$$
$$= d_1+(2X_P+3)$$

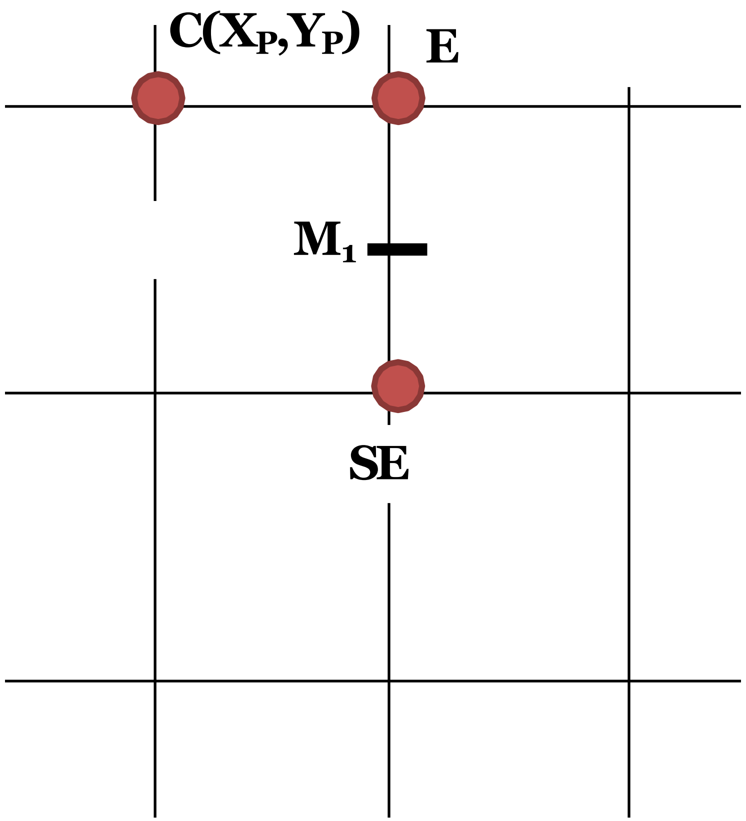
Bresenham's Mid Point Criteria : Successive Updating (for selecting E)

Every iteration after **selecting E**, we can successively update our decision variable with-

$$\mathbf{d_{NEW} = d_{OLD} + (2X_P + 3)}$$

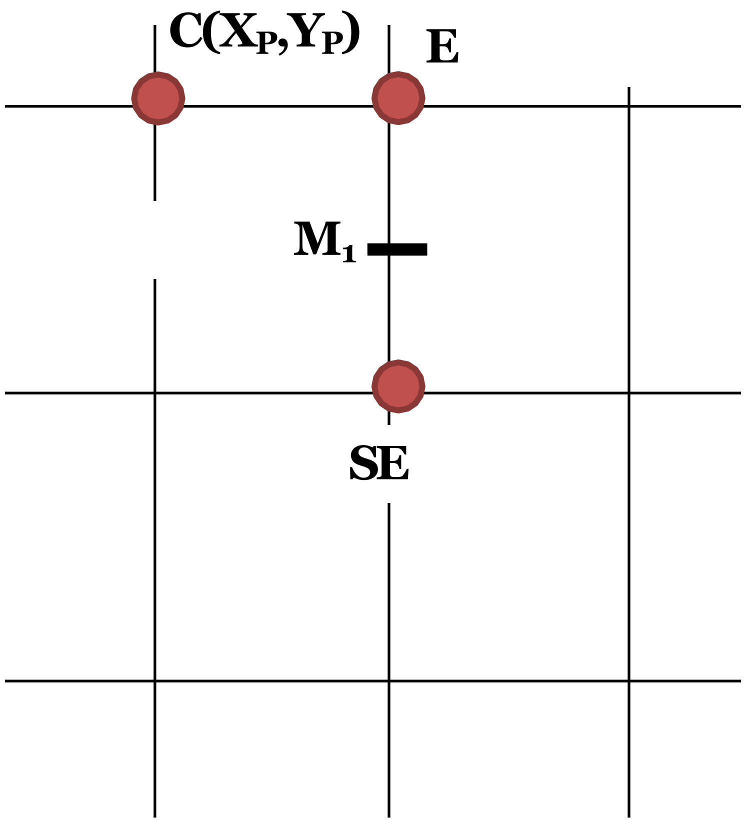
Bresenham's Mid Point Criteria : Successive Updating (for selecting SE)

$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$



Bresenham's Mid Point Criteria : Successive Updating (for selecting SE)

$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2 + (Y_P-0.5)^2 - R^2$$

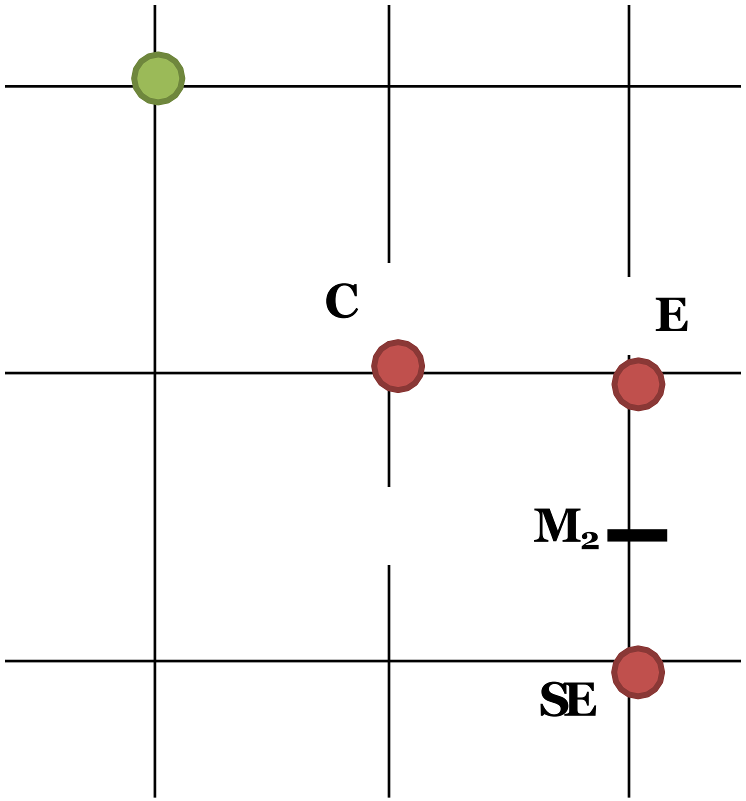


If $d_1 >= 0, SE(X_P = X_P+1, Y_P-1)$

Bresenham's Mid Point Criteria : Successive Updating (for selecting SE)

$$d_1 = F(M_1)$$
$$= F(X_P+1,\ Y_P-0.5)$$
$$= (X_P+1)^2+(Y_P-0.5)^2-R^2$$

If $d_1 >= 0$, $SE(X_P = X_P+1, Y_P-1)$



$$d_2 = F(M_2)$$

…. DIY….

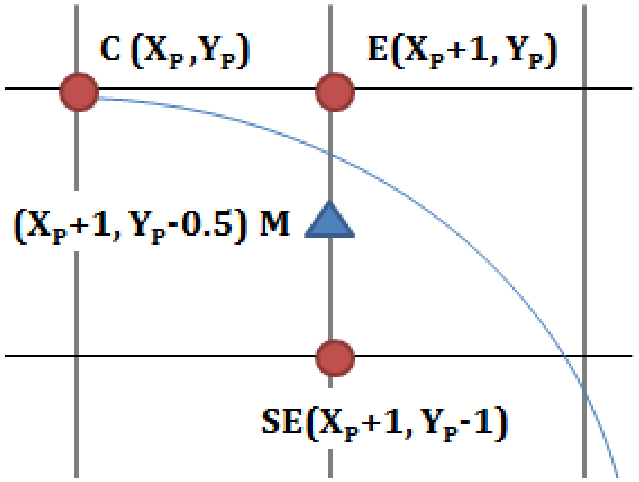
$$= d_1 + (2X_P - 2Y_P + 5)$$

Bresenham's Mid Point Criteria : Successive Updating (for selecting SE)

Every iteration after **selecting NE**, we can successively update our decision variable with-

$$\mathbf{d_{NEW} = d_{OLD} + (2X_P - 2Y_P + 5)}$$

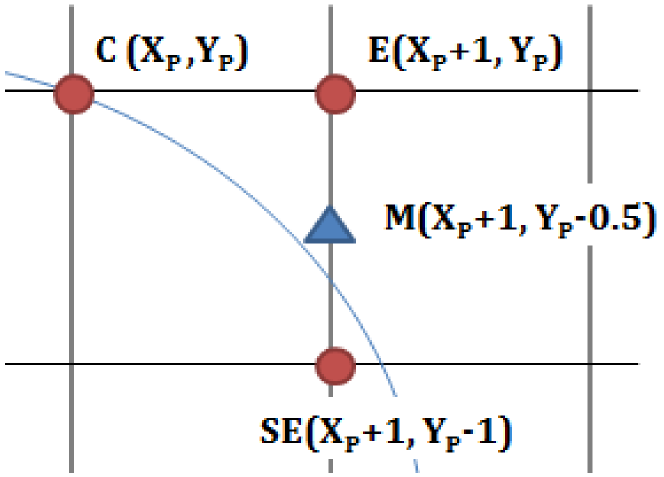
Bresenham's Mid Point Criteria : Successive Updating (summary)



If **d < 0**, then midpoint M is inside the circle, E is closer to the circumference,
So, **E** is selected and do-

$d = d + \Delta E$

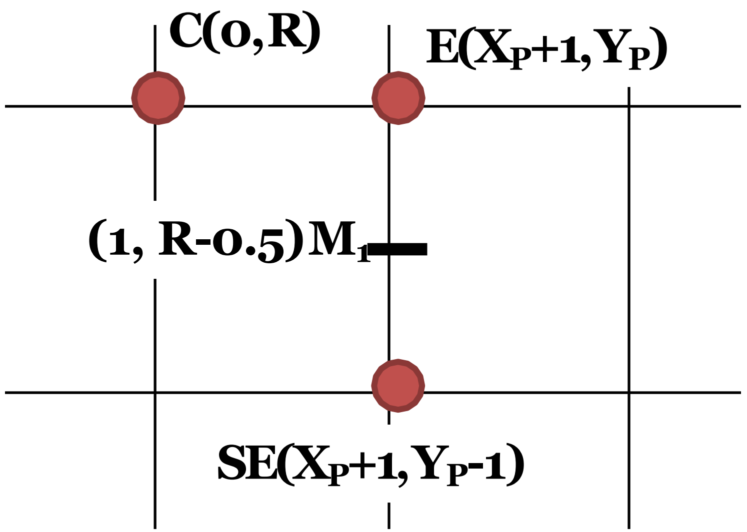
Where, $\Delta E = 2X_P + 3$



If **d >= 0**, then midpoint M is outside the circle, SE is closer to the circumference,
So, **SE** is selected and do-

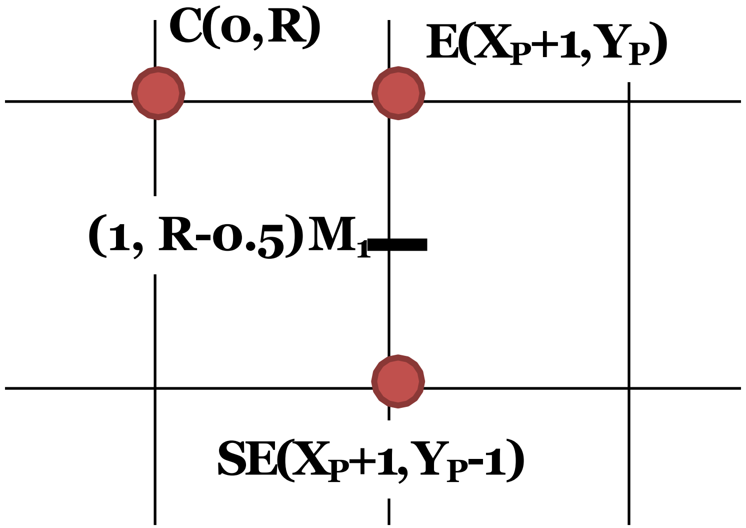
$d = d + \Delta SE$

Where, $\Delta SE = 2X_P - 2Y_P + 5$



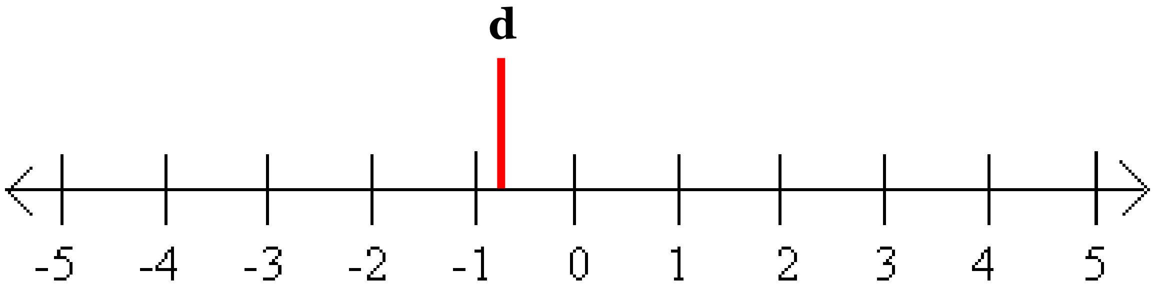
$$d_{INIT} = F(M_1)$$
$$= F(1, R-0.5)$$
$$= (1)^2 + (R-0.5)^2 - R^2$$
$$= 1 + R^2 - R + 0.25 - R^2$$
$$= 1.25 - R$$

Initialization



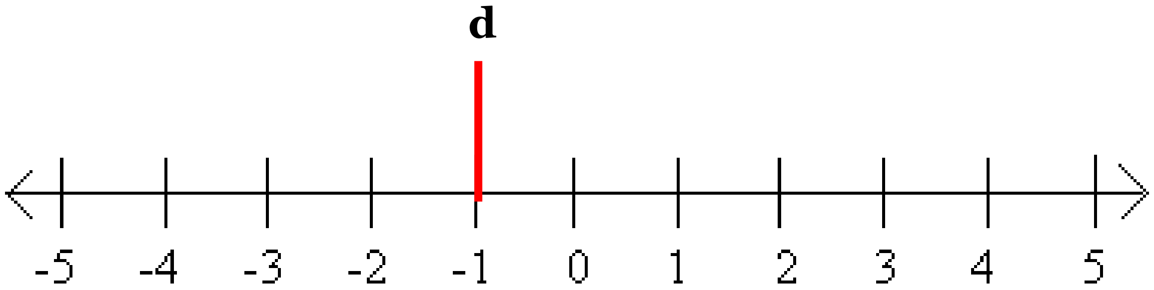
$$
\begin{aligned}
d_{INIT} &= F(M_1) \\
&= F(1, R-0.5) \\
&= (1)^2 + (R-0.5)^2 - R^2 \\
&= 1 + R^2 - R + 0.25 - R^2 \\
&= \color{red}{1.25} - R \\
&\color{red}{\approx 1-R}
\end{aligned}
$$

Initialization



R = 2

d = **1.25** – R = –0.75



R = 2

d = **1** – R = –1

So, finally…..

$\mathbf{d_{INIT} = 1 - R}$

If $\mathbf{d < 0}$, then **E** is selected, $\mathbf{d = d + \Delta E}$

If $\mathbf{d >= 0}$, then **SE** is selected, $\mathbf{d = d + \Delta SE}$

Where,

$\Delta E = 2X_P + 3$
$\Delta SE = 2X_P - 2Y_P + 5$

Algorithm

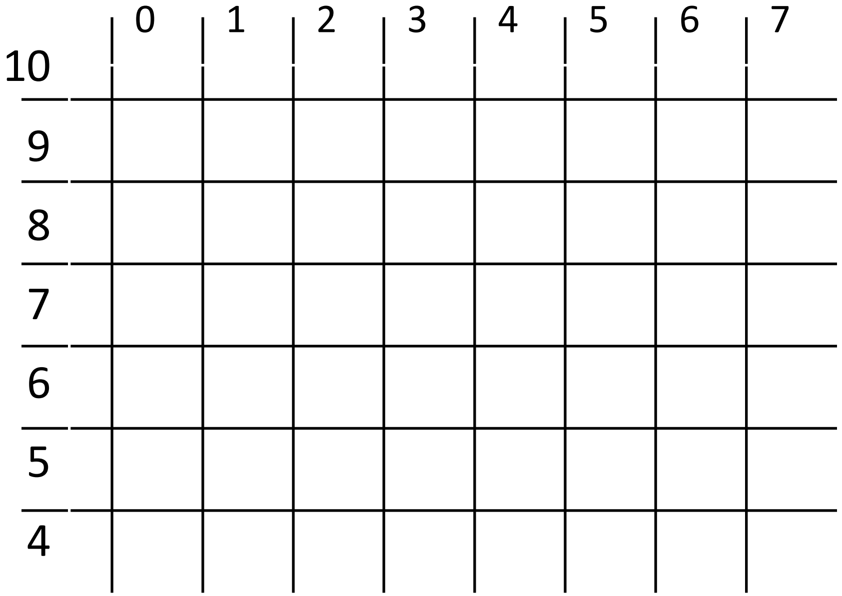
```
void MidpointCircle(int radius)
{
    int x = 0;
    int y = radius ;
    int d = 1 − radius ;
    CirclePoints(x, y);
    while (y > x)
    {
        if (d < 0)   /* Select E*/
                d = d + 2 * x + 3;
        else
        {       /* Select SE*/
            d = d + 2 * ( x − y ) + 5;
            y = y − 1;
        }
     x = x + 1;
     CirclePoints(x, y);
    }
}
```

Algorithm

```
void MidpointCircle(int radius)
{
    int x = 0;
    int y = radius ;
    int d = 1 - radius ;
    CirclePoints(x, y);
    while (y > x)
    {
        if (d < 0)   /* Select E*/
                d = d + 2 * x + 3;
        else
        {     /* Select SE*/
            d = d + 2 * ( x - y ) + 5;
            y = y - 1;
        }
     x = x + 1;
     CirclePoints(x,y);
    }
}
```

```
CirclePoints (x,y)
    Plotpoint(x,y) ;
    Plotpoint (x,-y) ;
    Plotpoint(-x,y) ;
    Plotpoint(-x, -y) ;
    Plotpoint(y,x) ;
    Plotpoint(y, -x) ;
    Plotpoint(-y, x) ;
    Plotpoint( -y, -x) ;
end
```

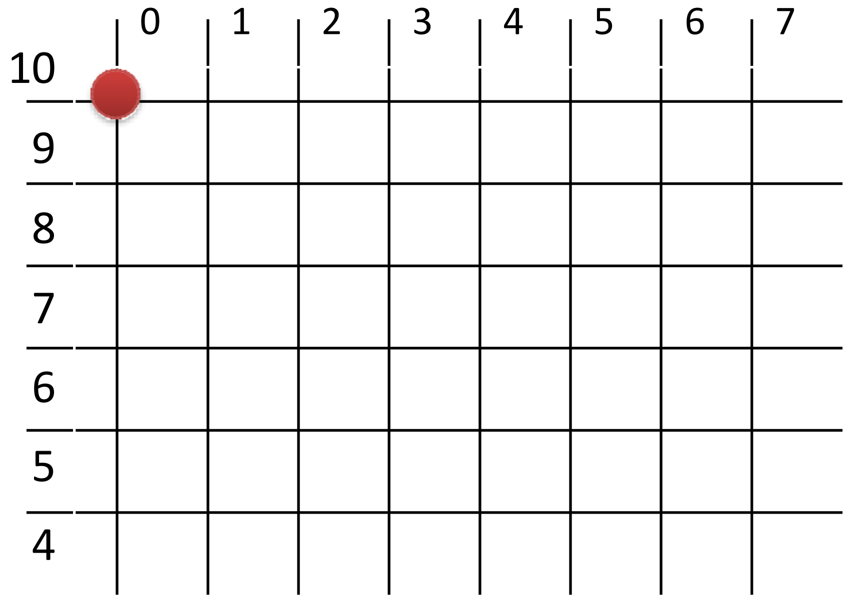
Example



Given:

Radius , R = 10

Example



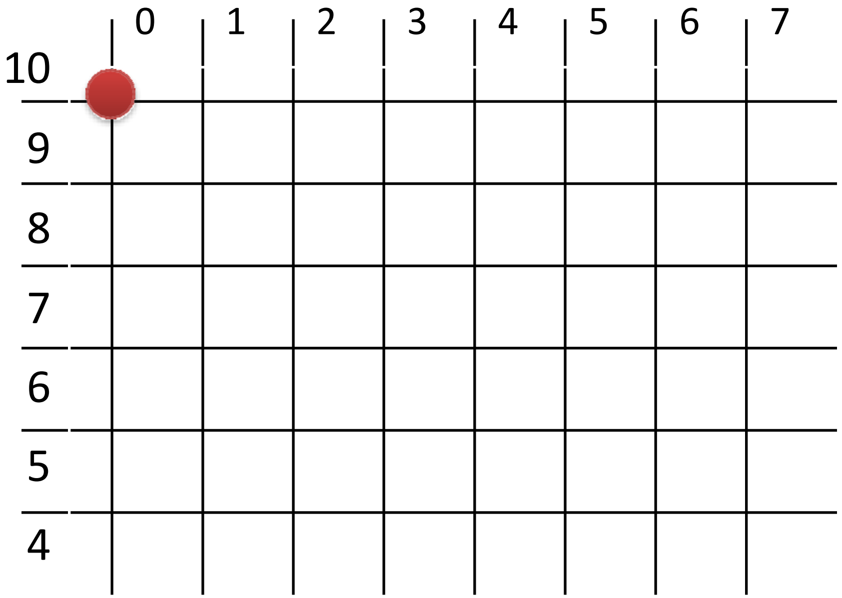
Given:

Radius , R=10

(x,y)=(0,10)

h=1 –R= -9

Example



Given:

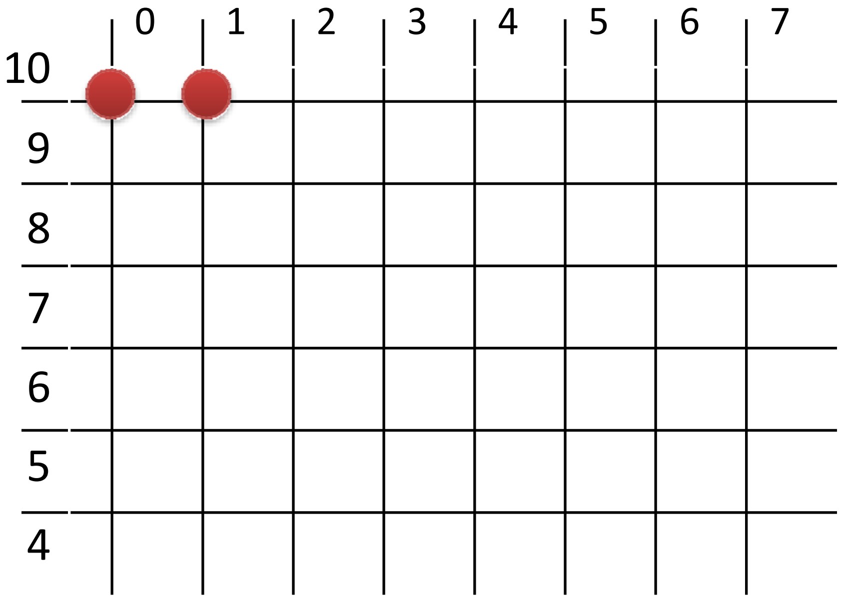
Radius , R=10

(x,y)=(0,10)

h =1 –R=-9

K	1						
2x	0						
2y	20						
h							
(x,y)							

Example



<u>Given:</u>

Radius , R=10

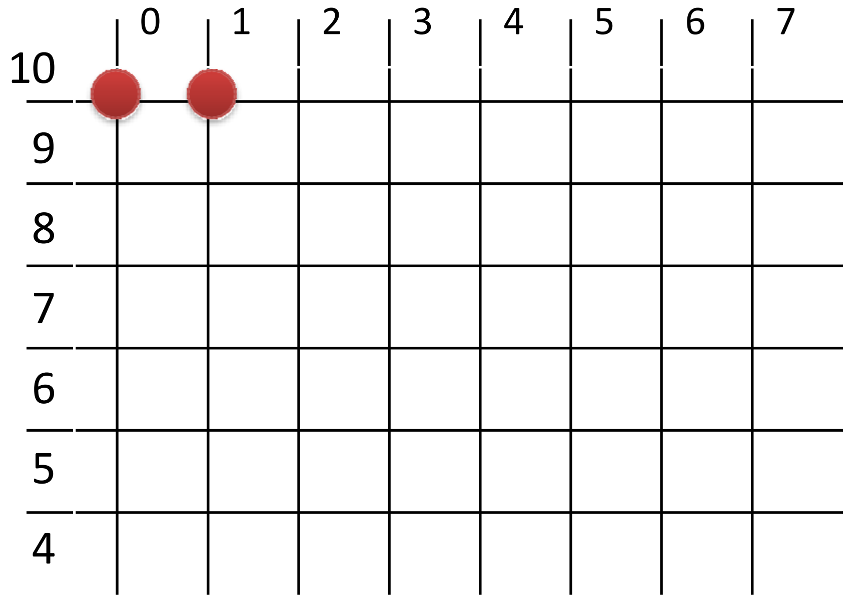
(x,y)=(0,10)

h =1 –R=-9

K	1						
2x	0						
2y	20						
h							
(x,y)	E(1,10)						

h <=0,E

Example



<u>Given:</u>

Radius , R = 10

(x,y)=(0,10)

h =1 –R= -9

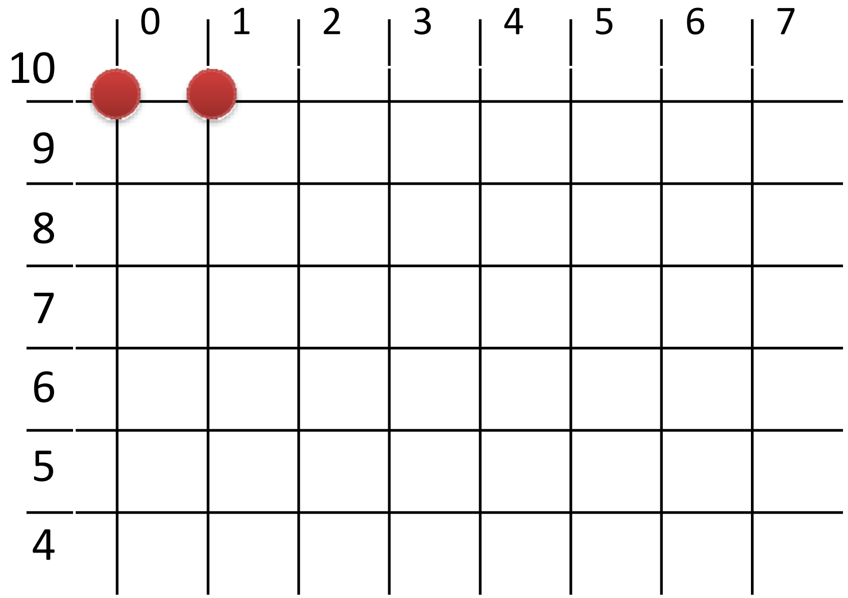
h =h+ΔE=h+2x+3

=-9+0+3

=-6

K	**1**						
2x	0						
2y	20						
h	-6						
(x,y)	E(1,10)						

Example



Given:

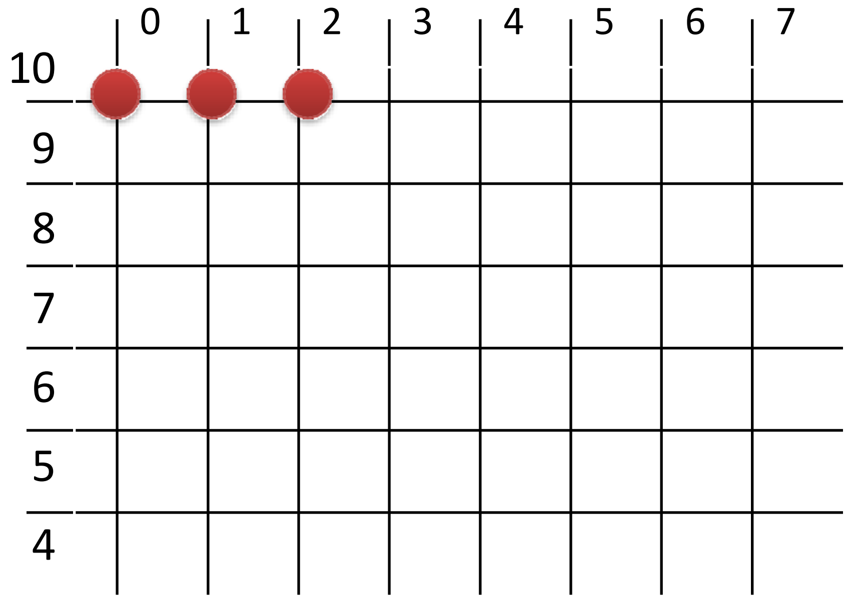
Radius , R=10

(x,y)=(0,10)

h =1 –R=-9

K	1	2					
2x	0	2					
2y	20	20					
h	-6						
(x,y)	E(1,10)						

Example



Given:

Radius , R=10

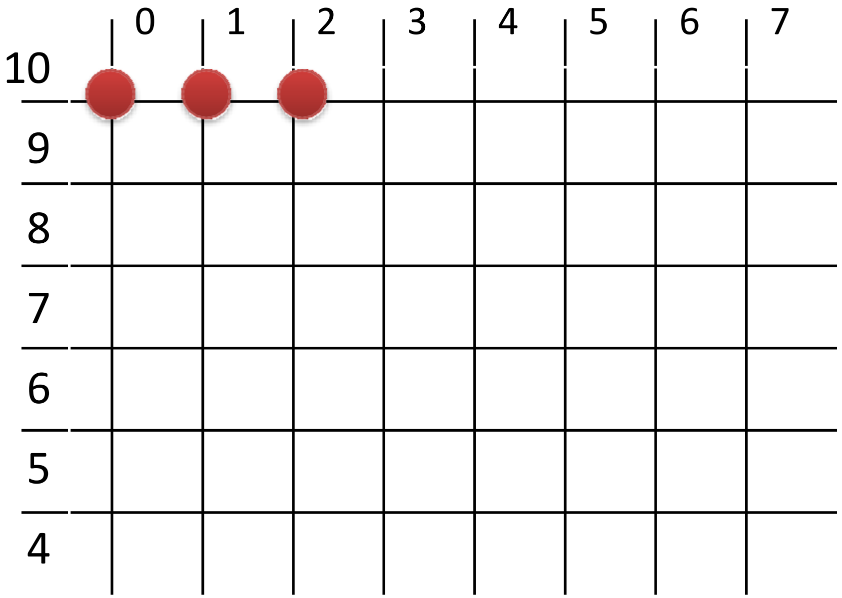
(x,y)=(0,10)

h =1 –R=-9

K	**1**	**2**					
2x	0	2					
2y	20	20					
h	-6						
(x,y)	E(1,10)	E(2,10)					

h <=0,E

Example



Given:

Radius , R = 10

(x,y)=(0,10)

h =1 –R= -9

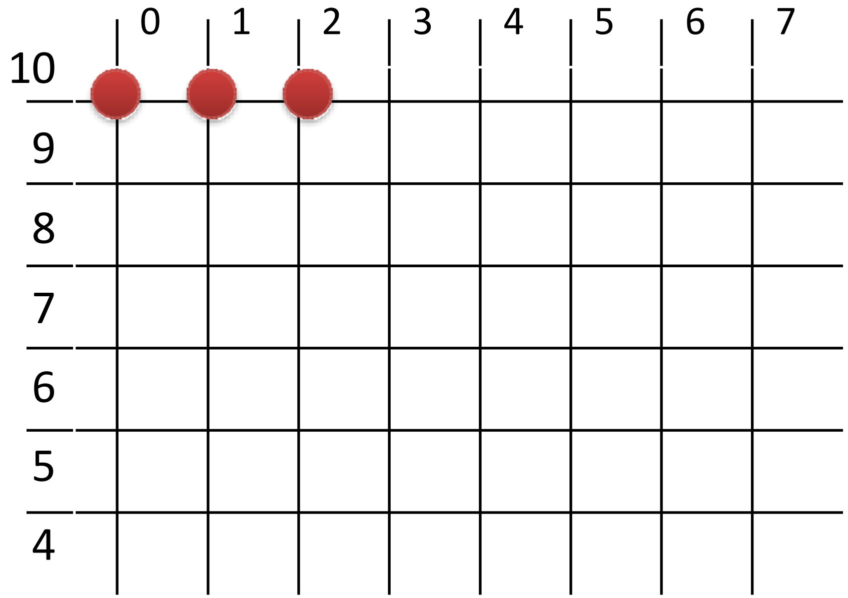
h =h+ΔE=h+2x+3

=-6+2+3

=-1

K	**1**	**2**					
2x	0	2					
2y	20	20					
h	-6	-1					
(x,y)	E(1,10)	E(2,10)					

Example



Given:

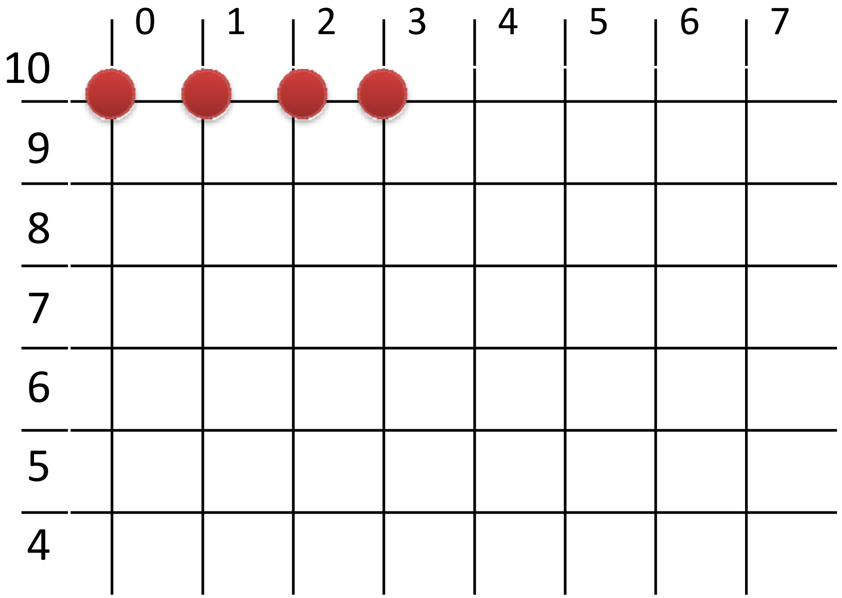
Radius , R = 10

(x,y)=(0,10)

h =1 –R=-9

K	**1**	**2**	**3**				
2x	0	2	4				
2y	20	20	20				
h	-6	-1					
(x,y)	E(1,10)	E(2,10)					

Example



Given:

Radius , R=10

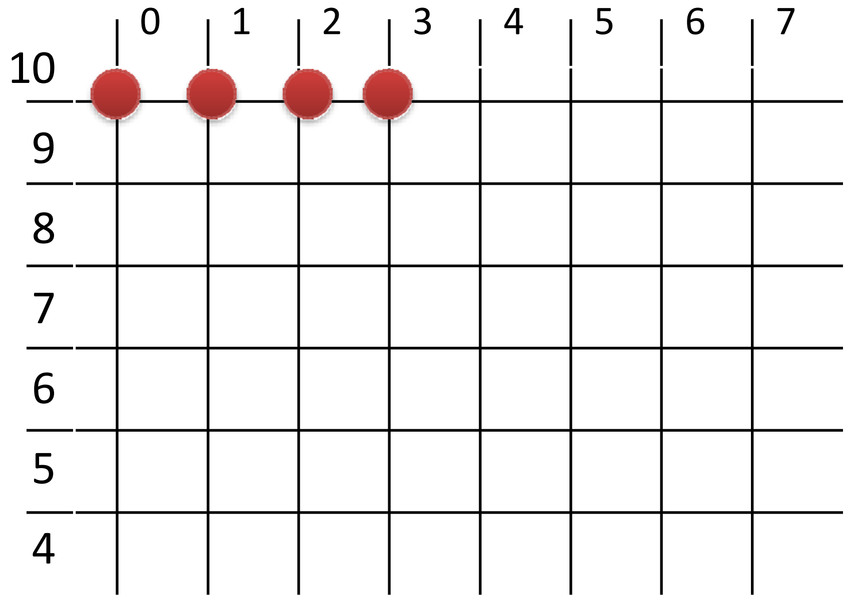
(x,y)=(0,10)

h =1 –R=-9

K	**1**	**2**	**3**				
2x	0	2	4				
2y	20	20	20				
h	-6	-1					
(x,y)	E(1,10)	E(2,10)	E(3,10)				

h <=0,E

Example



Given:

Radius , R = 10

(x,y)=(0,10)

h =1 –R=-9

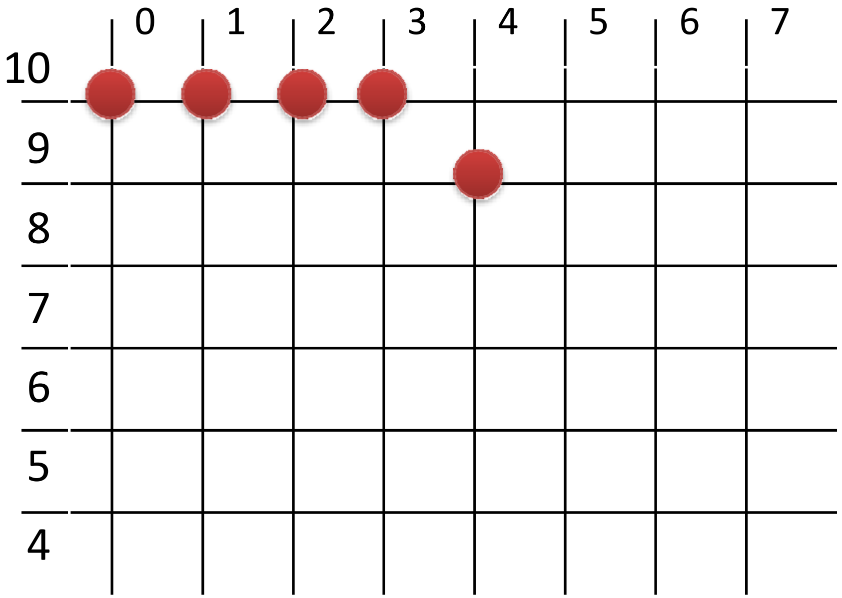
h =h+ΔE=h+2x+3

=-1+4+3

=6

K	**1**	**2**	**3**				
2x	0	2	4				
2y	20	20	20				
h	-6	-1	6				
(x,y)	E(1,10)	E(2,10)	E(3,10)				

Example



Given:

Radius , R = 10

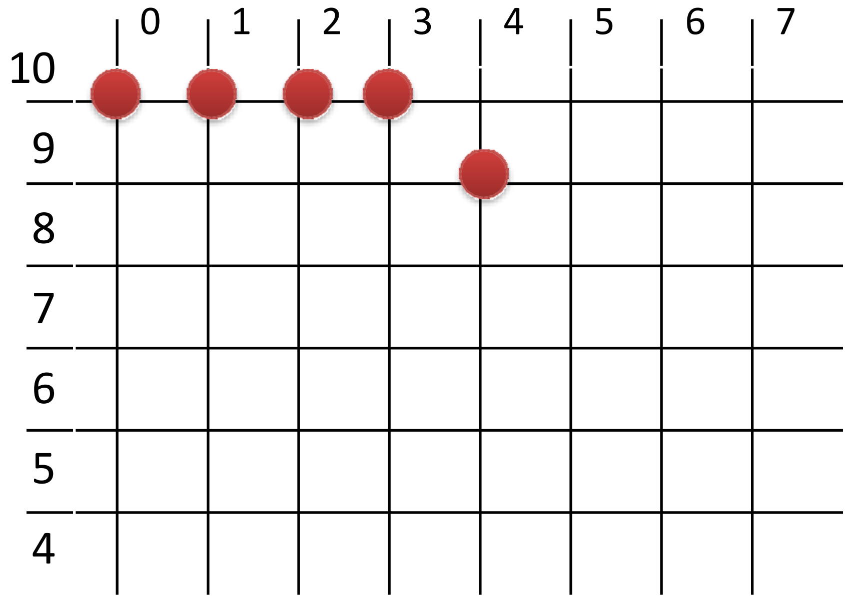
(x,y)=(0,10)

h =1 –R=-9

K	1	2	3	4			
2x	0	2	4	6			
2y	20	20	20	20			
h	-6	-1	6				
(x,y)	E(1,10)	E(2,10)	E(3,10)	S(4,9)			

h >0,SE

Example



Given:

Radius , R = 10

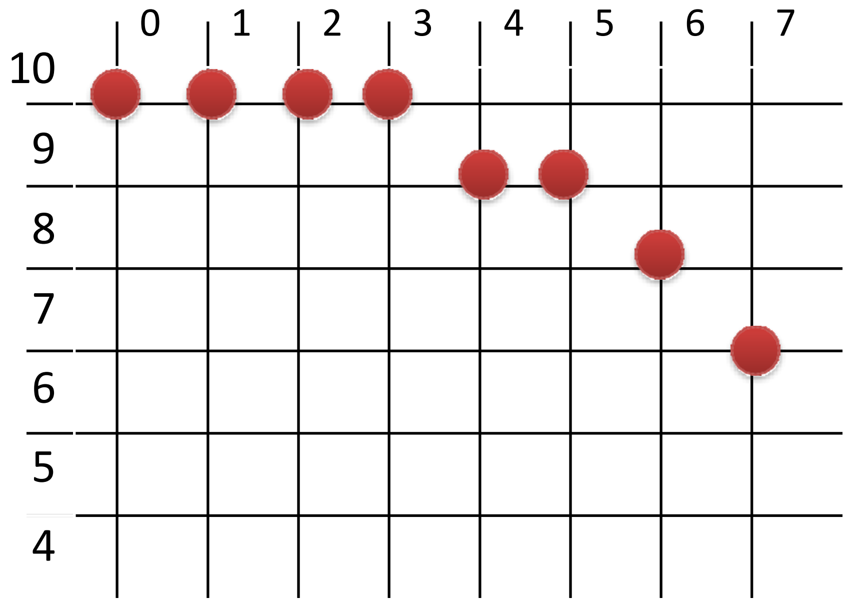
(x,y)=(0,10)

h =1 –R= -9

h =h+ΔSE=h+2x-2y+5
=6+6-20+5
=-3

K	1	2	3	4			
2x	0	2	4	6			
2y	20	20	20	20			
h	-6	-1	6	-3			
(x,y)	E(1,10)	E(2,10)	E(3,10)	S(4,9)			

Example



Given:

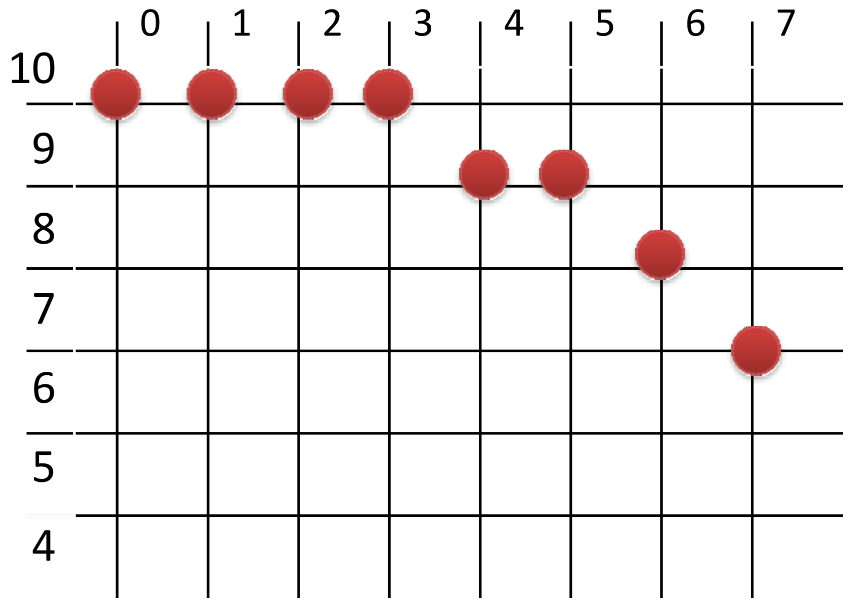
Radius , R=10

(x,y)=(0,10)

h =1 –R=-9

K	1	2	3	4	5	6	7
2x	0	2	4	6	8	10	12
2y	20	20	20	20	18	18	16
h	-6	-1	6	-3	8	5	6
(x,y)	E(1,10)	E(2,10)	E(3,10)	S(4,9)	E(5,9)	S(6,8)	S(7,7)

Example



Given:

Radius , R = 10

(x,y)=(0,10)

h =1 –R= -9

Untill y > x

K	1	2	3	4	5	6	7
2x	0	2	4	6	8	10	12
2y	20	20	20	20	18	18	16
h	-6	-1	6	-3	8	5	6
(x,y)	E(1,10)	E(2,10)	E(3,10)	S(4,9)	E(5,9)	S(6,8)	S(7,7)

Practice Problem

- Perform the midpoint algorithm to draw a circle's portion at 7^{th} octant which has center at (2,-3) and a radius of 7 pixels. Show each iterations and plot the points.