# Experiment No:5

**Title:** Algorithm For Filling A Polygon

**Aim:** Write a program to fill a polygon using Scan line Polygon filling Algorithm.

## Prerequisites:

Polygon Concepts, inside test of polygon, polygon filling

## Objectives:

Understand different polygon scan fill algorithms and implement them in laboratory.

## Theory:

Different types of Polygons

* Simple Convex
* Simple Concave
* Non-simple : self-intersecting
* With holes

Scan Line Polygon Fill Algorithms

* + A standard output primitive in general graphics package is a solid color or patterned polygon area:
  + There are two basic approaches to filling on raster systems.
  + Determine overlap Intervals for scan lines that cross that area.
  + Start from a given interior point and paint outward from this point until we encounter the boundary
  + The first approach is mostly used in general graphics packages, however second approach is used in applications having complex boundaries and interactive painting systems

## Input:

Enter a polygon of different.

## Output:

Output polygon should be properly painted.

## Questions:

1. Explain concave and convex polygons?
2. What is Polygon?
3. What are the different algorithms for polygon filling?
4. Explain the concept of polygon filling?
5. What is seed fill?
6. What is scan line algorithm?
7. What are the advantages of scan line algorithm?
8. What is raster scan?
9. What is simple convex polygon?
10. What Edge fill algorithm?

# Experiment No:6

**Title:** Windowing and Clipping

**Aim:** 2 D Clipping : i) Line Clipping – Cohen Sutherland Outcode Method ii) Sutherland Hodgman Polygon Clipping

## Prerequisites:

* 1. Concepts, Windowing, Clipping

## Objectives:

Understand and Implement concept of clipping and windowing in 2D.

## Theory:

**Algorithm1: Cohen Sutherland algorithm for Line Clipping**

* + 1. Start
    2. Input two endpoints of the line say p1 ( x 1 , y1 ) and p2 ( x 2 , y 2 )
    3. Input two corners (Let-top and right -bottom ) of the window , say(wx1

,wy1 and wx2 , wy2)

* + 1. Assign the region codes for two endpoints p1 and p2 using following steps : Initialize code with bits 0000

Set Bit1 =0 if (x < wx1 ) Set Bit2 =0 if (x < wx2 ) Set Bit3 =0 if (y < wy2 ) Set Bit4 =0 if (y < wy1 )

* + 1. Check for visibility of line
       1. If region codes for both endpoints p1 and p2 are zero then the line is completely visible. Hence draw the line and go to step 9
       2. If region codes for both endpoints are not zero and the logical ANDing of them is also nonzero then the line is completely invisible. So reject the ling and go to 9
       3. If region codes for two endpoints do not satisfy the condition in (4a and 4b the line is partially visible.

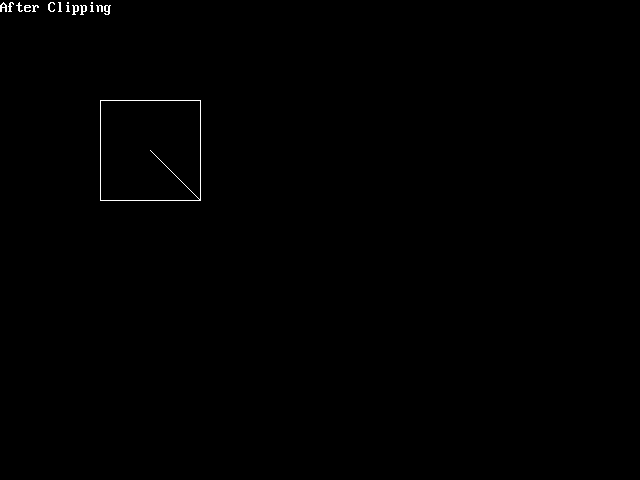
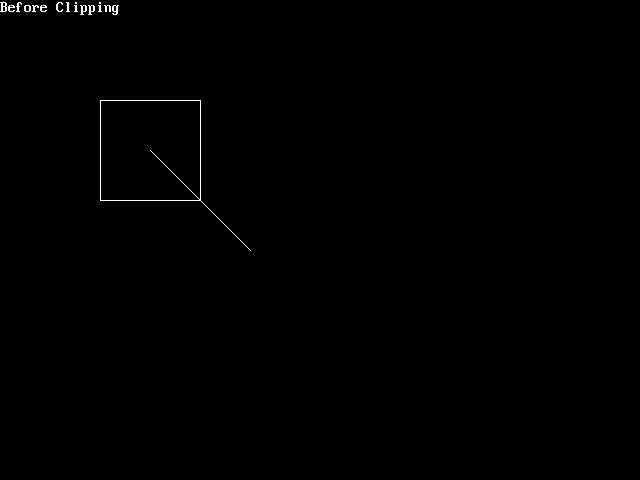
1. Determine the intersecting edge of the clipping window by inspecting the region codes of two endpoints
   1. If region codes for both endpoints are non- zero, find intersecting point p'1 and p'2 with boundary edges of clipping window with respect to point p1 and point p2 respectively
   2. If region codes for any one endpoints are non-zero, find intersecting point p'1 or p'2 with boundary edges of clipping window with respect to it.
2. Divide the Line segments considering intersection points
3. Reject the line segments if any one endpoint of it appears outsides the clipping window
4. Draw the remaining line segments
5. Stop

## Input:

Enter the window minimum coordinates: 100 100 Enter the window maximum coordinates: 200 200 Enter the starting point: 150 150

Enter the ending point: 200 200

## Output:

****

**Questions:**

* 1. What Is Cohen Sutherland clipping?
  2. Name the different algorithms of polygon clipping?
  3. What is mean of region code?
  4. Explain even odd and winding number method for visibility of line?
  5. Explain Sutherland Hodgeman Clipping?

# Experiment No:3

**Title:** Implement Following pattern using Line Drawing and Circle Drawing Algorithm.

**Aim:** Write a program to draw using following algorithms- Circle generation and Line Drawing.

## Prerequisites:

Line and Line segment, Slope of Line, General Equation of Line

## Objectives:

Understand DDA, Bresenham’s line drawing algorithm and Bresenham’s Circle drawing. Further using these algorithms to draw real time pictures.

## Theory:

**DDA Line Drawing:**

It is Digital Differential Analyzer (DDA).

It is also called as **Vector generation line drawing algorithm.**

Here we solve the differential equation for straight line. For drawing a line we need to turn ON the pixels which are on the line segment. For this we consider the slope of line.

For simplicity we divide the line segment in two types.

* + 1. Line with Gentle Slope.

Dx

Dy

Here Dx>Dy. We increment x by one unit and fine corresponding new value of y.

* + 1. Line with Sharpe Slope

Dy

Dx

Here Dy>Dx, we increment y by one unit and calculate corresponding new value of x.

## Algorithm:

**DDA line(Draw a figure)**

Read the line endpoints (X1, Y1) and (X2, Y2) such that they are not equal.

1. Calculate

dx = X2 – X1 dy = Y2 –Y1

1. if (dx >dy) then step = dx

else

end if

step = dy

1. Xinc = dx / step Yinc = dx / step
2. putpixel (X1, Y1,1) x = X1

y = Y1

5) i = 1

while (i <= step)

{

putpixel( x, ,y 1) x = x + dx

y = y + dy

i = i + 1

}

## Bresenham’s Line Drawing: (Draw a figure) Principle:

Either x or y is incremented by one unit depending on slope of line and increment in other variable is determined by checking the distance between true line segment and nearest pixel. This distance is called as **decision variable.**

Initially this decision variable s set as e = 2 \* dy - dx.

## Algorithm:

1. Read the line endpoints (X1, Y1) and (X2, Y2) such that they are not equal.
2. Calculate

dx = X2 – X1 dy = Y2 – Y1

1. Initialize

x = X1

y = Y1

1. Calculate decision variable e = 2 \* dy – dx

5) i = 1

putpixel ( x, y,1)

6) while ( e >= 0)

{

y = y + 1

e = e – 2 \* dx

}

x = x + 1

e = e + 2 \* dy

7) i = 1

1. if ( i <= dx) then go to step
2. Stop.

## Output:

**Questions:**

* 1. Explain drawbacks of DDA line drawing algorithm?
  2. Explain steps of Bresenham’s line drawing algorithm?
  3. Discuss DDA advantages and disadvantages?
  4. What is gentle slope and sharp slope?
  5. How to calculate Breasenham’s Circle generation?
  6. Explain mid-point circle drawing algorithm?
  7. What are the 8 symmetric points for any point (x,y)?