Name - Laxmikant S Babaleshwar
Class - SE -AI&DS- C1
Roll No - 20

```
//Cohen Sutherland Algorithm
#include <iostream>
#include <graphics.h>
using namespace std;

class Coordinate
{
	public:
	int x, y;
	char code[4] = {'0', '0', '0', '0'};
};

class Lineclip
{
	public:
	void drawWindow();
	void drawLine(Coordinate p1, Coordinate p2);
	Coordinate setCode(Coordinate p);
	int visibility(Coordinate p1, Coordinate p2);
	Coordinate clipEndpoint(Coordinate p1, Coordinate p2);
};

int main()
{
	Lineclip lc;
	int gd = DETECT, gm;
	Coordinate p1, p2;

	cout << "Enter x1, y1: ";
	cin >> p1.x >> p1.y;
	cout << "Enter x2, y2: ";
	cin >> p2.x >> p2.y;

	initgraph(&gd, &gm, NULL);

	lc.drawWindow();
	lc.drawLine(p1, p2);
	delay(2000);
	cleardevice();
```

```
	p1 = lc.setCode(p1);		// Set codes for endpoints
	p2 = lc.setCode(p2);

	int vis = lc.visibility(p1, p2);

	if (vis == 0)
	{			// if vis = 0  Line Fully visible
		lc.drawWindow();
		lc.drawLine(p1, p2);
	}
	else if (vis == 2)
	{			// if vis = 2 Line Partially visible
		p1 = lc.clipEndpoint(p1, p2);
		p2 = lc.clipEndpoint(p2, p1);
		lc.drawWindow();
		lc.drawLine(p1, p2);
	}
	else
	{			// if vis = 1 Line Fully invisible
		lc.drawWindow();
	}

	delay(2000);
	closegraph();
	return 0;
}

void Lineclip::drawWindow()
{
	rectangle(150, 100, 450, 350);		// Draw rectangular clipping window
}

void Lineclip::drawLine(Coordinate p1, Coordinate p2)
{
	line(p1.x, p1.y, p2.x, p2.y);	// Draw line between points
}

Coordinate Lineclip::setCode(Coordinate p)
{
	if (p.y < 100) p.code[0] = '1';		// Above top
	if (p.y > 350) p.code[1] = '1';		// Below bottom
	if (p.x > 450) p.code[2] = '1';		// Right of window
	if (p.x < 150) p.code[3] = '1';		// Left of window
	return p;
```

```
}

int Lineclip::visibility(Coordinate p1, Coordinate p2)
{
	bool allZero = true, anyOverlap = false;

	for (int i = 0; i < 4; i++)
	{
		if (p1.code[i] == '1' && p2.code[i] == '1')
		return 1;		// Fully invisible
		if (p1.code[i] == '1' || p2.code[i] == '1')
		anyOverlap = true;
	}

	return anyOverlap ? 2 : 0;// 2 = Partial, 0 = Fully visible,ternary (conditional)operator used
}

Coordinate Lineclip::clipEndpoint(Coordinate p1, Coordinate p2)
{
	float m = (float)(p2.y - p1.y) / (p2.x - p1.x); // Slope of the line

	if (p1.code[3] == '1')
	{			// Left
		p1.y += m * (150 - p1.x);
		p1.x = 150;
	}
	else if (p1.code[2] == '1')
	{			// Right
		p1.y += m * (450 - p1.x);
		p1.x = 450;
	}
	else if (p1.code[0] == '1')
	{			// Top
		p1.x += (100 - p1.y) / m;
		p1.y = 100;
	}
	else if (p1.code[1] == '1')
	{			// Bottom
		p1.x += (350 - p1.y) / m;
		p1.y = 350;
	}
	return p1;
}
```

COMMAND:-

```
base) ubuntu@ubuntu-OptiPlex-3000:~$ g++ cohen1.cpp -o cohen1 -lgraph
base) ubuntu@ubuntu-OptiPlex-3000:~$ ./cohen1
```

INPUT:-

```
(base) ubuntu@ubuntu-OptiPlex-3000:~$ g++ cohen1.cpp -o cohen1 -lgraph
(base) ubuntu@ubuntu-OptiPlex-3000:~$ ./cohen1
Enter x1, y1: 100
100
Enter x2, y2: 400
400
```

OUTPUT:-