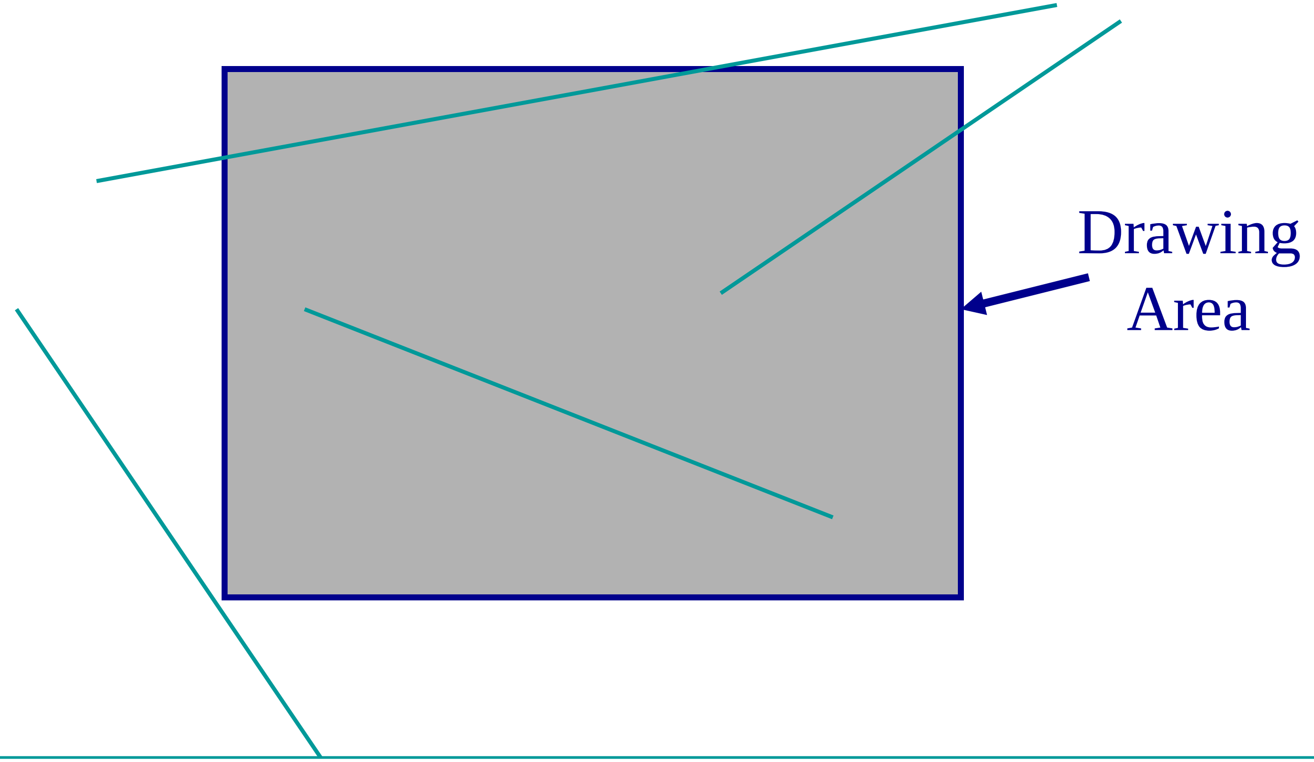
Clipping

Comp 535

Line Clipping

What happens when one or both endpoints of a line segment are not inside the specified drawing area?

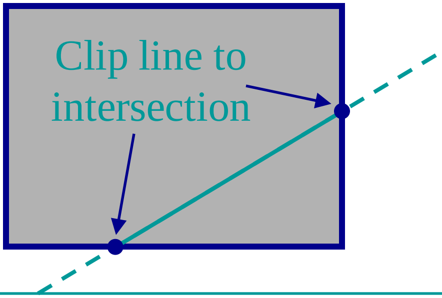


Line Clipping

- Strategies for clipping:
  a) Check (in inner loop) if each point is inside → Works, but ***slow***

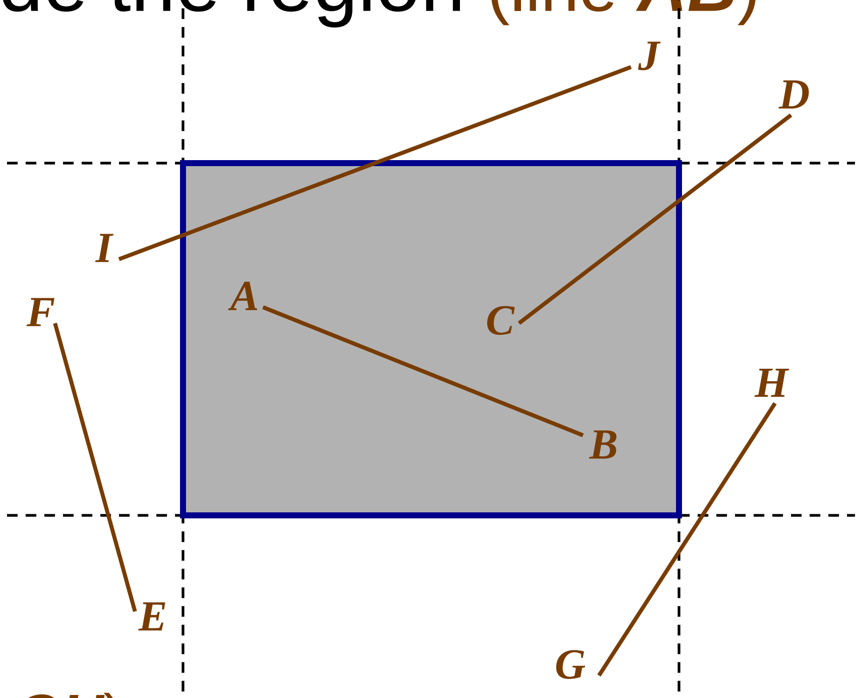
```
if (x ≥ xmin and x ≤ xmax and y ≥ ymin and y ≤ ymax)
  drawPoint(x,y,c);
```

  b) Find intersection of line with boundary → Correct



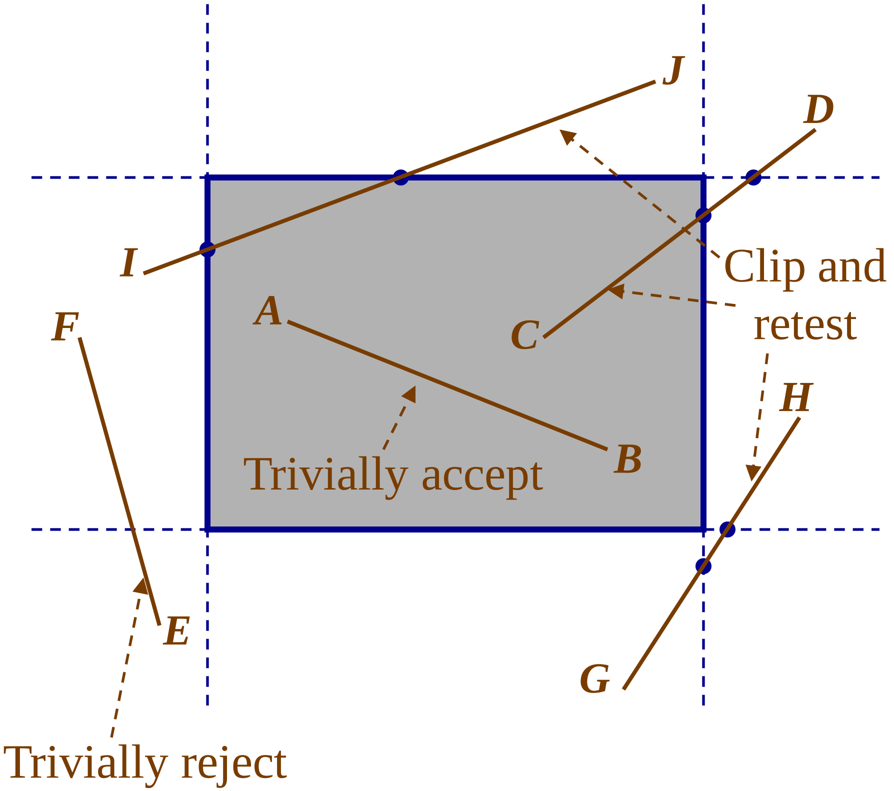
Line Clipping: Possible Configurations

1. Both endpoints are inside the region (line ***AB***)
   - No clipping necessary
2. One endpoint in, one out (line ***CD***)
   - Clip at intersection point
3. Both endpoints outside the region:
   a. No intersection (lines ***EF***, ***GH***)
   b. Line intersects the region (line ***IJ***)
      - Clip line at both intersection points



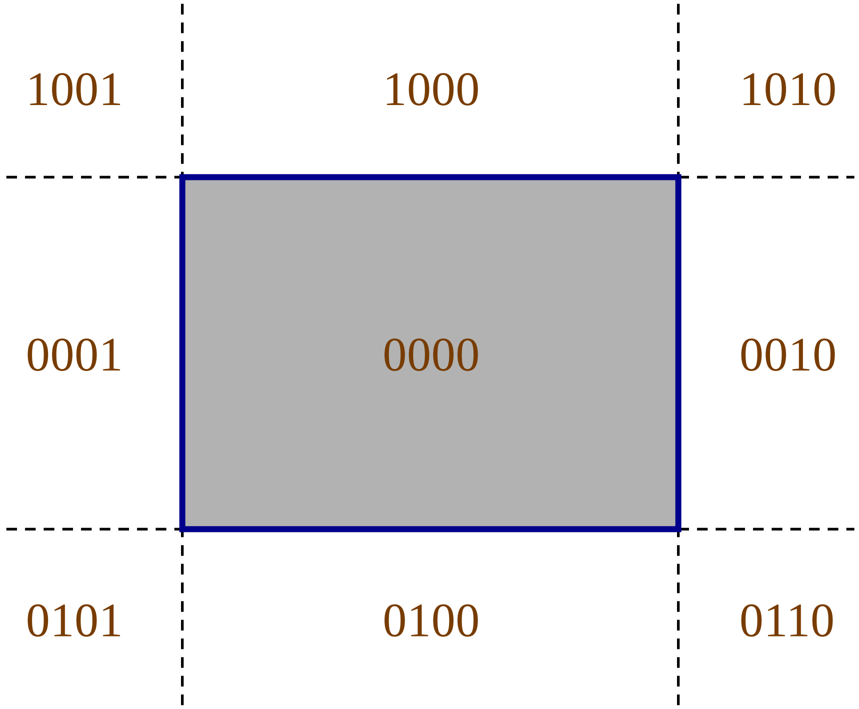
Line Clipping: Cohen-Sutherland

- Basic algorithm:
  - Accept lines that have both endpoints inside the region.
  - Reject lines that have both endpoints less than x_{min} or y_{min} or greater than x_{max} or y_{max}.
  - Clip the remaining lines at a region boundary and repeat the previous steps on the clipped line segments.



Cohen-Sutherland: Accept/Reject Tests

- Assign a 4-bit code to each endpoint c_0, c_1 based on its position:
  - 1st bit (1000): if $y > y_{max}$
  - 2nd bit (0100): if $y < y_{min}$
  - 3rd bit (0010): if $x > x_{max}$
  - 4th bit (0001): if $x < x_{min}$

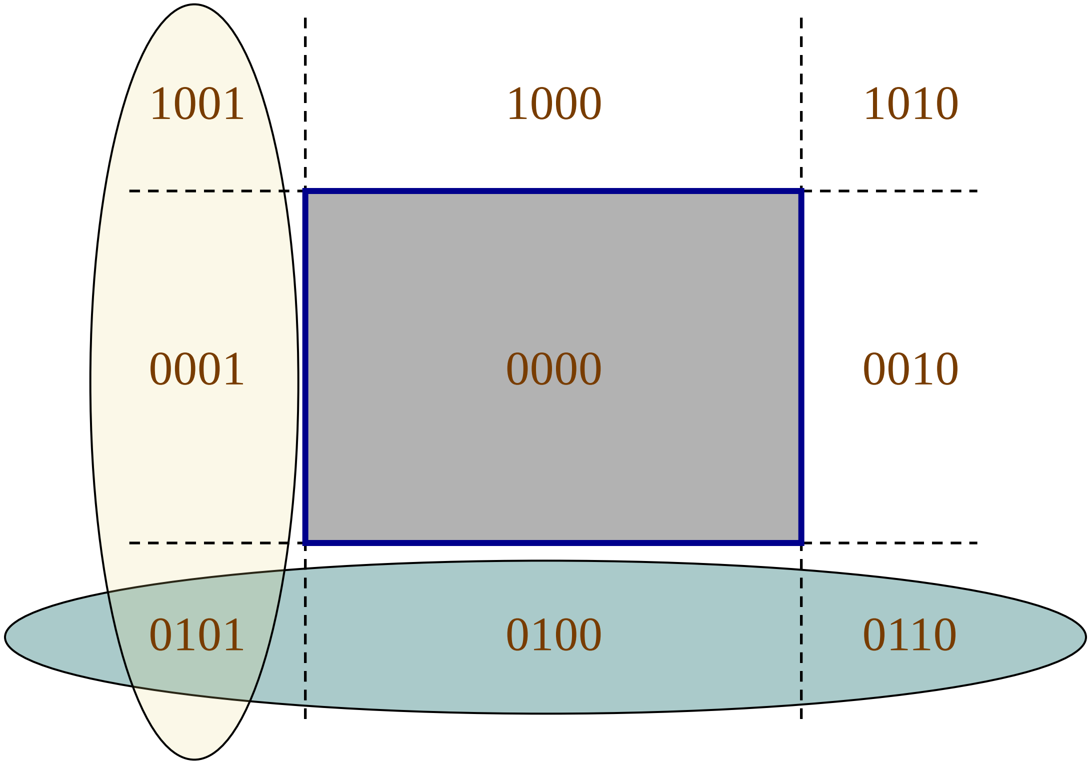


- Test using bitwise functions

if $c_0 \mid c_1 = 0000$
    accept (draw)
else if $c_0 \,\&\, c_1 \neq 0000$
    reject (don't draw)
else clip and retest

Cohen-Sutherland Accept/Reject

- Accept/reject/redo all based on bit-wise Boolean ops.



Cohen-Sutherland: Overview

1. Choose an endpoint outside the clipping region.
2. Clip to a boundary using a consistent ordering (top to bottom, left to right).
3. Set the new line to have as endpoints the new intersection point and the other original endpoint.
4. You may need to run this a few times on a single line.

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

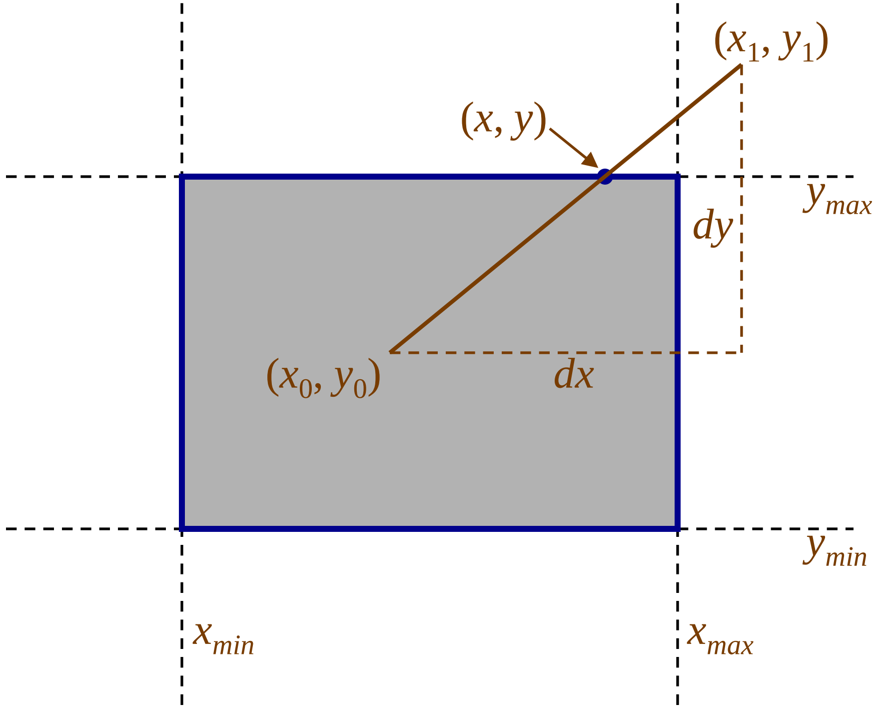
else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

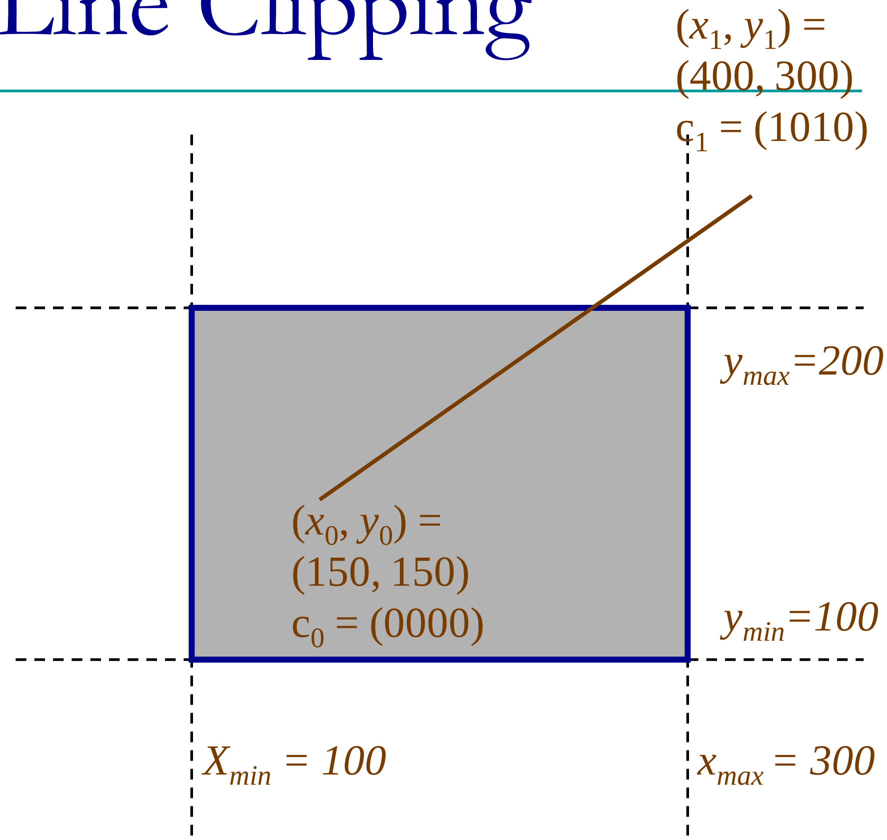
else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
$x_0 = x$; $y_0 = y$;

else
$x_1 = x$; $y_1 = y$;



Cohen-Sutherland: Line Clipping



Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000   // y_{max}
  $x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

else if c & 0100   // y_{min}
  $x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010   // x_{max}
  $y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else   // x_{min}
  $y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
  $x_0 = x$; $y_0 = y$;

else
  $x_1 = x$; $y_1 = y$;

_c_ __dx__ __dy__ __x__ __y__

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

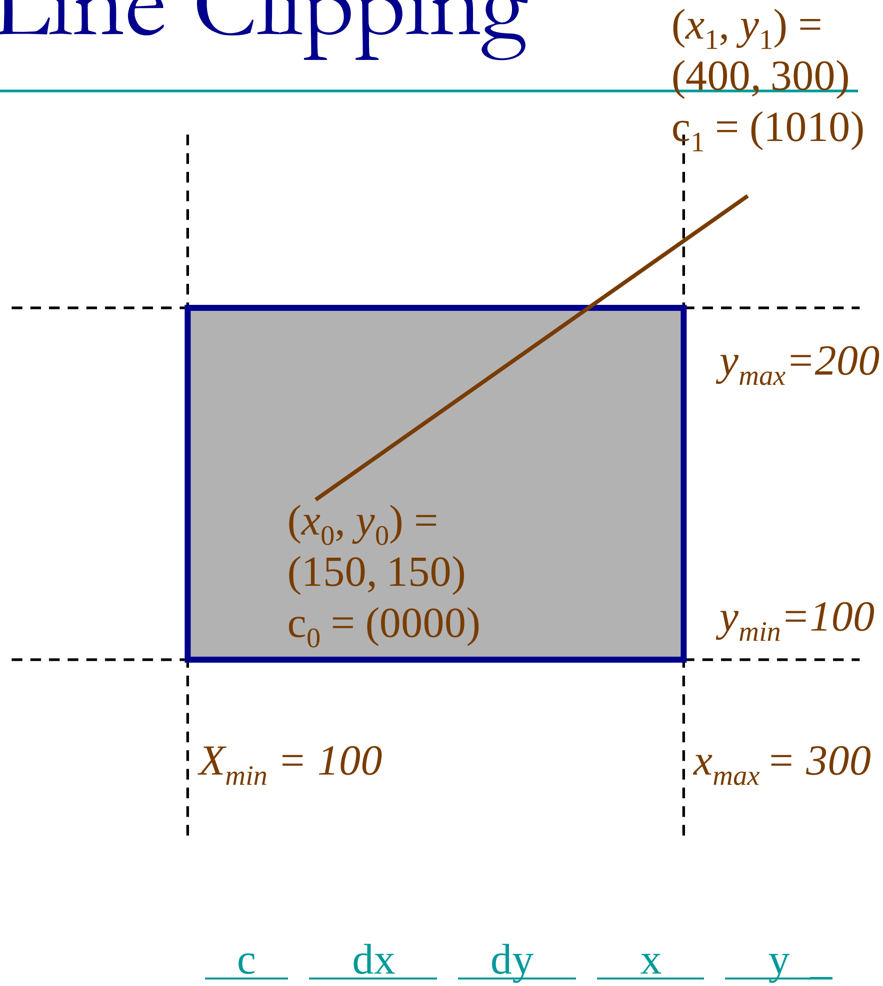
else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
$x_0 = x$; $y_0 = y$;

else
$x_1 = x$; $y_1 = y$;



Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
 $x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

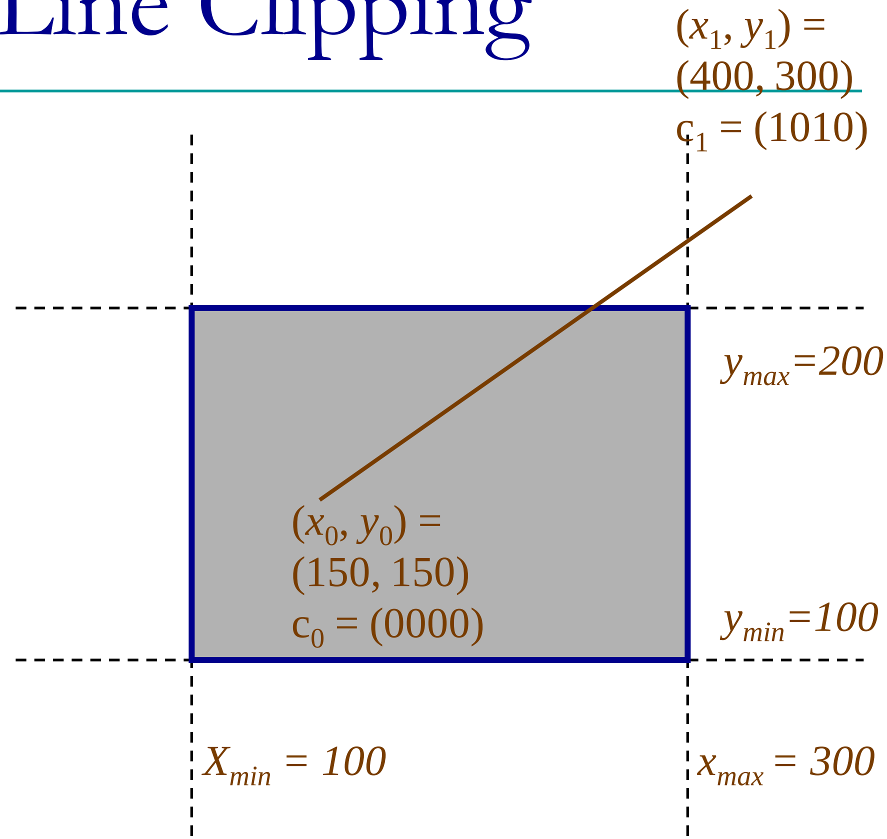
else if c & 0100 // y_{min}
 $x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
 $y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
 $y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
 $x_0 = x$; $y_0 = y$;

else
 $x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010				

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

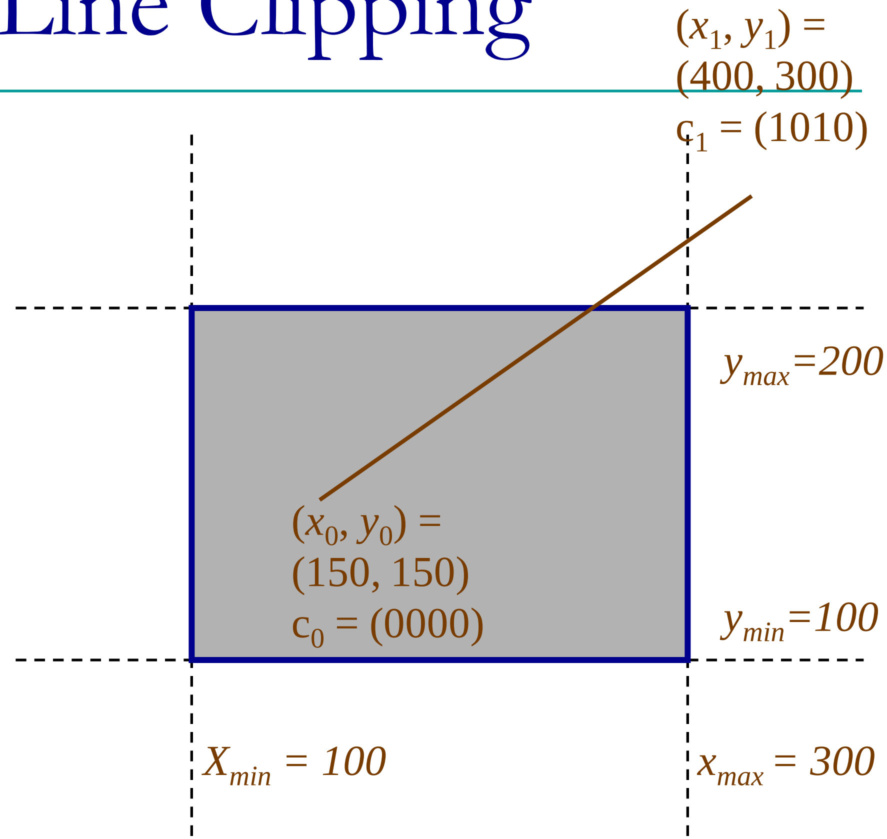
else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
$x_0 = x$; $y_0 = y$;

else
$x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010	250	150		

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
  $x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

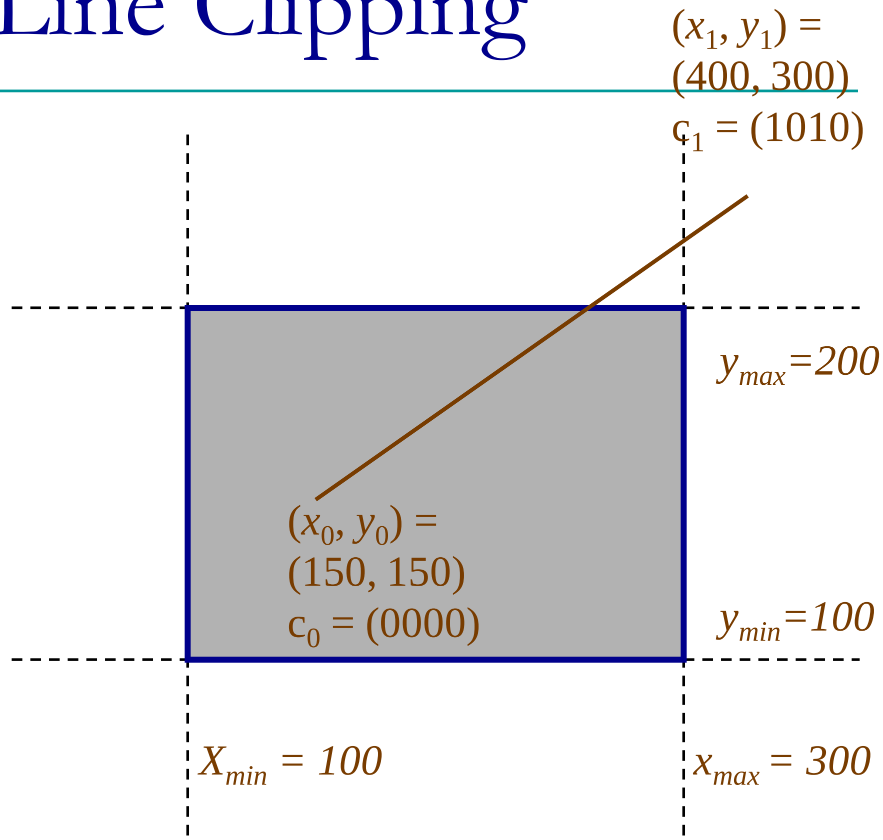
else if c & 0100 // y_{min}
  $x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
  $y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
  $y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
  $x_0 = x$; $y_0 = y$;

else
  $x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010	250	150		

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

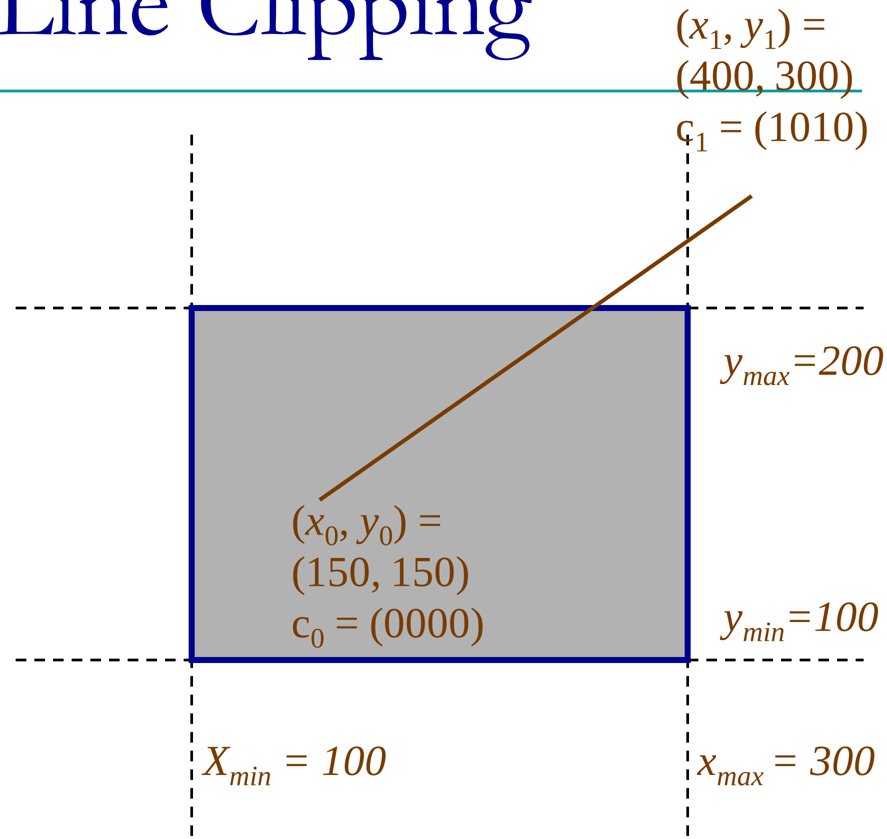
else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
$x_0 = x$; $y_0 = y$;

else
$x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010	250	150	233	200

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

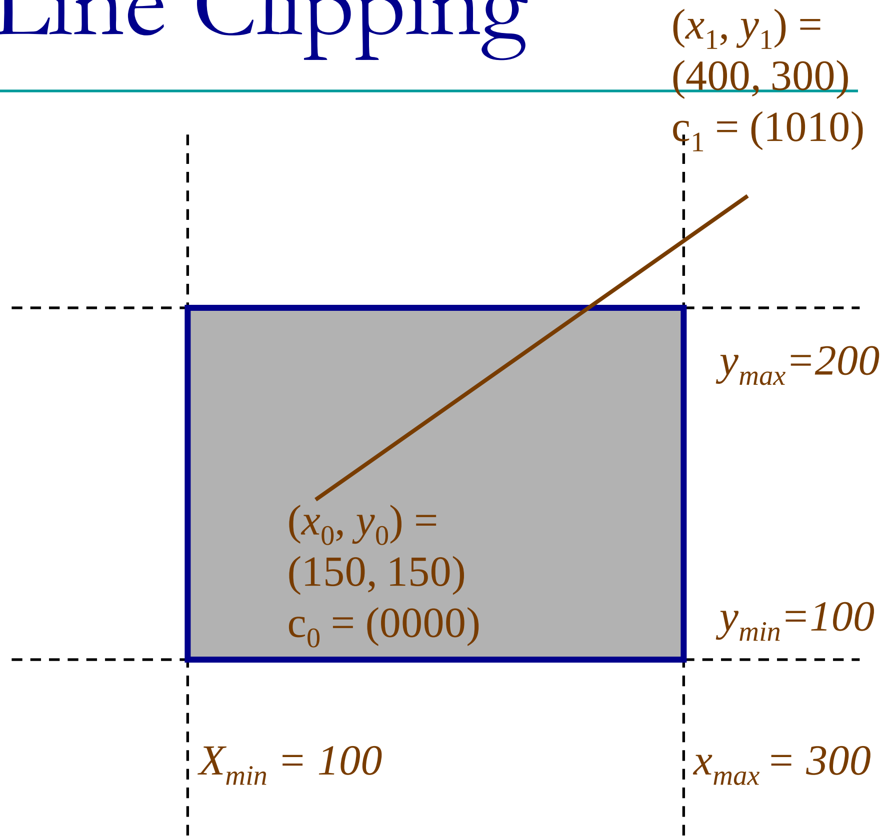
else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$

$x_0 = x$; $y_0 = y$;

else
$x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010	250	150	233	200

Cohen-Sutherland: Line Clipping

Intersection algorithm:

if $c_0 \neq 0000$ then $c = c_0$;
else $c = c_1$;

$dx = x_1 - x_0$; $dy = y_1 - y_0$
if c & 1000 // y_{max}
$x = x_0 + dx * (y_{max} - y_0) / dy$; $y = y_{max}$;

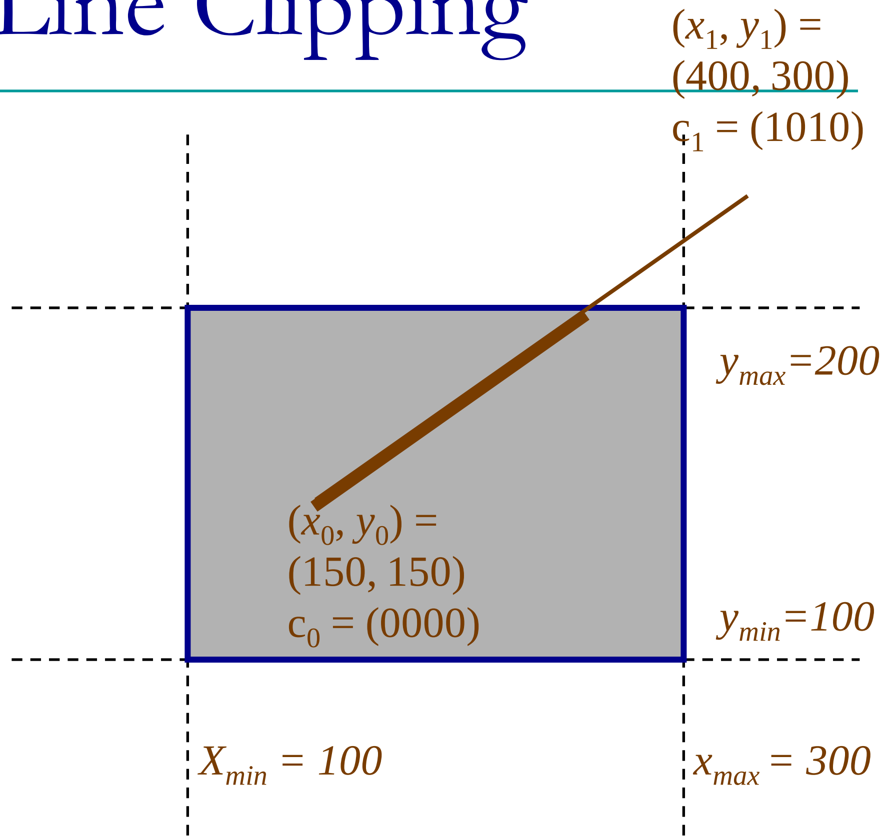
else if c & 0100 // y_{min}
$x = x_0 + dx * (y_{min} - y_0) / dy$; $y = y_{min}$;

else if c & 0010 // x_{max}
$y = y_0 + dy * (x_{max} - x_0) / dx$; $x = x_{max}$;

else // x_{min}
$y = y_0 + dy * (x_{min} - x_0) / dx$; $x = x_{min}$;

if $c = c_0$
$x_0 = x$; $y_0 = y$;

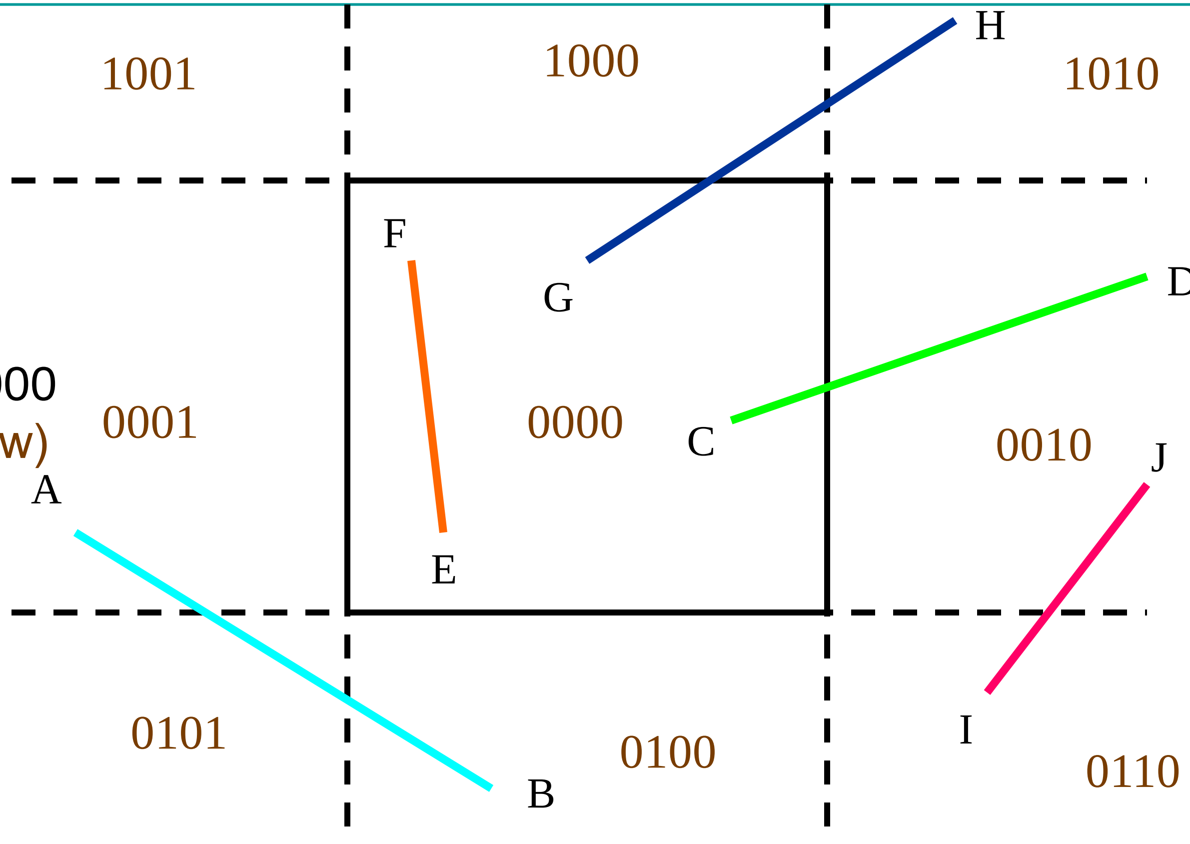
else
$x_1 = x$; $y_1 = y$;



c	dx	dy	x	y
1010	250	150	233	200

Cohen-Sutherland Line Clip Examples

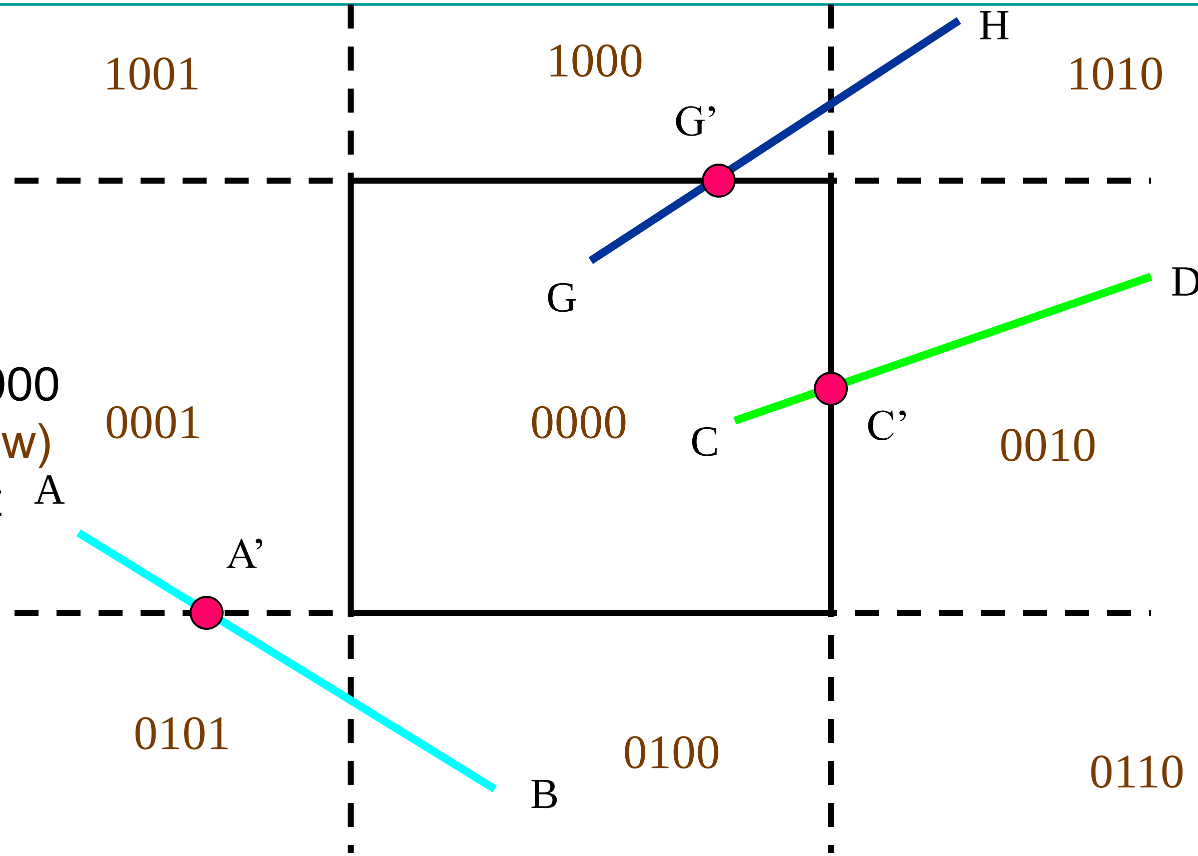
if $c0$ | $c1$ = 0000
    accept (draw)
else if $c0$ & $c1 \neq$ 0000
    reject (don't draw)
else clip and retest



A 0001	C 0000	E 0000	G 0000	I 0110
B 0100	D 0010	F 0000	H 1010	J 0010
OR 0101	OR 0010	OR 0000	OR 1010	OR 0110
AND 0000	AND 0000	AND 0000	AND 0000	AND 0010
clip	**clip**	**accept**	**clip**	**reject**

Cohen-Sutherland Line Clip Examples

if $c0 \mid c1 = 0000$
    accept (draw)
else if $c0 \ \& \ c1 \neq 0000$
    reject (don't draw)
else clip and retest



remove
A'B

A 0001
A' 0001
OR 0001
AND 0001
reject

remove
C'D

C 0000
C' 0000
OR 0000
AND 0000
accept

remove
G'H

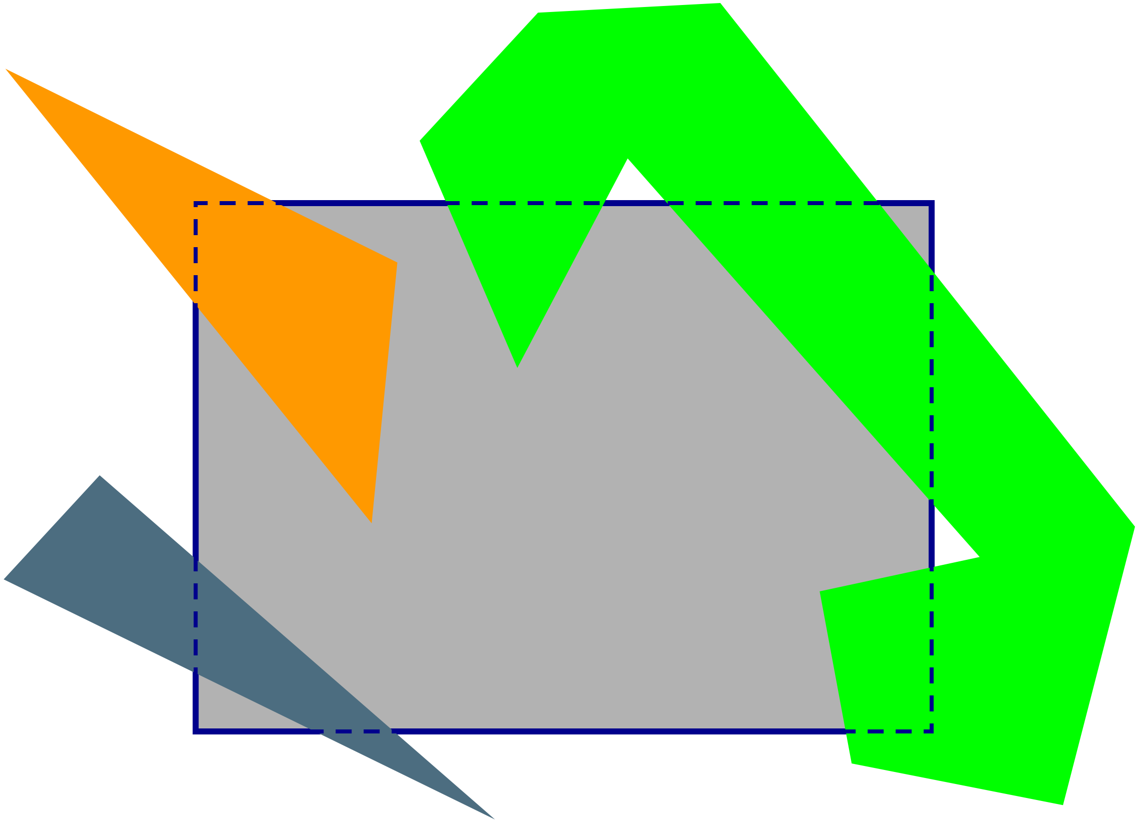
G 0000
G' 0000
OR 0000
AND 0000
accept

Cohen-Sutherland: Summary

1. Choose an endpoint outside the clipping region.
2. Clip to a boundary using a consistent ordering (top to bottom, left to right).
3. Set the new line to have as endpoints the new intersection point and the other original endpoint.
4. You may need to run this a few times on a single line.

Polygon Clipping

What about polygons?



Polygon Clipping: Algorithm

- Clip polygon to y_{min} and y_{max}:
  - Create empty output vertex list
  - Process input list $(\boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_n)$ where $\boldsymbol{v}_0 = \boldsymbol{v}_n$
  - For each input vertex $(\boldsymbol{v}_i$ where $0 \leq i \leq n-1)$:
    - If $\boldsymbol{v}_i$ is inside region → Add $\boldsymbol{v}_i$ to output list.
    - If the line between $\boldsymbol{v}_i$ and $\boldsymbol{v}_{i+1}$ intersects clipping boundaries → Add intersection point(s) to output list.
- Repeat: Clip to x_{min} and x_{max}
- Post-process:
  - Remove degenerate sections that have collapsed to region boundary.

Polygon Clipping: Example

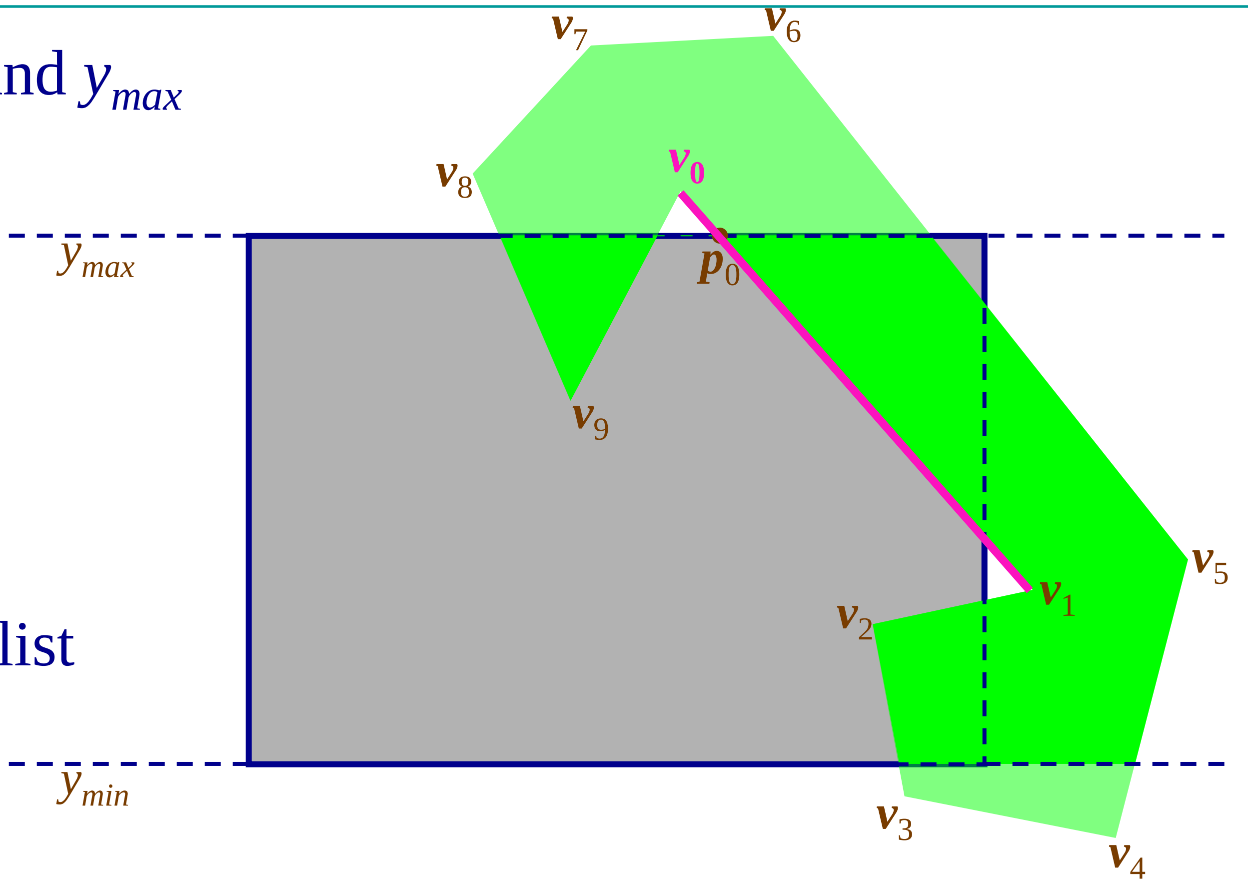
Clip first to y_{min} and y_{max}

vertex: v_0

Inside region: No

Line intersect boundary: Yes

Add p_0 to output list



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: p_0

Polygon Clipping: Example

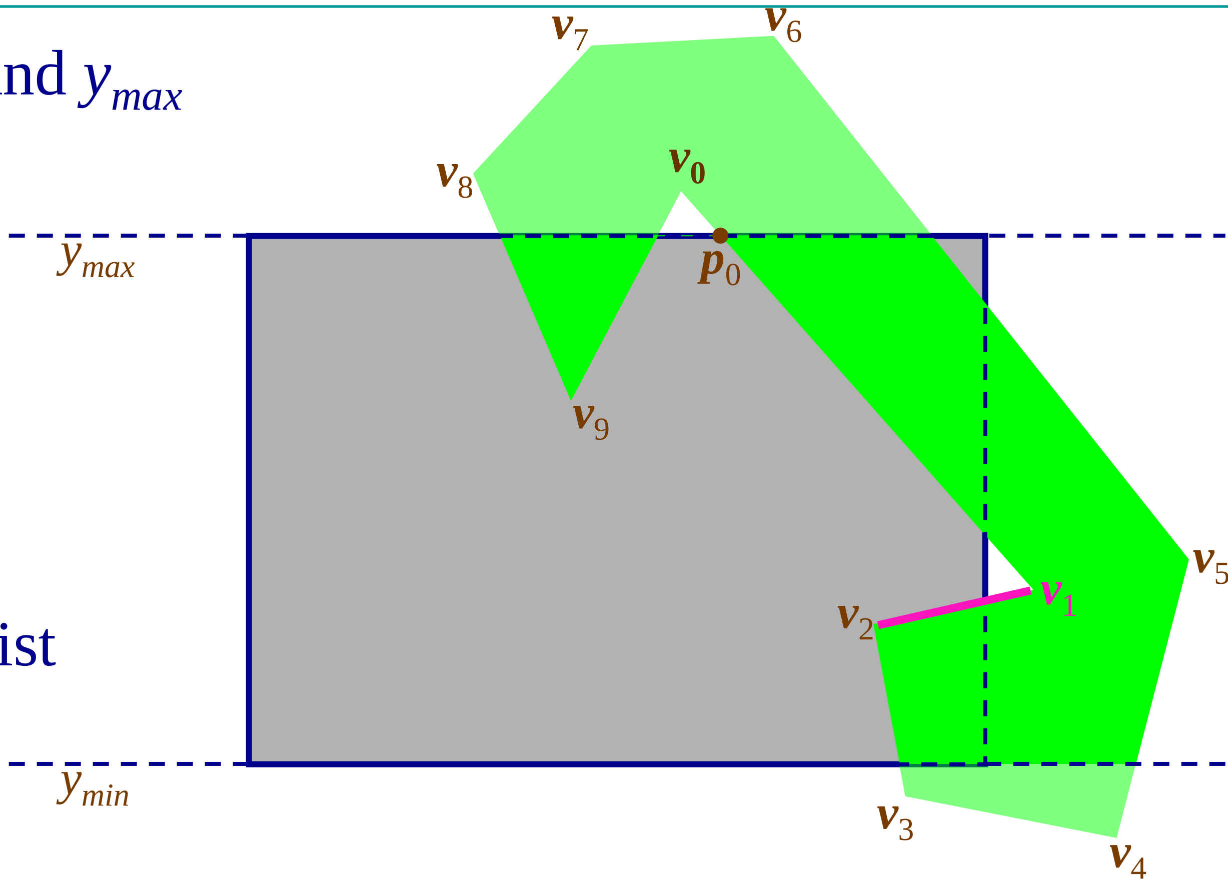
Clip first to y_{min} and y_{max}

vertex: v_1

inside region: yes

line intersect
boundary: no

add v_1 to output list



Input vertex list: (v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)

Output vertex list: p_0 , v_1

Polygon Clipping: Example

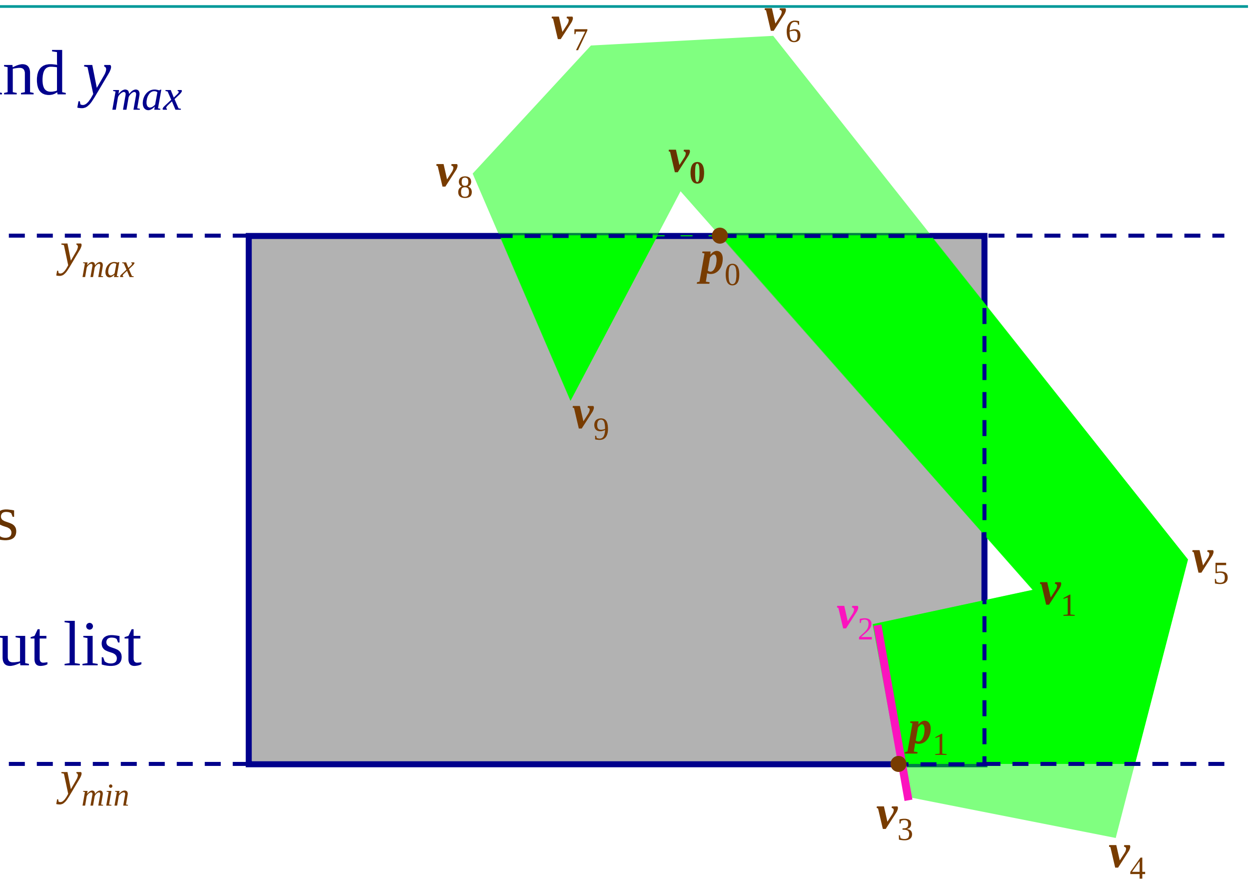
Clip first to y_{min} and y_{max}

vertex: v_2

inside region: yes

line intersect
boundary: yes

add v_2, p_1 to output list



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: p_0, v_1, v_2, p_1

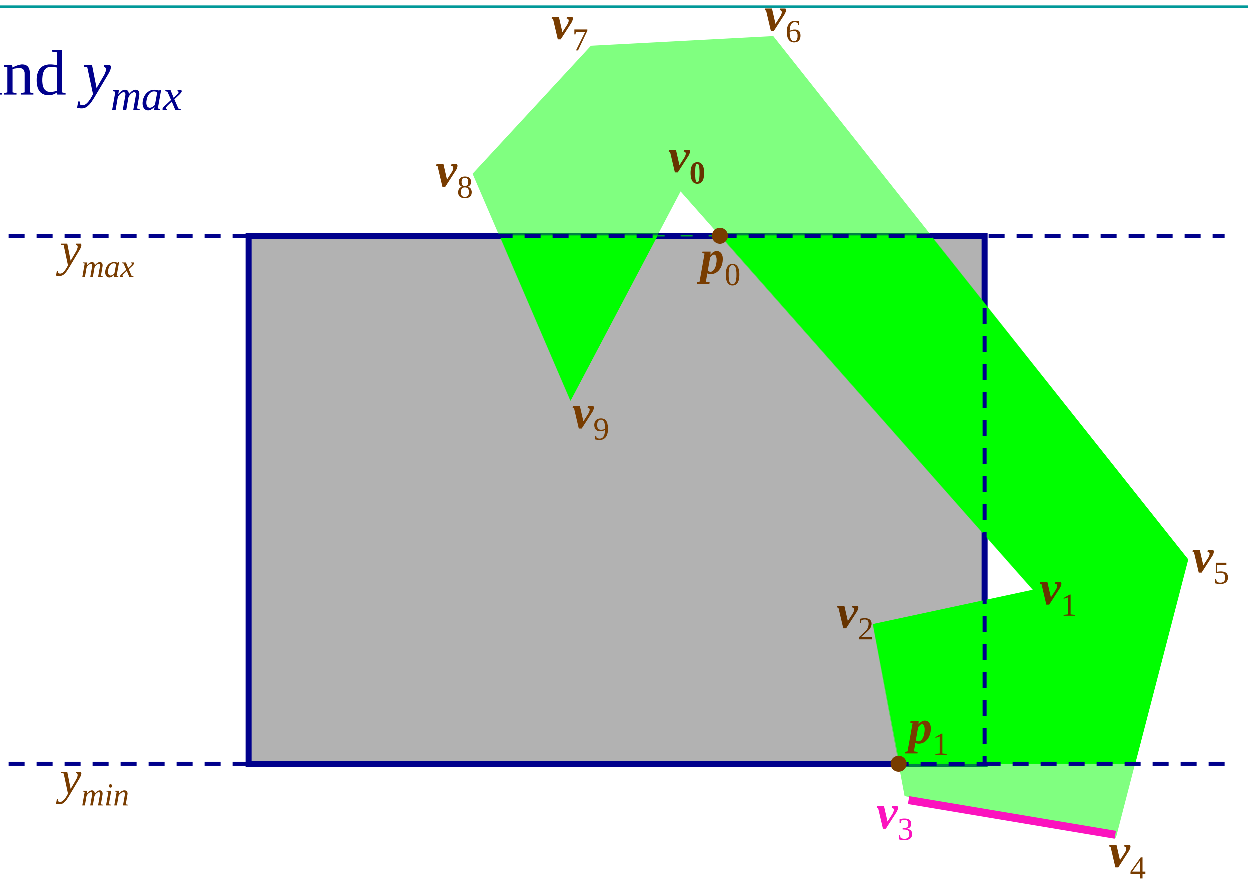
Polygon Clipping: Example

Clip first to y_{min} and y_{max}

vertex: $\boldsymbol{v_3}$

inside region: no

line intersect
boundary: no



Input vertex list: $(\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4, \boldsymbol{v}_5, \boldsymbol{v}_6, \boldsymbol{v}_7, \boldsymbol{v}_8, \boldsymbol{v}_9)$

Output vertex list: $\boldsymbol{p}_0, v_1, v_2, p_1$

Polygon Clipping: Example

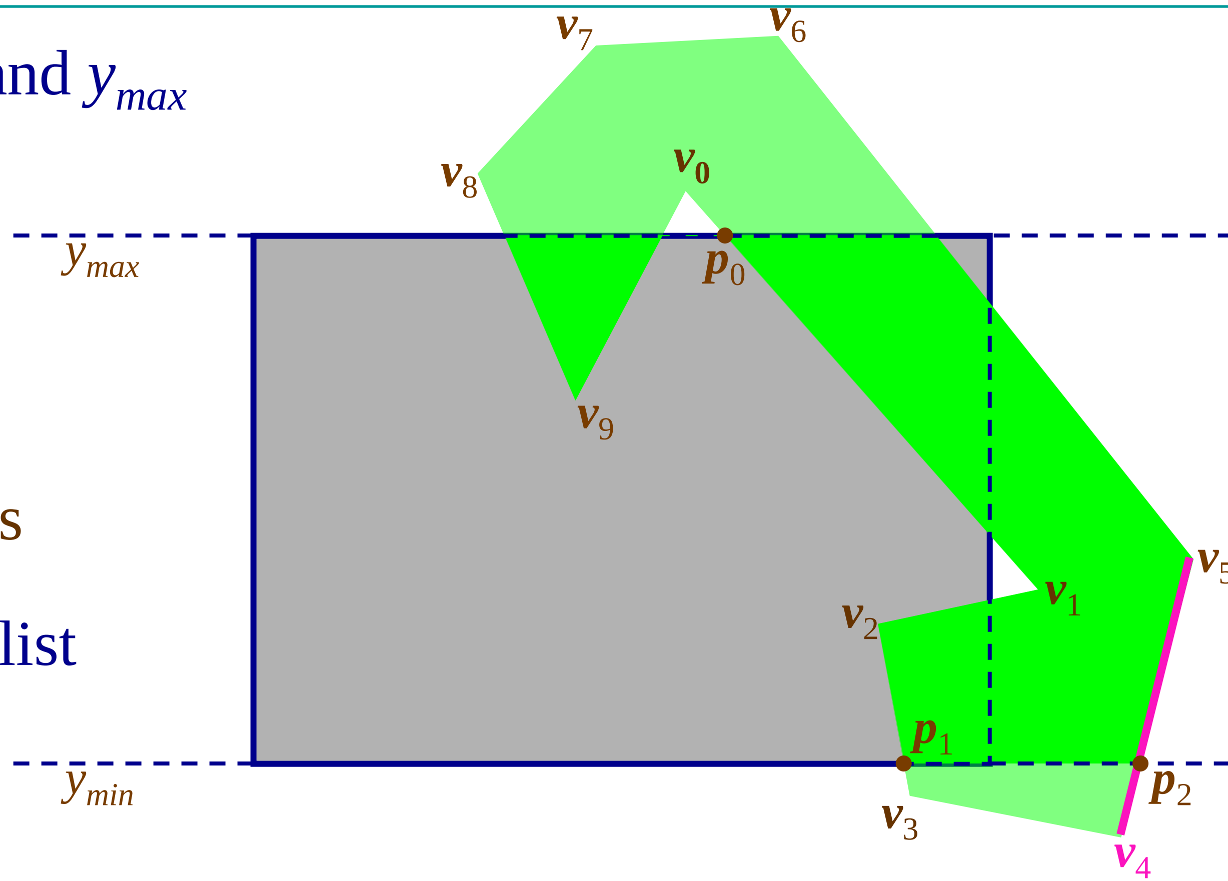
Clip first to y_{min} and y_{max}

vertex: $\boldsymbol{v_4}$

inside region: no

line intersect boundary: yes

add p_2 to output list



Input vertex list: (v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)

Output vertex list: p_0, v_1, v_2, p_1, p_2

Polygon Clipping: Example

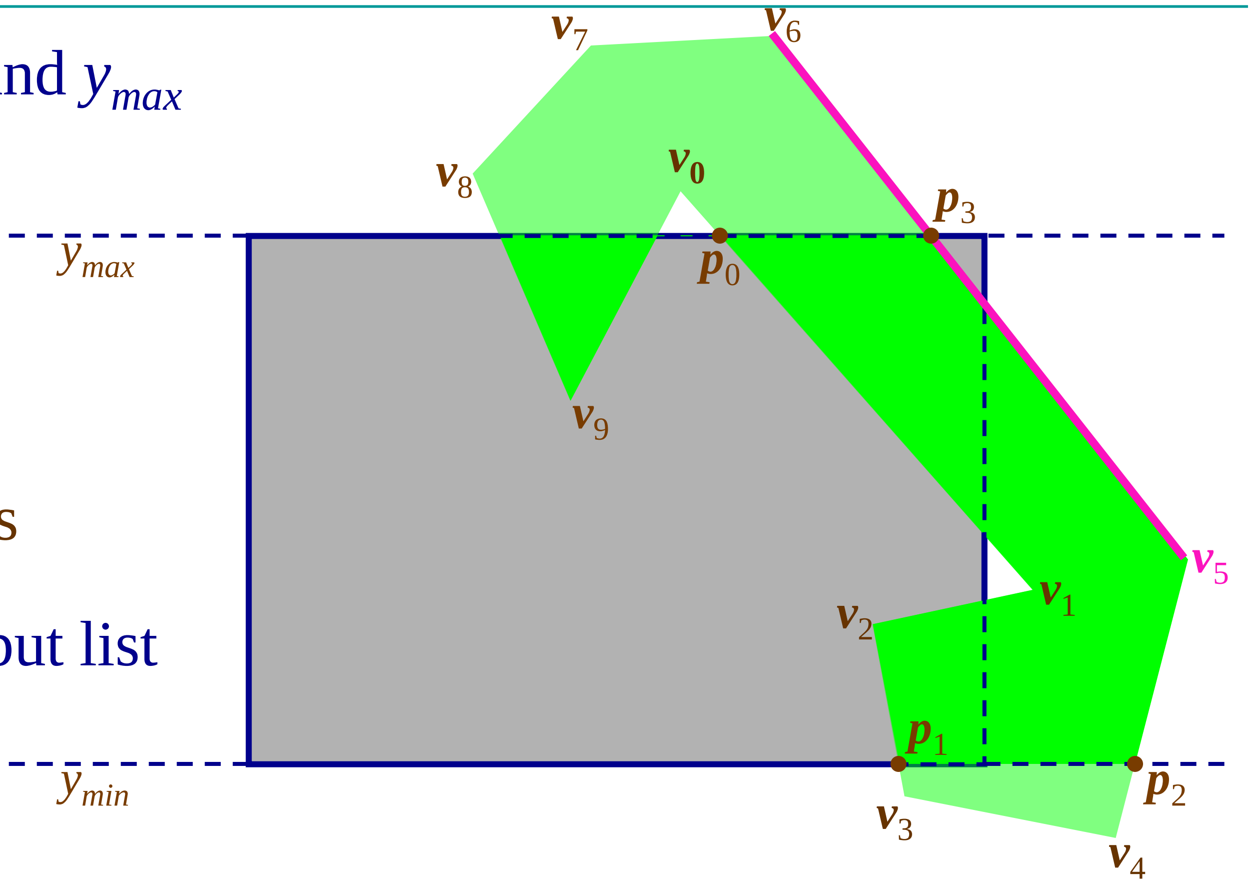
Clip first to y_{min} and y_{max}

vertex: v_5

inside region: yes

line intersect
boundary: yes

add v_5, p_3 to output list



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: $p_0, v_1, v_2, p_1, p_2, v_5, p_3$

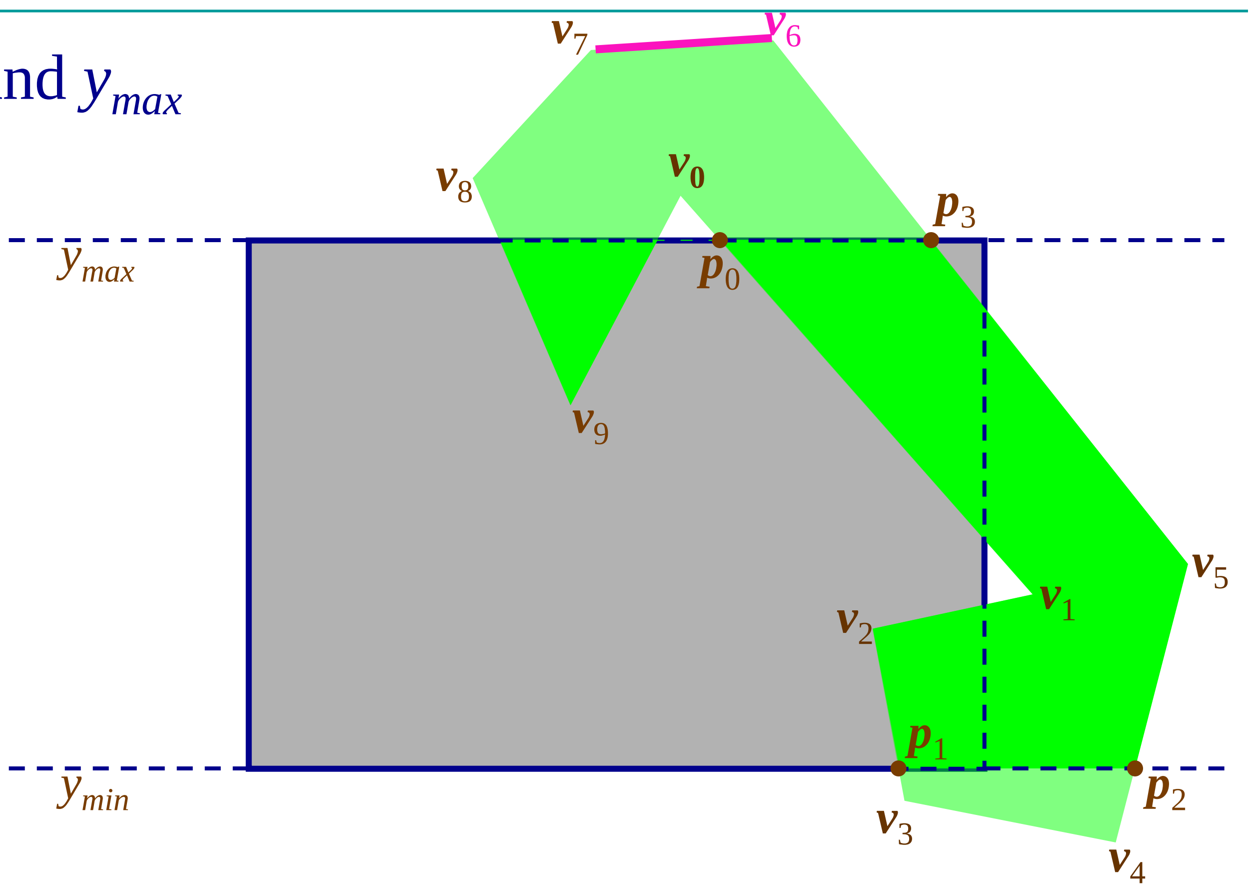
Polygon Clipping: Example

Clip first to y_{min} and y_{max}

vertex: $\boldsymbol{v_6}$

inside region: no

line intersect
boundary: no



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: $p_0, v_1, v_2, p_1, p_2, v_5, p_3$

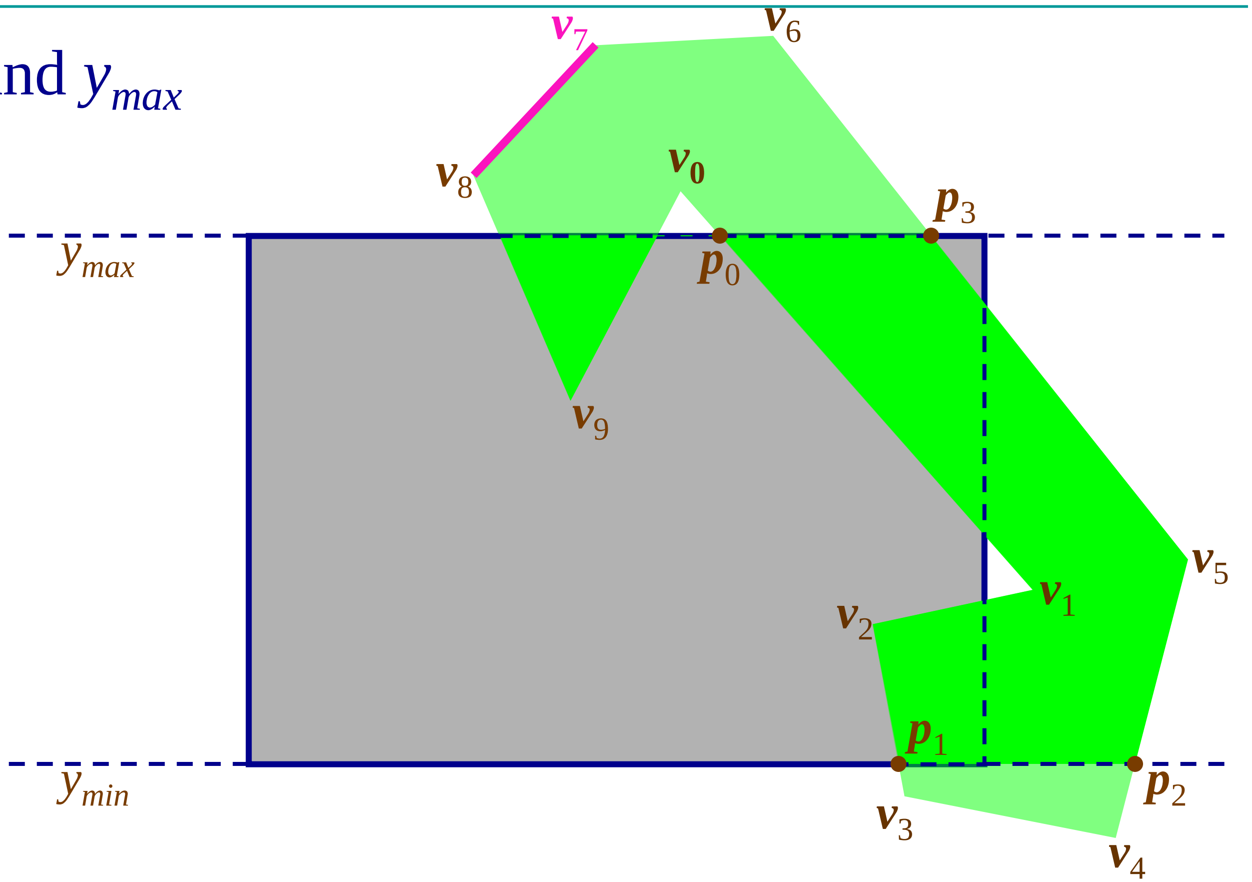
Polygon Clipping: Example

Clip first to y_{min} and y_{max}

vertex: v_7

inside region: no

line intersect
boundary: no



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: $p_0, v_1, v_2, p_1, p_2, v_5, p_3$

Polygon Clipping: Example

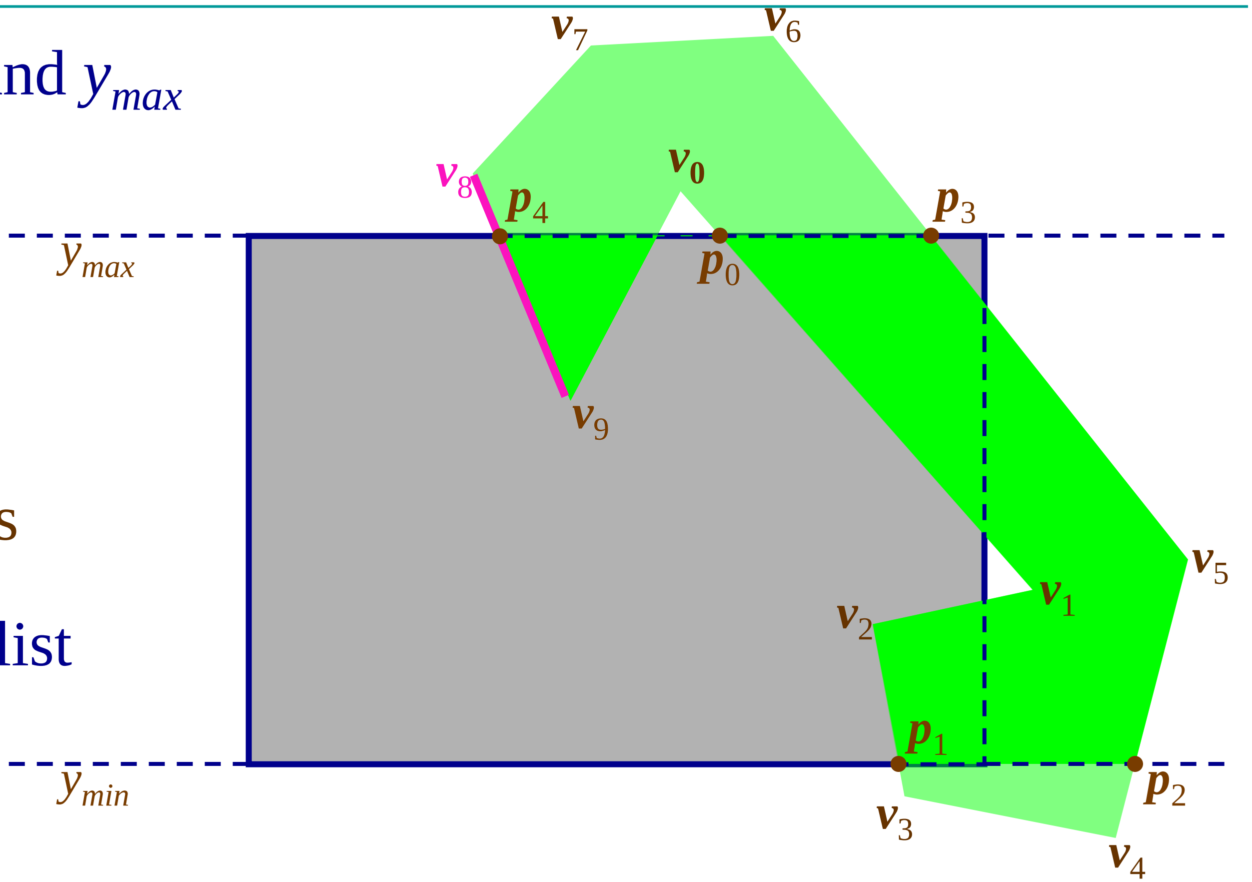
Clip first to y_{min} and y_{max}

vertex: v_8

inside region: no

line intersect
boundary: yes

add p_4 to output list



Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: $p_0, v_1, v_2, p_1, p_2, v_5, p_3, p_4$

Polygon Clipping: Example

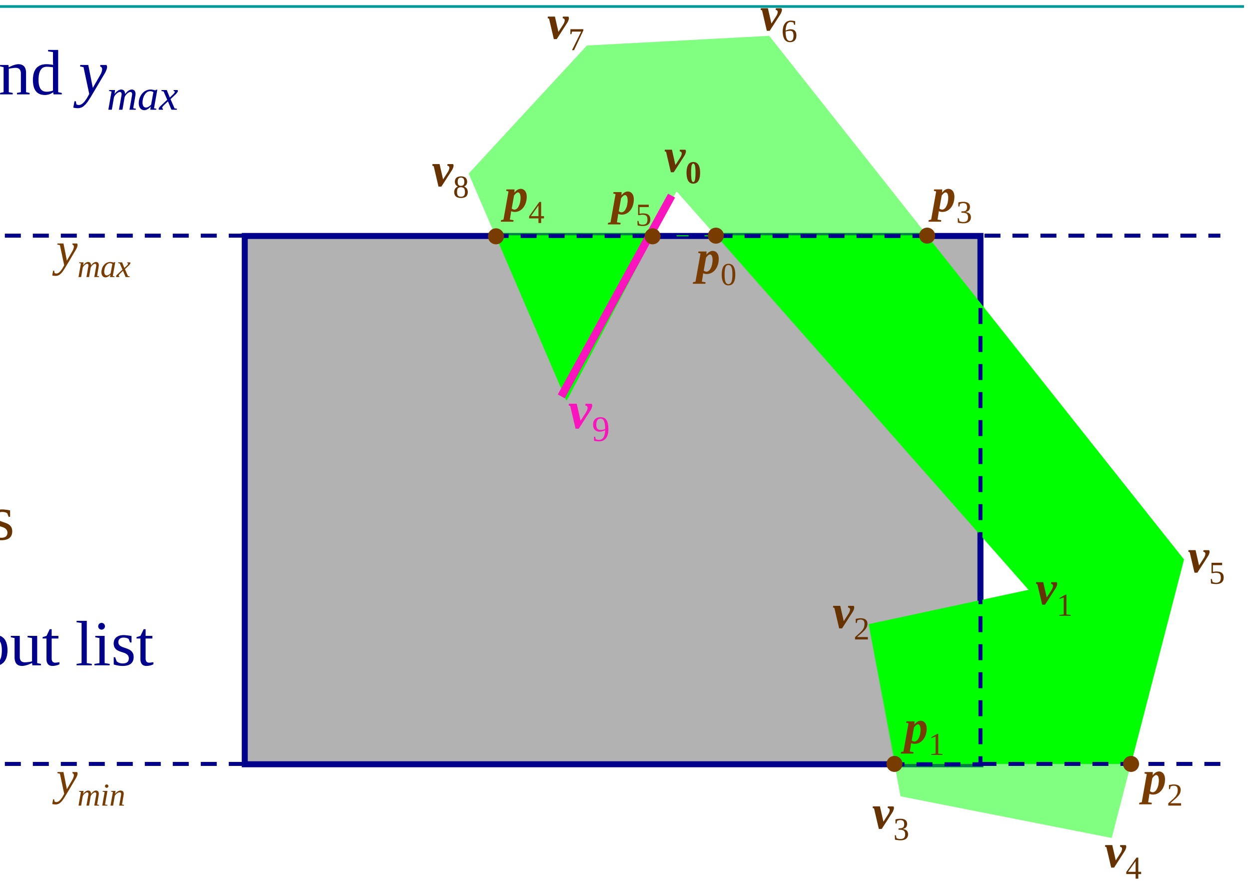
Clip first to y_{min} and y_{max}

vertex: $\boldsymbol{v_9}$

inside region: yes

line intersect
boundary: yes

add v_9, p_5 to output list

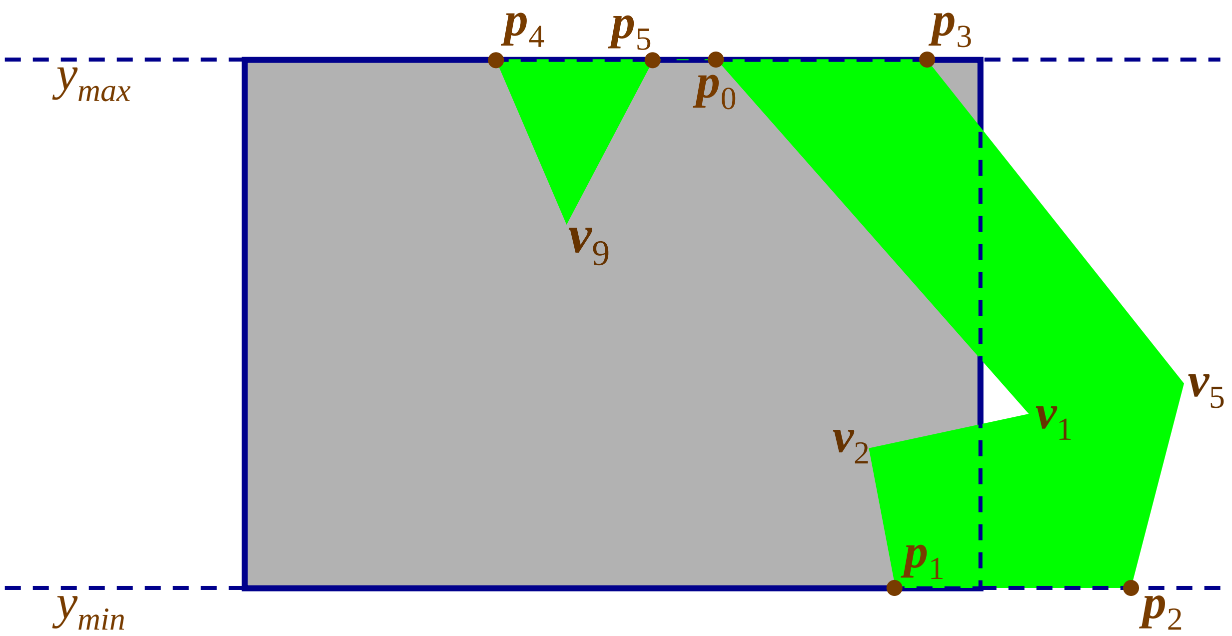


Input vertex list: $(v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9)$

Output vertex list: $p_0, v_1, v_2, p_1, p_2, v_5, p_3, p_4, v_9, p_5$

Polygon Clipping: Example

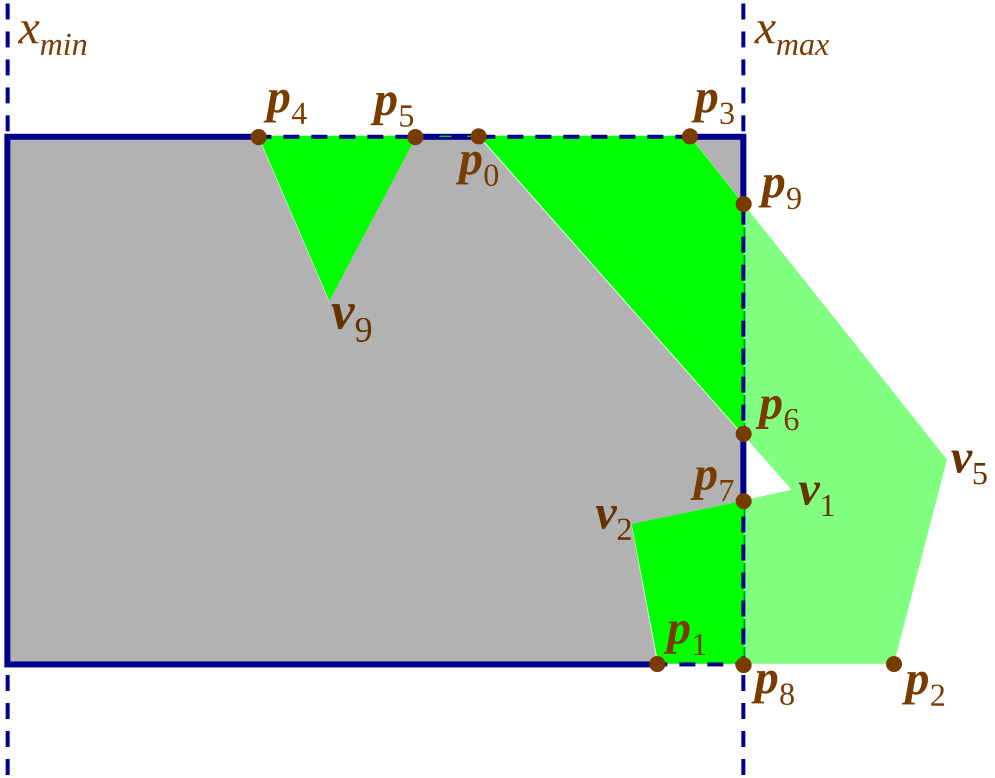
This gives us a new polygon



with vertices: $(\boldsymbol{p}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{v}_5, \boldsymbol{p}_3, \boldsymbol{p}_4, \boldsymbol{v}_9, \boldsymbol{p}_5)$

Polygon Clipping: Example (cont.)

Now clip to x_{min} and x_{max}

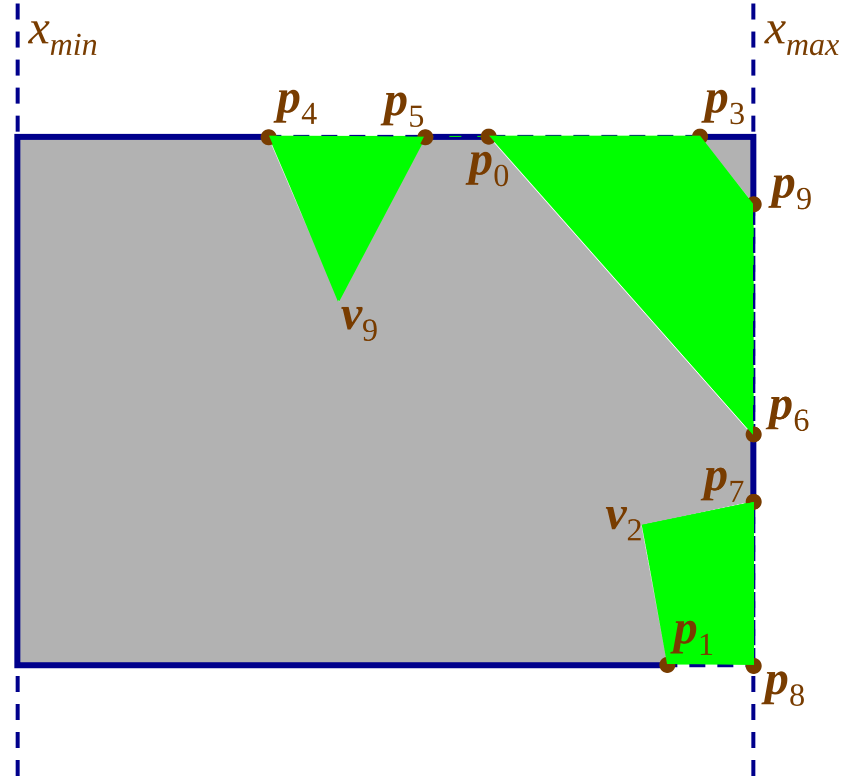


Input vertex list: $= (p_0, v_1, v_2, p_1, p_2, v_5, p_3, p_4, v_9, p_5)$

Output vertex list: $(p_0, p_6, p_7, v_2, p_1, p_8, p_9, p_3, p_4, v_9, p_5)$

Polygon Clipping: Example (cont.)

Now post-process



Output vertex list: $(\boldsymbol{p}_0, \boldsymbol{p}_6, \boldsymbol{p}_7, \boldsymbol{v}_2, \boldsymbol{p}_1, \boldsymbol{p}_8, \boldsymbol{p}_9, \boldsymbol{p}_3, \boldsymbol{p}_4, \boldsymbol{v}_9, \boldsymbol{p}_5)$

Post-process: $(\boldsymbol{p}_0, \boldsymbol{p}_6, \boldsymbol{p}_9, \boldsymbol{p}_3,)$ and $(\boldsymbol{p}_7, \boldsymbol{v}_2, \boldsymbol{p}_1, \boldsymbol{p}_8)$ and $(\boldsymbol{v}_4, \boldsymbol{v}_9, \boldsymbol{p}_5)$