#include<GL/glut.h>  
#include <math.h>  
#include<stdlib.h>  
#include<stdio.h>  
#define MIN(X,Y) ((X) < (Y) ?  (X) : (Y))  
#define MAX(X,Y) ((X) > (Y) ?  (X) : (Y))  
int mat[1000][1000] ;  
int sign(float x)  
{  
    if(x<0.0)  
        return -1 ;  
    else if (x>0.0)  
        return 1;  
    else  
        return 0 ;  
}  
  
void DDA(float x1,float y1,float x2,float y2,float size)  
{  
   
    float delx,dely ;  
    delx=abs(x2-x1) ;  
    dely=abs(y2-y1) ;  
    float x=0.0,y=0.0 ;  
    int s1=sign(x2-x1) ;  
    int s2=sign(y2-y1) ;  
    x=x1 ;  
    y=y1 ;  
    int interchange=0 ;  
    if(dely>delx)  
    {  
        float temp=delx ;  
        delx=dely ;  
        dely=temp ;  
        interchange=1 ;  
    }  
    int i=1 ;  
    float e =2\*dely-delx ;  
    for(i=1;i<=delx;i++)  
    {  
        
            glPointSize(size) ;  
            glBegin(GL\_POINTS);  
            glColor3f(0.0, 1.0, 0.0);  
        
            glVertex2d((float)(x),(float)(y));  
            glEnd();  
        //glFlush ();  
        while(e>0)  
        {  
            if(interchange==1)  
                x=x+s1 ;  
            else  
                y=y+s2 ;  
            e=e-2\*delx ;  
        }  
        if(interchange==1)  
            y=y+s2 ;  
        else  
            x=x+s1 ;  
        e=e+2\*dely ;  
       // i++ ;  
    }  
  
}  
  
void poly\_color(float x1,float y1,float x2,float y2)  
{  
    float dely=y2-y1 ;  
    float delx=x2-x1 ;  
    float m=dely/delx ;  
    if(delx==0)  
        m=-2 ;  
    float c=y1-m\*x1 ;  
    int max=600 ;  
    float ymin=MIN(y1,y2) ;  
    float ymax=MAX(y1,y2) ;  
    int i,valx,j ;  
    printf("m%f ",m) ;  
    for(i=(int)floor(ymin);i<ymax;i++ )  
    {  
        if(m!=-2)  
        valx=(float)(i-c)/(float)m ;  
        else  
            valx=x1 ;  
        //printf("v%d ",valx) ;   
        //glPointSize(size) ;  
        for(j=(int)valx;j<600;j++)  
        {  
            if(mat[j][i]==0)  
            {  
                glColor3f(0.0, 1.0, 0.0);  
                glBegin(GL\_POINTS);  
                glVertex2d((float)(j),(float)(i));  
                //printf("I%d j%d ",i,j) ;  
                glEnd();  
                mat[j][i]=1 ;  
            }  
            else  
            {  
                glColor3f(1.0, 1.0, 1.0);  
                glBegin(GL\_POINTS);  
                glVertex2d((float)(j),(float)(i));  
                //printf("I%d j%d ",i,j) ;  
                glEnd();  
                mat[j][i]=0 ;  
            }  
        }  
    }  
  
}  
void Primitives(void)  
{  
//clear all pixels  
glClear (GL\_COLOR\_BUFFER\_BIT);  
  
DDA(100,100,600,100,4.0) ;  
DDA(600,100,600,700,4.0) ;  
DDA(600,700,300,400,4.0) ;  
DDA(300,400,100,600,4.0) ;  
DDA(100,600,100,100,4.0) ;  
  
poly\_color(100,100,600,100) ;  
poly\_color(600,100,600,700) ;  
poly\_color(600,700,300,400) ;  
poly\_color(300,400,100,600) ;  
poly\_color(100,600,100,100) ;  
  
glFlush ();  
}  
  
  
  
void Init()  
  
{  
  
glClearColor(1.0,1.0,1.0,0);  
  
    //glColor3f(1.0,0.0,0.0); // red  
  
//glViewport(0 , 0 , 800 , 800);  
glMatrixMode(GL\_PROJECTION);  
  
glLoadIdentity();  
  
gluOrtho2D(0 , 800 , 0 , 800);  
  
}  
  
int main(int argc, char \*\*argv)  
  
{  
  
glutInit(&argc,argv);  
int i,j ;  
for(i=0;i<1000;i++)  
{  
    for(j=0;j<1000;j++)  
    mat[i][j]=0 ;  
}  
  
glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  
  
glutInitWindowSize(800,800);  
  
glutInitWindowPosition(80,80);  
  
glutCreateWindow("Primitives");  
  
Init();  
  
glutDisplayFunc(Primitives);  
  
glutMainLoop();  
  
return 0;  
  
}

