#include <iostream>  
#include "stdafx.h"  
#include<stdlib.h>  
#include<GL/glut.h>  
#include <math.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 boundry[1000][1000] ;  
int vis[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));  
            boundry[(int)x][(int)y]=1 ;  
            mat[(int)x][(int)y]=1 ;  
            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 seed\_fill(float x,float y)  
{  
              
                  
            if(x>300 || x<250)  
                return ;  
            if(y>300 || y<250)  
                return ;  
            if(mat[(int)x][(int)y]==1)  
                return ;  
            glPointSize(4.0f) ;  
            glBegin(GL\_POINTS);  
            glColor3f(0.0, 1.0, 0.0);  
            glVertex2d((x),(y));  
            mat[(int)x][(int)y]=1 ;  
            glEnd() ;  
            glFlush();  
            //printf("%f %f ",x,y);  
          
          
        if(boundry[(int)x+1][(int)y]!=1 && vis[(int)x+1][(int)y]!=1)  
        {  
            seed\_fill(x+1,y) ;  
            vis[(int)x+1][(int)y]=1 ;  
        }  
        if(boundry[(int)x][(int)y+1]!=1 && vis[(int)x][(int)y+1]!=1)  
        {  
            seed\_fill(x,y+1) ;  
            vis[(int)x][(int)y+1]=1 ;  
        }  
        if(boundry[(int)(x-1)][(int)y]!=1 && vis[(int)x-1][(int)y]!=1)  
        {  
            seed\_fill(x-1,y) ;  
            vis[(int)x-1][(int)y]=1 ;  
        }  
        if(boundry[(int)x][(int)y-1]!=1 && vis[(int)x][(int)y-1]!=1)  
        {  
            seed\_fill(x,y-1) ;  
            vis[(int)x][(int)y-1]=1 ;  
        }  
              
}  
void Primitives(void)  
{  
//clear all pixels  
glClear (GL\_COLOR\_BUFFER\_BIT);  
/\*  
glBegin(GL\_LINES);  
glVertex2d(100,100);  
glVertex2d(600,100);  
glVertex2d(600,700);  
glVertex2d(300,400);  
glVertex2d(100,600);  
//glVertex2d(500,400);  
glEnd() ;  
\*/  
// draws a colorful triangle  
//DDA(0,0,70,20) ;  
//DDA(0,0,10,10);  
//DDA(6,3,4,3) ;  
//DDA(4,3,3,5) ;  
//DDA(3,5,4,7) ;  
//DDA(4,7,6,7) ;  
//DDA(6,7,7,5) ;  
//DDA(7,5,6,3) ;  
/\*  
DDA(500,400,450,313,2.0) ;  
DDA(450,313,350,313,2.0) ;  
DDA(350,313,300,400,2.0) ;  
DDA(300,400,350,487,2.0) ;  
DDA(350,487,450,487,2.0) ;  
DDA(450,487,500,400,2.0) ;  
  
DDA(520,400,460,296,4.0);  
  
DDA(460,296,340,296,4.0);  
DDA(340,296,280,400,4.0);  
DDA(280,400,340,504,4.0);  
DDA(340,504,460,504,4.0);  
DDA(460,504,520,400,4.0);  
  
DDA\_dash(550,400,475,270,4.0) ;  
DDA\_dash(475,270,325,270,4.0) ;  
DDA\_dash(325,270,250,400,4.0) ;  
DDA\_dash(250,400,325,530,4.0) ;  
DDA\_dash(325,530,475,530,4.0) ;  
DDA\_dash(475,530,550,400,4.0) ;  
  
DDA\_dot(580,400,490,244,8.0);  
DDA\_dot(490,244,310,244,8.0);  
DDA\_dot(310,244,220,400,8.0);  
DDA\_dot(220,400,310,556,8.0);  
DDA\_dot(310,556,490,556,8.0);  
DDA\_dot(490,556,580,400,8.0);  
\*/  
//circle\_fq(500,500,250);  
//elipse(400,575,25,35);  
//elipse(600,575,25,35);  
//elipse(500,350,50,40);  
/\*  
DDA(100,100,300,100,4.0) ;  
DDA(300,100,300,400,4.0) ;  
DDA(300,400,150,200,4.0) ;  
DDA(150,200,100,250,4.0) ;  
DDA(100,250,100,100,4.0) ;  
vis[120][120]=1 ;  
seed\_fill(120,120) ;  
\*/  
DDA(250,250,300,250,4.0) ;  
DDA(300,250,300,300,4.0) ;  
DDA(300,300,250,300,4.0) ;  
DDA(250,300,250,250,4.0) ;  
vis[260][260]=1 ;  
seed\_fill(260,260) ;  
/\*  
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) ;  
\*/  
//circle\_fq(350,150,100);  
//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]=boundry[i][j]=vis[i][j]=0 ;  
      
}  
  
glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  
  
glutInitWindowSize(800,800);  
  
glutInitWindowPosition(80,80);  
  
glutCreateWindow("Primitives");  
  
Init();  
  
glutDisplayFunc(Primitives);  
  
glutMainLoop();  
  
return 0;  
  
}

