Computer Graphics

Module - 4

Two-Dimensional Viewing and Clipping

CSC305

By Prof. Sunil Katkar

Department of Computer Engineering, VCET

Module -4 Two-Dimensional Viewing and Clipping

Objective

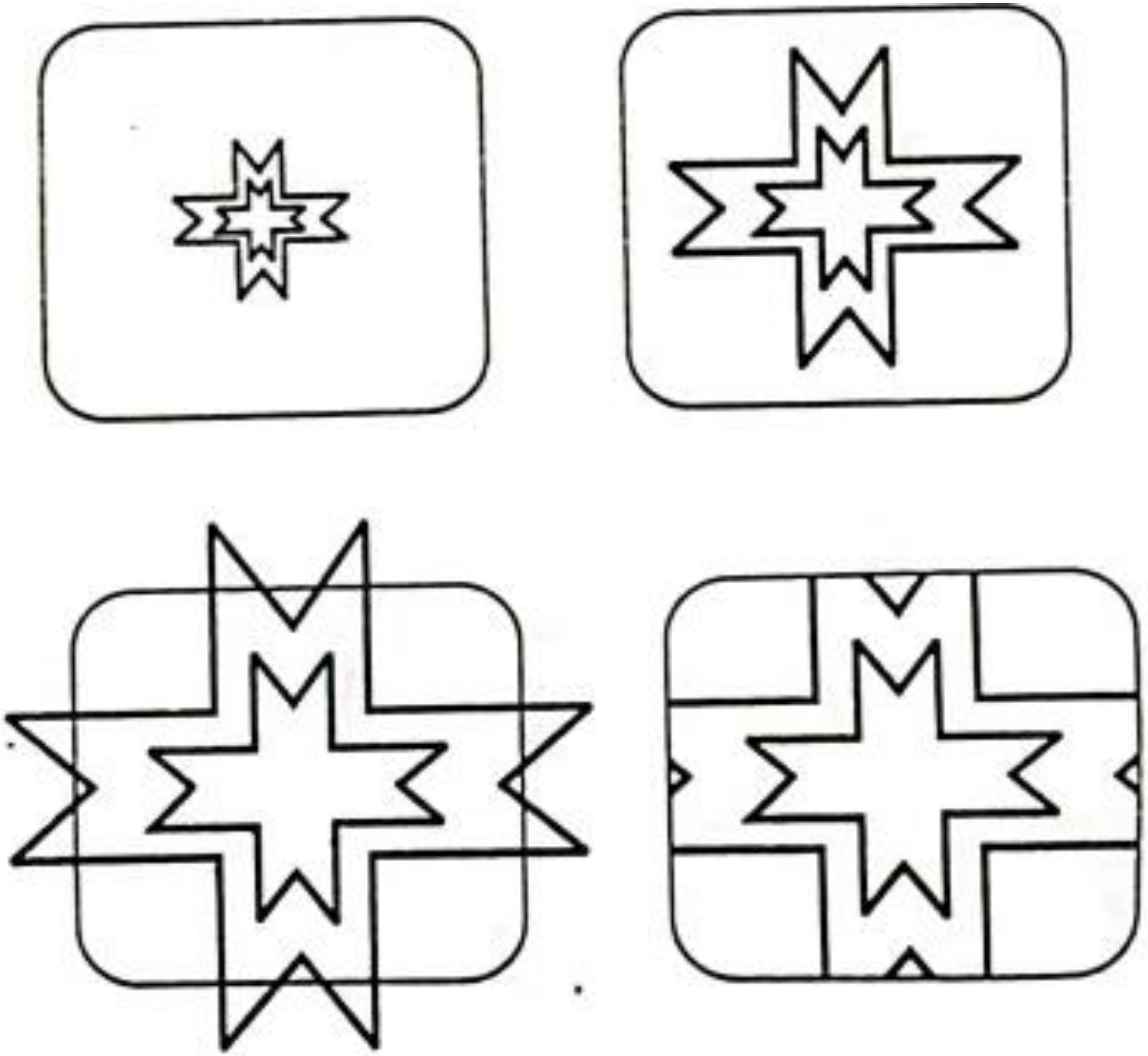
To emphasize on implementation aspect of Computer Graphics Algorithms.

Outcome

At the end of the course student will be able to:

apply line and polygon clipping algorithms on 2D graphical objects.

Two-Dimensional Viewing and Clipping



Viewing Transformation

The process of selecting and viewing the picture with different views is called windowing.

A process which divides each element of the picture into its visible and invisible portions, allowing the invisible portion to be discarded is called clipping.

The picture is stored in the computer memory using world coordinate system.

The picture is displayed on the display device it is measured in physical device coordinate system.

The viewing transformation which maps picture coordinates in the WCS to display coordinates in PDCS is performed by the the transformation -

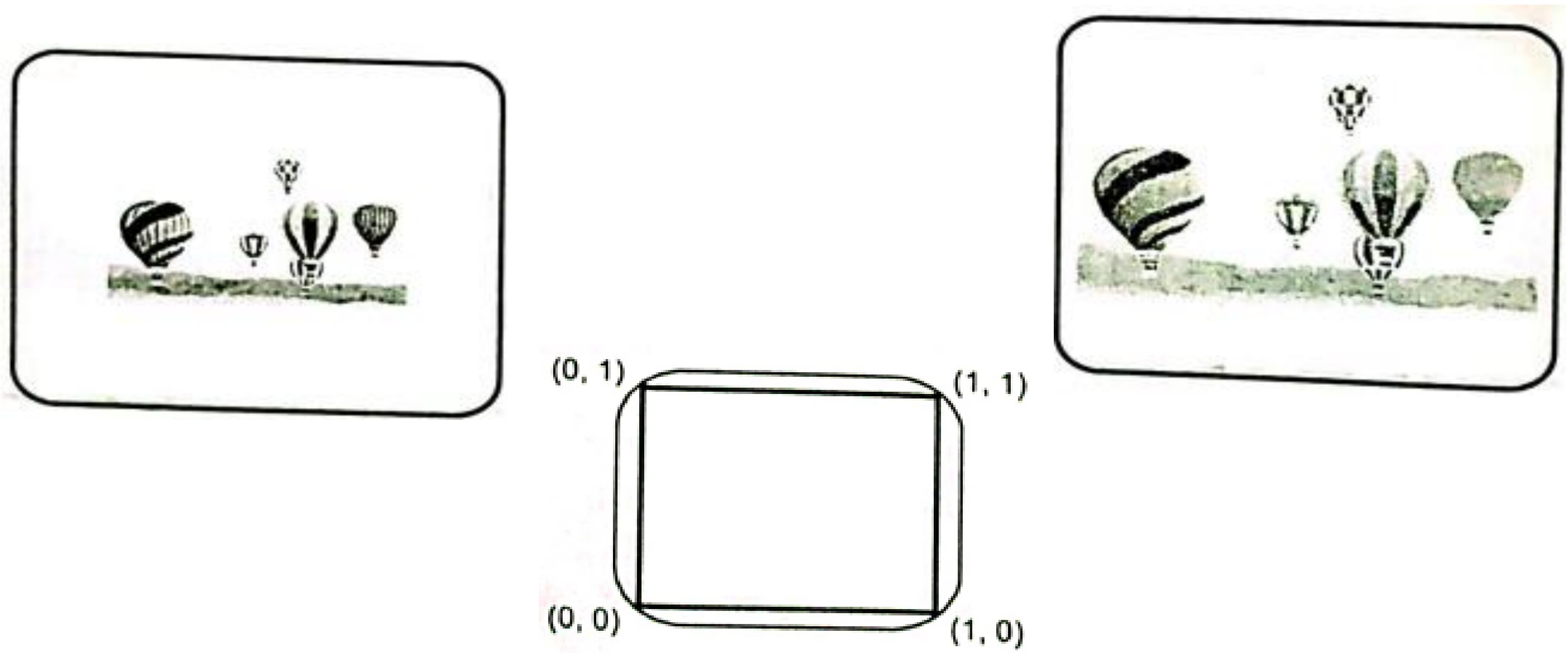
- Normalization transformation(N)
- Workstation transformation(W)

Normalization Transformation

To make our programs to be device independent,

- we define the picture coordinates in some units other than pixels.
- we use the interpreter to convert these coordinates to appropriate pixel values for the particular display device.

The device independent units are called the normalized device coordinates.



Normalization Transformation

The interpreter uses a simple linear formula to convert the normalized device coordinates to the actual device coordinates.

$$x = x_n * x_w$$

$$y = y_n * y_w$$

Where, x = Actual device x-coordinate

y = Actual device y-coordinate

x_n = Normalized x-coordinate

y_n = Normalized y-coordinate

x_w = Width of actual screen in pixels.

y_w = Height of actual screen in pixels.

The transformation which maps the world coordinates to normalized device coordinate is called normalization transformation.

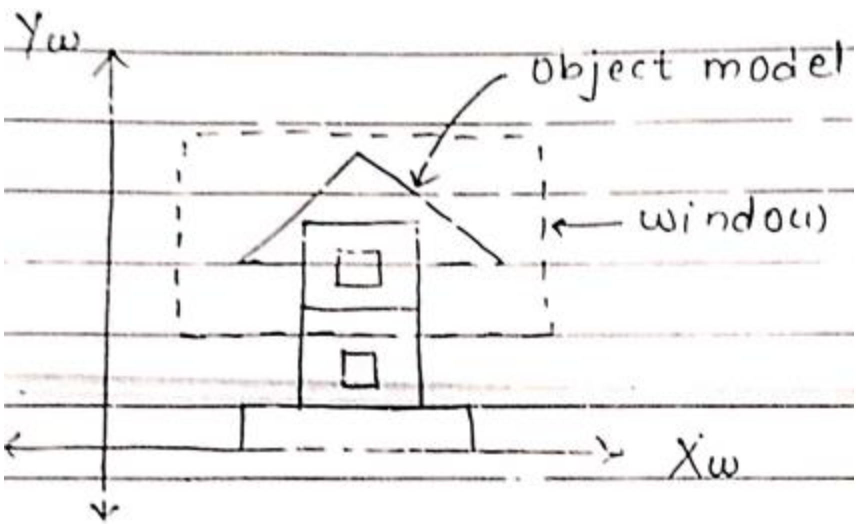
Workstation Transformation

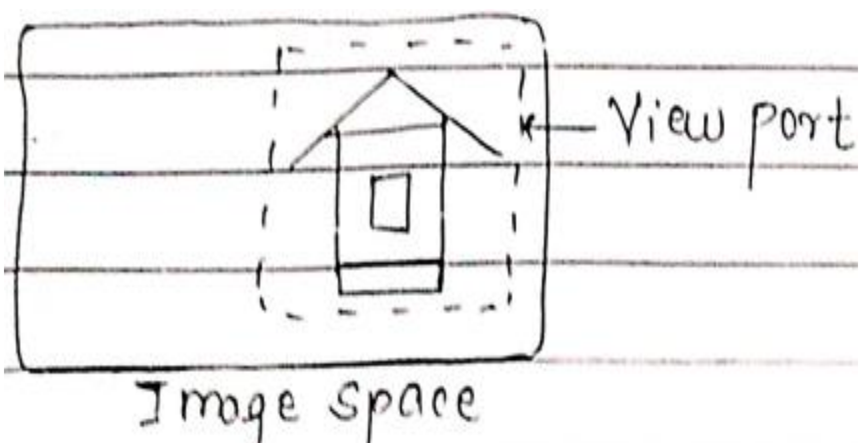
An area selected in world coordinate system is called **WINDOW**.

Window defines what is to be viewed.

An area on a display on a device to which a window is mapped is called a **VIEWPORT**.

Viewport defines where it is to be displayed.





Workstation Transformation

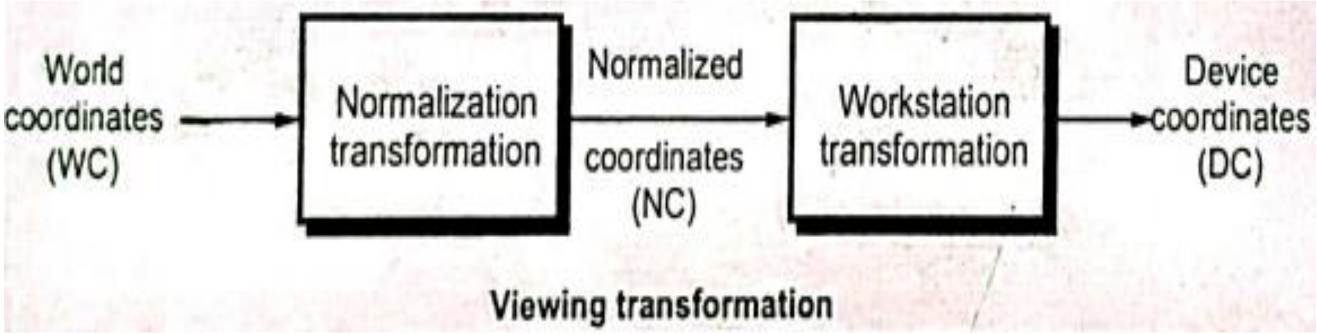
The window define in world coordinates is first transformed into the normalized device coordinates.

The normalized window is then transformed into the viewport coordinates.

This window to viewport coordinate transformation is known as workstation transformation.

The viewing transformation is the combination of normalization transformation and workstation transformations -

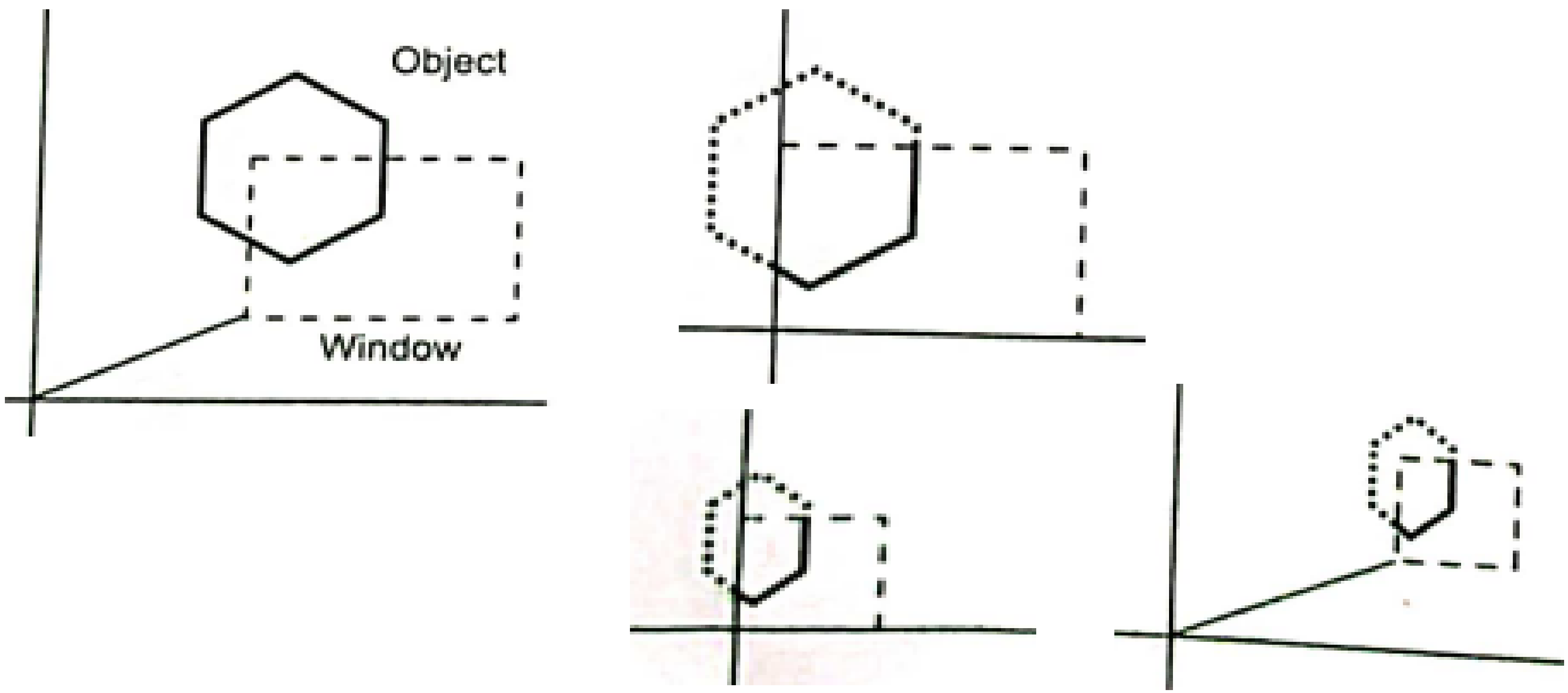
$$V = N.\ W$$



Window to Viewport Transformation

Steps -

- The object together with its window is translated until the lower left corner of the window is at the origin.
- Object and window are scaled until the window has the dimensions of the viewport.
- Translate the viewport to its correct position on the screen.



Window to Viewport Transformation

$\therefore$ workstation transformation is given as

$$W = T \cdot S \cdot T^{-1}$$

$$T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -x_{wmin} & -y_{wmin} & 1 \end{pmatrix}$$

$$S = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where, $s_x = (x_{vmax} - x_{vmin}) / (x_{wmax} - x_{wmin})$

$s_y = (y_{vmax} - y_{vmin}) / (y_{wmax} - y_{wmin})$

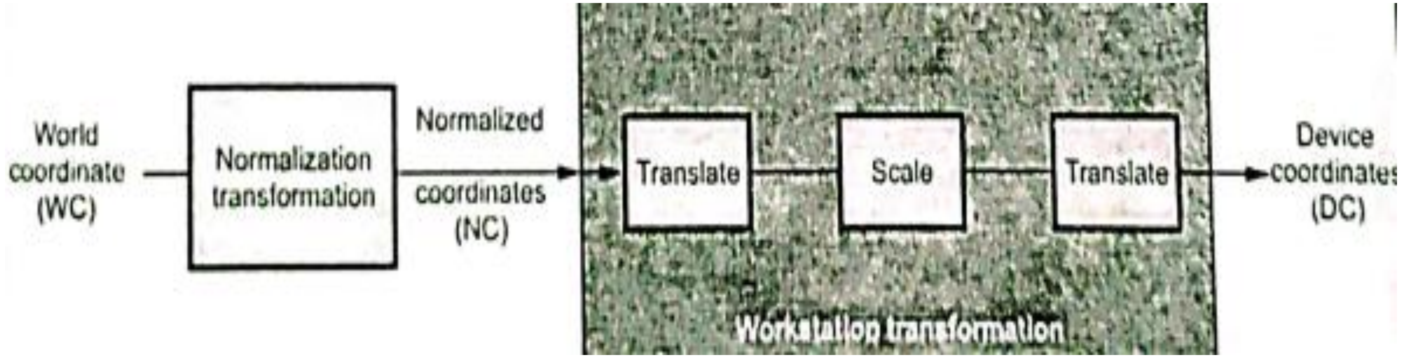
Window to Viewport Transformation

$$T^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_{vmin} & y_{vmin} & 1 \end{pmatrix}$$

The overall transformation matrix for W is given as

$$W = T \,.\, S \,.\, T^{-1}$$

$$W = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ x_{vmin} - x_{wmin}\, s_x & y_{vmin} - y_{wmin}\, s_y & 1 \end{pmatrix}$$



Window to Viewport Transformation

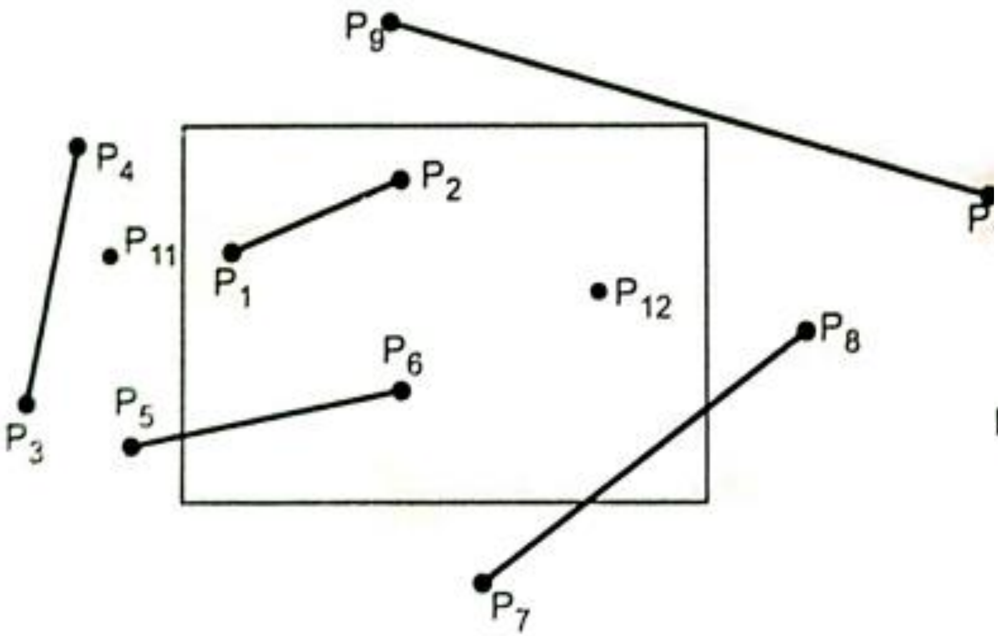
Find the normalization transformation window to viewport, with window, lower left corner at (1, 1), and upper right corner at (3, 5) onto a viewport with lower left corner at (0, 0) and upper right corner at (0.5, 0.5).

$$W = \begin{pmatrix} \mathbf{0.25} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0.125} & \mathbf{0} \\ \mathbf{-0.25} & \mathbf{-0.125} & \mathbf{1} \end{pmatrix}$$

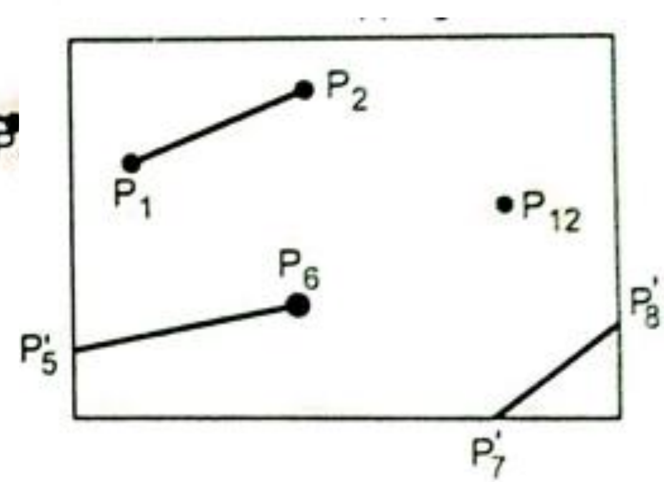
2D Clipping

The procedure that identifies the portions of a picture that are either inside or outside of a specified region of space is referred to as clipping.

The region against which an object is to be clipped is called clip window or clipping window.



(a) Before clipping



(b) After clipping

Line Clipping

Consider the line segment with endpoints $A(x_{1,} y_1)$ and $B(x_{2,} y_2)$

1. Any point p(x, y) is inside the window if all the following inequalities are satisfied -

$$x_{wmin} \leq x \leq x_{wmax}$$

$$y_{wmin} \leq y \leq y_{wmax}$$

2. If both the endpoints of a line segment are within the window then the line segment is visible.

3. If the line segment satisfies any one of the following condition then the line segment is not visible.

$$x_{1,} x_2 < x_{wmin}$$

$$x_{1,} x_2 > x_{wmax}$$

$$y_{1,} y_2 < y_{wmin}$$

$$y_{1,} y_2 > y_{wmax}$$

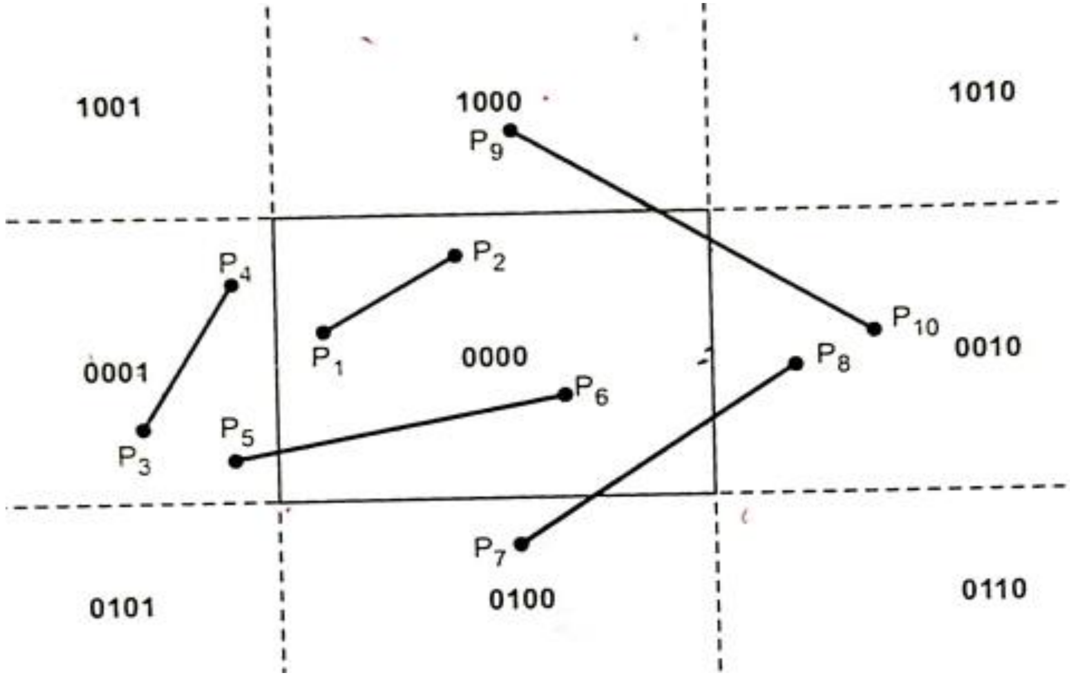
Line Clipping

4. If the line segment is neither of category 2 or 3, then it is a clipping candidate.

Line Clipping Algorithm

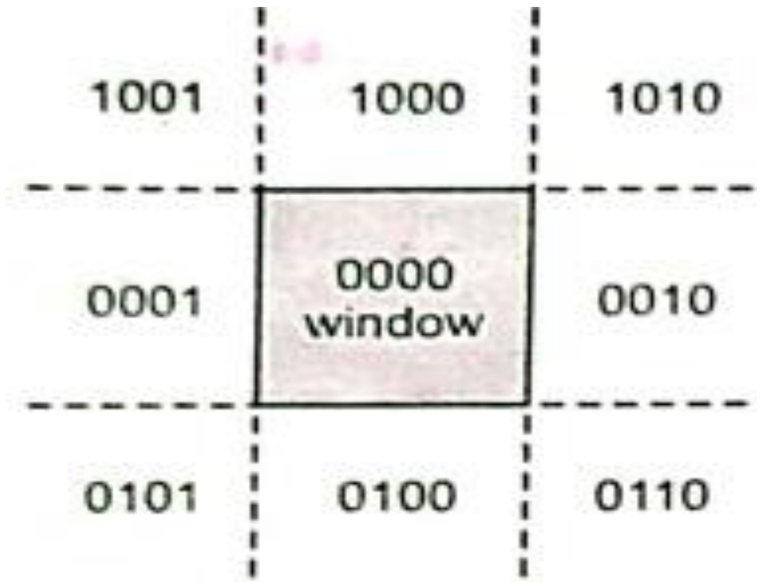
- Cohen-Sutherland Line Clipping Algorithm
- Liang-Barsky Line Clipping Algorithm

Line Clipping



Line	End Point Codes		Logical ANDing	Result
$P_1 P_2$	0000	0000	0000	Completely visible
$P_3 P_4$	0001	0001	0001	Completely invisible
$P_5 P_6$	0001	0000	0000	Partially visible
$P_7 P_8$	0100	0010	0000	Partially visible
$P_9 P_{10}$	1000	0010	0000	Partially visible

Cohen-Sutherland Line Clipping



* Algorithm has nine regions and
* Uses 4-bit code to indicate which of the 9-regions contains the endpoint of a line.
* These codes identify the location of the point relative to the boundaries of the clipping rectangle
* Each bit position in the region code is used to indicate one of the four relative coordinate positions of the point with respect to the clipping window - to the left, right, top or bottom.
* The rightmost bit is the first bit and the bits are set to '1' based on following rules

Set Bit-1 - if the end point is to the **left** of the window

Set Bit-2 - if the end point is to the **right** of the window

Set Bit-3 - if the end point is to the **below** the window

Set Bit-4 - if the end point is to the **above** the window

Otherwise, the bit is set to Zero

Cohen-Sutherland Line Clipping Algorithm

1. Read two end points of the line say P1(x_1, y_1), P2(x_2, y_2).
2. Read two corners (left-top and right-bottom) of the window.
3. Assign the 4-bit code to each endpoints of the line segment i. e. $B_4B_3B_2B_1$

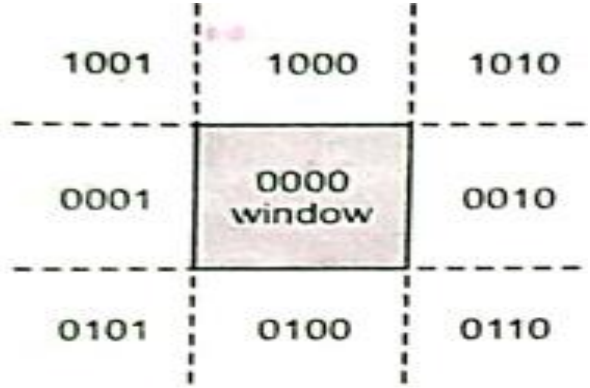
If $x < x_{wmin}$ then $B_1 = 1$ else $B_1 = 0$.

If $x < x_{wmax}$ then $B_2 = 1$ else $B_2 = 0$.

If $y < y_{wmin}$ then $B_3 = 1$ else $B_3 = 0$.

If $y < y_{wmax}$ then $B_4 = 1$ else $B_4 = 0$

The code is determine according to which of the following 9-regions of the plane the endpoint lies.



Cohen-Sutherland Line Clipping Algorithm

4. a) If both the endpoint codes are 0000 then the line is visible.

   Display the line segment.

   Stop.

   b) If logical AND of the endpoint codes are not 0000 then the line segment is not visible.

   Discard the line segment.

   Stop.

   c) If logical AND of the endpoint codes is 0000 then the line segment is clipping candidate.

5. Determine the intersecting boundaries -

   If $B_1 = 1$ intersect with $x = x_{wmin}$.

   If $B_2 = 1$ intersect with $x = x_{wmax}$.

   If $B_3 = 1$ intersect with $y = y_{wmin}$.

   If $B_4 = 1$ intersect with $y = y_{wmax}$.

Cohen-Sutherland Line Clipping Algorithm

6. Determine the intersecting point coordinate (x', y') -

   The equation of line passing through P1(x_1, y_1), P2(x_2, y_2). and (x', y') is

$(x' - x_1)/(x_2 - x_1) = (y' - y_1) / (y_2 - y_1)$

$(x' - x_1) = [(x_2 - x_1) / (y_2 - y_1)] * (y' - y_1)$

$(x' - x_1) = 1 / m * (y' - y_1)$

$x' = x_1 + [1 / m * (y' - y_1)]$ 1

Here $y' = y_{wmin}$ or $y' = y_{wmax}$

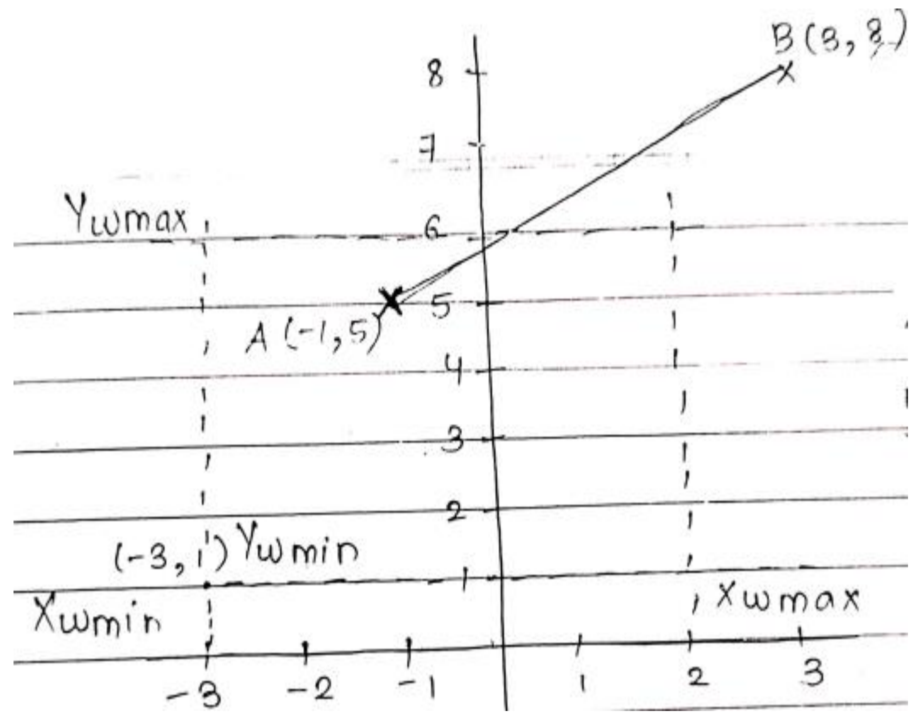
Similarly $y' = y_1 + m * (x' - x_1)$ 2

$x' = x_{wmin}$ or $x' = x_{wmax}$

where $m = (y_2 - y_1) / (x_2 - x_1)$

7. Go to step 3
8. Draw the remaining line segments.
9. Stop

Find the clipping coordinates of the line joining A(-1, 5), and B(3, 8). Given $x_{wmin} = -3$, $x_{wmax} = 2$, $y_{wmin} = 1$, $y_{wmax} = 6$.



Code for A(-1, 5)

B4B3B2B1 = 0000

∴ Point A is visible

Similarly, Code for B(3, 8)

B4B3B2B1 = 1010

∴ Point B is not visible.

Now

0000 AND 1010 = 0000

∴ Line AB is clipping candidate.

<u>Now Intersecting boundary</u>

Code for B(3, 8) = 1010

If B2 = 1, then intersect with $x = x_{wmax}$

If B4 = 1, then intersect with $y = y_{wmax}$

Here we select $x = x_{wmax} = 2$ is a clipping boundary.

For intersecting point coordinates

$$x' = x_{wmax} = 2$$

$$y' = y_1 + m * (x' - x_1)$$

$$m = (y_2 - y_1) / (x_2 - x_1) = (8 - 5) / (3 + 1) = 3/4$$

$$y' = 5 + 3/4 * (2 + 1) = 29/4$$

$$\therefore I'(x', y') = I'(2, 29/4)$$

$\therefore$ Now AI' is the clipped line segment.

Code for A(-1, 5) is

B4B3B2B1 = 0000

$\therefore$ Point A is visible

Similarly, Code for I'(2, 29/4) is

B4B3B2B1 = 1000

$\therefore$ Point I' is not visible.

Now

0000 AND 1000 = 0000

$\therefore$ Line AI' is clipping candidate.

Now Intersecting boundary

Code for I’(2, 29/4) = 1000

If B4 = 1, then intersect with $y = y_{wmax}$

Here we select $y = y_{wmax} = 6$ is a clipping boundary.

For intersecting point coordinates

$y' = y_{wmax} = 6$

$x' = x_1 + [1 / m * (y' - y_1)]$

$x' = -1 + 4/3 * (6- 5) = 1/3$

∴ I’’(x’, y’) = I’(1/3, 6)

∴ Now AI’’ is the clipped line segment.

Code for A(-1, 5) is

B4B3B2B1 = 0000

∴ Point A is visible

Similarly, Code for I’’(1/3, 6) is

B4B3B2B1 = 0000

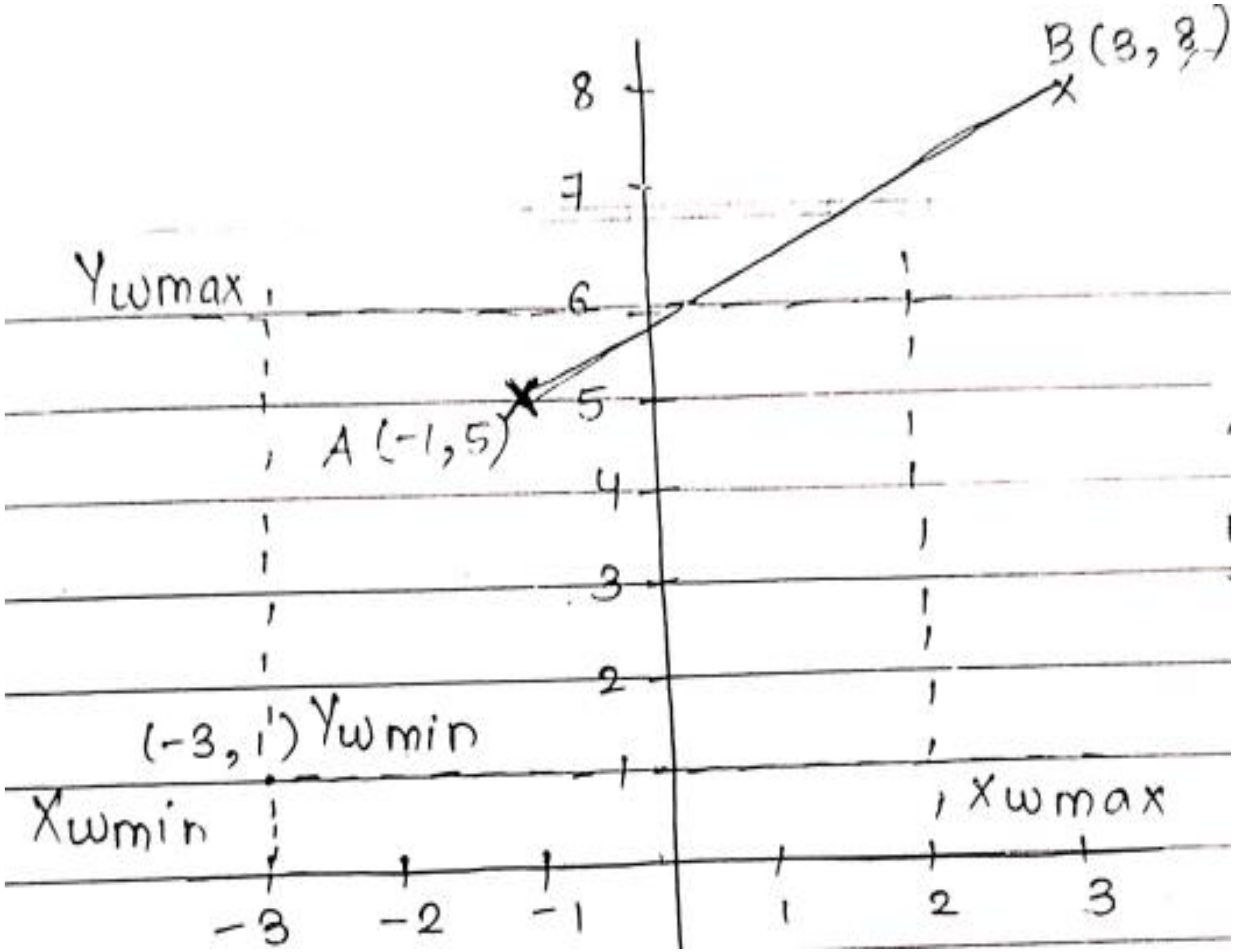
∴ Point I’’ is visible.

Now

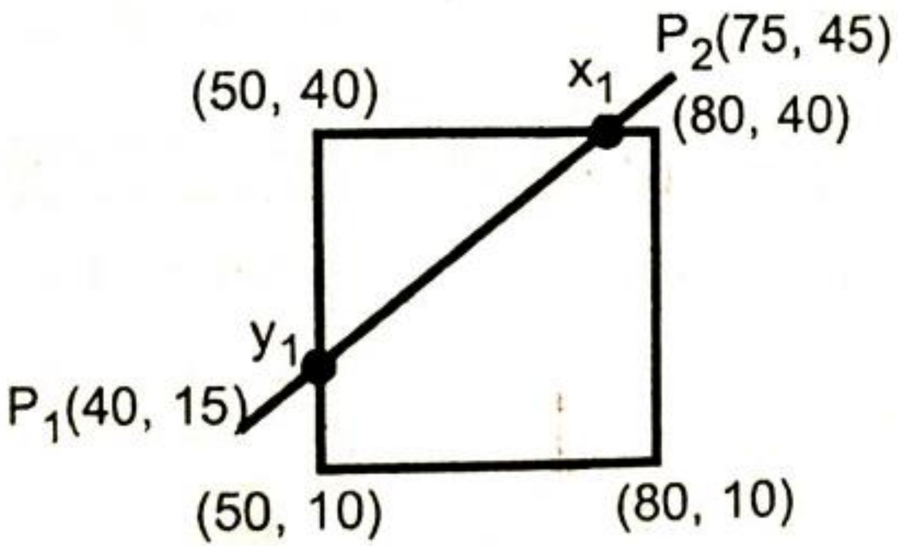
0000 AND 0000 = 0000

∴ **Line segment AI’’ is visible.**

Find the clipping coordinates of the line joining A(-1, 5), and B(3, 8). Given x_{wmin} = -3, x_{wmax}= 2 , y_{wmin} = 1, y_{wmax} = 6.



Find the clipping coordinates of the line joining A(40, 15), and B(75, 45). Given $x_{wmin} = 50$, $x_{wmax} = 80$, $y_{wmin} = 10$, $y_{wmax} = 40$.



Code for A(40, 15)

B4B3B2B1 = 0001

∴ Point A is not visible

Similarly, Code for B(75, 45)

B4B3B2B1 = 1000

∴ Point B is not visible.

Now

0001 AND 1000 = 0000

∴ Line AB is clipping candidate.

Now Intersecting boundary

Code for A(40, 15) = 0001

If B1 = 1, then intersect with $x = x_{wmin}$

Here we select $x = x_{wmin}$ is a clipping boundary.

For intersecting point coordinates

$$x' = x_{wmin} = 50$$

$$y' = y_1 + m * (x' - x_1)$$

$m = (y_2 - y_1) / (x_2 - x_1) = (45 - 15) / (75 - 40) = 0.8571$

$y' = 15 + 0.8571 * (50 - 40) = 23.571$

$\therefore$ I'(x', y') = I'(50, 23.571)

$\therefore$ Now I'B is the clipped line segment.

$\therefore$ Code for I'(50, 23.571) is

$\therefore$ B4B3B2B1 = 0000

$\therefore$ Point I' is visible

$\therefore$ Similarly, Code for B(75, 45) is

$\therefore$ B4B3B2B1 = 1000

$\therefore$ Point B is not visible.

Now

0000 AND 1000 = 0000

$\therefore$ Line I'B is clipping candidate.

Now Intersecting boundary

Code for B(75, 45) = 1000

If B4 = 1, then intersect with $y = y_{wmax}$

Here we select $y = y_{wmax}$ is a clipping boundary.

For intersecting point coordinates

$y' = y_{wmax} = 40$

$x' = x_1 + [1 / m * (y' - y_1)]$

$x' = 40 + 1.1667 * (40 - 15) = 69.1675$

$\therefore$ I''(x', y') = I''(69.1675, 40)

$\therefore$ Now I'I'' is the clipped line segment

Code for I'(50, 23.571) is

B4B3B2B1 = 0000

$\therefore$ Point I' is visible

Code for I''(69.1675, 40) is

B4B3B2B1 = 0000

$\therefore$ Point I'' is visible

$\therefore$ **Line segment I'I'' is visible.**

How the line between (2, 2), and (12, 9) is clipped against window with $(x_{wmin}, y_{wmin}) = (4, 4)$ and $(x_{wmax}, y_{wmax}) = (9, 8)$.

Liang-Barsky Line Clipping Algorithm

1. Read two end points of the line say $P1(x_1, y_1)$, $P2(x_2, y_2)$.
2. Read two corners (left-top and right-bottom) of the window.
3. Calculate p_k and q_k for $k = 1, 2, 3, 4$

$p_1 = -\Delta_x$ $\quad\quad$ $q_1 = x_1 - x_{wmin}$

$p_2 = \Delta_x$ $\quad\quad$ $q_2 = x_{wmax} - x_1$

$p_3 = -\Delta_y$ $\quad\quad$ $q_3 = y_1 - y_{wmin}$

$p_4 = \Delta_y$ $\quad\quad$ $q_4 = y_{wmax} - y_1$

4. If $p_k = 0$ then

{ the line is parallel to k^{th} boundary

If $q_k < 0$ then

the line is outside the boundary

Discard the line segment

Stop

If $q_k \geq 0$ then

the line is inside the parallel boundary

}

Liang-Barsky Line Clipping Algorithm

5. Calculate $r_k = q_k / p_k$ for $k = 1, 2, 3, 4$

6. Determine u_1 for all $p_k < 0$ from the set consisting $\{r_k , 0\}$

   select r_k for all $p_k < 0$

   then $u_1 = \{r_k , 0\}_{max}$

   Determine u_2 for all $p_k > 0$ from the set consisting $\{r_k , 1\}$

   select r_k for all $p_k > 0$

   then $u_2 = \{r_k , 1\}_{min}$

7. If $u_1 > u_2$ then

{ the line is completely outside the boundary

Discard the line segment

stop

}

Liang-Barsky Line Clipping Algorithm

8. Calculate the endpoints of the clipped line segment

$$x' = x_1 + u_1\Delta_x$$

$$y' = y_1 + u_1\Delta_y$$

$$\therefore\ I_1(x', y')$$

Similarly

$$x'' = x_1 + u_2\Delta_x$$

$$y'' = y_1 + u_2\Delta_y$$

$$\therefore\ I_2(x'', y'')$$

9. Display the line segment I_1I_2

10. Stop

Find the clipping coordinates of the line joining A(7,5), and B(9,7) using Liang-Barsky Line clipping algorithm against the window having Xwmin = 4, Xwmax= 10 , Ywmin = 4, Ywmax = 9.

$\Delta_x = x_2 - x_1 = 9 - 7 = 2$

$\Delta_y = y_2 - y_1 = 7 - 5 = 2$

Calculate p_k and q_k , for k = 1, 2, 3, 4

$p_1 = - \Delta_x = - 2$ and $q_1 = x_1 - x_{wmin} = 7 - 4 = 3$

$p_2 = \Delta_x = 2$ and $q_2 = x_{wmax} - x_1 = 10 - 7 = 3$

$p_3 = - \Delta_y = - 2$ and $q_3 = y_1 - y_{wmin} = 5 - 4 = 1$

$p_4 = \Delta_y = 2$ and $q_4 = y_{wmax} - y_1 = 9 - 5 = 4$

As $p_k \neq 0$, Calculate $r_k = q_k / p_k$, for k = 1, 2, 3, 4

For k =1, $p_1 < 0 \rightarrow$ $r_1 = q_1 / p_1 = 3 / - 2 = - 3/2$

For k =2, $p_2 < 0 \rightarrow$ $r_2 = q_2 / p_2 = 3 / 2 = 3/2$

For k =3, $p_3 < 0 \rightarrow$ $r_3 = q_3 / p_3 = 1 / - 2 = - 1/2$

For k =4, $p_4 < 0 \rightarrow$ $r_4 = q_4 / p_4 = 4 / 2 = 2$