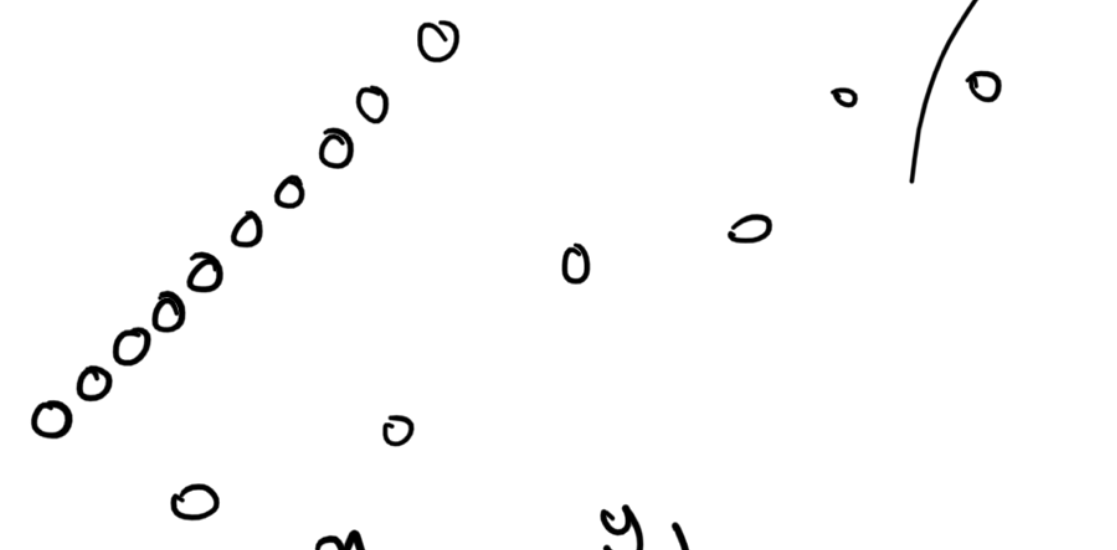
Week 2

midpoint 8 way Symmetry =

Problems!

$m > 1$



$x_0 \quad y_0$

$(30, -20)$ to $(-90, 40)$ $\quad x_1 \quad y_1$

$(-90, 40)$

$(30, -20)$

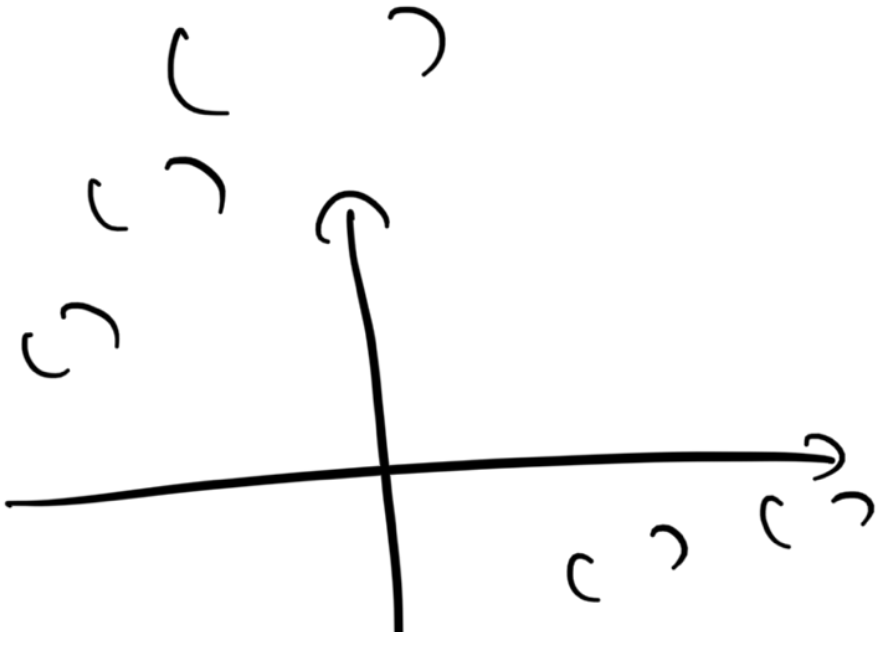
$x++$

-2^0

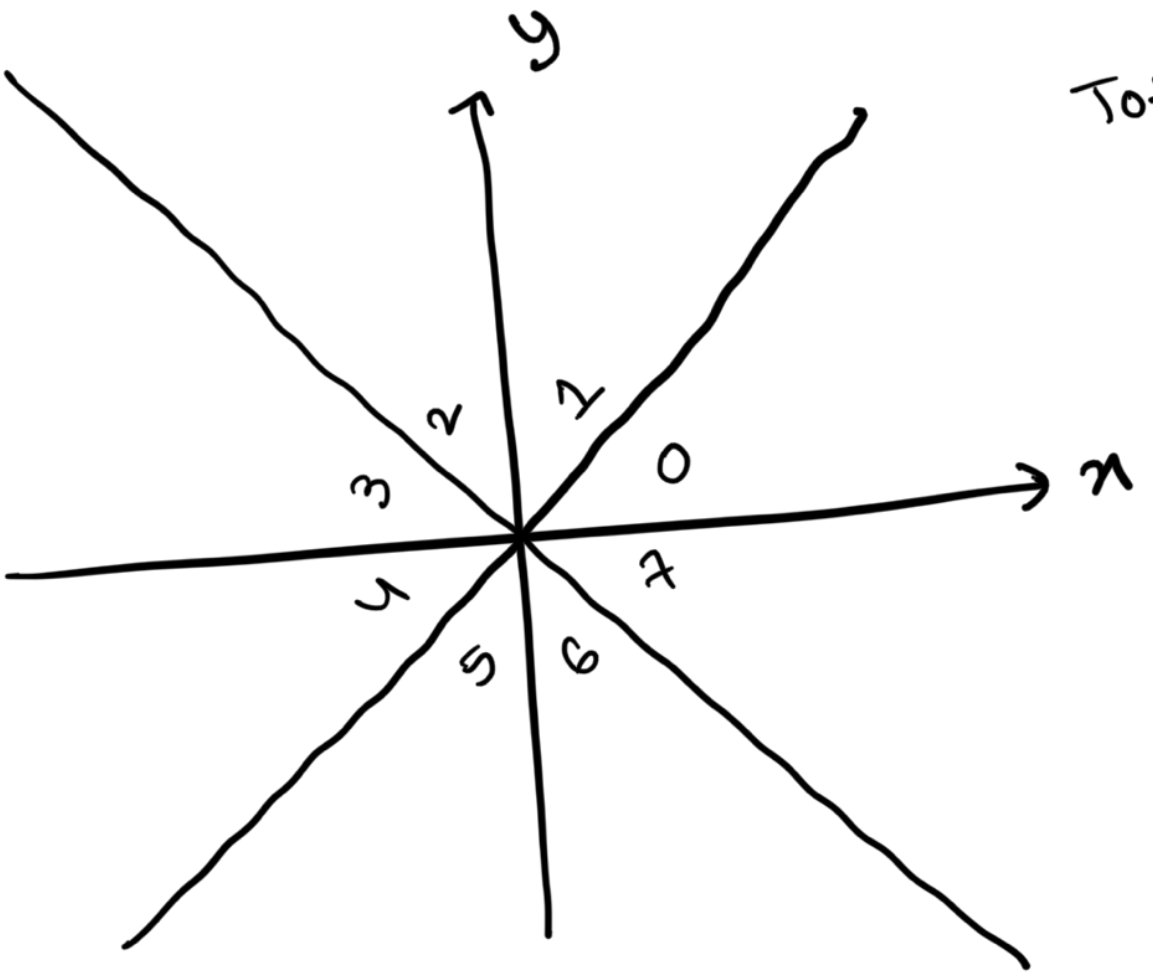
$y--$

$x++$

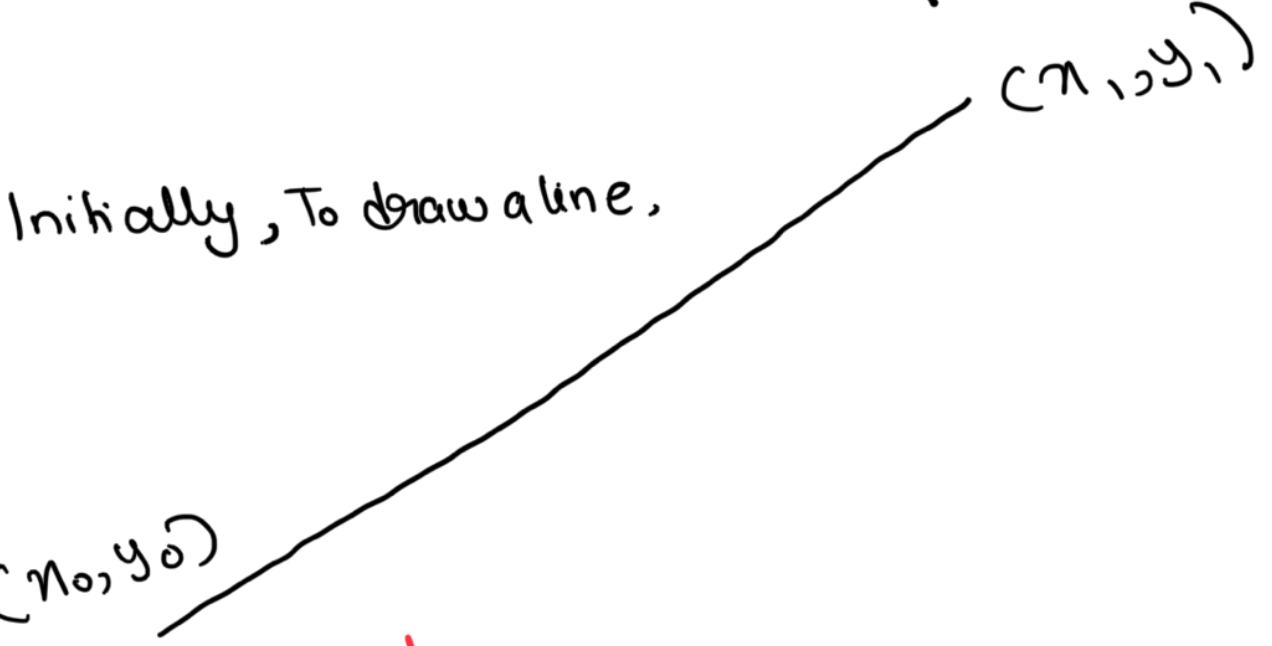
$y++$



Eight-way Symmetry



Total 8 zones.

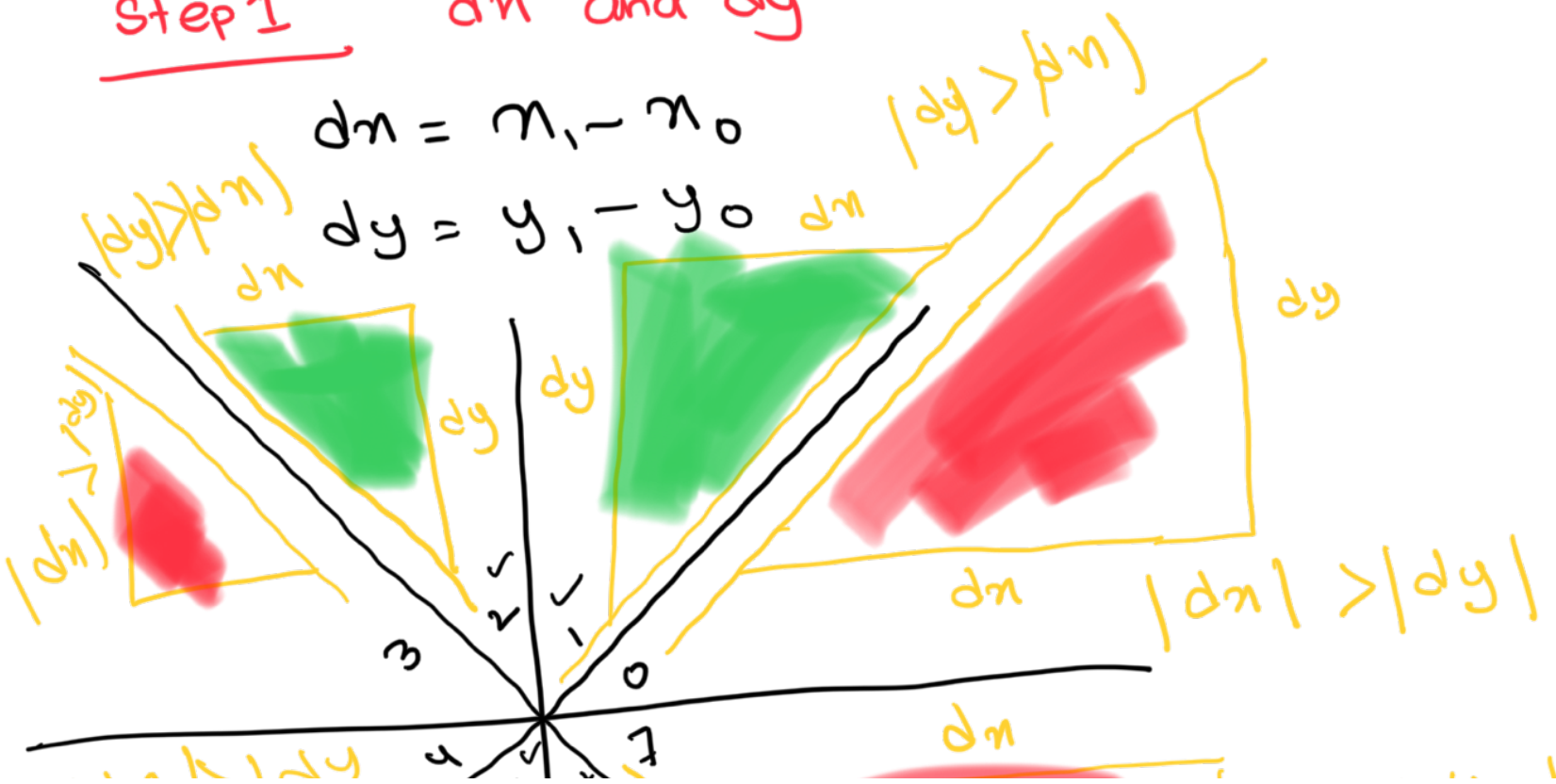


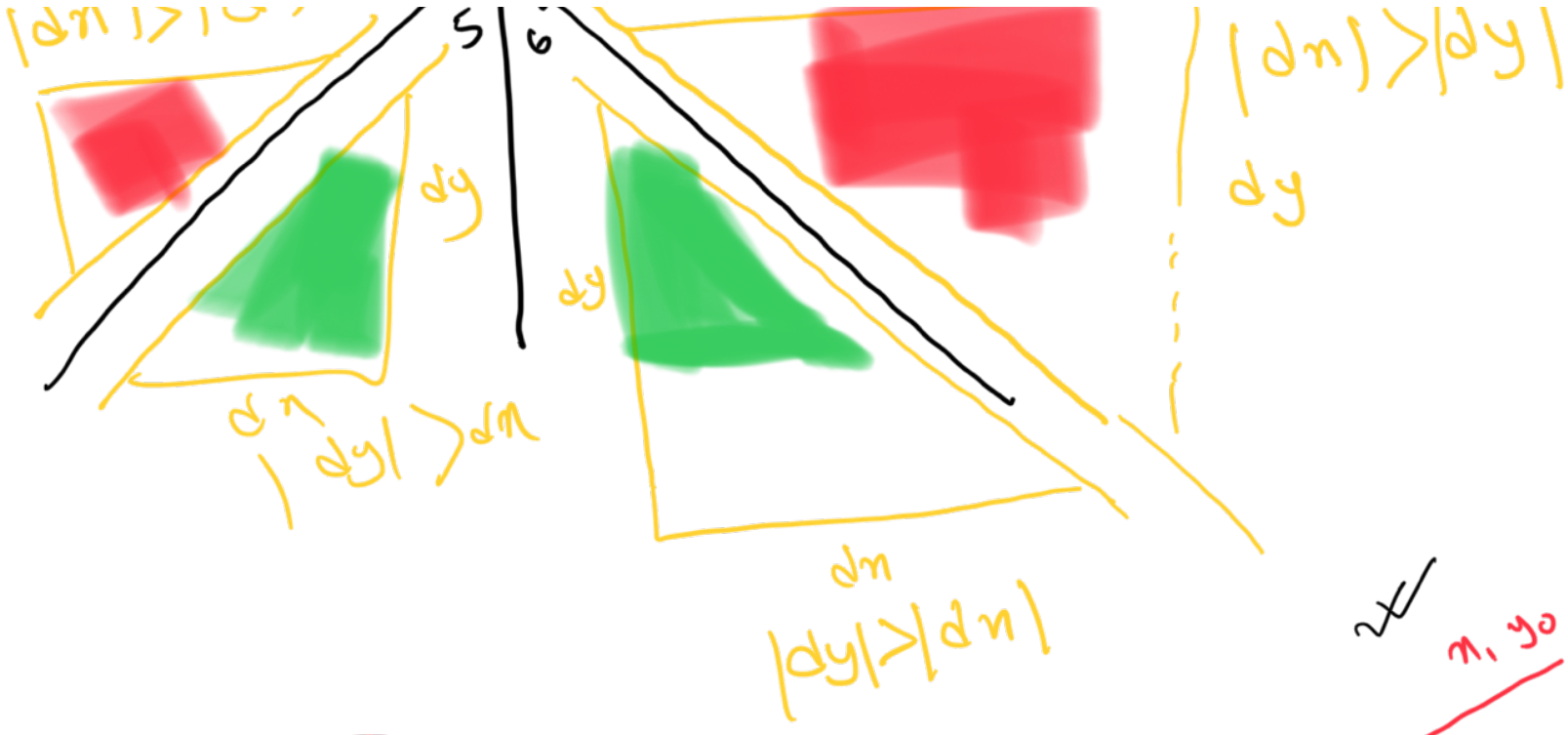
Initially, To draw a line,

Step 1 Find dx and dy

$$dx = x_1 - x_0$$

$$dy = y_1 - y_0$$





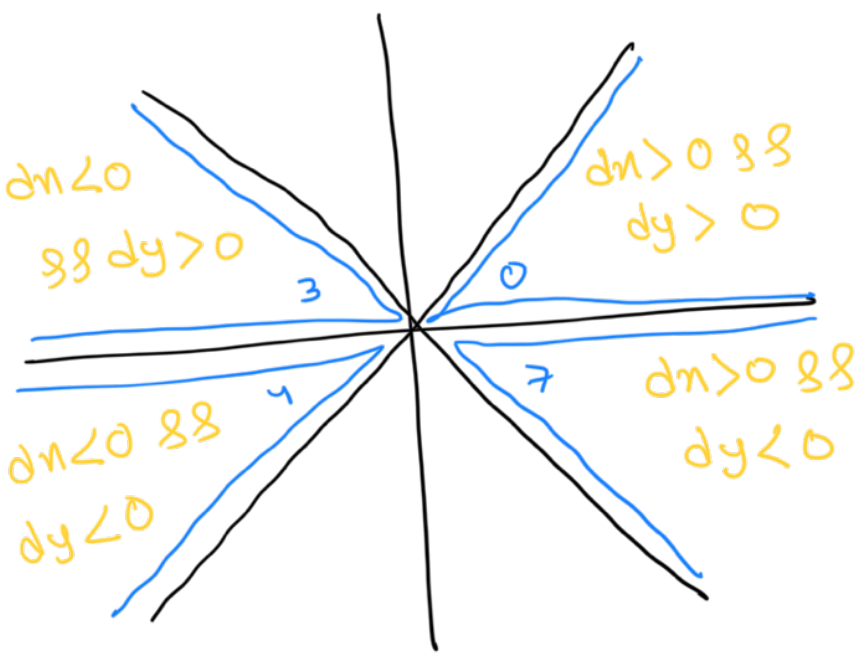
Step2 Findzone

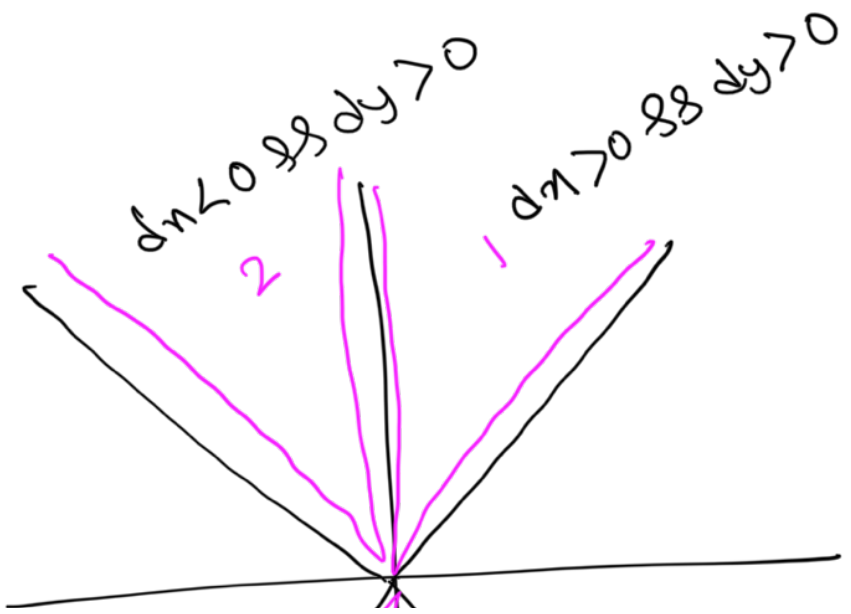
Two types of zone

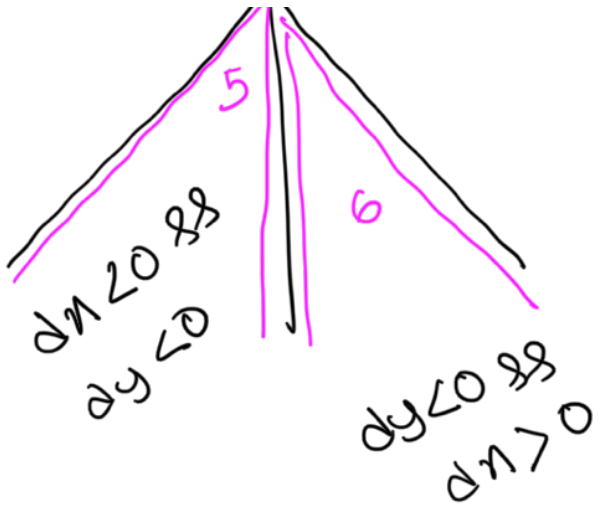
zone 1, 2, 6, 5 — |dy| > |dx|

zone 0, 7, 3, 4 — |dx| > |dy|


```
findzone (int x0, int y0, int x1, int y1)
    int dx = x1 - x0
    int dy = y1 - y0
    if (|dx| > |dy|) {          //zone 0,3,4,7
        if (dx>=0 && dy>=0)
            zone 0
        elif (dx<=0 and dy>=0)
            zone = 3
        elif (dx>=0 and dy<=0)
            zone = 7
        elif (dx<=0 && dy<=0)
            zone = 4
    }
    else:                       //1,2,5,6
        if (dx>=0 && dy>=0)
            zone=1
        elif (dx<=0 and dy>=0)
```







zone=2

elif (dx>=0 && dy<=0)

zone=6

elif (dx<=0 && dy>=0)

zone=5.

For example: (-30, -10) to (-10, 30)

dx = -10 - (-30) = 20. dx>0

dy = 30 - (-10) = 40. dy>0

dy > dx

zone = 1 ← ↓ ending

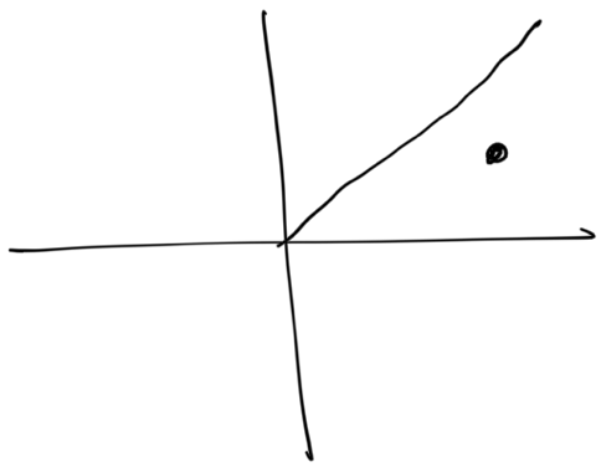
Step 3 Convert to zone 0 (int x, y, zone)

X = 0

Y = 0

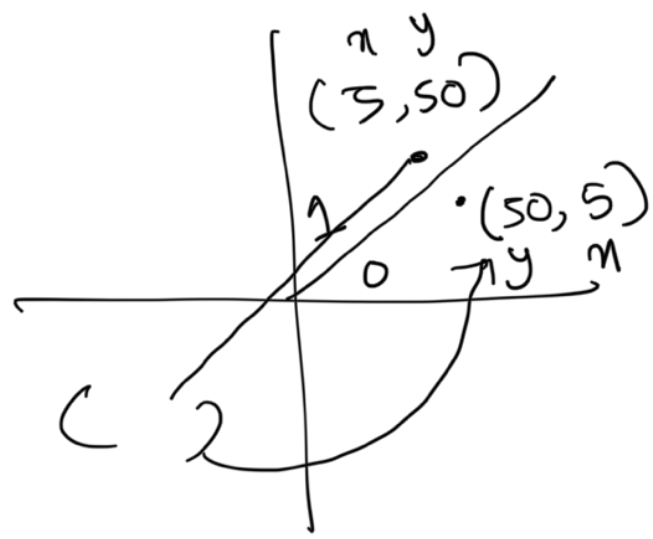
if (zone == 0)

X = x , Y = y



if (zone == 1)

X = y , Y = x



if (zone == 2)

X = y

Y = -x

if (zone == 3)

X = -x

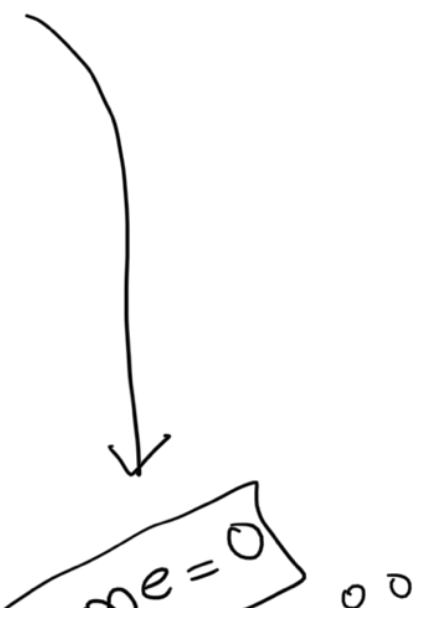
```
Y = y
if (zone == 4)

X = -x
Y = -y
if (zone == 5)

X = -y
y = -x
if (zone == 6)

X = -y
Y = x
if (zone == 7)

X = x
Y = -y
```

New Coordinate,

(X0, Y0) (X1, Y1)

Step 4 Apply Midpoint line Algorithm

```
midpoint (X0, Y0, X1, Y1)
    dx = X1 - X0
    dy = Y1 - Y0
    D = 2 * dy - dx
    DNE = 2 * (dy - dx)
    DE = 2 * dy
```



DE = 2 (y)

x = x0 , y = y0

while (x <= x1) {

Draw(x, y);

x++;

if (D < 0) {

D = D + DE; }

else { D = D + DNE; }

y++;

}

end

//E

Step 5 zeroToOriginalZone(zone, x, y)

x = 0

y = 0

if (zone == 0)

x = x y = y

if (zone == 1)

x = y

y = x

if (zone == 2)

x = -y

y = x

if (zone == 3)

x = -x

y = y

if (zone == 4)

x = -x

step 3

OriginalZoneTo Zero

x = y , y = x

x = -y

x = -x

y = y

x = -x

y = -y

```
y = -y
if (zone == 5)
    x = -y
    y = -x
if (zone == 6)
    x = y
    y = -x
if (zone == 7)
    x = x
    y = -y
```

Overall Step:

Two coordinates given:

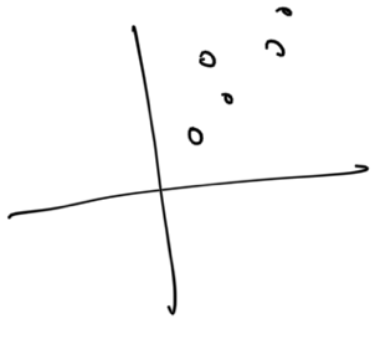
(I) Find Δx, Δy

(II) Find zone with conditions.

(III) Convert (x, y) to zone 0

(IV) Midpoint line Algorithm.

(V) Convert to Original zone.



Example

$(-10, -20)$ to $(-20, 70)$

x_0 y_0 x_1 y_1

Step1

$dx = -20 - (-10) = -10$

$dy = 70 - (-20) = 90 \quad dy > 0$

$dy > dx$

$|dy| > |dx|$

$dx < 0$ && $dy > 0$

Step2

Zone = 2

Step3

$(-10, -20) \quad (-20, 70)$

$x = y, \; y = -x$

$(-10, -20)$	$(-20, 10)$
$(-20, 70)$	$(70, 20)$
$x_0 \; y_0$	$x_1 \; y_1$

Step4

$dx = 70 + 20 = 90$

$dy = 20 - 10 = 10$

$\rightarrow D = 2 \times 10 - 90 = -70$

$DNE = 2 \times (10 - 90) = -100$

$DE = 2 \times 10 = 20$

$x = -20, \; y = 10$

$x^{-20} < x^{70}$ ← ① Pixel $(-20, 10)$

← $x = -19$

// $-70 < 0$

$D < 0$

$d = -70 + 20 = -50$

② pixel $(-19, 10)$

$x = -18$

$-50 < 0$ // E

Zone 0

$(-20, 10)$

$(-19, 10)$

$(-18, 10)$

$d = -50 + 20$

Zone 2

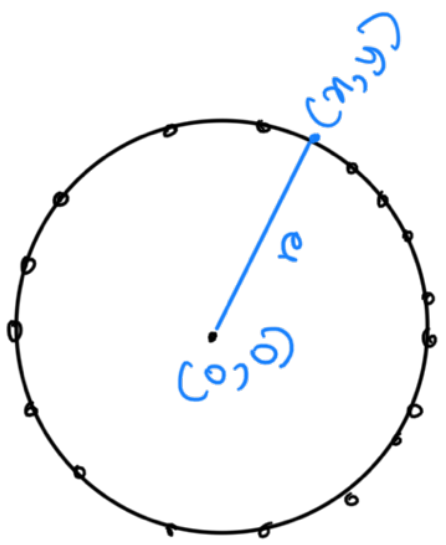
$(-y, x) = -30$

$(-10, -20)$ ③ pixel $(-18, 10)$

$(-10, -19)$

$(-10, -18)$

Midpoint Circle



$$r = \sqrt{(x-0)^2 + (y-0)^2}$$

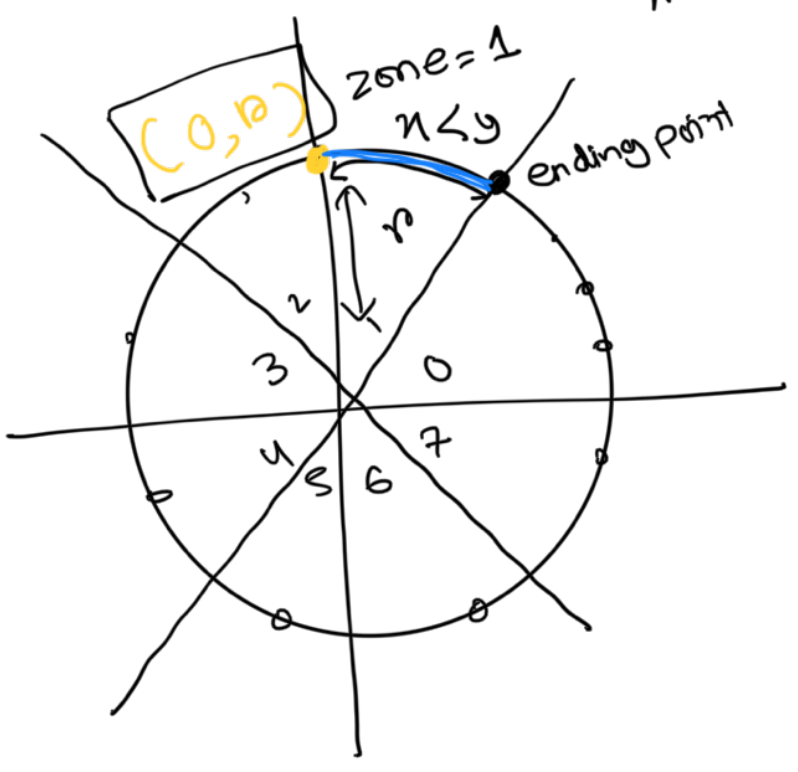
$$r^2 = x^2 + y^2$$

$$x^2 + y^2 - r^2 = 0$$

$$f(x,y) = x^2 + y^2 - r^2$$

If points are on the boundary of circle

$$f(x,y) = 0$$



$x = y$

- arc length same
- equally divided into sector

If midpoint outside the circle

$$f(x,y) > 0$$

Choose Lower Pixel / SE

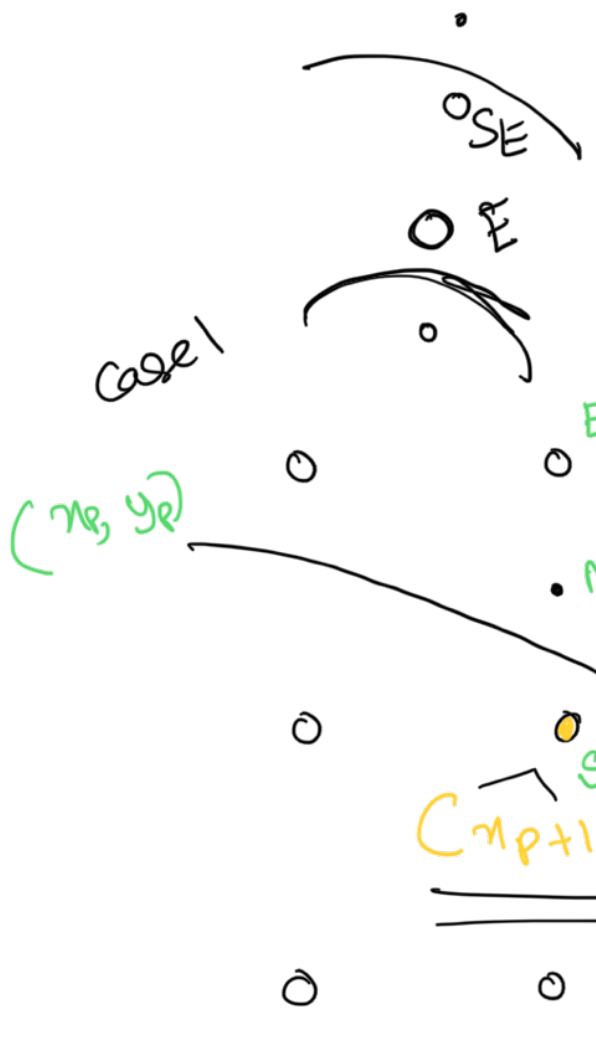
If midpoint is inside the circle

Choose upper Pixel / E

$$f(x,y) < 0$$

$f(M)$	Pixel
$f(M) \geq 0$	Lower Pixel (SE)

$f(M) \leq 0$ Upper Pixel (E)

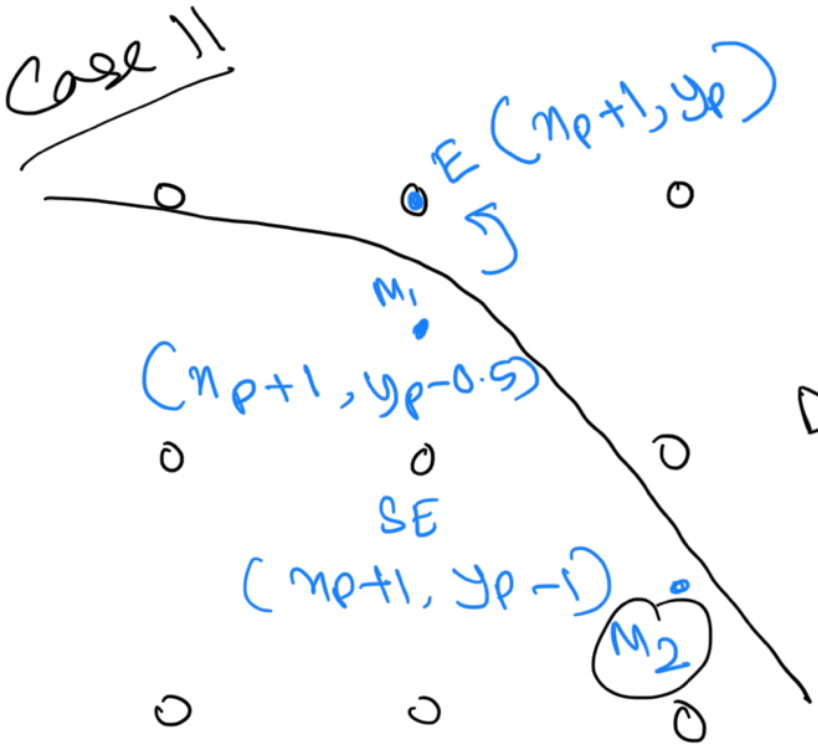


If we choose SE
NEXT MIDPOINT

$D_{new} = (x_p+1+1, y_p-0.5-1)$

$D_{new} = (x_p+2, y_p-1.5)$

Case II



If we choose E
Next Midpoint

$D_{new} = (x_{p+1}+1, y_p-0.5)$

$= (x_p+2, y_p-0.5)$

$D = f(x_p+1, y_p-0.5)$ $x^2+y^2-r^2$

$= (x_p+1)^2 + (y_p-0.5)^2 - r^2$

$= \boxed{x_p^2+2x_p+1+y_p^2-y_p+0.25-r^2}$

If Pixel SE, // $d > 0$

$D_{new} = f(x_p + 2, y_p - 1.5)$

$= (x_p+2)^2 + (y_p - 1.5)^2 - r^2$

$= x_p^2 + 4x_p + 4 + y_p^2 - 3y_p + 2.25 - r^2$

$D_{new} - D = 2x_p + 3 - 2y_p + 2$

$D_{new} = D + 2x_p - 2y_p + 5$ // SE ← $d > 0$

If E // $d < 0$

$D_{new} = f(x_p + 2, y_p - 0.5)$

$D_{new} = (x_p+2)^2 + (y_p - 0.5)^2 - r^2$

$= x_p^2 + 4x_p + 4 + y_p^2 - y_p + 0.25 - r^2$

$D_{new} - D = x_p^2 + 4x_p + 4 + y_p^2 - y_p + 0.25 - r^2$

$- x_p^2 - 2x_p - 1 - y_p^2 - y_p - 0.25 + r^2$

$D_{new} = D + 2x_p + 3$ ←

Starting $x_p = 0$, $y_p = r$

$D_{in} = f(x_p + 1, y_p - 1/2)$

$= f(1, r - 0.5)$

$= 1^2 + (r - 0.5)^2 - r^2$ ($x^2 + y^2 - r^2$)

$= 1 + r^2 - r + 0.25 - r^2$

$D = 1.25 - r$

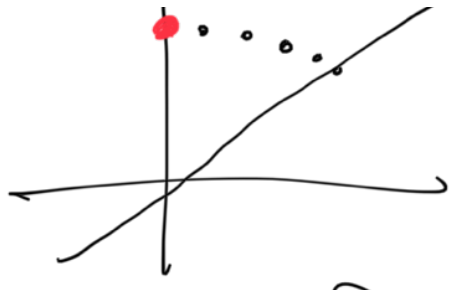
Round off $D = 1 - r$

```
Func Midpoint Circle (int r, x, y)
  int x, y, d;
```

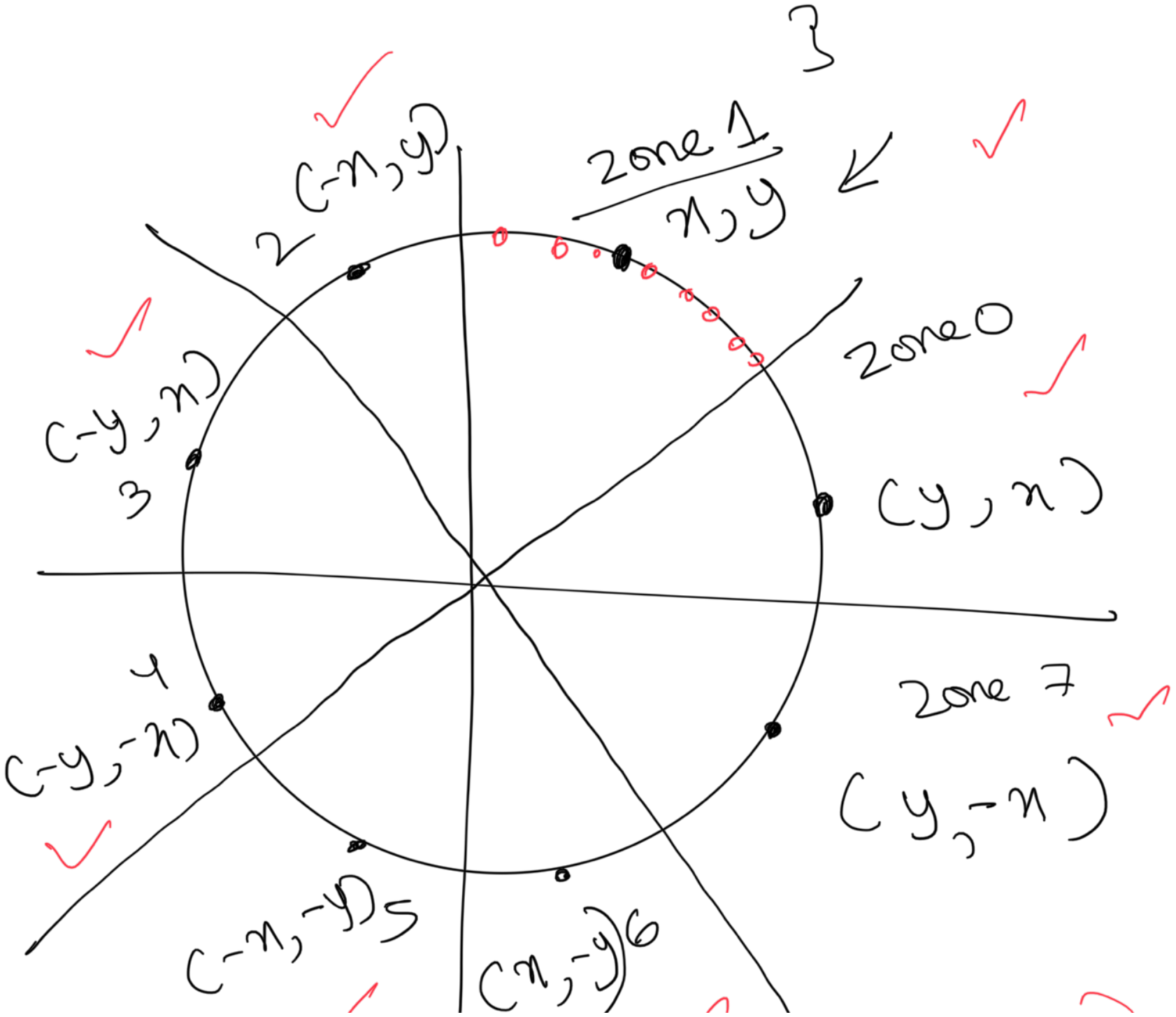
$d = 1 - r$

$x = 0$

$y = r$



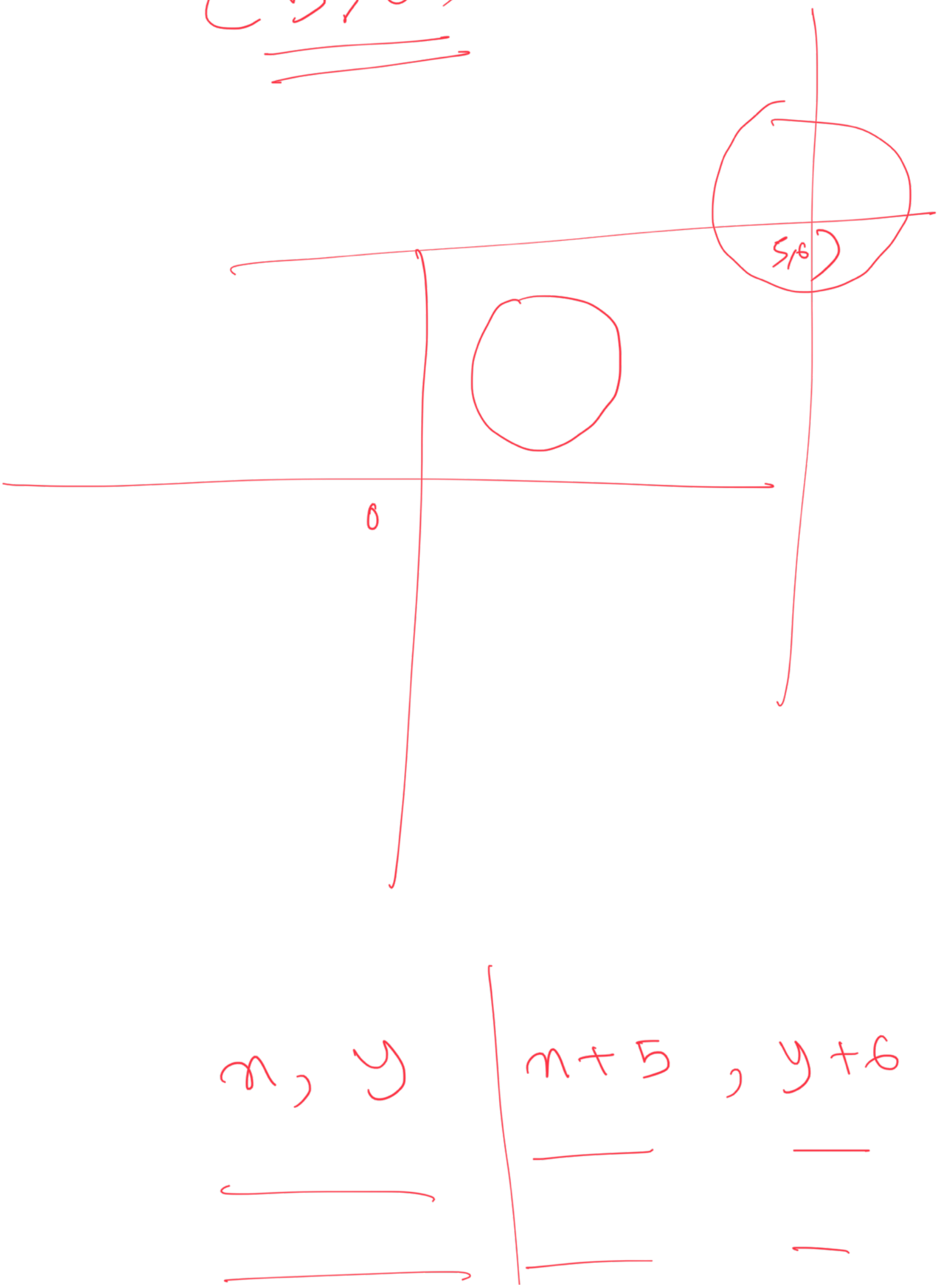
```
CirclePoints (x, y, value)
while x < y {
    if (d < 0) { // E
        d = d + 2x + 3
        x = x + 1 }
    else { // SE
        d = d + 2x - 2y + 5
        x = x + 1
        y = y - 1 }
    Circle (x, y, value) }
```



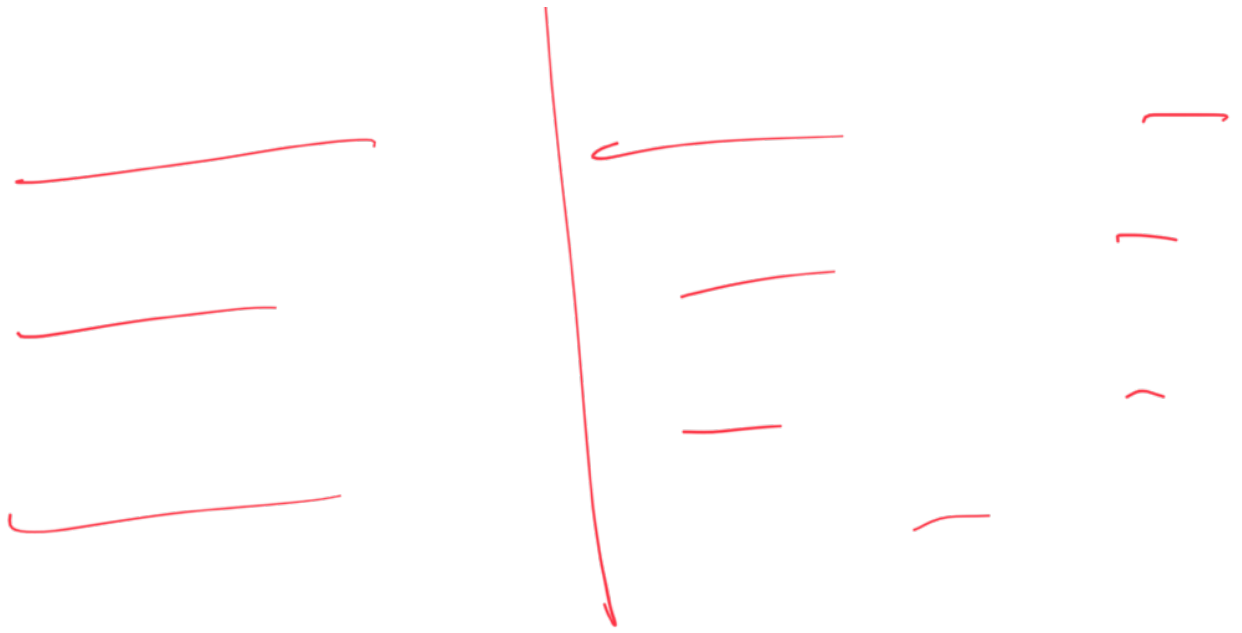
(0,5)

Centre (0,0)

(5,6)



x, y	$x+5, y+6$



radius = 15

x	y	d	E/SE	d
0	15	-14	d<0 E	-11
1	15	-11	d<0 E	-6
2	15	-6	d<0 E	1
3	15	1	d>0 SE	-18
4	14	-18	d<0 E	-7