include <stdio.h>  
#include <iostream>  
#include <math.h>  
#include <GL/glut.h>  
  
using namespace std;  
  
int sign(int x)  
{  
    if (x>0)  
        return 1;  
    if(x<0)  
        return -1;  
    else  
        return 0;  
}  
  
  
void dda\_algo(float a,float b,float c,float d)  
{  
     int len,i=1;  
     float x0,xt,y0,yt,dx,dy,x,y;  
     //length=200  
     x0=a;y0=b;  
     xt=c;yt=d;  
  
     if(abs(xt-x0)>abs(yt-y0))  
         len=abs(xt-x0);  
     else  
         len=abs(yt-y0);  
  
     dx=(xt-x0)/len;  
     dy=(yt-y0)/len;  
     x=x0+(0.5\*sign(dx));  
     y=y0+(0.5\*sign(dy));  
     glPointSize(3.0f);  
     glBegin(GL\_POINTS);  
     glColor3f(1.0,1.0,1.0);  
     while(i<=len)  
     {  
           glVertex2d(int(x),int(y));  
           x=x+dx;  
           y=y+dy;  
           i++;  
     }  
     glEnd();  
     glFlush();  
}  
  
void dda\_algo1(float a,float b,float c,float d)  
{  
     int len,i=1;  
     float x0,xt,y0,yt,dx,dy,x,y;  
     //length=200  
     x0=a;y0=b;  
     xt=c;yt=d;  
  
     if(abs(xt-x0)>abs(yt-y0))  
         len=abs(xt-x0);  
     else  
         len=abs(yt-y0);  
  
     dx=(xt-x0)/len;  
     dy=(yt-y0)/len;  
     x=x0+(0.5\*sign(dx));  
     y=y0+(0.5\*sign(dy));  
     glPointSize(3.0f);  
     //glColor3f(1.0,1.0,0.0);  
     glBegin(GL\_POINTS);  
     glColor3f(1.0,0.0,0.0);  
     while(i<=len)  
     {  
           glVertex2d(int(x),int(y));  
           x=x+dx;  
           y=y+dy;  
           i++;  
     }  
     glEnd();  
     glFlush();  
}  
  
void Clip(float a,float b,float c,float d,float xmin,float ymin,float xmax,float ymax)  
{  
    float mx,my,px,py,mxd,myd,qx,qy,afinal,bfinal,cfinal,dfinal;  
    px=c;  
    py=d;  
    qx=(a+c)/2;  
    qy=(b+d)/2;  
    afinal=a;  
    bfinal=b;  
    cfinal=c;  
    dfinal=d;  
      
  
    while(true)  
    {  
        mx=(a+c)/2;  
        my=(b+d)/2;  
  
        if(mx == xmin || my == ymin)  
        {  
            dda\_algo1(afinal,bfinal,mx,my);  
            break;  
        }  
  
        else if(mx > xmin && my > ymin && mx < xmax && my < ymax)  
        {  
            c=mx;  
            d=my;  
        }  
        else  
        {  
            a=mx;  
            b=my;  
        }  
    }  
  
    while(true)  
    {  
        mx=(px+qx)/2;  
        my=(py+qy)/2;  
  
        if(mx == xmax || my == ymax)  
        {  
            dda\_algo1(mx,my,cfinal,dfinal);  
            break;  
        }  
  
        else if(mx > xmin && my > ymin && mx < xmax && my < ymax)  
        {  
            qx=mx;  
            qy=my;  
        }  
        else  
        {  
            px=mx;  
            py=my;  
        }  
    }  
  
  
}  
  
void LineClip()  
{  
    dda\_algo(100,100,400,100);  
    dda\_algo(400,100,400,300);  
    dda\_algo(400,300,100,300);  
    dda\_algo(100,300,100,100);  
    dda\_algo(50,150,500,250);  
  
    Clip(50,150,500,250,100,100,400,300);  
  
  
}  
  
void Init()  
{  
    glClearColor(1.0,1.0,1.0,0);  
    //glColor3f(1.0,1.0,1.0); // white  
    glViewport(0 , 0 , 640 , 480);  
    glMatrixMode(GL\_PROJECTION);  
    glLoadIdentity();  
    gluOrtho2D(0 , 640 , 0 , 480);  
}  
  
  
int main(int argc, char \*\*argv)  
{  
    glutInit(&argc,argv);  
    glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  
    glutInitWindowSize(640,480);  
    glutInitWindowPosition(0,0);  
    glutCreateWindow("Line Clipping");  
    Init();  
    float a,b,c,d;  
    glutDisplayFunc(LineClip);  
    glutMainLoop();  
    return 0;  
}

