Sutherland line clipping algorithm

#include<iostream>

#include<graphics.h>

#include<conio.h>

using namespace std ;

const int INSIDE = 0 ; const int LEFT = 1 ;

const int RIGHT = 2 ; const int BOTTOM = 4 ; const int TOP = 8 ;

const int x\_min = 100 ; const int x\_max = 300 ;

const int y\_min = 100 ; const int y\_max = 300 ;

int code\_compute(double , double ) ;

void suther\_land\_clipping(double , double , double , double) ;

int main()

{ int gd = DETECT , gm ;

int x1 , y1 , x2 , y2 ;

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

rectangle(x\_min , y\_min , x\_max , y\_max) ; getch();

cout<<"Enter points(x1,y1) & (x2 ,y2) of line :"<<endl ;

cin>>x1>>y1>>x2>>y2 ;

cout<<"Before clipping :"<<endl ; line(x1, y1 , x2 , y2 ) ; getch() ;

suther\_land\_clipping(x1, y1, x2 ,y2) ; getch() ;

closegraph(); return 0 ; }

void suther\_land\_clipping(double x1 , double y1 , double x2 , double y2)

{ int code1 = code\_compute(x1, y1) ;

int code2 = code\_compute(x2 , y2) ;

bool accept = false ; int x , y , code\_out ;

while(true)

{ if(code1 == 0 && code2 == 0)

{ accept = true ; break ;}

else if(code1&code2)

{break ; }

else

{ if(code1 !=0 )

{ code\_out = code1 ; }

else

{ code\_out = code2 ; }

if(code\_out&TOP)

{ y = y\_max ; x = x1 + (x2 - x1)\*(y\_max - y1)/(y2 - y1) ; }

else if(code\_out&BOTTOM)

{ y = y\_min ; x = x1 + (x2 -x1)\*(y\_min -y1)/(y2-y1) ; }

else if(code\_out&LEFT)

{ x = x\_min ; y = y1 + (y2-y1)\*(x\_min-x1)/(x2-x1) ; }

else if (code\_out&RIGHT)

{ x = x\_max ; y = y1 + (y2-y1)\*(x\_max-x1)/(x2-x1) ; }}

if(code\_out == code1)

{ x1 = x ; y1 = y ; code1 = code\_compute(x1, y1) ; }

else

{ x2 = x ; y2 = y ; code2 = code\_compute(x2, y2) ; } }

if(accept ==true)

{ cout<<"After clipping"<<endl ; cleardevice() ;

rectangle(x\_min , y\_min , x\_max , y\_max) ; line(x1,y1,x2,y2) ; } }

int code\_compute(double x , double y )

{ int code = INSIDE ;

if(x<x\_min) code |= LEFT ; else if(x>x\_max) code |= RIGHT ;

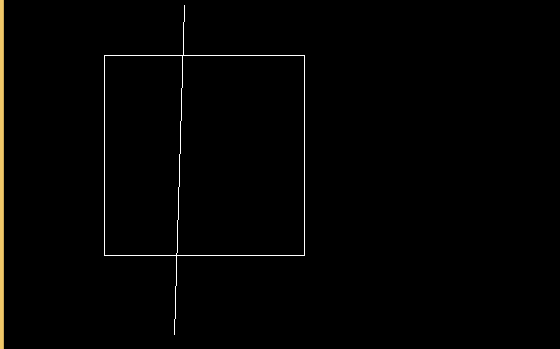
else if(y<y\_min) code |= BOTTOM ; else if(y>y\_max) code |= TOP ;

return code ; }

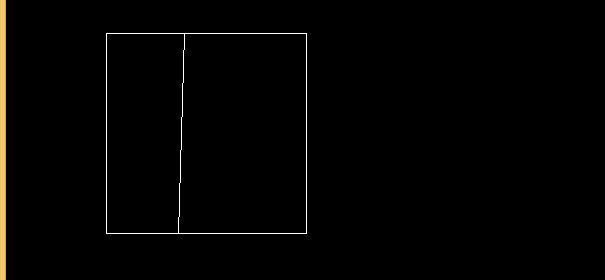
**INPUT GIVEN:-**

**Screenshot (75).png**

**BEFORE CLIPPING:-**

****

**AFTER CLIPPING:-**

****

**Polygon Filling Algorithm**

**Flooding filling:-**

#include<iostream>

#include<graphics.h>

#include<conio.h>

#include<dos.h>

using namespace std;

void flood(int , int , int , int );

int main()

{

int gd = DETECT , gm ;

initgraph(&gd, &gm , "C:\\TC\\BGI");

rectangle(50 , 50 , 200 , 200); getch();

flood(55,150,12,0); getch(); closegraph(); return 0 ; }

void flood(int x , int y , int fill , int old)

{ if( getpixel(x,y) == old)

{ putpixel(x,y,fill);

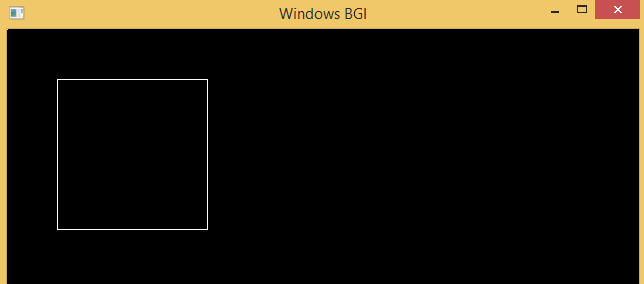
flood(x+1,y, fill, old);

flood(x-1,y, fill , old);

flood(x,y+1,fill , old);

flood(x , y-1 , fill, old); } }

**BEFORE:-**

****

**AFTER:-**

****

Polygon Clipping

Program:-

#include<iostream>

#include<conio.h>

#include<graphics.h>

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

using namespace std ;

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; } }

int main()

{ int gdriver=DETECT,gmode,n,poly[20] , i =0 ;

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

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 :";

for(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:\\TC\\BGI");

setcolor(RED);

rectangle(xmin,ymax,xmax,ymin);

cout<<"\t\tUNCLIPPED POLYGON";

setcolor(WHITE);

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]);

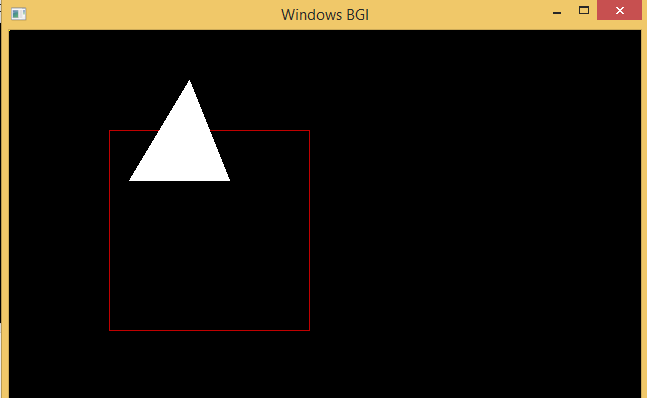
for(i=0;i < k;i++) poly[i]=round(arr[i]);

if(k) fillpoly(k/2,poly); setcolor(RED);

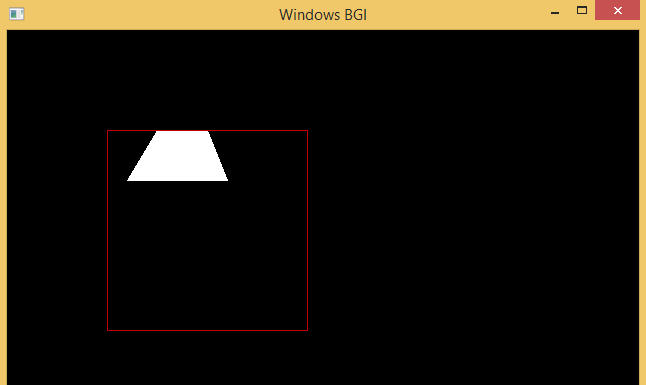
rectangle(xmin,ymax,xmax,ymin);

cout<<"\tCLIPPED POLYGON"; getch(); closegraph(); return 0 ; }

**BEFORE:-**

****

**AFTER:-**

****

**Scan line Polygon Filling**

**Program:-**

#include <conio.h>

#include <graphics.h>

#include<iostream>

using namespace std ;

int main()

{int n,i,j,k,gd,gm,dy,dx;

int x,y,temp;

int a[20][2],intersect[20];

float slope[20];

cout<<"\n\n\tEnter the no. of edges of polygon : " ; cin>>n ;

cout<<"\n\n\tEnter the cordinates of polygon :\n\n\n " ;

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

{cout<<"\tX"<<i<<" Y"<<i;

cin>>a[i][0]>>a[i][1] ; }

a[n][0]=a[0][0]; a[n][1]=a[0][1];

initgraph(&gd,&gm,"c:\\tc\\bgi");

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

{ line(a[i][0],a[i][1],a[i+1][0],a[i+1][1]); } getch() ;

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

{dy=a[i+1][1]-a[i][1];

dx=a[i+1][0]-a[i][0];

if(dy==0) slope[i]=1.0;

if((dy!=0))

{slope[i]=(float) dx/dy;} }

for(y=480;y>0;y--)

{k=0;

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

{if( ((a[i][1]<=y)&&(a[i+1][1]>y))||

((a[i][1]>y)&&(a[i+1][1]<=y)))

{ intersect[k]=(int)(a[i][0]+slope[i]\*(y-a[i][1])); k++; } }

for(j=0;j<k-1;j++)

for(i=0;i<k-1;i++)

{ if(intersect[i]>intersect[i+1])

{temp=intersect[i];

intersect[i]=intersect[i+1];

intersect[i+1]=temp; } }

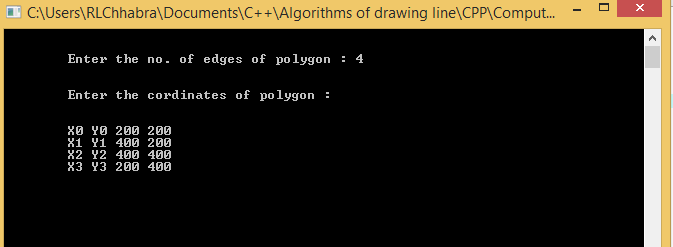
setcolor(WHITE);

for(i=0;i<k;i+=2)

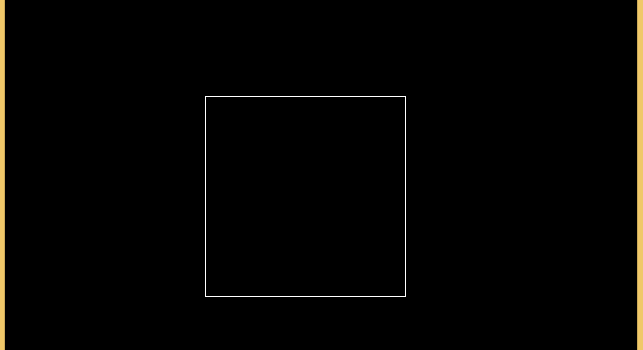
{ line(intersect[i],y,intersect[i+1]+1,y); delay(5); } }

getch(); closegraph(); return 0; }

INPUT GIVEN:-



BEFORE:-



AFTER:-

