**Experiment No** : 9

**Aim** : To implement Cohen-Sutherland Line Clipping Algorithm.

**Theory** :
In computer graphics, line clipping is the process of removing lines or portions of lines outside an area of interest. Typically, any line or part there of which is outside of the viewing area is removed. A line-clipping method consists of various parts. Tests are conducted on a given line segment to find out whether it lies outside the view volume. Afterwards, intersection calculations are carried out with one or more clipping boundaries. Determining which portion of the line is inside or outside of the clipping volume is done by processing the endpoints of the line with regards to the intersection.

**Algorithm**
1. Initially, every line endpoint in a picture is assigned a four-digit binary value, called a region code, and each bit position is used to indicate whether the point is inside or outside one of the clipping-window boundaries
2. Using region codes determine which lines are completely outside the clipping boundary and eliminate them.
3. Using region codes determine which lines are completely inside the clipping boundary and save them.
4. To determine whether a line crosses a selected clipping boundary, we can check corresponding bit values in the two endpoint region codes. If one of these bit values is 1 and the other is 0, the line segment crosses that boundary.
5. Use line equations to determine intersection of line and boundary.
6. Clip the line to the intersection point and save it

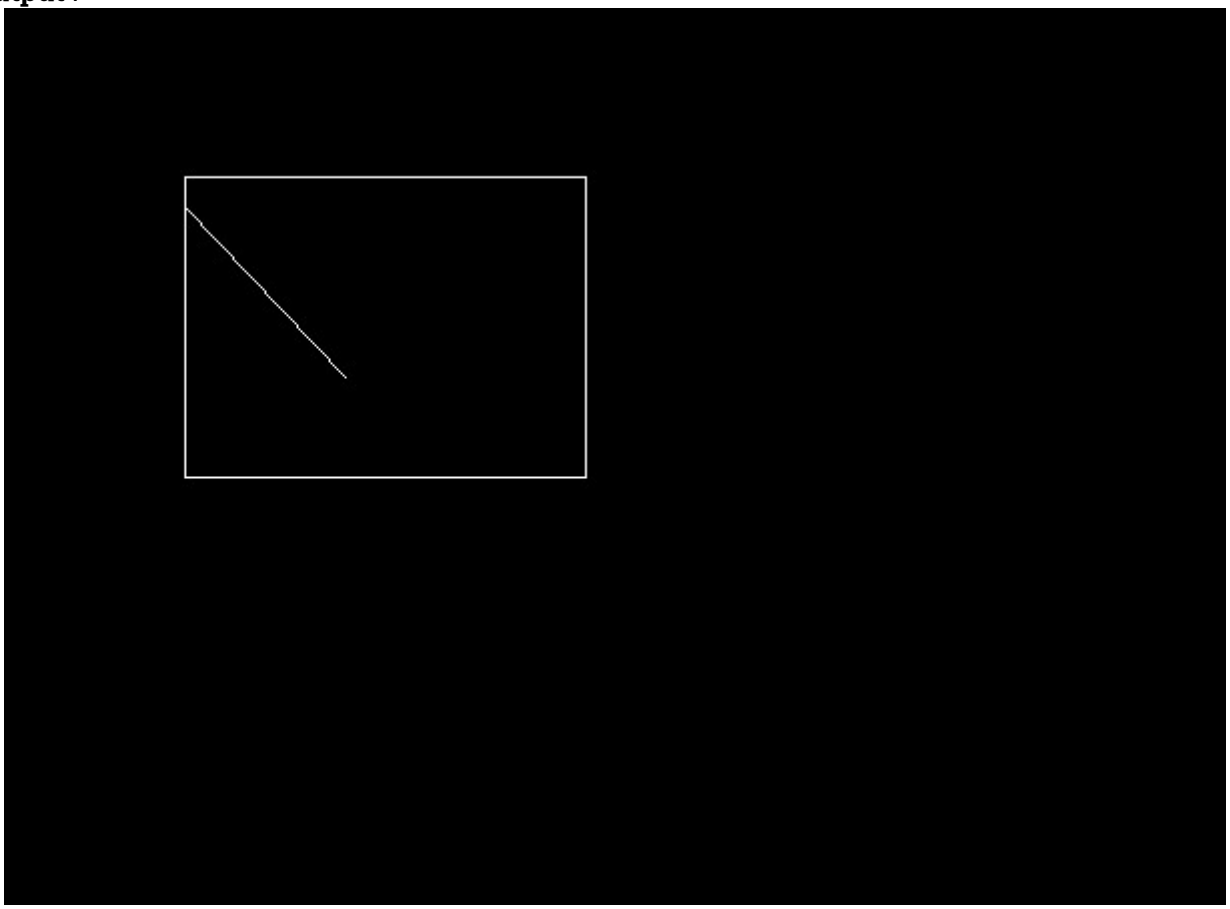
**Code & Output** :
```
#include<graphics.h>
#include<iostream>
using namespace std;
void get_region_code(float x1, float y1, float x2, float y2, int rc0[], int rc1[]);
void cohen_sutherland(float x1, float y1, float x2, float y2);
void clip_top(float &x,float &y, float a, float b);
void clip_bottom(float &x,float &y, float a, float b);
void clip_right(float &x,float &y, float a, float b);
void clip_left(float &x,float &y, float a, float b);
bool find_and_operation(int rc0[], int rc1[]);
float m = 0;
int XWmin = 100;
int YWmin = 100;
int XWmax = 300;
int YWmax = 250;

int main()
{
  float x1,y1,x2,y2;

  cout<<"Program to demonstrate Cohen Sutherland Line clipping algorithm"<<endl;
  cout<<endl;

  cout<<"Enter x1 : ";
  cin>>x1;
  cout<<"Enter y1 : ";
  cin>>y1;
  cout<<endl;
```

```
  cout<<"Enter x2 : ";
  cin>>x2;
  cout<<"Enter y2 : ";
  cin>>y2;
  m = (y2 - y1)/(x2 - x1);
  cohen_sutherland(x1,y1,x2,y2);


  return 0;
}

void cohen_sutherland(float x1,float y1, float x2, float y2)
{
   int rc0[4];
   int rc1[4];
   float a0,b0,a1,b1;
   a0 = x1;
   b0 = y1;
   a1 = x2;
   b1 = y2;
   get_region_code(a0,b0,a1,b1,rc0,rc1);

if((rc0[0]==0)&&(rc0[1]==0)&&(rc0[2]==0)&&(rc0[3]==0)&&(rc1[0]==0)&&(rc1[1]==0)&&(rc1[2]==0)&&(rc1[
3]==0))
   {
      int gd=DETECT,gm;
      initgraph(&gd,&gm,NULL);
      line(100,100,100,250); // Xmin = 100
      line(100,100,300,100); // Ymin = 100
      line(100,250,300,250); // Xmax = 300
      line(300,100,300,250); // Ymax = 250

      line(a0,b0,a1,b1);

      delay(1000000);
      closegraph();
   }
   else
   {
      if(find_and_operation(rc0,rc1)==true)
      {
         cout<<endl;
         cout<<"Line is discarded"<<endl;
      }
      else
      {
         if((rc0[0] != 0) || (rc0[1] != 0) || (rc0[2] != 0) || (rc0[3] != 0))
         {
            if(rc0[3] != 0)
            {
               clip_top(a0,b0,a1,b1);
            }
            else if(rc0[2] != 0)
            {
               clip_bottom(a0,b0,a1,b1);
            }
            else if(rc0[1] != 0)
            {
               clip_right(a0,b0,a1,b1);
            }
```

```
            else if(rc0[0] != 0)
            {
               clip_left(a0,b0,a1,b1);
            }
         }
         if((rc1[0] != 0) || (rc1[1] != 0) || (rc1[2] != 0) || (rc1[3] != 0))
         {
            if(rc1[3] != 0)
            {
               clip_top(a1,b1,a0,b0);
            }
            else if(rc1[2] != 0)
            {
               clip_bottom(a1,b1,a0,b0);
            }
            else if(rc1[1] != 0)
            {
               clip_right(a1,b1,a0,b0);
            }
            else if(rc1[0] != 0)
            {
               clip_left(a1,b1,a0,b0);
            }
         }

         cohen_sutherland(a0,b0,a1,b1);
      }
   }
}

void clip_top(float &x,float &y, float a, float b)
{
   y = YWmax;
   x = a + ((y-b)/m);
}

void clip_bottom(float &x,float &y, float a, float b)
{
   y = YWmin;
   x = a + ((y-b)/m);
}

void clip_right(float &x,float &y, float a, float b)
{
   x = XWmax;
   y = b + m*(x-a);
}

void clip_left(float &x,float &y, float a, float b)
{
   x = XWmin;
   y = b + m*(x-a);
}

bool find_and_operation(int rc0[], int rc1[])
{
   if((rc0[0]==1)&&(rc1[0]==1))
      return true;
   else if((rc0[1]==1)&&(rc1[1]==1))
      return true;
```

```
    else if((rc0[2]==1)&&(rc1[2]==1))
        return true;
    else if((rc0[3]==1)&&(rc1[3]==1))
        return true;
    else
        return false;

}

void get_region_code(float x1, float y1, float x2, float y2, int rc0[], int rc1[]){   // For first point
    if(x1 - XWmin >= 0.0)
      rc0[0] = 0;
    else
      rc0[0] = 1;

    if(XWmax - x1 >= 0.0)
      rc0[1] = 0;
    else
      rc0[1] = 1;

    if(y1 - YWmin >= 0.0)
      rc0[2] = 0;
    else
      rc0[2] = 1;

    if(YWmax - y1 >= 0.0)
      rc0[3] = 0;
    else
      rc0[3] = 1;

    //For second point
    if(x2 - XWmin >= 0.0)
      rc1[0] = 0;
    else
      rc1[0] = 1;

    if(XWmax - x2 >= 0.0)
      rc1[1] = 0;
    else
      rc1[1] = 1;

    if(y2 - YWmin >= 0.0)
      rc1[2] = 0;
    else
      rc1[2] = 1;

    if(YWmax - y2 >= 0.0)
      rc1[3] = 0;
    else
      rc1[3] = 1;

}
```

**Output** :



**Conclusion** : Program to implement Cohen-Sutherland line clipping was successfully written and executed

**Deepraj Bhosale**
**181105016**