

code blocks with MinGW setup

1. CG Basics

2. OpenGL Basics

3. Three algorithms - DDA (Digital Differential Analyser), Bresenham's Line or midpoint line, Bresenham's circle or Midpoint circle

6.2.19

* Explain Raster-scan systems & Random-scan Displays.

* Explain CRT systems

Explain basic primitives with an example (points, lines, linestrip, line loop)

OpenGL Basics

gl.h - graphics library ; Primitive fncs how to draw line, circle

glu.h - graphics library utility ; contains fncs of gl.h & windowing fncs.

glut.h - graphics library utility Toolkit ; gl.h + glu.h + other properties

How to draw a point, line, linestrip, line loop, polygon etc

First OpenGL program.

* Point

```
#include <GL / glut.h>
```

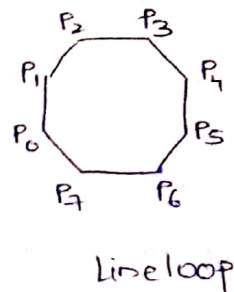
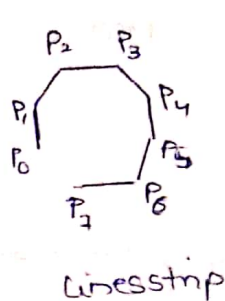
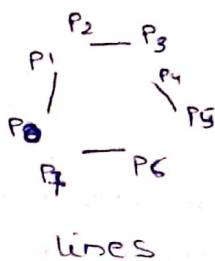
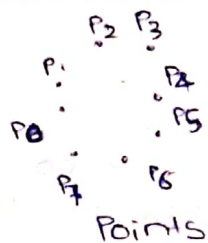
```
void display()
{
    glPointSize(10);
    glBegin ( GL_POINTS);
    glVertex2f (-0.5, -0.5);
    glVertex2f (0.5, -0.5);
    glVertex2f (0.5, 0.5);
    glVertex2f (-0.5, 0.5);
    glEnd();
    glFlush();
}
```

```
3
void main (int argc, char ** argv)
```

```
{
    glutInit (&argc, argv); // initializing openGL/graphics system so that system
                             // understands our graphics applications
    glutCreateWindow ("Points Demo"); // OpenGL will create a window with
                                     // title 'Points Demo'
    glutDisplayFunc (display); // It is a call back function which renders
                              // pixels to be drawn on to screen. We need to
                              // register a function which actually draws/execute
    glutMainLoop();           // as a parameter to this glutDisplayFunc callback function.
}
```

→ Run the output forever

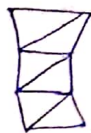
* Primitives



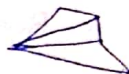
GL_Triangles



Triangle Strip



Triangle fan



GL_Polygon

Polygon with colour



GL_Quad



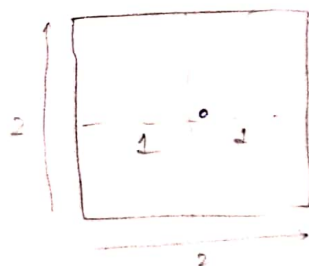
GL_Quadstrip



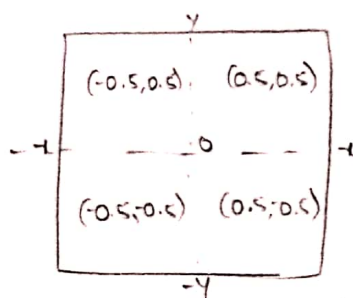
- In OpenGL programs default bgcolor is black
- pixel colour is white by default
- Entry point for any prog is `int main()`

1st Program

- canvas/window size 2×2 by default



Aim is to draw



* Applying custom background colour

- 1) `glClearColor(0, 0, 1, 1)` // setting color
R G B or transparency parameter
It can be 1 or 0

- 2) `glClear(GL_COLOR_BUFFER_BIT)` // applying color
constant

In OpenGL, uppercase : constant

```
void display()
```

```
{ glClearColor (0,0,1,1);
```

```
glClear (GL_COLOR_BUFFER_BIT);
```

```
glBegin (GL_POINTS);
```

```
=
```

```
} O/P Blue bg color  
white pixel color
```

* Applying custom font color

```
glColor3f(1,0,1);
```

```
void display()
```

```
{ glClearColor (0,0,1,1);
```

```
glClear (GL_COLOR_BUFFER_BIT);
```

```
glColor (1,1,0);
```

```
=
```

```
}
```


O/P Blue Bg color

Yellow pixel color

LINES

```
glBegin(GL_LINES);
```

```
glLineWidth(10); //thickness
```

O/P 

LINE STRIP

```
glBegin(GL_LINE_STRIP);
```

O/P 

LINE LOOP

```
glBegin (GL_LINE_LOOP);
```

O/P 

Algorithms

1) DDA

2) Bresenham's line

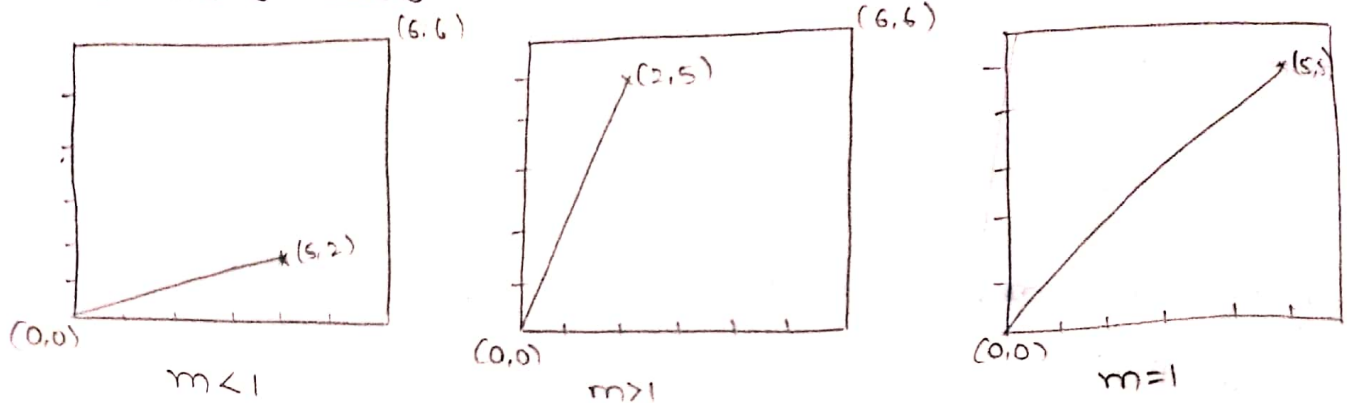
3) Bresenham's circle

17 DDA Line Digital Differential Analyser

* screen is full of pixels

* aim: to determine intermediate pixels b/w start & end point

we have 3 cases



we know, line eq is $y = mx + c$ where $m = \frac{y_2 - y_1}{x_2 - x_1}$ — (1)

for any two points (x_1, y_1) & (x_2, y_2)

In general, let us assume

$$\left. \begin{array}{l} \text{current pixel} = (x_k, y_k) \\ \therefore \text{Next pixel} = (x_{k+1}, y_{k+1}) \end{array} \right\} m = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} \quad \text{--- (2)}$$

we have 3 cases,

case 1: $m < 1$

x always gets changed unit wise.

$y = ?$

$$\boxed{x_{k+1} = x_k + 1}$$

$y_{k+1} = ?$

$$(2) \Rightarrow m = \frac{y_{k+1} - y_k}{1}$$

$$\therefore \boxed{y_{k+1} = y_k + m}$$

case 2: $m > 1$

y always gets changed unit wise

$x = ?$

$$\boxed{y_{k+1} = y_k + 1}$$

$x_{k+1} = ?$

$$(2) \Rightarrow m = \frac{1}{x_{k+1} - x_k}$$

$$\therefore \boxed{x_{k+1} = x_k + \frac{1}{m}}$$

case 3: $m = 1$

y changes unit wise
 x changes unit wise

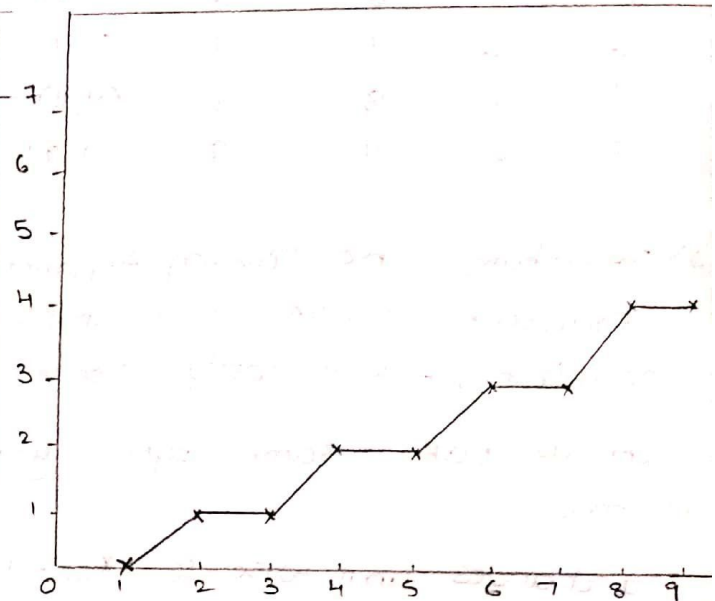
$$\begin{aligned} x_{k+1} &= x_k + 1 \\ y_{k+1} &= y_k + 1 \end{aligned}$$

Example 1: Draw a line between $(1,0)$ to $(9,4)$ using DDA.

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{4 - 0}{9 - 1} = \frac{1}{2} < 1$$

since $m < 1$,
$$\begin{aligned} x_{k+1} &= x_k + 1 \\ y_{k+1} &= y_k + m \end{aligned}$$

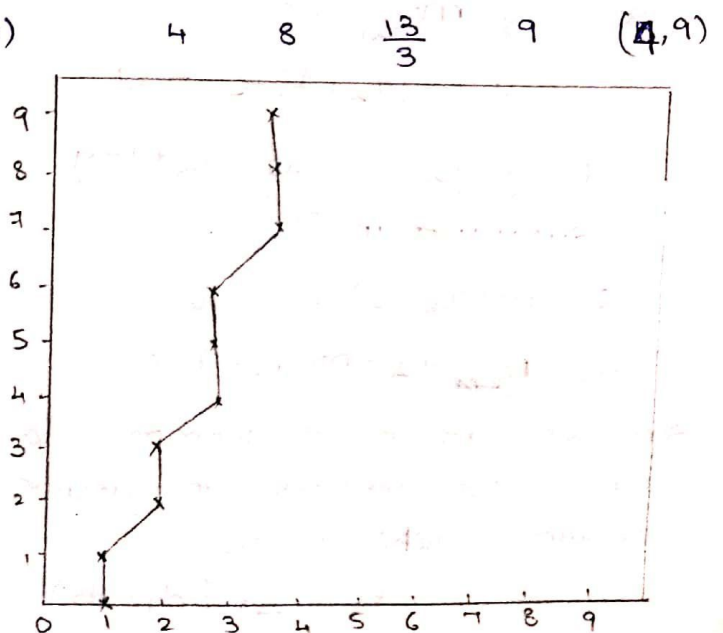
x_k	y_k	x_{k+1}	y_{k+1}	round (actual point (x,y))
1	0	1	0	(1,0)
1	0	2	0.5	(2,1)
2	0.5	3	1	(3,1)
3	1	4	1.5	(4,2)
4	1.5	5	2	(5,2)
5	2	6	2.5	(6,3)
6	2.5	7	3	(7,3)
7	3	8	3.5	(8,4)
8	3.5	9	4	(9,4)



2: $(1,0)$ to $(4,9)$

$$m = 9/3 = 3 > 1 ; \begin{aligned} x_{k+1} &= x_k + \frac{1}{m} \\ y_{k+1} &= y_k + 1 \end{aligned}$$

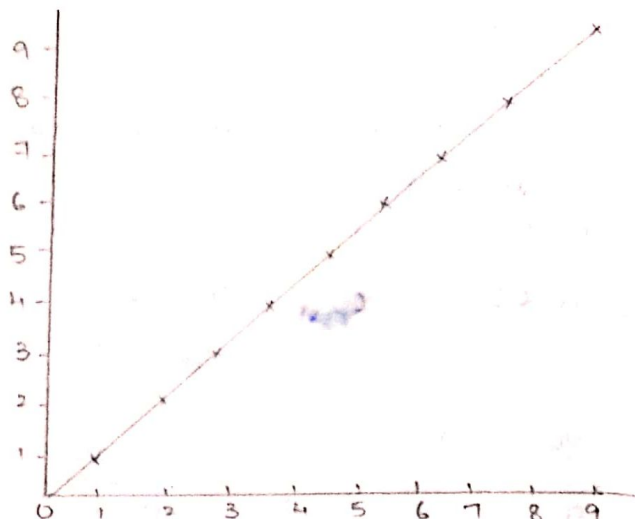
x_k	y_k	x_{k+1}	y_{k+1}	round (x,y)
1	0	1	0	(1,0)
1	0	$4/3$	1	(2,1)
$4/3$	1	$5/3$	2	(2,2)
$5/3$	2	2	3	(2,3)
2	3	$8/3$	4	(3,4)
$8/3$	4	3	5	(3,5)
3	5	$10/3$	6	(3,6)
$10/3$	6	$11/3$	7	(4,7)
$11/3$	7	4	8	(4,8)



3) (1,1) to (9,9)

$$m = 1 \quad x_{k+1} = x_k + 1 \\ y_{k+1} = y_k + 1$$

x_k	y_k	x_{k+1}	y_{k+1}	(x, y)
1	1	1	1	(1, 1)
1	1	2	2	(2, 2)
2	2	3	3	(3, 3)
3	3	4	4	(4, 4)
4	4	5	5	(5, 5)
5	5	6	6	(6, 6)
6	6	7	7	(7, 7)
7	7	8	8	(8, 8)
8	8	9	9	(9, 9)



2) Bresenham's line Drawing Algorithm: (Midpoint Line Algorithm)

Drawback of DDA is it involves rounding operation & floating pt operation which is costly. Hence we have Bresenham's line equation.

Let us take same eqn $y = mx + c$; 3 cases $m < 1, m > 1$ & $m = 1$

i) $m < 1$

x changes unit wise, so $x_{k+1} = x_k + 1$ (no dilemma)

$y_{k+1} = ?$ (there is a confusion between to choose y_{k+1} or y_k)

WKT $y = mx + c$, $m = \frac{\Delta y}{\Delta x}$

since it is $m < 1$ case, we know $x_{k+1} = x_k + 1$

so, $y = mx_{k+1} + c$

$$y = m(x_k + 1) + c \quad \text{--- ①}$$

$$d_1 = y - y_k \quad d_2 = y_k + 1 - y$$

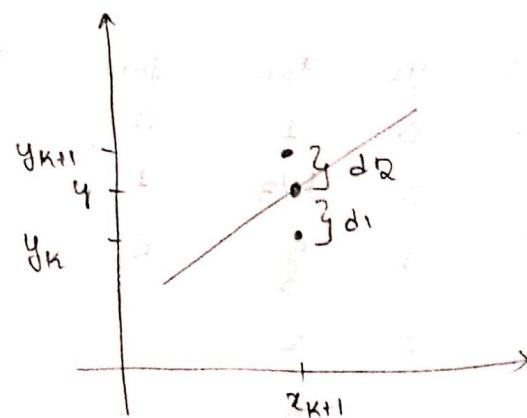
substitute in ①

$$d_1 = m(x_k + 1) + c - y_k$$

$$d_2 = y_k + 1 - m(x_k + 1) - c$$

Since we are in a dilemma to choose b/w $y_k + 1$ or y_k , let us (consider) calculate decision parameter P_k which helps us in deciding y_{k+1} or y_k .

$$P_k = \Delta x (d_1 - d_2) \quad \text{--- ②}$$



Let us calculate $(d_1 - d_2)$

$$\begin{aligned} d_1 - d_2 &= m(x_k + 1) + c - y_k - y_k + 1 + m(x_k + 1) + c \\ &= 2m(x_k + 1) + 2c - 2y_k - 1 \end{aligned}$$

substitute in ②,

$$\begin{aligned} P_k &= \Delta x [2m(x_k + 1) - 2y_k + 2c - 1] \\ &= \Delta x \left[2 \cdot \frac{\Delta y}{\Delta x} (x_k + 1) - 2y_k + 2c - 1 \right] \end{aligned}$$

$$P_k = 2\Delta y(x_k + 1) - 2\Delta x y_k + \Delta x(2c - 1) \quad \text{--- ③}$$

eqn ③ $\Rightarrow P_k$ is initial decision parameter. In order to find the continuous decisions, we have to find the next P_k .

$$P_{k+1} = 2\Delta y(x_{k+1} + 1) - 2\Delta x y_{k+1} + \Delta x(2c - 1) \quad \text{--- ④}$$

From now the next decision parameters will always be difference b/w P_{k+1} & P_k .

$$\text{④} - \text{③} \Rightarrow P_{k+1} - P_k = 2\Delta y(x_{k+1} + 1) - 2\Delta x y_{k+1} + \Delta x(2c - 1) - 2\Delta y(x_k + 1) + 2\Delta x y_k - \Delta x(2c - 1)$$

$$= 2\Delta y x_{k+1} + 2\Delta y - 2\Delta x y_{k+1} - 2\Delta y x_k - 2\Delta y + 2\Delta x y_k$$

$$= 2\Delta y(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$$

$$= 2\Delta y - 2\Delta x(y_{k+1} - y_k)$$

$$P_{k+1} = 2\Delta y - 2\Delta x(y_{k+1} - y_k) + P_k \quad \text{--- ⑤}$$

The initial point to be plotted is (x_k, y_k)

Let us substitute (x_k, y_k) for (x, y) in initial decision parameter.

$$\text{Eqn ③} \quad y = mx + c \Rightarrow c = y - mx$$

$$P_k = 2\Delta y(x_k + 1) - 2\Delta x y_k + \Delta x(2c - 1)$$

$$= 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + \Delta x(2(y - mx) - 1)$$

$$= 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + \Delta x(2y_k - 2\frac{\Delta y}{\Delta x} x_k - 1)$$

$$= 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + 2\Delta x y_k - 2\Delta y x_k - \Delta x$$

$$P_k = 2\Delta y - \Delta x \rightarrow \text{⑥}$$

Eqn ⑥ is initial decision parameter

$$(1) P_k = 2\Delta y - \Delta x \quad // \text{ apply once initially}$$

$$(2) P_{k+1} = P_k + 2\Delta y - 2\Delta x (y_{k+1} - y_k) \quad // \text{ from next iteration}$$

Conclusion: If ($P_k \geq 0$)

$$\{ x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k + 1$$

}

else

$$\{ x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k$$

}

eg 1: Draw a line between (1,0) , (9,4) using Bresenham's

$$m = \frac{4-0}{9-1} = 0.5 < 1.$$

$$\Delta y = 4, \Delta x = 8, 2\Delta y = 8, 2\Delta x = 16$$

$$P_k = 2\Delta y - \Delta x = 8 - 8 = 0$$

Let us calculate Initial decision parameter P_k ,
since this IDP P_k happens to be zero, see the if loop

x_k	y_k	P_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})	
—	—	—	1	0	(1, 0)	
1	0	(1) $8 - 8$ 0	2	1	(2, 1)	
2	1	(2) $0 + 8 - 16(1-0)$ -8	3	1	(3, 1)	$y_{k+1} = 1 \quad y_k = 0$
3	1	0	4	2	(4, 2)	$-8 + 8 - 16(0)$
4	2	-8	5	2	(5, 2)	$0 + 8 - 16(1)$
5	2	0	6	3	(6, 3)	$-8 + 8 - 16(0)$
6	3	-8	7	3	(7, 3)	$0 + 8 - 16(3-1)$
7	3	0	8	4	(8, 4)	$-8 + 8 - 16(3-3)$
8	4	-8	9	4	(9, 4)	$0 + 8 - 16(4-3)$

2. (3,2) to (9,6)

$$m = \frac{6-2}{9-3} = \frac{4}{6} = \frac{2}{3} < 1$$

$$\Delta y = 4, \Delta x = 6, 2\Delta y = 8, 2\Delta x = 12$$

$$P_k = 2(4) - 6 = 2$$

x_k	y_k	P_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})
—	—	—	3	2	(3,2)
3	2	2	4	3	(4,3)
4	3	-2	5	3	(5,3)
5	3	+6	6	4	(6,4)
6	4	2	7	5	(7,5)
7	5	-2	8	5	(8,5)
8	5	6	9	6	(9,6)

$$2 + 8 - 12 (1)$$

$$-2 + 8 - 12 (0)$$

$$6 + 8 - 12 (1)$$

$$2 + 8 - 12 (1)$$

$$-2 + 8 - 12 (0)$$

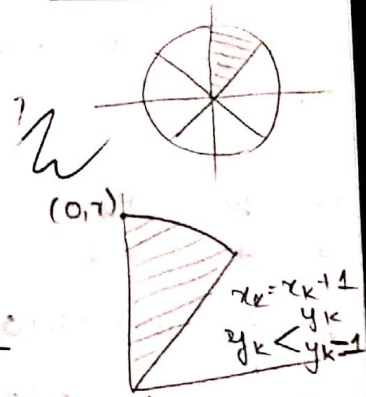
3) Mid point circle [Bresenham's circle algorithm]

Circle follows symmetrical property. As radius is same for all quadrants any one quadrant calculation is fine. In one quadrant one octant is fine

next coordinates may be (x_k+1, y_k) or (x_k+1, y_k-1)

$$\text{mid point} = \frac{x_k+1 + x_{k+1}}{2}, \frac{y_k + y_{k+1}}{2}$$

$$= (x_k+1, y_k - \frac{1}{2})$$



we have to apply this circle formula $x^2 + y^2 = r^2$

$$P_k = (x_k+1)^2 + (y_k - \frac{1}{2})^2 - r^2$$

① This is Initial decision parameter To find next, calculate P_{k+1}

$$P_{k+1} = (x_{k+1}+1)^2 + (y_{k+1} - \frac{1}{2})^2 - r^2$$

From now on the next dp can be calculated by diff b/w P_{k+1} & P_k

$$P_{k+1} - P_k = ((x_{k+1}+1)+1)^2 + (y_{k+1} - \frac{1}{2})^2 - (x_k+1)^2 - (y_k - \frac{1}{2})^2$$

$$= (x_k+1)^2 + 1 + 2(x_k+1) + (y_{k+1}^2) + \frac{1}{4} - y_{k+1} - (x_k+1)^2 - y_k^2 - \frac{1}{4} + y_k$$

$$= 2(x_k+1) + y_{k+1}^2 - y_k^2 - (y_{k+1} - y_k) + 1$$

$$\star P_{k+1} = P_k + 2(x_k+1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1$$

Initial dp at ① starting from $(0, r)$

$$P_k = (0+1)^2 + \left(r - \frac{1}{2}\right)^2 - r^2 = 1 + r^2 + \frac{1}{4} - r^2 - r$$

$$= \frac{5}{4} - r$$

★ $P_k = 1 - r \rightarrow$ Initial dp

Conclusion:

$\bar{y}_k (P_k \geq 0)$

$$\{ \quad x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k - 1$$

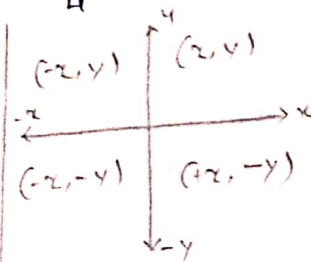
}

else

$$\{ \quad x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k$$

}



Quadrant 1: calculate till $(2, 0)$
Quadrant 2: write all values with $-x$

$$3: \quad -x, -y$$

$$4: \quad +x, -y$$

Q. Draw a circle of radius 8 units using M-PCircle algorithm.

given $r = 8$

Initial dp $P_k = 1 - r = -7$

$$P_{k+1} = P_k + 2(x_k + 1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1$$

x_k	y_k	P_k	x_{k+1}	y_{k+1}
0	8	-7	1	8
1	8	$-7 + 2(0+1) + (8^2 - 8^2) - (8-8) + 1 = -7 + 2 + 1 = -4$	2	8
2	8	$-4 + 2(0+2) + (0 - 0) - (0) + 1 = -4 + 4 + 1 = 1$	3	7
3	7	$1 + 2(3) + (49 - 64) - (-1) + 1 = 1 + 6 + (-15) + 1 = -7$	4	7
4	7	$-7 + 2(4) + (0 - 0) - 0 + 1 = -7 + 8 + 1 = 2$	5	6
5	6	$2 + 2(5) + (36 - 49) - (-1) + 1 = 2 + 10 + (-13) + 1 = -9$	6	5
6	5	$-9 + 2(6) + (0 - 0) - 0 + 1 = -9 + 12 + 1 = 4$	7	3
7	3	$4 + 2(7) + (9 - 25) - 2 + 1 = 4 + 14 - 16 - 1 = 11$	8	2
8	2	$11 + 2(8) + (4 - 9) - 1 + 1 = 11 + 16 - 5 = 22$	8	1
8	1	$22 + 2(9) + (1 - 4) - 1 + 1 = 22 + 18 - 3 = 37$	8	0

a. $r=12$

$P_k = 1-12 = -11$

x_k	y_k	P_k	x_{k+1}	y_{k+1}
0	12	-11	1	12
1	12	$-11 + 2(1) + (12^2 - 12^2) - (0) + 1 = -11 + 2 + 1 = -8$	2	12
2	12	$-8 + 2(2) + 0 - 0 + 1 = -8 + 4 + 1 = -3$	3	12
3	12	$-3 + 2(3) + 0 - 0 + 1 = -3 + 6 + 1 = 4$	4	11
4	11	$4 + 2(4) + 121 - 144 - (11 - 12) + 1 = 4 + 8 - 23 + 1 + 1 = -9$	5	11
5	11	$-9 + 2(5) + 0 - 0 + 1 = -9 + 10 + 1 = 2$	6	10
6	10	$2 + 2(6) + 100 - 121 - (-1) + 1 = 2 + 12 - 21 + 2 = -5$	7	10
7	10	$-5 + 2(7) + 0 - 0 + 1 = -5 + 14 + 1 = 10$	8	9
8	9	$10 + 2(8) + 81 - 100 + 1 + 1 = 10 + 16 - 19 + 2 = 9$	9	8
			10	7
			10	6
			11	5
			11	4
			12	3
			12	2
			12	1
			12	0

Quadrant 2: x_{k+1} with $(-)$

3: $-x, -y$

4: $-x, y$

