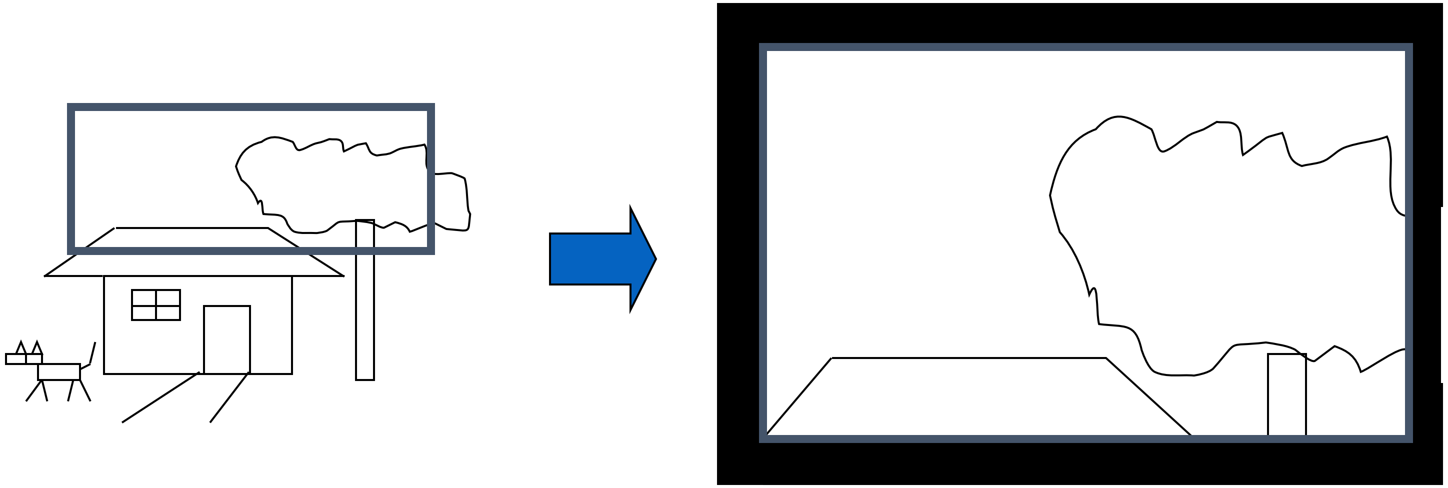
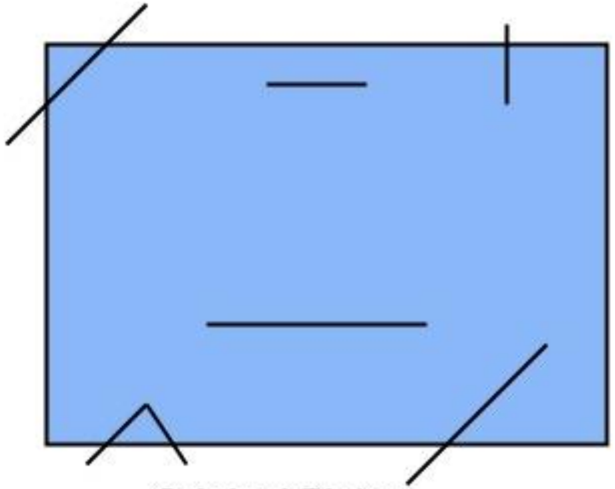
Clipping

Cohen-Sutherland Algo

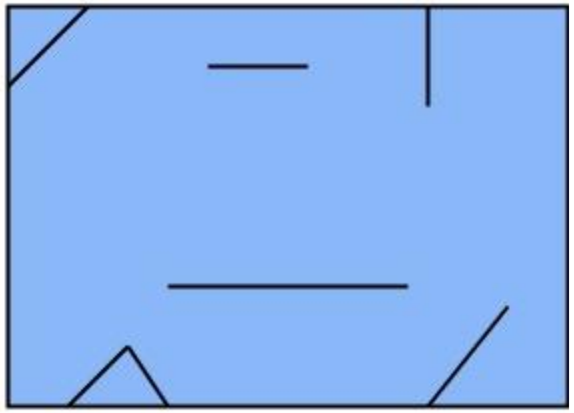
Clipped view in screen



Line Clipping



Original Picture
or
Before Clipping

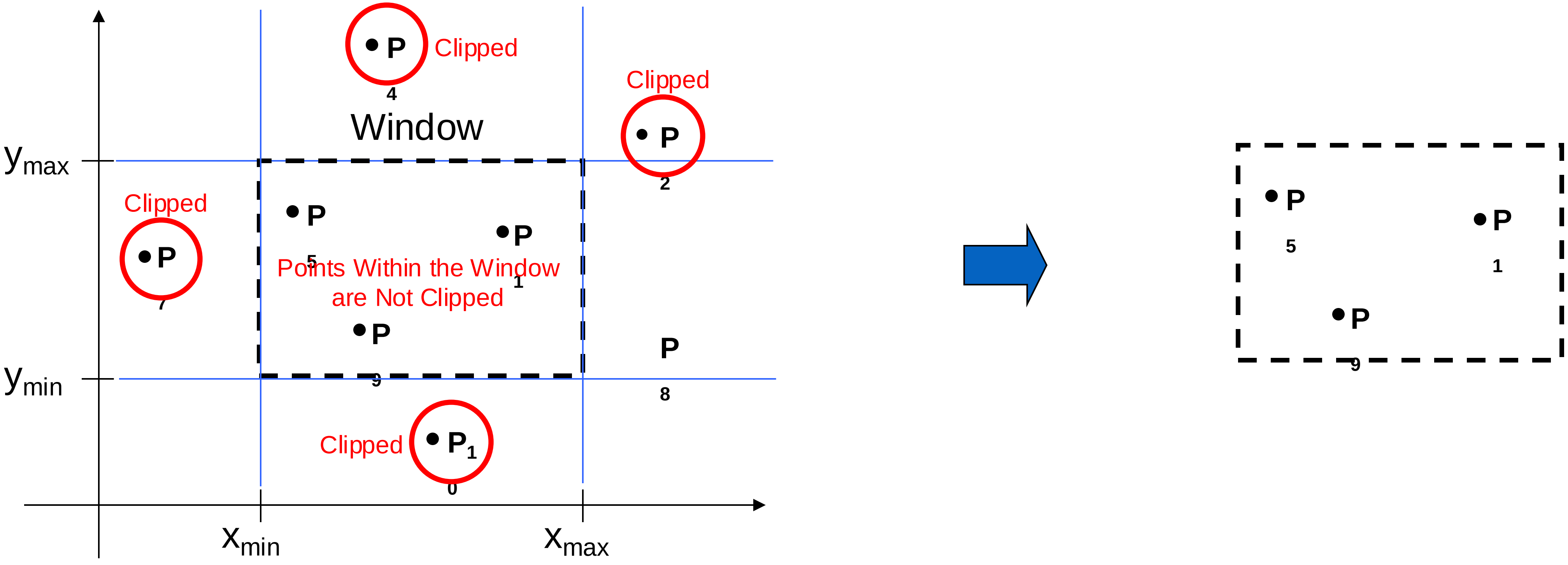


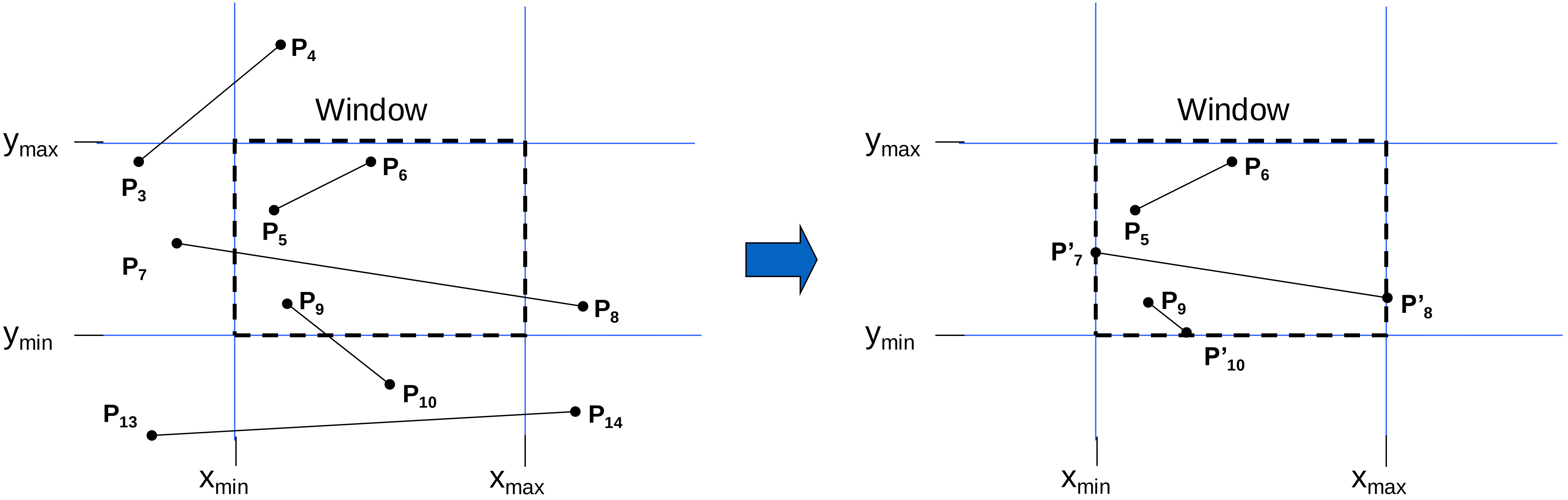
After Clipping

A point (x,y) is not clipped if:

$$x_{min} \le x \le x_{max} \text{ AND } y_{min} \le y \le y_{max}$$

otherwise it is clipped





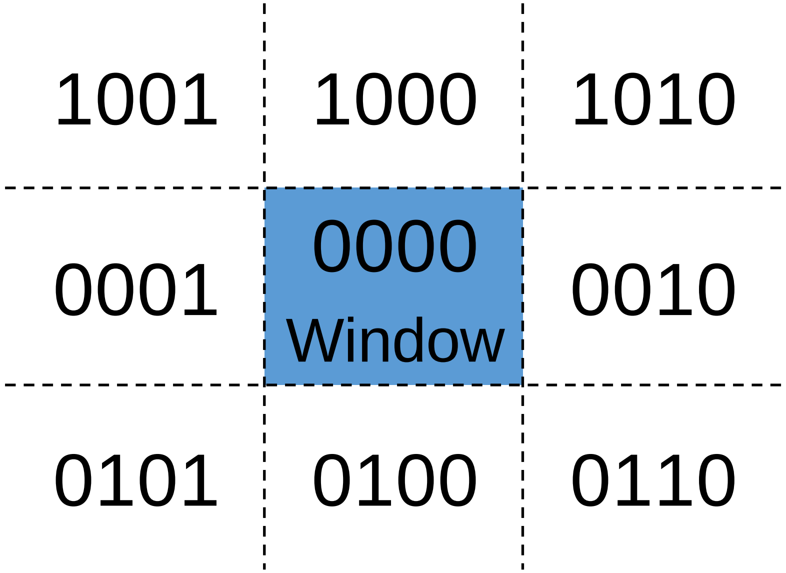
P4
Window
ymax
P3
P6
P5
P7
P9
P8
ymin
P10
P13
P14
xmin
xmax
Window
ymax
P6
P5
P'7
P9
P'8
ymin
P'10
xmin
xmax

World space is divided into regions based on the window boundaries

- Each region has a unique four bit outcode
- Outcodes indicate the position of the regions with respect to the window

3	2	1	0
above	below	right	left

Region Code



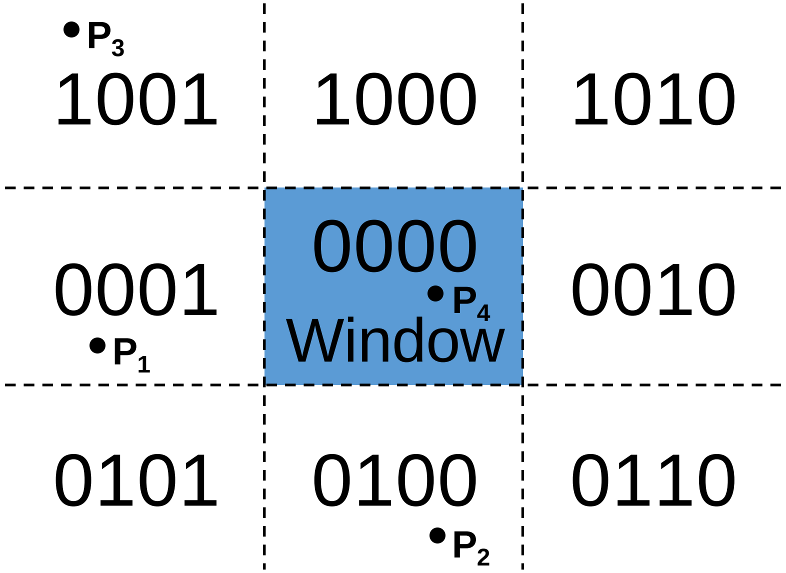
1001	1000	1010
0001	0000 Window	0010
0101	0100	0110

World space is divided into regions based on the window boundaries

- Each region has a unique four bit outcode
- Outcodes indicate the position of the regions with respect to the window

3	2	1	0
abov e	belo w	right	left

Region Code



1001	1000	1010
0001	0000 Window	0010
0101	0100	0110

```
Calculate_outcode(x,y){
    if ($x<x_{min}$) bit0 =1
    else bit0=0
    if ($x>x_{max}$) bit1=1
    else bit1=0
    if($y<y_{min}$) bit2=1
    else bit2=0
    if ($y>y_{max}$) bit3=1
    else bit3=0
}
```

3	2	1	0
above	below	right	left

Region Code

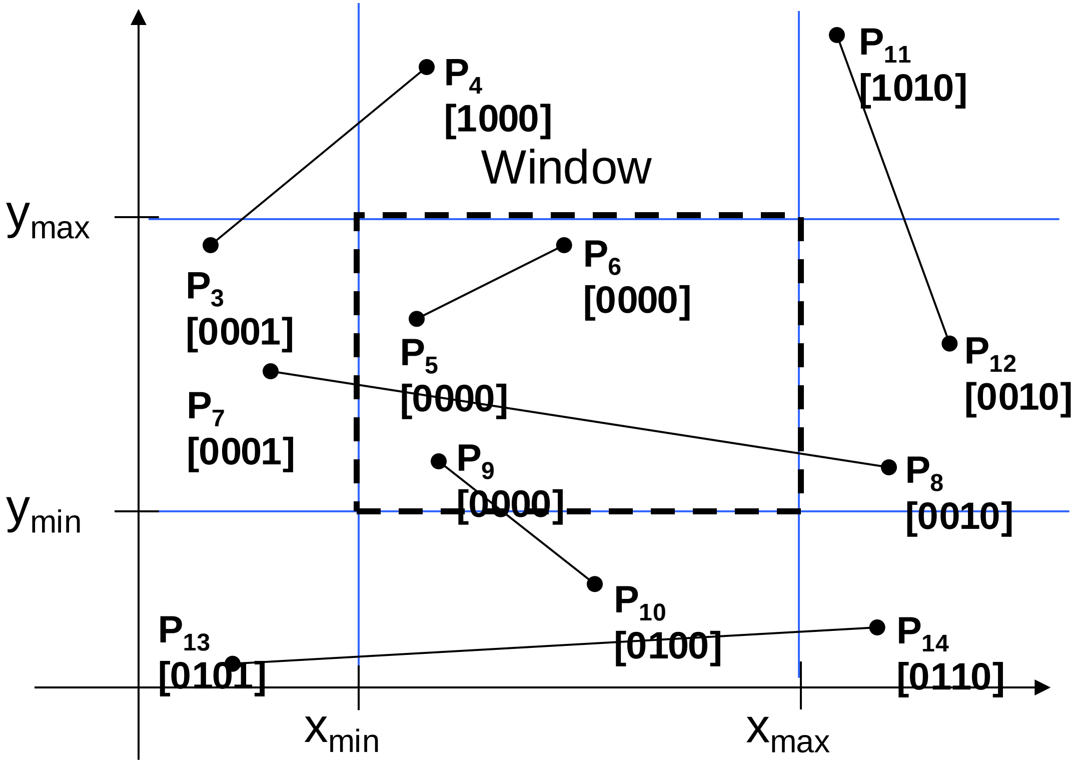
```
Calculate_Outcode_3D(x,y,z){
    if ($x<x_{min}$) bit0 =1
    else bit0=0
    if ($x>x_{max}$) bit1=1
    else bit1=0
    if($y<y_{min}$) bit2=1
    else bit2=0
    if ($y>y_{max}$) bit3=1
    else bit3=0
    if($z<z_{min}$) bit4=1
    else bit4=0
    if ($z>z_{max}$) bit5=1
    else bit5=0
}
```

5	**4**	**3**	**2**	**1**	**0**
Near	Far	Above	Below	Right	Left
Region Code					

```
Calculate_outcode(x,y){
    if ($x<x_{min}$) bit0 =1
    else bit0=0
    if ($x>x_{max}$) bit1=1
    else bit1=0
    if($y<y_{min}$) bit2=1
    else bit2=0
    if ($y>y_{max}$) bit3=1
    else bit3=0
}
```

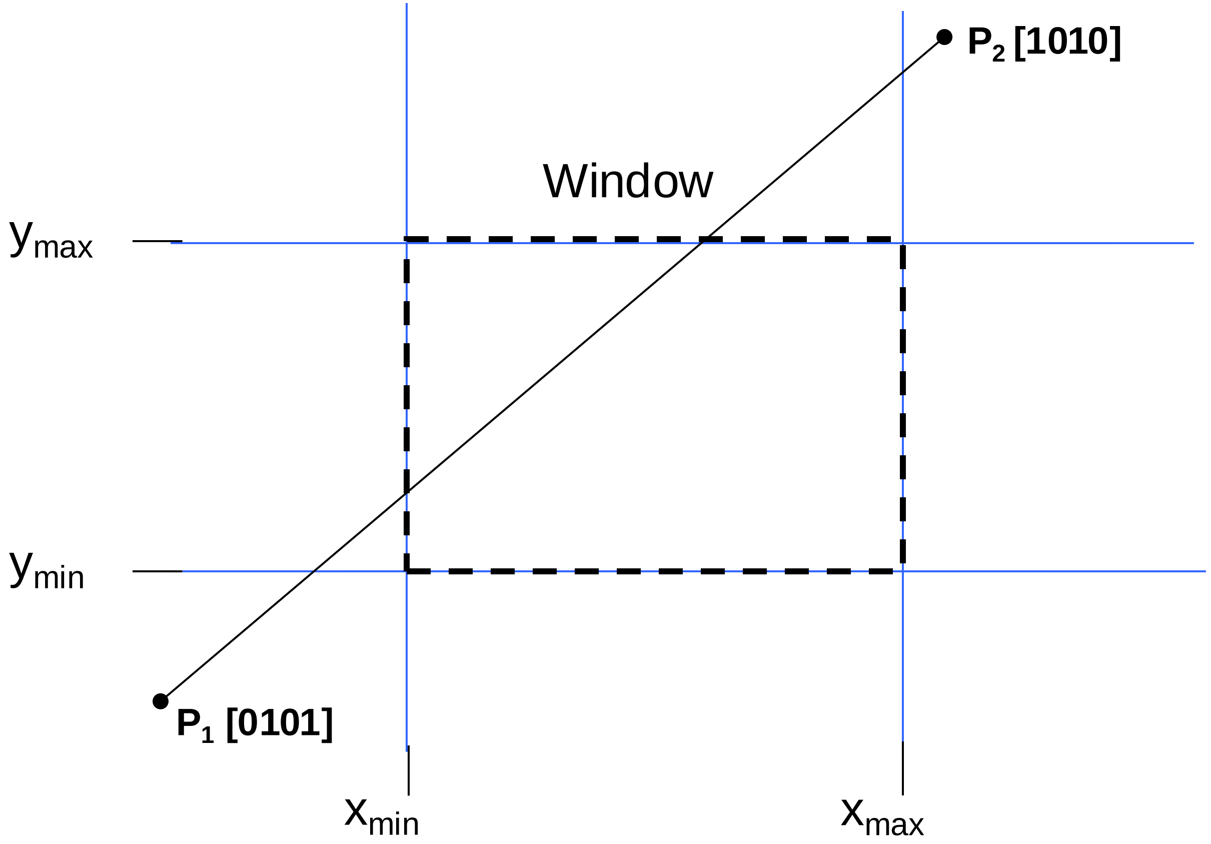
3	2	1	0
above	below	right	left

Region Code

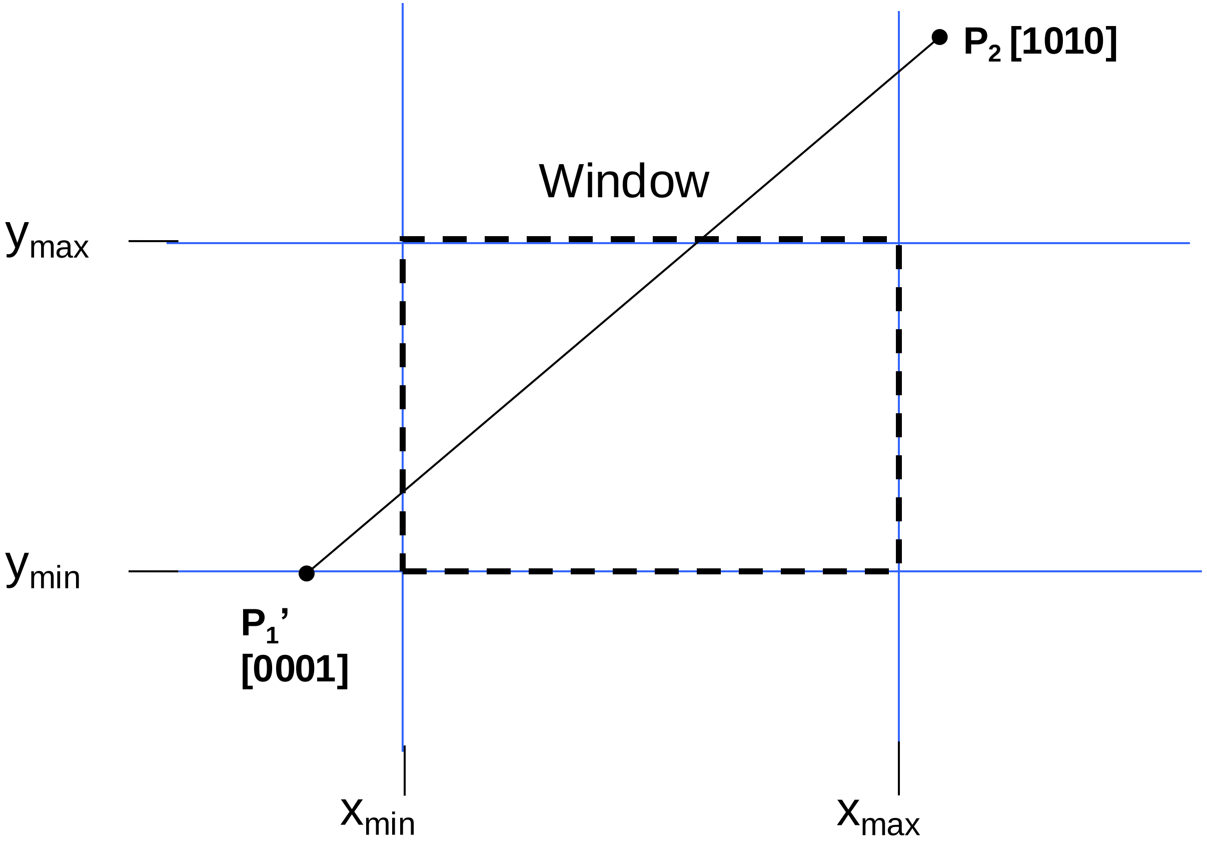


```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
    if(oc1 == oc2 == 0000) {
        //declare completely inside
        output (x1, y1), (x2, y2) as clipped line
        break
    }
    else if((oc1 AND oc2)!=0000){ // condition to check matching bit
        //declare completely outside and clip
        break
    }
    else{
        if(oc1 != 0000){
                (x1, y1) = find intersection point of line
                       and the boundary corresponding
                       to non-zero bit of oc1
                oc1 = calculate_outcode(x1, y1)
        }
        else{
                (x2, y2) = find intersection point of line
                       and the boundary corresponding
                       to non-zero bit of oc2
                oc2 = calculate_outcode(x2, y2)
        }
        continue
}
```

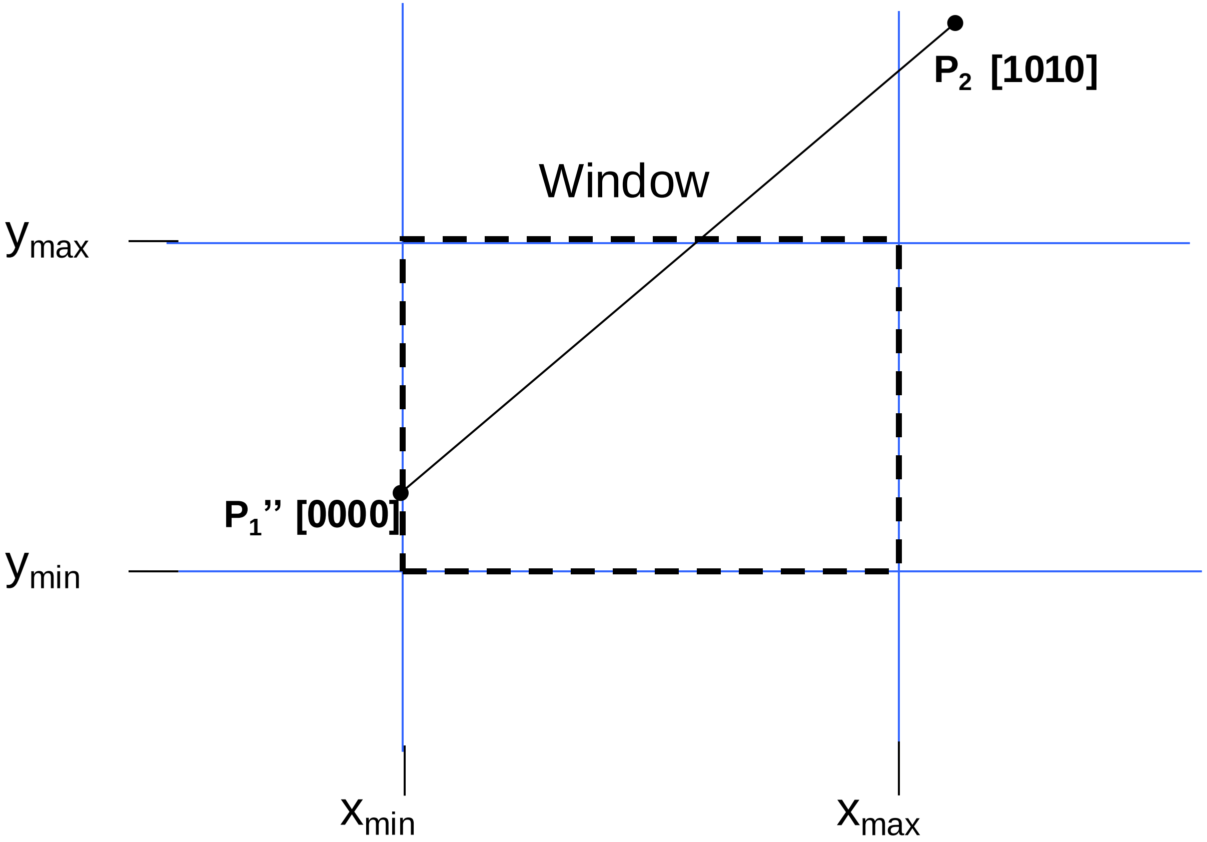
```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
        if (oc1 == oc2 == 0000) {
                //declare completely inside
                output (x1, y1), (x2, y2) as clipped line
                break
        }
        else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
                //declare completely outside and clip
                break
        }
        else{
                if(oc1 != 0000){
                                (x1, y1) = find intersection point of line
                                        and the boundary corresponding
                                        to non-zero bit of oc1
                                oc1 = calculate_outcode(x1, y1)
                }
                else{
                                (x2, y2) = find intersection point of line
                                           and the boundary corresponding
                                           to non-zero bit of oc2
                                oc2 = calculate_outcode(x2, y2)
                }
                continue
}
```



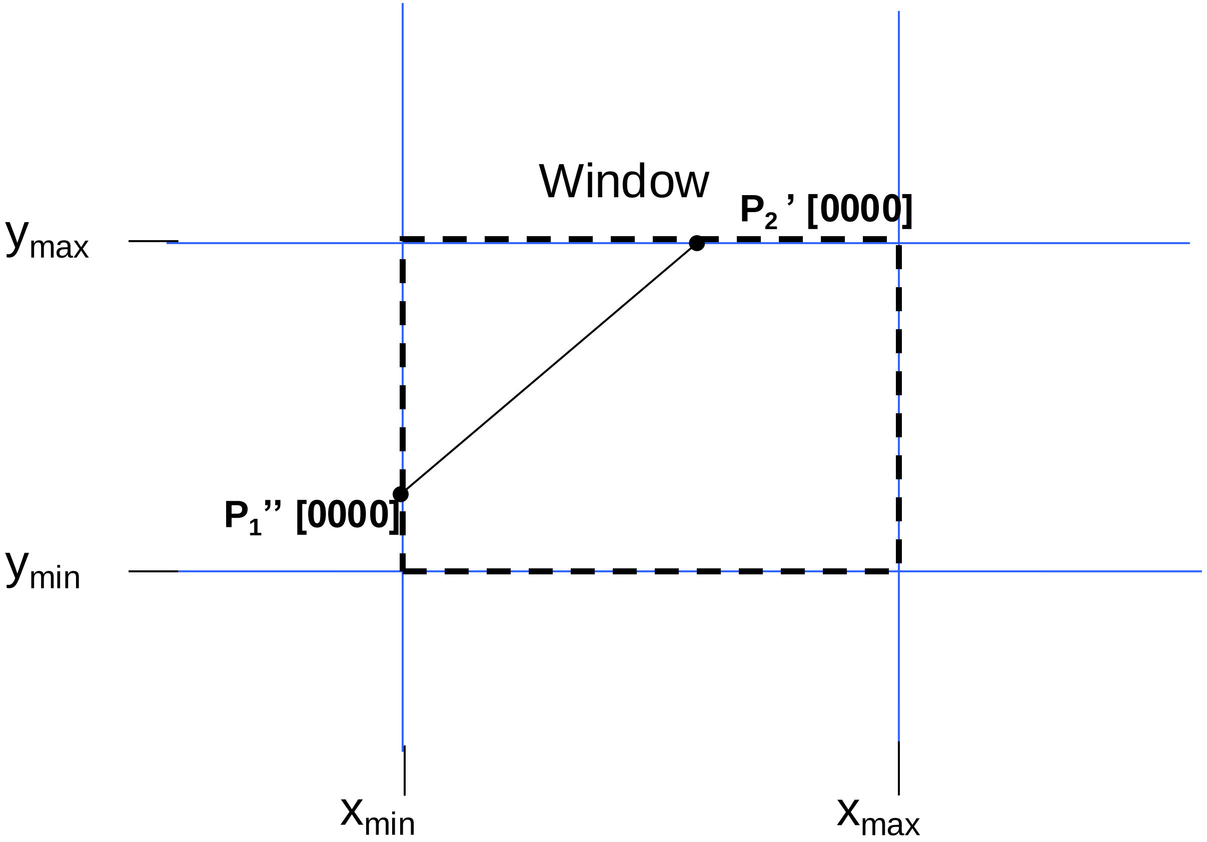
```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
    if (oc1 == oc2 == 0000) {
        //declare completely inside
        output (x1, y1), (x2, y2) as clipped line
        break
    }
    else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
        //declare completely outside and clip
        break
    }
    else{
        if(oc1 != 0000){
                (x1, y1) = find intersection point of line
                    and the boundary corresponding
                    to non-zero bit of oc1
                oc1 = calculate_outcode(x1, y1)
        }
        else{
                (x2, y2) = find intersection point of line
                      and the boundary corresponding
                      to non-zero bit of oc2
                oc2 = calculate_outcode(x2, y2)
        }
        continue
}
```



```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
    if (oc1 == oc2 == 0000) {
        //declare completely inside
        output (x1, y1), (x2, y2) as clipped line
        break
    }
    else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
        //declare completely outside and clip
        break
    }
    else{
        if(oc1 != 0000){
                (x1, y1) = find intersection point of line
                    and the boundary corresponding
                    to non-zero bit of oc1
                oc1 = calculate_outcode(x1, y1)
        }
        else{
                (x2, y2) = find intersection point of line
                      and the boundary corresponding
                      to non-zero bit of oc2
                oc2 = calculate_outcode(x2, y2)
        }
        continue
}
```



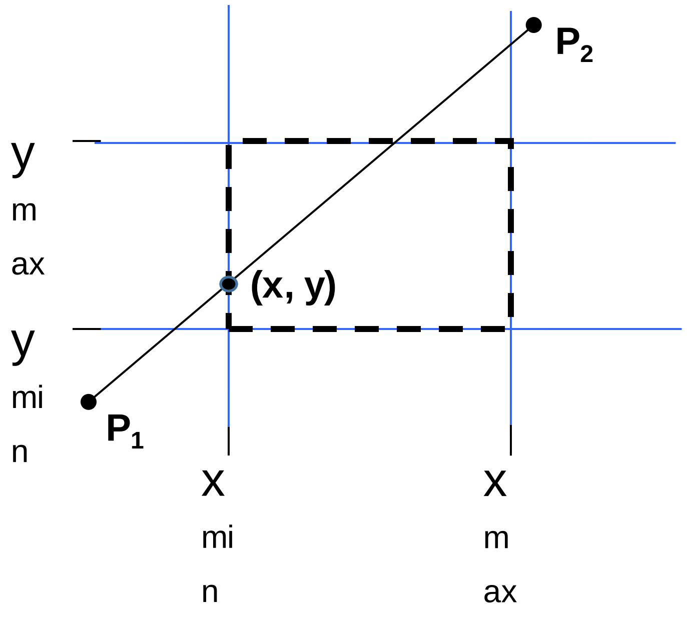
```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
    if (oc1 == oc2 == 0000) {
        //declare completely inside
        output (x1, y1), (x2, y2) as clipped line
        break
    }
    else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
        //declare completely outside and clip
        break
    }
    else{
        if(oc1 != 0000){
                (x1, y1) = find intersection point of line
                     and the boundary corresponding
                      to non-zero bit of oc1
                oc1 = calculate_outcode(x1, y1)
        }
        else{
                (x2, y2) = find intersection point of line
                        and the boundary corresponding
                        to non-zero bit of oc2
                oc2 = calculate_outcode(x2, y2)
        }
        continue
}
```



LEFT boundary intersection:

$x = x_{min}$

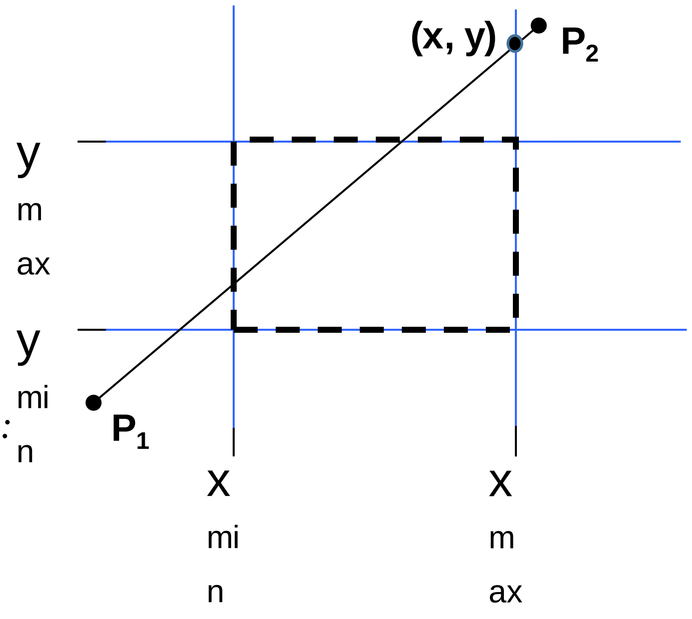
$y = y_1 + m\,(x_{min} - x_1)$



RIGHT boundary intersection:

$x = x_{max}$

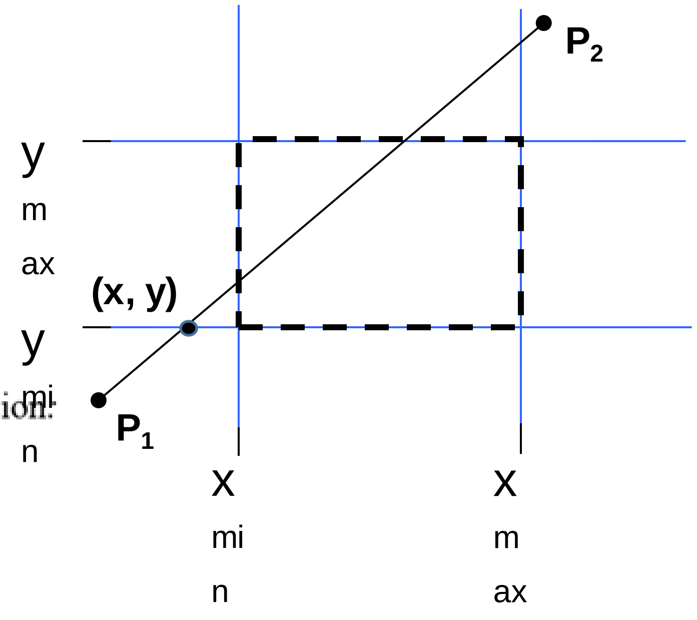
$y = y_1 + m\,(x_{max} - x_1)$



BOTTOM boundary intersection:

$y = y_{min}$

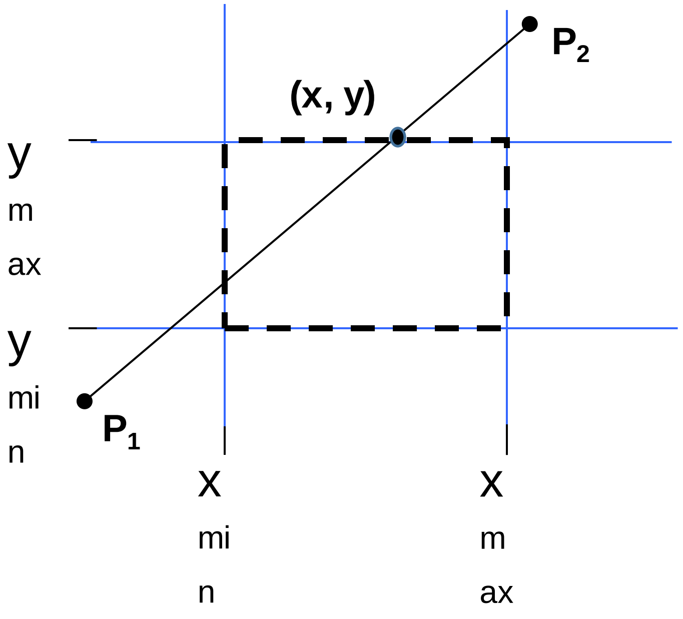
$x = x_1 + \frac{1}{m}.(y_{min} - y_1)$



TOP boundary intersection:

$y = y_{max}$

$x = x_1 + \frac{1}{m}.(y_{max} - y_1)$



Determine whether the following line are accepted/rejected/partial using Cohen Sutherland line clipping algorithm.

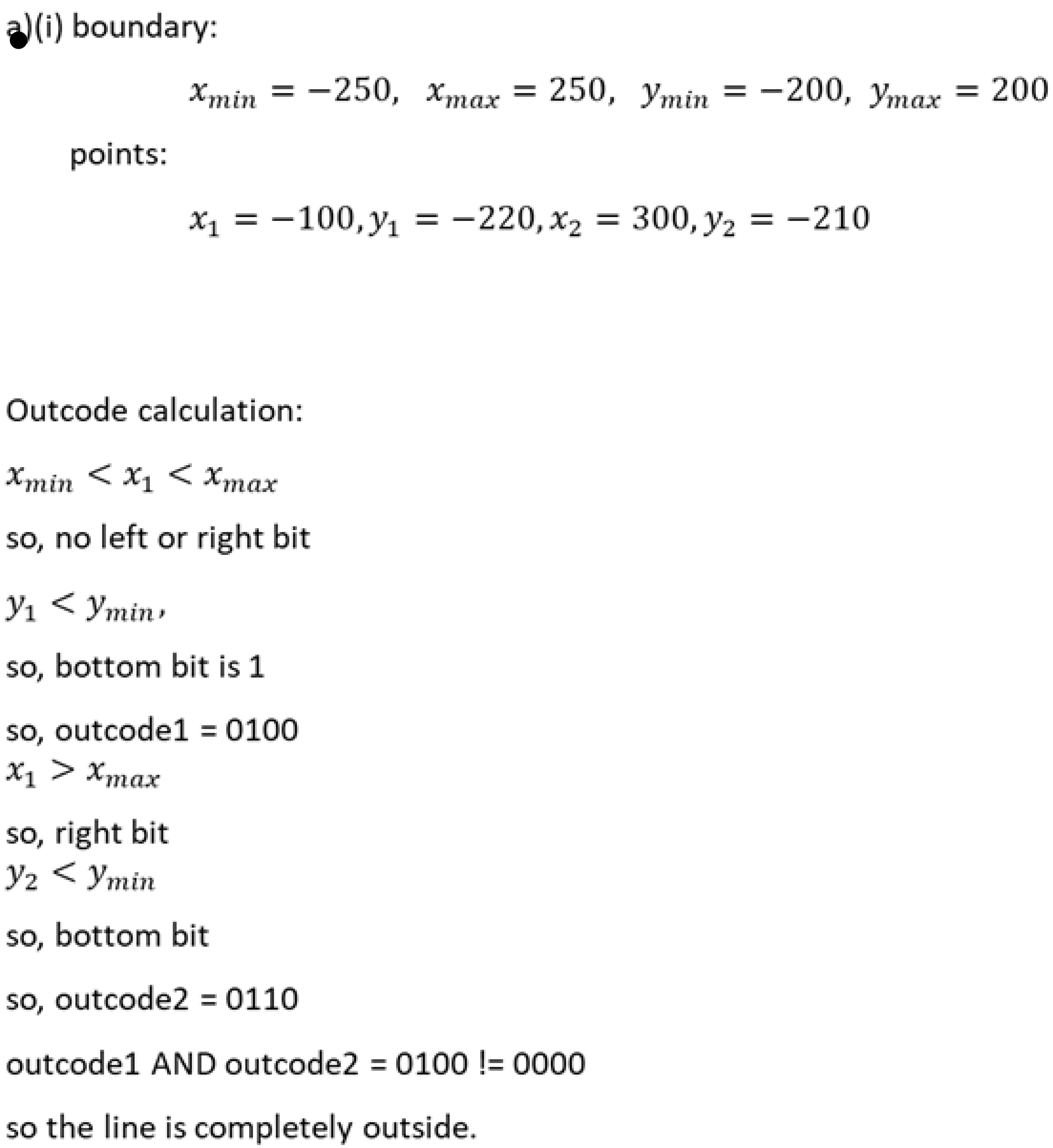
a) Given (-250,-200) to (250,200) be the clip region.

- (i) (-100, -220) to (300, -210).
- (ii) (-250, 200) to (250, -200).

b) Given (0,0) to (300,200) be the clip region.

- (i) (50, -125) to (-100, 225).
- (ii) (-250, 200) to (250, -200).

If they are partially accepted/rejected find the line segment within the clipping window.



a)(i) boundary:

$$x_{min} = -250, \ \ x_{max} = 250, \ \ y_{min} = -200, \ \ y_{max} = 200$$

points:

$$x_1 = -100, y_1 = -220, x_2 = 300, y_2 = -210$$

Outcode calculation:

$x_{min} < x_1 < x_{max}$

so, no left or right bit

$y_1 < y_{min}$,

so, bottom bit is 1

so, outcode1 = 0100

$x_1 > x_{max}$

so, right bit

$y_2 < y_{min}$

so, bottom bit

so, outcode2 = 0110

outcode1 AND outcode2 = 0100 != 0000

so the line is completely outside.

```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
        if (oc1 == oc2 == 0000) {
                //declare completely inside
                output (x1, y1), (x2, y2) as clipped line
                break
        }
        else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
                //declare completely outside and clip
                break
        }
        else{
                if(oc1 != 0000){
                                (x1, y1) = find intersection point of line
                                        and the boundary corresponding
                                        to non-zero bit of oc1
                                oc1 = calculate_outcode(x1, y1)
                }
                else{
                                (x2, y2) = find intersection point of line
                                           and the boundary corresponding
                                           to non-zero bit of oc2
                                oc2 = calculate_outcode(x2, y2)
                }
                continue
}
```

● a)(ii) boundary:

$$x_{min} = -250, \;\; x_{max} = 250, \;\; y_{min} = -200, \;\; y_{max} = 200$$

points:

$$x_1 = -250, y_1 = 200, x_2 = 250, y_2 = -200$$

Outcode calculation:
$x_{min} \leq x_1 < x_{max}$

so, no left or right bit

$y_{min} < y_1 \leq y_{max}$,

so, no top or bottom bit

so, outcode1 = 0000

similarly, outcode2 = 0000

since outcode1 & outcode 2 both are 0000

so the line is completely inside.

```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
    if (oc1 == oc2 == 0000) {
        //declare completely inside
        output (x1, y1), (x2, y2) as clipped line
        break
    }
    else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
        //declare completely outside and clip
        break
    }
    else{
        if(oc1 != 0000){
                (x1, y1) = find intersection point of line
                    and the boundary corresponding
                    to non-zero bit of oc1
                oc1 = calculate_outcode(x1, y1)
        }
        else{
                (x2, y2) = find intersection point of line
                        and the boundary corresponding
                        to non-zero bit of oc2
                oc2 = calculate_outcode(x2, y2)
        }
        continue

}
```

b)(i) boundary:

$$x_{min} = 0, \quad x_{max} = 300, \quad y_{min} = 0, \quad y_{max} = 200$$

points:

$$x_1 = 50, y_1 = -125, x_2 = -100, y_2 = 225$$

Outcode calculation:

outcode1 = 0100

outcode2 = 1001

outcode1 AND outcode2 = 0000

so partially inside

outcode1 != 0000

outcode1 has bottom bit

Applying bottom intersection:

$y_1 = y_{min} = 0$

$x_1 = x_1 + \frac{1}{m}.(y_{min} - y_1) = 50 + \frac{-150}{350}.(0 + 125) =$ -3.57

outcode1 = 0001 [recalculated]

outcode2 = 1001

outcode1 AND outcode2 = 0001

so completely outside

```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
        if (oc1 == oc2 == 0000) {
                //declare completely inside
                output (x1, y1), (x2, y2) as clipped line
                break
        }
        else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
                //declare completely outside and clip
                break
        }
        else{
                if(oc1 != 0000){
                                (x1, y1) = find intersection point of line
                                        and the boundary corresponding
                                        to non-zero bit of oc1
                                oc1 = calculate_outcode(x1, y1)
                }
                else{
                                (x2, y2) = find intersection point of line
                                           and the boundary corresponding
                                           to non-zero bit of oc2
                                oc2 = calculate_outcode(x2, y2)
                }
                continue

}
```

b)(ii) boundary:

$$x_{min} = 0, \;\; x_{max} = 300, \;\; y_{min} = 0, \;\; y_{max} = 200$$

points:

$$x_1 = -250, y_1 = 200, x_2 = 250, y_2 = -200$$

Outcode calculation:

outcode1 = 0001

outcode2 = 0100

outcode1 AND outcode2 = 0000

so partially inside

outcode1 != 0000

outcode1 has left bit

applying left intersection:
$x_1 = x_{min} = 0$

$y_1 = y_1 + m.(x_{min} - x_1) = 200 + \frac{-400}{500}.(0 + 250) = 0$

outcode1 = 0000 [recalculated]

so (x1, y1) has been clipped to (0, 0)

outcode2 = 0100

so partially inside

```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
        if (oc1 == oc2 == 0000) {
                //declare completely inside
                output (x1, y1), (x2, y2) as clipped line
                break
        }
        else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
                //declare completely outside and clip
                break
        }
        else{
                if(oc1 != 0000){
                                (x1, y1) = find intersection point of line
                                       and the boundary corresponding
                                        to non-zero bit of oc1
                                oc1 = calculate_outcode(x1, y1)
                }
                else{
                                (x2, y2) = find intersection point of line
                                          and the boundary corresponding
                                          to non-zero bit of oc2
                                oc2 = calculate_outcode(x2, y2)
                }
                continue
}
```

● b)(ii) boundary:

$$x_{min} = 0,\ \ x_{max} = 300,\ \ y_{min} = 0,\ y_{max} = 200$$

points:

$$x_1 = 0, y_1 = 0, x_2 = 250, y_2 = -200$$

(continued)

outcode1 = 0000 [recalculated]

outcode2 = 0100

so partially inside

outcode1 = 0000

so going into else codeblock,

outcode 2 has bottom bit

applying bottom intersection:
$y_2 = y_{min} = 0$

$$x_2 = x_2 + \frac{1}{m}.(y_{min} - y_2) = 250 + \frac{250}{-200}.(0 + 200) = 0$$

[Note: m has been recalculated, you can skip recalculation too]

outcode2 = 0000

outcode1=0000

so completely inside

The clipped segment is between (0, 0) to (0, 0) which is just a single point.

```
cohen-Sutherland(x1, y1, x2, y2):
oc1 = calculate_outcode(x1, y1),
oc2 = calculate_outcode(x2, y2);
while(true) {
        if (oc1 == oc2 == 0000) {
                //declare completely inside
                output (x1, y1), (x2, y2) as clipped line
                break
        }
        else if ((oc1 AND oc2)!=0000) { // condition to check matching bit
                //declare completely outside and clip
                break
        }
        else{
                if(oc1 != 0000){
                        (x1, y1) = find intersection point of line
                                and the boundary corresponding
                                to non-zero bit of oc1
                        oc1 = calculate_outcode(x1, y1)
                }
                else{
                        (x2, y2) = find intersection point of line
                                and the boundary corresponding
                                to non-zero bit of oc2
                        oc2 = calculate_outcode(x2, y2)
                }
                continue
}
```