Clipping

- When we have to display a large portion of the picture, then not only scaling and translation is necessary, the visible part of picture is also identified
- The window against which object is clipped called a CLIP WINDOW. It can be CURVED or RECTANGULAR in shape.
- For deciding the visible and invisible portion, a particular process called clipping is used. Clipping determines each element into the visible and invisible portion. Visible portion is selected. An invisible portion is discarded.
- Clipping can be applied to world co-ordinates. The contents inside the window will be mapped to device co-ordinates. Another alternative is a complete world co-ordinates picture is assigned to device co-ordinates, and then clipping of viewport boundaries is done.
- Clipping algorithms can be 2D or 3D
- For a 2d transformation viewing, a clipping algorithm can be applied in 2 ways:

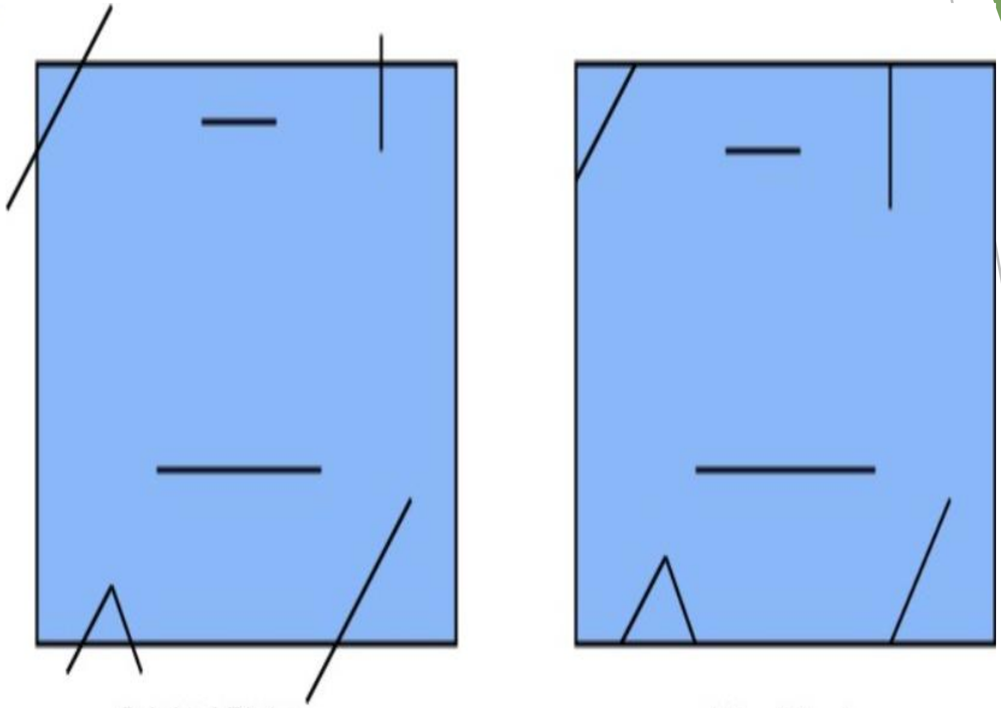
  1) world co-ordinate clipping. 2) viewport clipping.

Clipping….

- **Clipping** is a procedure that identifies those portions of a picture that are either inside or outside of a specified region of space.

- **Clip Window** is the region against which an object is to be clipped.
- It may be a polygon or curved boundaries. But **rectangular clip regions** are preferred.

- For viewing transformations, picture parts within window area are displayed. Everything outside window area is discarded.



Clipping….

Applications of clipping

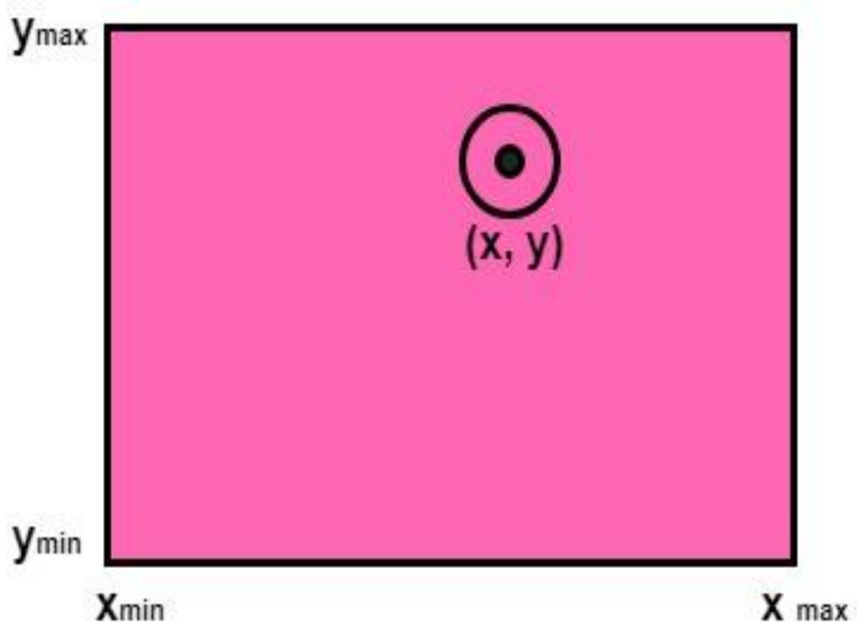
- It will extract part we desire.
- For identifying the visible and invisible area in the 3D object.
- For creating objects using solid modeling.
- For drawing operations.
- Operations related to the pointing of an object.
- For deleting, copying, moving part of an object
- Displaying multi-window environment

Types of Clipping:

- Point Clipping
- Line Clipping
- Area Clipping (Polygon)
- Curve Clipping (Arc Clipping)
- Text Clipping

Point Clipping

- Point Clipping is used to determining, whether the point is inside the window or not. For the point to be inside following two condition should hold
- xmin ≤ x ≤ xmax
- ymin ≤ x ≤ ymax
- The (x, y) is coordinate of the point. If anyone from the above inequalities is false, then the point will fall outside the window and will not be considered to be visible.(the point is clipped i.e not saved for display)



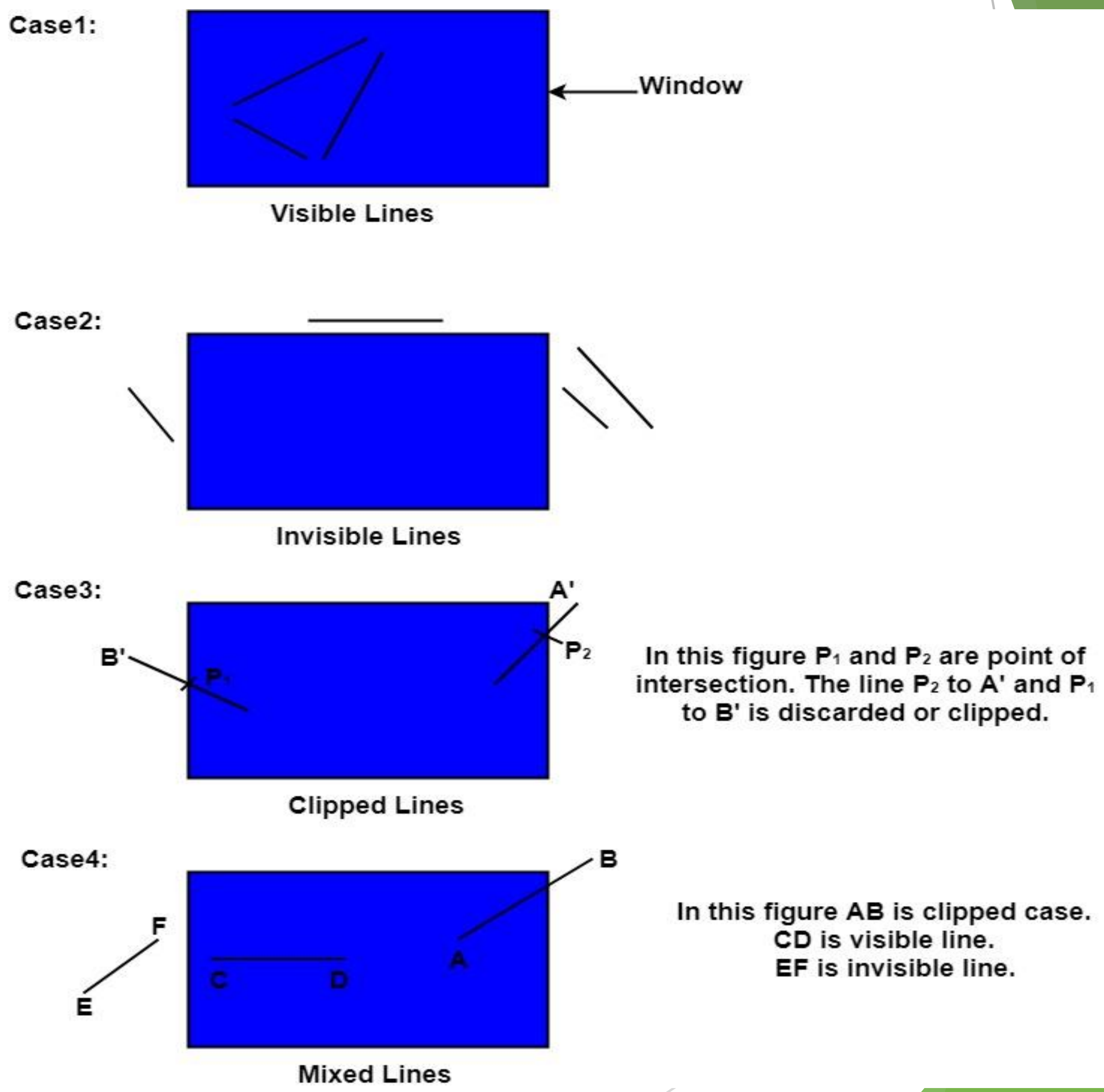
Line Clipping

It is performed by using the line clipping algorithm. The line clipping algorithms are:

- Cohen Sutherland Line Clipping Algorithm
- Midpoint Subdivision Line Clipping Algorithm
- Liang-Barsky Line Clipping Algorithm

All lines come under any one of the following categories:

- Visible
- Not Visible
- Clipping Case



In this figure P_1 and P_2 are point of intersection. The line P_2 to A' and P_1 to B' is discarded or clipped.

In this figure AB is clipped case.
CD is visible line.
EF is invisible line.

Line Clipping…..

- **Case1 : Visible lines:** If a line lies within the window, i.e., both endpoints of the line lies within the window.it will be accepted and displayed.
- **Case 2: Not Visible or invisible lines**:i f a line lies outside the window it will be invisible and rejected. Such lines will not display. If any one of the following inequalities is satisfied, then the line is considered invisible. Let A (x1,y2) and B (x2,y2) are endpoints of line.

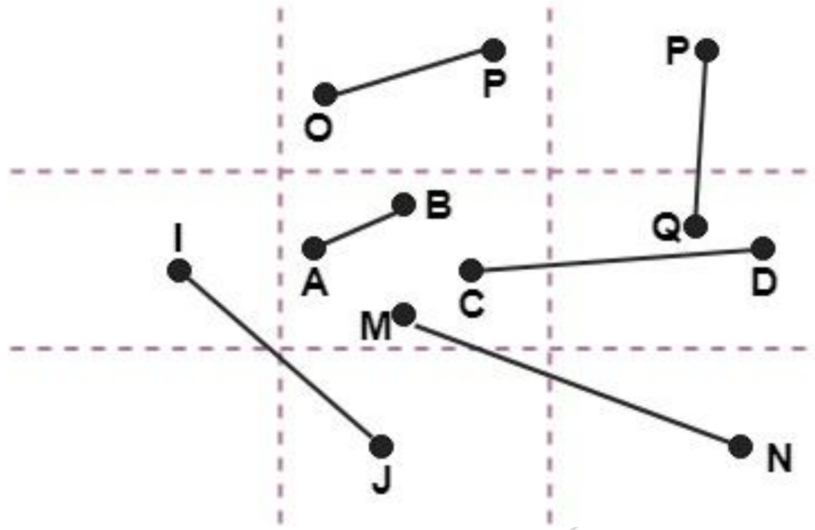
xmin,xmax are coordinates of the window.

ymin,ymax are also coordinates of the window.

x1>xmax , x2>xmax , y1>ymax , y2>ymax , x1<xmin , x2<xmin , y1<ymin , y2<ymin

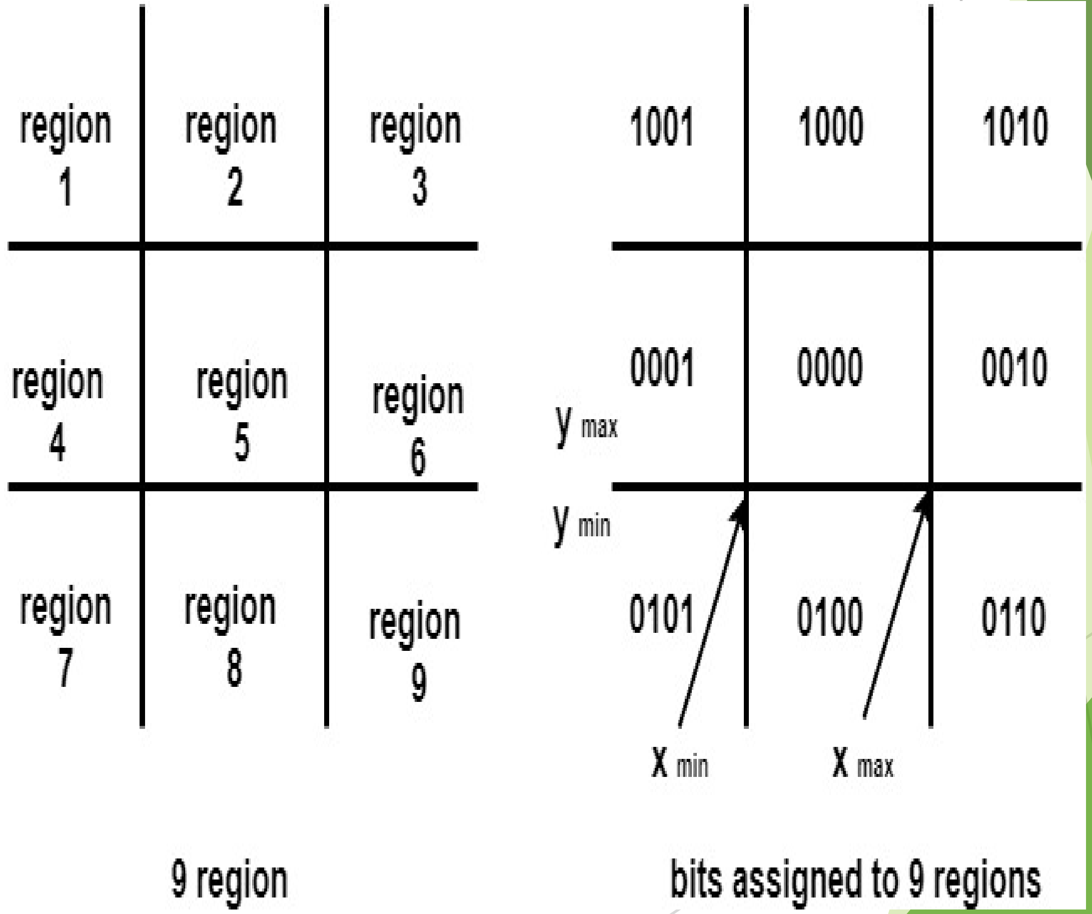
- **Case 3 :Clipping Case** : If the line is neither visible case nor invisible case. It is considered to be clipped case.

- Line AB is the visible case (accepted line)
- Line OP is an invisible case(rejected line)
- Line PQ is an invisible line (rejected line)
- Line IJ are clipping candidates
- Line MN are clipping candidate
- Line CD are clipping candidate



Cohen Sutherland Line Clipping Algorithm

- We will use 4-bits code(also known as region code) to divide the entire region .
- Clipping window has code 0000
- Bit1 = 1 if $y > ymax$ i.e. 1000 (above clipping window)
- Bit2 = 1 if $y < ymin$ i.e. 0100 (below clipping window)
- Bit3 = 1 if $x > xmax$ i.e. 0010 (right to clipping window)
- Bit4 = 1 if $x < xmin$i.e. 0001 (left to clipping window)



Algorithm of Cohen Sutherland Line Clipping

- Step 1 – Assign a region code for each endpoints.
- Step 2 – If both endpoints have a region code 0000 then accept this line.
- Step 3 – Else, perform the logical AND operation for both region codes.
- Step 3.1 – If the result is not 0000, then reject the line.
- Step 3.2 – Else you need clipping.
- Step 3.2.1 – Choose an endpoint of the line that is outside the window.
- Step 3.2.2 – Find the intersection point at the window boundary base on region code.
- Step 3.2.3 – Replace endpoint with the intersection point and update the region code.
- Step 3.2.4 – Repeat step 2 until we find a clipped line either trivially accepted or trivially rejected.
- Step 4 – Repeat step 1 for other lines

Cohen Sutherland algo….

- <u>Logical AND operation</u>

A AND B is true if both A and B are true

Example

1010 AND 0010 = 1010

1010 AND 0000= 0000

- <u>Advantages of Cohen Sutherland algo</u>

It calculates end-points very quickly and rejects and accepts lines quickly.

It can clip pictures much large than screen size.

- <u>Calculating intersection points with clipping window</u>

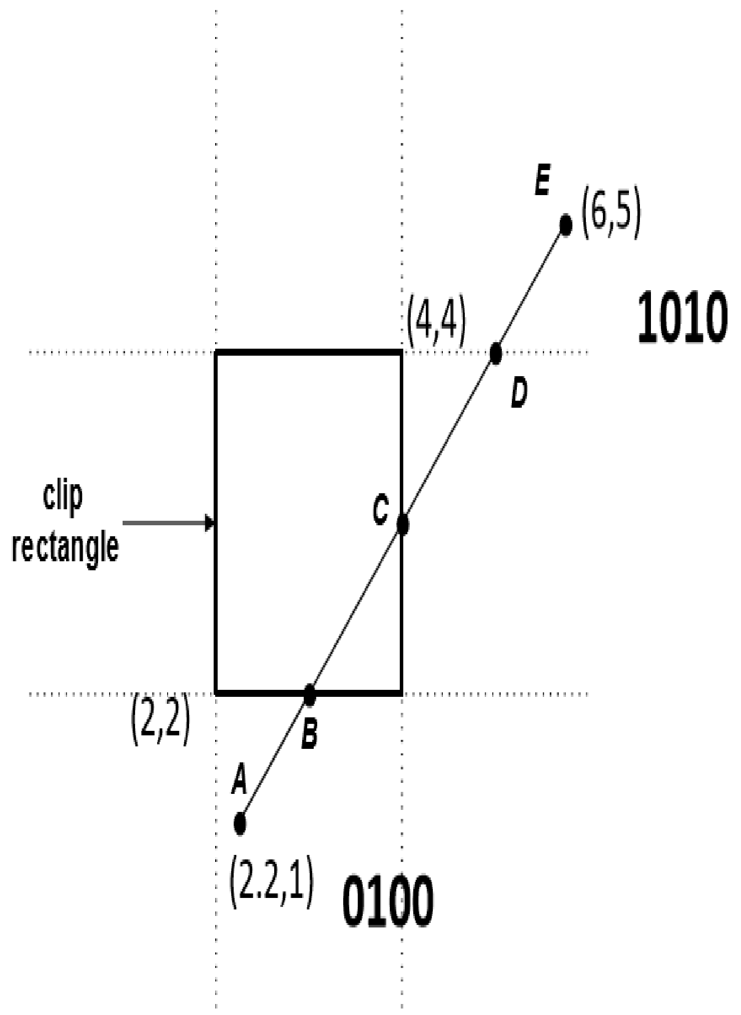
$$\left.\begin{aligned} x_i &= x_{\min} \text{ or } x_{\max} \\ y_i &= y_1 + m(x_i - x_1) \end{aligned}\right\} \text{If edge line is vertical}$$

OR

$$\left.\begin{aligned} x_i &= x_1 + (y_i - y_1)/m \\ y_i &= y_{\min} \text{ or } y_{\max} \end{aligned}\right\} \text{If edge line is horizontal}$$

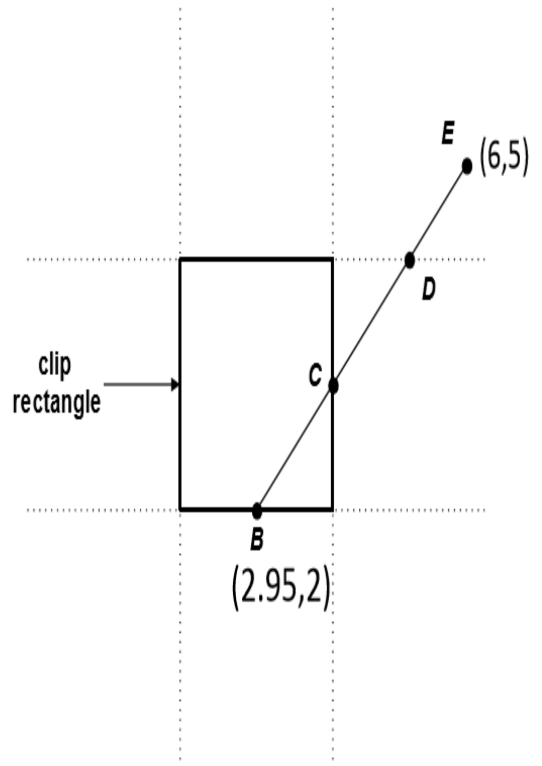
$$\textit{where}, \quad m = (y_2 - y_1)/(x_2 - x_2)$$

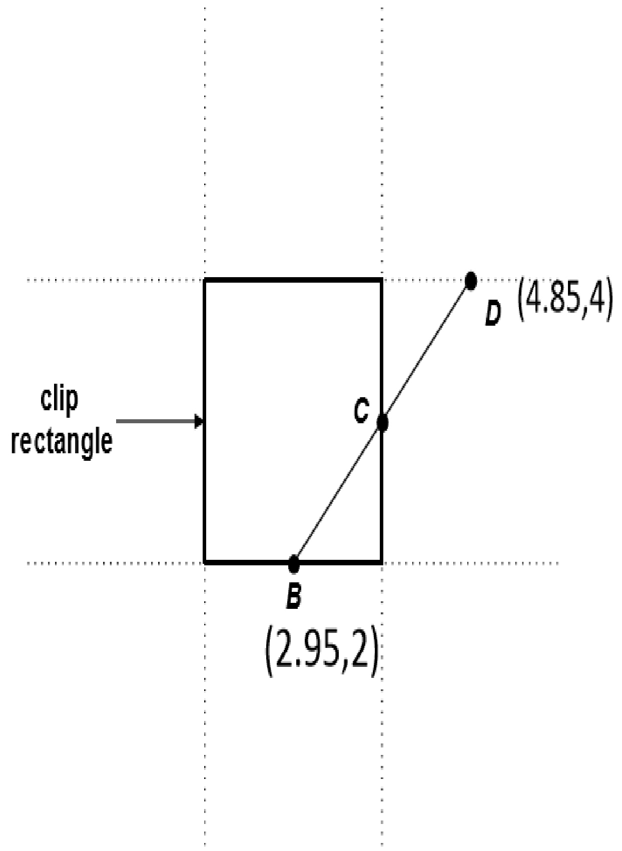
Cohen Sutherland Example 1



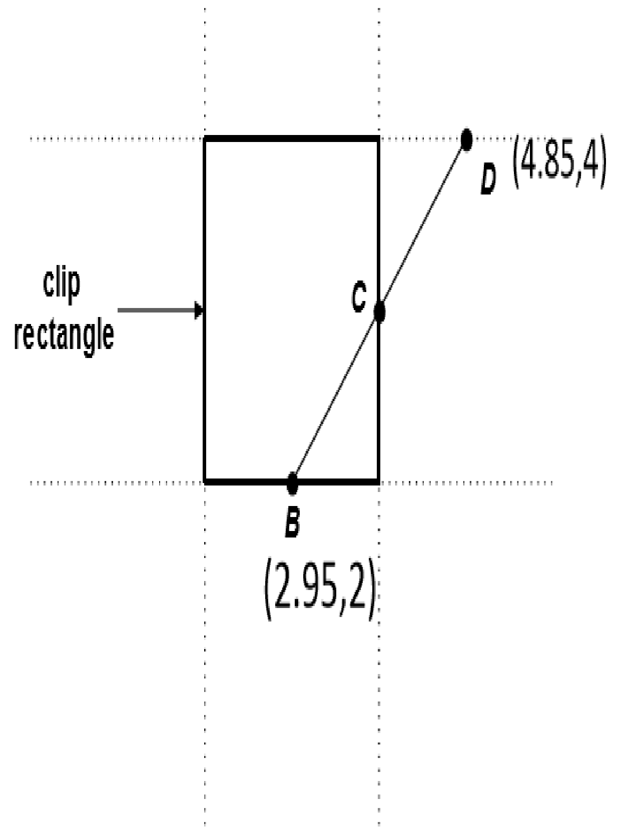
$$m=\frac{y_2-y_1}{x_2-x_1}=\frac{5-1}{6-2.2}=\frac{4}{3.8}=1.05$$

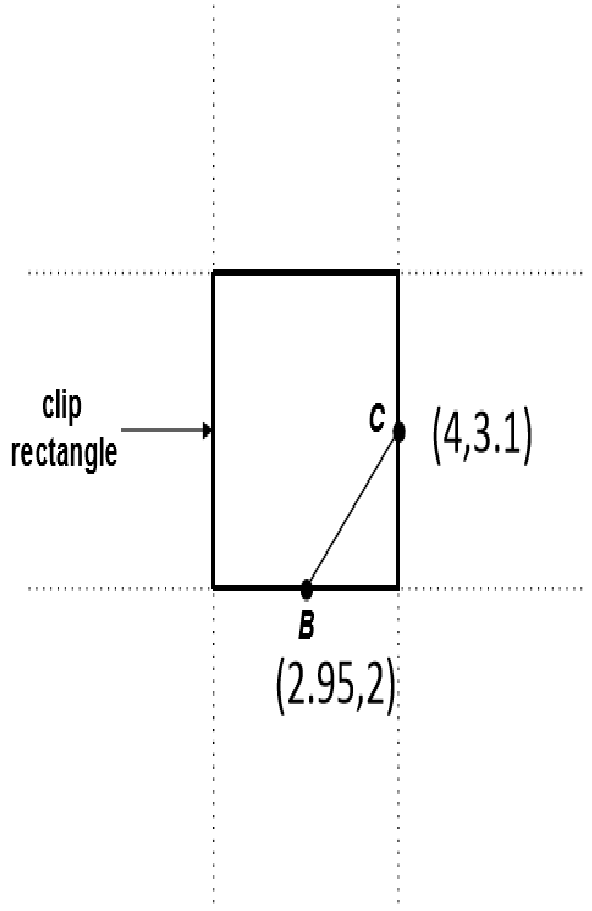
Example 1 cont…



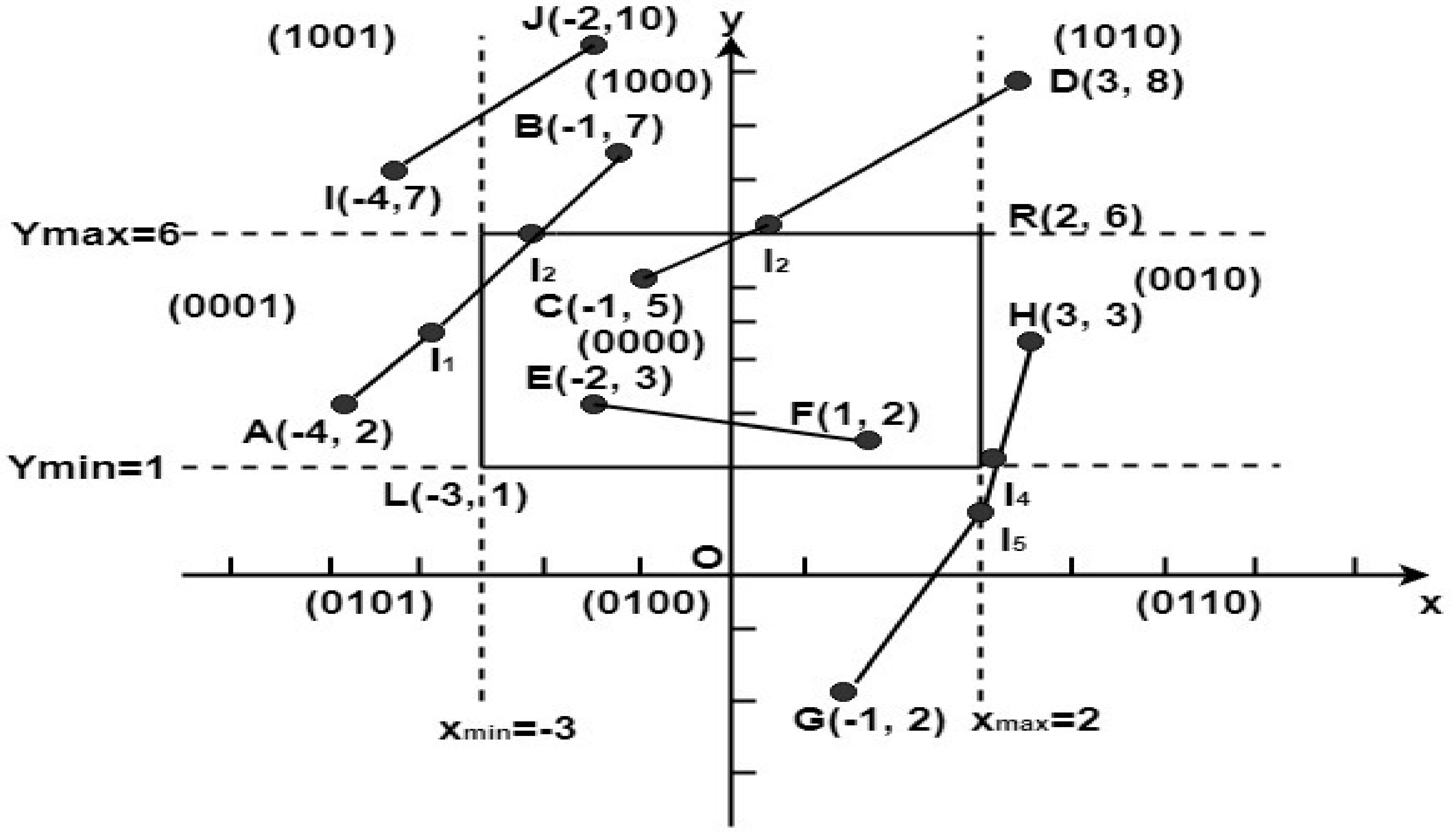


Example 1 cont..





Example 2: Let R be the rectangular window whose lower left-hand corner is at L (-3, 1) and upper right-hand corner is at R (2, 6). Find the region codes for the endpoints in fig:



Example 2(Detailed Explanation)

- Assign region codes to all end points using scheme of assigning codes to region

A (-4, 2)→ 0001	F (1, 2)→ 0000
B (-1, 7) → 1000	G (1, -2) →0100
C (-1, 5)→ 0000	H (3, 3) → 0100
D (3, 8) → 1010	I (-4, 7) → 1001
E (-2, 3) → 0000	J (-2, 10) → 1000

- Category1 (visible): EF since the region code for both endpoints is 0000.
- Category2 (not visible): IJ since (1001) AND (1000) =1000 (which is not 0000).
- Category 3 (candidate for clipping): AB since (0001) AND (1000) = 0000, CD since (0000) AND (1010) =0000, and GH. since (0100) AND (0010) =0000.
- The candidates for clipping are AB, CD, and GH.

Example 2(Detailed Explanation) (LINE : AB)

- In clipping AB, the code for A is 0001. To push the 1 to 0, we clip against the boundary line xmin=-3. The resulting intersection point I1(3,11/3) .
- We clip (do not display) AI1 and I1 B. The code for is 1001. The clipping category for I1B is 3 since (0000) AND (1000) is (0000). Now B is outside the window (i.e., its code is 1000), so we push the 1 to a 0 by clipping against the line ymax=6. he resulting intersection is l2 (-9/5,6). Thus I2 B is clipped. The code for I2 is 0000. The remaining segment I1 I2 is displayed since both endpoints lie in the window (i.e., their codes are 0000).

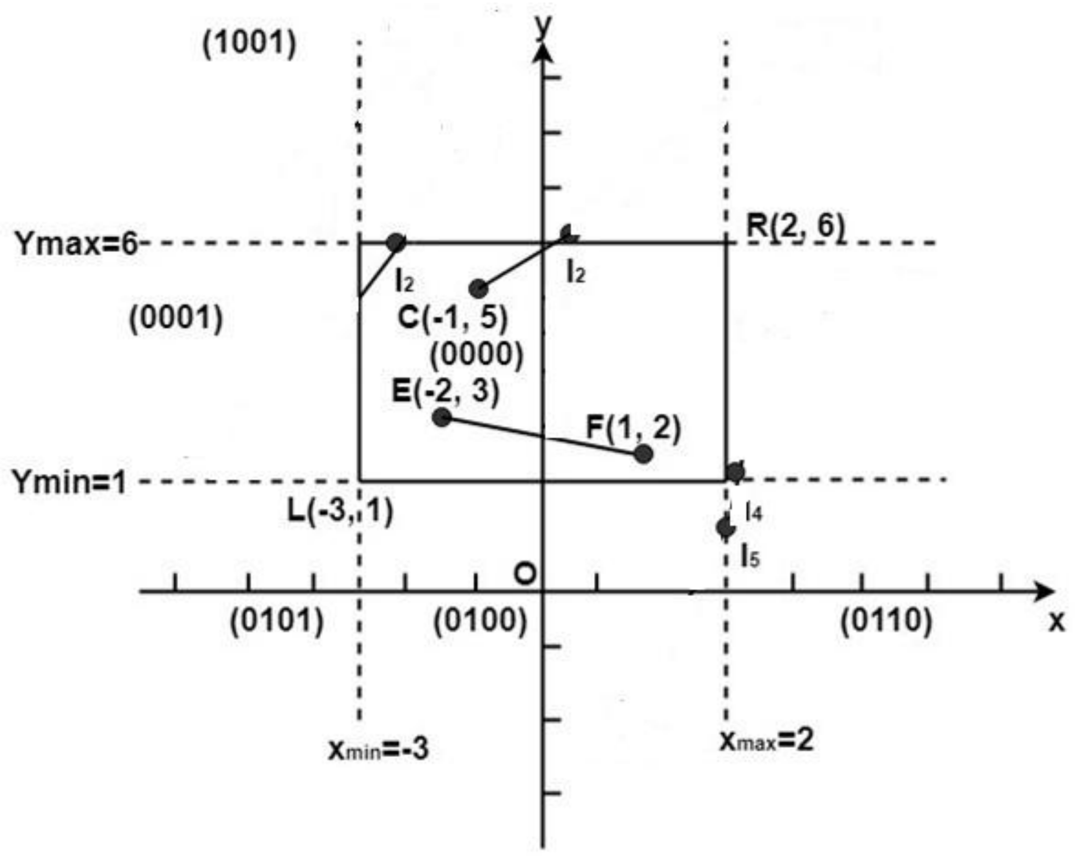
Example 2(Detailed Explanation) (LINE : CD)

- For clipping CD, we start with D since it is outside the window. Its code is 1010. We push the first 1 to a 0 by clipping against the line ymax=6. The resulting intersection I3 is (3/5,6),and its code is 0000. Thus I3 D is clipped and the remaining segment CI3 has both endpoints coded 0000 and so it is displayed.

Example 2(Detailed Explanation) (LINE : GH)

- For clipping GH, we can start with either G or H since both are outside the window. The code for G is 0100, and we push the 1 to a 0 by clipping against the line ymin=1.The resulting intersection point is I4 (11/5,1)and its code is 0010. We clip GI4 and work on I4 H. Segment I4 H is not displaying since (0010) AND (0010) =0010.

Final result (Clipped lines)


(1001)
y
Ymax=6
R(2, 6)
I2
I2
(0001)
C(-1, 5)
(0000)
E(-2, 3)
F(1, 2)
Ymin=1
L(-3, 1)
I4
I5
O
(0101)
(0100)
(0110)
x
xmin=-3
xmax=2