**Practical No 8.C**

**Implementation of SUTHERLAND HODGEMAN POLYGON clipping algorithms.**

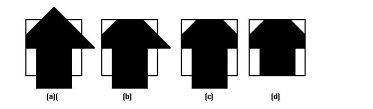
**Aim: Write a program to implement a Sutherland Hodgeman Polygon clipping algorithm.**

**Theory:**

The Sutherland - Hodgman algorithm performs a clipping of a polygon against each window edge in turn. It accepts an ordered sequence of vertices v1, v2, v3, ..., vn and puts out a set of vertices defining the clipped polygon. The following figure given below represents a polygon (the large, solid, upward pointing arrow) before clipping has occurred.



The following figures below show how this algorithm works at each edge, clipping the polygon.



1. Clipping against the left side of the clip window.

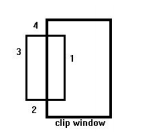
2. Clipping against the top side of the clip window.

3. Clipping against the right side of the clip window.

4. Clipping against the bottom side of the clip window.

**Four Types of Edges**

As the algorithm goes around the edges of the window, clipping the polygon, it encounters four types of edges. All four edge types are illustrated by the polygon in the following figure. For each edge type, zero, one, or two vertices are added to the output list of vertices that define the clipped polygon. The four types of edges are:

1. Edges that are totally inside the clip window: - add the second inside vertex point.

2. Edges that are leaving the clip window: - add the intersection point as a vertex.

3. Edges that are entirely outside the clip window: - add nothing to the vertex output list.

4. Edges that are entering the clip window: - save the intersection and inside points as vertices.

**How To Calculate Intersections**

Assume that we're clipping a polygon’s edge with vertices at (x1,y1) and (x2,y2) against a clip window with vertices at (xmin, ymin) and (xmax,ymax).

The location (IX, IY) of the intersection of the edge with the left side of the window is:

1. IX = xmin

2. IY = slope\*(xmin-x1) + y1, where the slope = (y2-y1)/(x2-x1)

The location of the intersection of the edge with the right side of the window is:

1. IX = xmax

2. IY = slope\*(xmax-x1) + y1, where the slope = (y2-y1)/(x2-x1)

The intersection of the polygon's edge with the top side of the window is:

1. IX = x1 + (ymax - y1) / slope

2. IY = ymax

Finally, the intersection of the edge with the bottom side of the window is:

1. IX = x1 + (ymin - y1) / slope

2. IY = ymin

**Conclusion: We have implemented Sutherland Hodgeman Polygon clipping algorithm.**

**Code:**

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#define round(a) ((int)(a+0.5))

int k;

float xmin,ymin,xmax,ymax,arr[20],m;

void clipl(float x1,float y1,float x2,float y2)

{

if(x2-x1)

m=(y2-y1)/(x2-x1);

else

m=100000;

if(x1 >= xmin && x2 >= xmin)

{

arr[k]=x2; arr[k+1]=y2; k+=2;

}

if(x1 < xmin && x2 >= xmin)

{

arr[k]=xmin;

arr[k+1]=y1+m\*(xmin-x1);

arr[k+2]=x2;

arr[k+3]=y2;

k+=4;

}

if(x1 >= xmin && x2 < xmin)

{

arr[k]=xmin;

arr[k+1]=y1+m\*(xmin-x1);

k+=2;

}

}

void clipt(float x1,float y1,float x2,float y2)

{

if(y2-y1)

m=(x2-x1)/(y2-y1);

else

m=100000;

if(y1 <= ymax && y2 <= ymax)

{

arr[k]=x2;

arr[k+1]=y2;

k+=2;

}

if(y1 > ymax && y2 <= ymax)

{

arr[k]=x1+m\*(ymax-y1);

arr[k+1]=ymax; arr[k+2]=x2;

arr[k+3]=y2;

k+=4;

}

if(y1 <= ymax && y2 > ymax)

{

arr[k]=x1+m\*(ymax-y1);

arr[k+1]=ymax;

k+=2; } }

void clipr(float x1,float y1,float x2,float y2)

{

if(x2-x1)

m=(y2-y1)/(x2-x1);

else

m=100000;

if(x1 <= xmax && x2 <= xmax)

{

arr[k]=x2; arr[k+1]=y2; k+=2;

}

if(x1 > xmax && x2 <= xmax)

{

arr[k]=xmax;

arr[k+1]=y1+m\*(xmax-x1);

arr[k+2]=x2;

arr[k+3]=y2;

k+=4;

}

if(x1 <= xmax && x2 > xmax)

{

arr[k]=xmax;

arr[k+1]=y1+m\*(xmax-x1);

k+=2; } }

void clipb(float x1,float y1,float x2,float y2)

{

if(y2-y1)

m=(x2-x1)/(y2-y1);

else

m=100000;

if(y1 >= ymin && y2 >= ymin) {

arr[k]=x2;

arr[k+1]=y2; k+=2; }

if(y1 < ymin && y2 >= ymin) {

arr[k]=x1+m\*(ymin-y1);

arr[k+1]=ymin;

arr[k+2]=x2;

arr[k+3]=y2;

k+=4;

}

if(y1 >= ymin && y2 < ymin)

{

arr[k]=x1+m\*(ymin-y1); arr[k+1]=ymin;

k+=2; } }

void main() {

int gdriver=DETECT,gmode,n,poly[20];

float xi,yi,xf,yf,polyy[20];

clrscr();

cout<<"Coordinates of rectangular clip window :\nxmin,ymin: ";

cin>>xmin>>ymin;

cout<<"xmax,ymax: ";

cin>>xmax>>ymax;

cout<<"\n\nPolygon to be clipped \nNumber of sides: ";

cin>>n;

cout<<"Enter the coordinates: \n";

for(int i=0;i<2\*n;i++)

cin>>polyy[i];

polyy[i]=polyy[0];

polyy[i+1]=polyy[1];

for(i=0;i < 2\*n+2;i++)

poly[i]=round(polyy[i]);

initgraph(&gdriver,&gmode,"C:\\TURBOC3\\BGI");

setcolor(RED);

rectangle(xmin,ymax,xmax,ymin);

cout<<"\t\tUNCLIPPED POLYGON";

setcolor(BLUE);

fillpoly(n,poly);

getch();

cleardevice();

k=0;

for(i=0;i < 2\*n;i+=2)

clipl(polyy[i],polyy[i+1],polyy[i+2],polyy[i+3]);

n=k/2;

for(i=0;i < k;i++)

polyy[i]=arr[i];

polyy[i]=polyy[0];

polyy[i+1]=polyy[1];

k=0;

for(i=0;i < 2\*n;i+=2)

clipt(polyy[i],polyy[i+1],polyy[i+2],polyy[i+3]);

n=k/2;

for(i=0;i < k;i++)

polyy[i]=arr[i];

polyy[i]=polyy[0];

polyy[i+1]=polyy[1];

k=0;

for(i=0;i < 2\*n;i+=2)

clipr(polyy[i],polyy[i+1],polyy[i+2],polyy[i+3]);

n=k/2;

for(i=0;i < k;i++)

polyy[i]=arr[i];

polyy[i]=polyy[0];

polyy[i+1]=polyy[1];

k=0;

for(i=0;i < 2\*n;i+=2)

clipb(polyy[i],polyy[i+1],polyy[i+2],polyy[i+3]);

for(i=0;i < k;i++)

poly[i]=round(arr[i]);

if(k)

fillpoly(k/2,poly);

setcolor(RED);

rectangle(xmin,ymax,xmax,ymin);

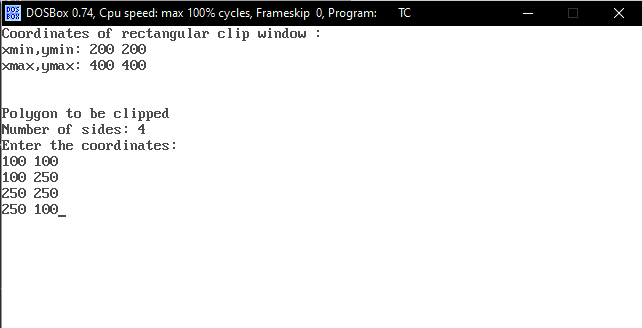
cout<<"CLIPPED POLYGON";

getch();

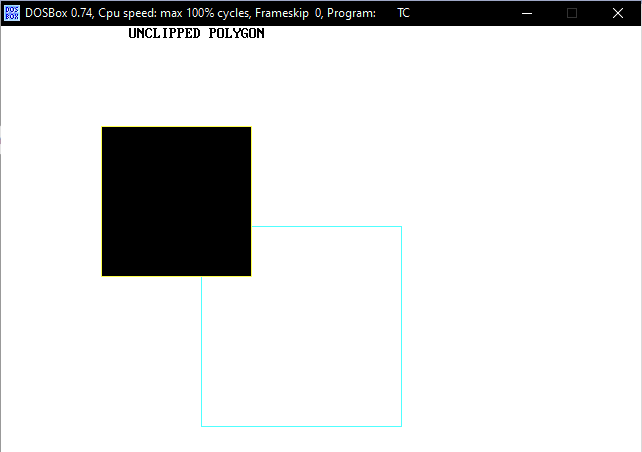
closegraph();

}

**Output:**



**Unclipped polygon:**



**Clipped polygon:**

