CG Assignment 21118

Title: Cohen-Sutherland line clipping algorithm

Problem statement: write a C++ program to implement Cohen-Sutherland line clipping algorithm.

Learning Outcomes:
1. learn Cohen-Sutherland line clipping algorithm
2. Implement the algorithm

Software & Hardware Requirements.
64-bit opensource OS creator
Windows 10 operating System.
8GB RAM, 512 GB SSD with 64-based architecture.

Theory:
1. Cohen-Sutherland line clipping algorithm is a 2d line clipping algorithm.
2. The main advantage of the algorithm is that it vastly reduces the number of line intersections that were to be calculated in scan conversions.
3. It operates in two phases:
   a) Region of code generation
   b) line clipping

* Region code-generation
It divides plane into a region & determines the visible portion using outcodes at the end points outcode is a four-bit number. Values of the bit is set to 1 if condition is satisfied.

$y>y_{max}$	$y<y_{max}$	$x>x_{max}$	$x<x_{min}$

a) set first bit if a point lies above the clipping window ie $y>y_{max}$

(b) Set second bit of a point lies to the right of the window i.e. $x > x_{max}$

(c) Set third bit of a point lies to the right of the window i.e. $y < y_{min}$

(d) Set the fourth bit if a point lies left of window i.e. $x < x_{min}$

1001	1000	1010	y_{max}
0001	0000	0010	
0101	0100	0110	y_{min}
x_{min}	x_{max}		

4) Clipping procedure

The algorithm quickly detects two cases if both end points are inside the window then line is completely visible & is accepted. It needs no clipping

Case I: The line is completely inside window, if logical OR operation yields 0000.

Case II: If logical OR is not 0000 then there are two possibilities. Either it is completely invisible or partially visible

- The line is completely outside the clipping window if logical AND operation of outcome doesn't yield to 0000.
- Above tests are performed in the same order i.e. case II is applicable only if case I is not true.

Case III:

- if logical AND of the outcode is not 0000 the line is partially invisible therefore it needs to be clipped
- if logical AND of outcode is 0000 the line may/may not pass from clipping window.
- Now compare the endpoints of the endpoints of boundry that have much line needs to be accepted

- consider the line with endpoints (x_1, y_1) (x_2, y_2)

$$\text{slope } (m) = \frac{y_2 - y_1}{x_2 - x_1}$$

- Using $y = mx + c$, compute $c = y - mx$.
- As both the points are known 'c' can be known.
- lets put $(x_1, y_1) \Rightarrow c = y_1 - mx_1$

The co-ordinate of an intersection with a vertical window boundry can be calculated as $y = mx + c$. x i.e. either x_{min} or x_{max}, depends on the which vertical line we are calculating y.

$y = mx_{min} + c$ .. left edge

$y = mx_{max} + c$ .. left edge

- Similarly, x-coordinate of an intersection with a horizontal window can be computed as.

$$x = \frac{y_{min} - c}{m} \quad \text{.. bottom edge}$$

$$x = \frac{y_{max} - c}{m} \quad \text{.. top edge}$$

Algorithm

1. Start
2. Read (x_{min}, y_{min}) & (x_{max}, y_{max}) for lowerleft & topright corner of clipping window.
3. Read $A(x_1, y_1)$ & $B(x_2, y_2)$ end points of the line.
4. Compute outcode of A & B

if $y_1 > y_{max}$ then, outcode $A(1) = 1$ else 0.

if $y_1 < y_{min}$ then, outcode $A(2) = 1$ else 0

if $x_1 > x_{max}$ then, outcode $A(3) = 1$ else 0

if $x_1 < x_{min}$ then, outcode $A(4) = 1$ else 0

if $y_2 > y_{max}$ then, outcode $A(5) = 1$ else 0

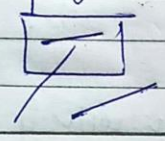
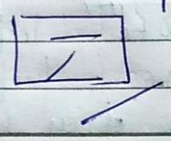
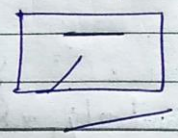
if $x_2 > x_{max}$ then outcode $A(6) = 1$ else 0

if $y_2 < y_{min}$ then, outcode $A(7) = 1$ else 0

if $x_2 < x_{min}$ then, outcode $A(8) = 1$ else 0.

5. If outcode A or outcode B = 0000 then display entire line, go to step 5.
else if outcode A & outcode B ≠ 0000 then reject the entire line.
Else compute the intersection point with window boundaries.
6. Repeat step 4.
6. End program

Test Cases

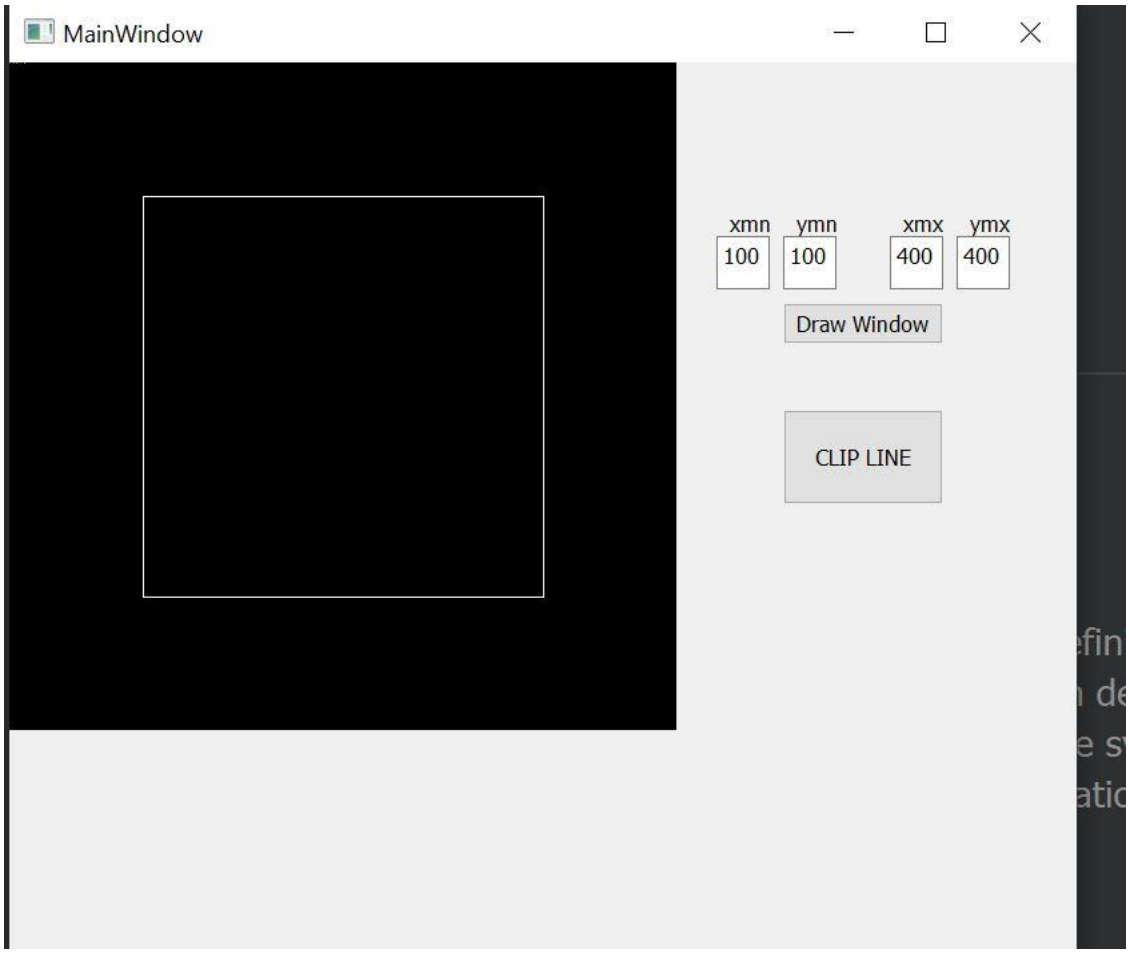
Test No.	Input	Expected output	Actual o/p	Result
1				pass

Conclusion

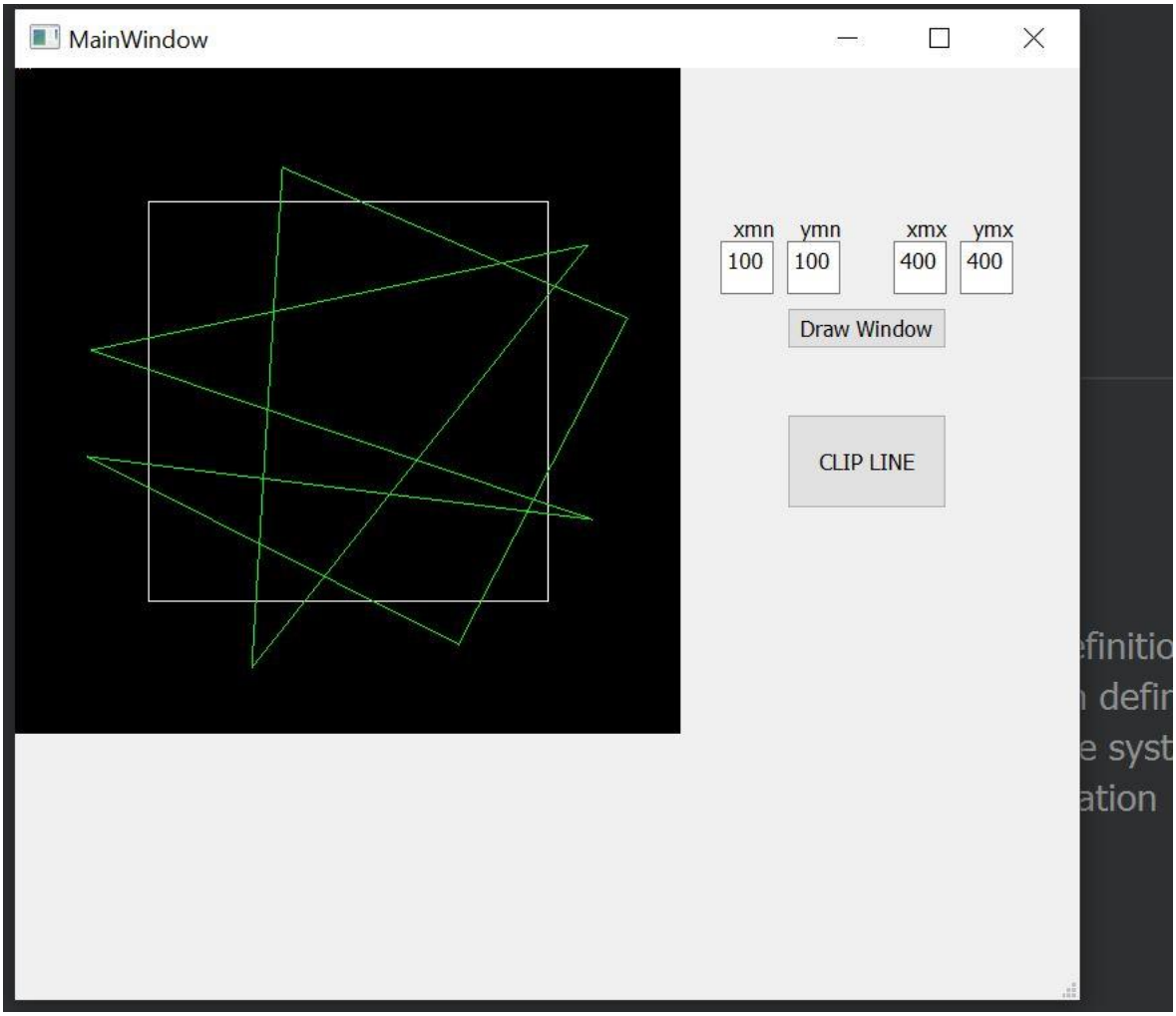
We understood the concept of Cohen-Sutherland line clipping algorithm & were able to implement it.

Output:

1. Drawing Window



2. Drawing Polygon



3. Clipping Polygon using cohen Sutherland line clipping algorithm