Name : **Shawn Louis** SE Comps 31

Experiment 7

Title	IMPLEMENTATION OF COHEN SUTHERLAND LINE CLIPPING AND WINDOW TO VIEWPORT TRANSFORMATION
Objective	To write a C program to implement Cohen Sutherland line clipping algorithm and to perform window to viewport transformation.
Algorithm	Line Clipping: 1. Start 2. Read the rectangular frame coordinates values rx1 , ry1, rx2, ry2 3. Read the line coordinate(x,y) 4. Find the slope of the line and the corresponding intersection values with the rectangular frames 5. Clip the portion of the line which lies outside the rectangular frame and retain the portion of the line that lies inside the rectangular frame 6. Stop Window to Viewport Transformation: 1. Start 2. Assign the viewport coordinates in the variables v1, v2, v3, v4 3. To perform transformation use the corresponding line coordinates 4. Determine the scaling factors using the formula sx = v3- v1 / rx2 – rx1 & sy = v4- v2 / ry2 – ry1 5. Map the line to the viewport using the line coordinate and the scaling factors 6. Display the viewport 7. Stop

Program	#include <stdio.h>
	#include <stdlib.h>
	#include <math.h>
	#include <graphics.h>
	#include <dos.h>
	
	typedef struct coordinate
	{
	int x, y;
	char code[4];
	} PT;
	
	void drawwindow();
	void drawline(PT p1, PT p2);
	PT setcode(PT p);
	int visibility(PT p1, PT p2);
	PT resetendpt(PT p1, PT p2);
	
	void main()
	{
	int gd = DETECT, v, gm;
	PT p1, p2, p3, p4, ptemp;

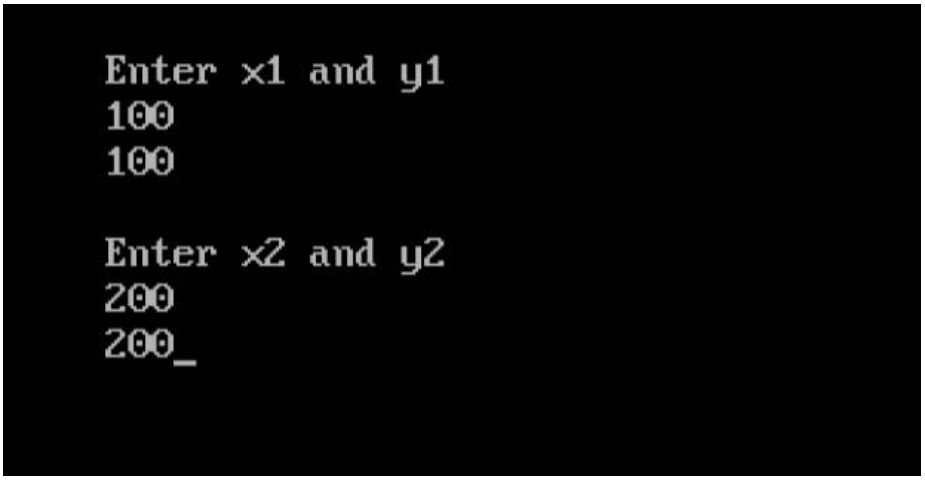
```
    printf("\nEnter x1 and y1\n");
    scanf("%d %d", &p1.x, &p1.y);
    printf("\nEnter x2 and y2\n");
    scanf("%d %d", &p2.x, &p2.y);

    initgraph(&gd, &gm, "c:\\turboc3\\bgi");
    drawwindow();
    delay(500);

    drawline(p1, p2);
    delay(500);
    cleardevice();

    delay(500);
    p1 = setcode(p1);
    p2 = setcode(p2);
    v = visibility(p1, p2);
    delay(500);

    switch (v)
    {
    case 0:
       drawwindow();
       delay(500);
       drawline(p1, p2);
       break;
    case 1:
       drawwindow();
       delay(500);
       break;
    case 2:
       p3 = resetendpt(p1, p2);
       p4 = resetendpt(p2, p1);
       drawwindow();
       delay(500);
       drawline(p3, p4);
       break;
    }

    delay(5000);
    closegraph();
}

void drawwindow()
{
    line(150, 100, 450, 100);
    line(450, 100, 450, 350);
    line(450, 350, 150, 350);
    line(150, 350, 150, 100);
}
```

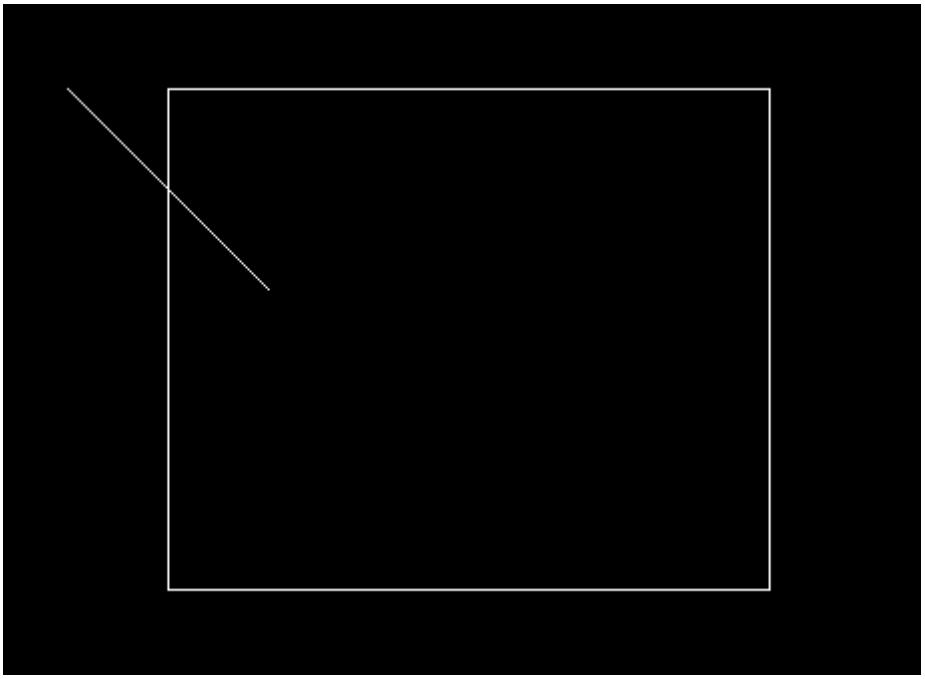
```
void drawline(PT p1, PT p2)
{
   line(p1.x, p1.y, p2.x, p2.y);
}

PT setcode(PT p) //for setting the 4 bit code
{
   PT ptemp;

   if (p.y < 100)
      ptemp.code[0] = '1'; //Top
   else
      ptemp.code[0] = '0';

   if (p.y > 350)
      ptemp.code[1] = '1'; //Bottom
   else
      ptemp.code[1] = '0';

   if (p.x > 450)
      ptemp.code[2] = '1'; //Right
   else
      ptemp.code[2] = '0';

   if (p.x < 150)
      ptemp.code[3] = '1'; //Left
   else
      ptemp.code[3] = '0';

   ptemp.x = p.x;
   ptemp.y = p.y;

   return (ptemp);
}

int visibility(PT p1, PT p2)
{
   int i, flag = 0;

   for (i = 0; i < 4; i++)
   {
      if ((p1.code[i] != '0') || (p2.code[i] != '0'))
         flag = 1;
   }

   if (flag == 0)
      return (0);

   for (i = 0; i < 4; i++)
   {
      if ((p1.code[i] == p2.code[i]) && (p1.code[i] == '1'))
```

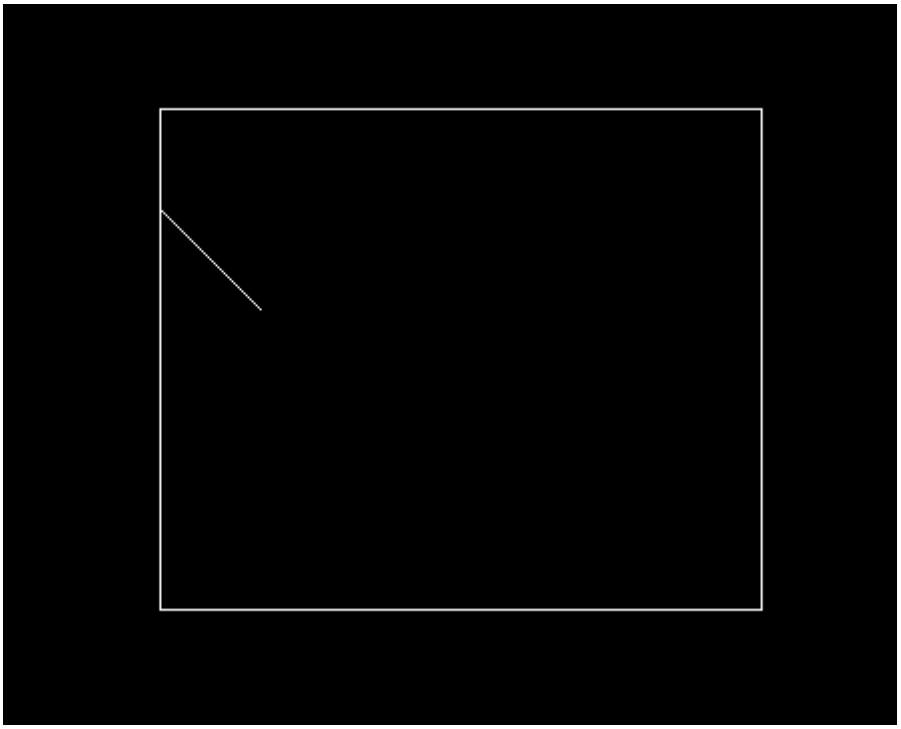
```
            flag = '0';
    }

    if (flag == 0)
        return (1);

    return (2);
}

PT resetendpt(PT p1, PT p2)
{
    PT   temp;
    int  x,  y,  i;
    float m, k;

    if (p1.code[3] == '1')
        x = 150;

    if (p1.code[2] == '1')
        x = 450;

    if ((p1.code[3] == '1') || (p1.code[2] == '1'))
    {
        m = (float)(p2.y - p1.y) / (p2.x - p1.x);
        k = (p1.y + (m * (x - p1.x)));
        temp.y = k;
        temp.x = x;

        for (i = 0; i < 4; i++)
            temp.code[i] = p1.code[i];

        if (temp.y <= 350 && temp.y >= 100)
            return (temp);
    }

    if (p1.code[0] == '1')
        y = 100;

    if (p1.code[1] == '1')
        y = 350;

    if ((p1.code[0] == '1') || (p1.code[1] == '1'))
    {
        m = (float)(p2.y - p1.y) / (p2.x - p1.x);
        k = (float)p1.x + (float)(y - p1.y) / m;
        temp.x = k;
        temp.y = y;

        for (i = 0; i < 4; i++)
            temp.code[i] = p1.code[i];
```

```
        return (temp);
    }
    else
        return (p1);
}
```

Output





	
Conclusion	Thus a C program to implement Cohen Sutherland line clipping algorithm and to perform window to viewport transformation was written and executed.