**DDA**

#include <GL/glut.h>  
#include<math.h>  
  
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 length ;  
    if(abs(x2-x1)>=abs(y2-y1))  
    {  
        length=abs(x2-x1) ;  
    }  
    else  
    {  
        length=abs(y2-y1) ;  
    }  
      
    float delx,dely ;  
    delx=(x2-x1)/length ;  
    dely=(y2-y1)/length ;  
    float x=0.0,y=0.0 ;  
    x=x1+0.5\*sign(delx) ;  
    y=y1+0.5\*sign(dely) ;  
    int i=1 ;  
    while(i<=length)  
    {  
        glBegin(GL\_POINTS);  
        glColor3f(0.0, 0.0, 1.0);  
        glPointSize(4.0f) ;  
        glVertex2d((float)floor(x),(float)floor(y));  
        glEnd();  
        //glFlush ();  
        x=x+delx ;  
        y=y+dely ;  
        i++ ;  
    }  
  
}  
void DDA\_dot(float x1,float y1,float x2,float y2)  
{  
    float length ;  
    if(abs(x2-x1)>=abs(y2-y1))  
    {  
        length=abs(x2-x1) ;  
    }  
    else  
    {  
        length=abs(y2-y1) ;  
    }  
      
    float delx,dely ;  
    delx=(x2-x1)/length ;  
    dely=(y2-y1)/length ;  
    float x=0.0,y=0.0 ;  
    x=x1+0.5\*sign(delx) ;  
    y=y1+0.5\*sign(dely) ;  
    int i=1 ;  
    while(i<=length)  
    {  
        glBegin(GL\_POINTS);  
        glColor3f(0.0, 0.0, 1.0);  
        glPointSize(6.0f) ;  
        glVertex2d((float)floor(x),(float)floor(y));  
        glEnd();  
        //glFlush ();  
        x=x+delx ;  
        y=y+dely ;  
        i=i+1 ;  
    }  
  
}  
  
  
  
void Primitives(void)  
{  
//clear all pixels   
glClear (GL\_COLOR\_BUFFER\_BIT);  
  
  
DDA(500,400,450,313) ;  
DDA(450,313,350,313) ;  
DDA(350,313,300,400) ;  
DDA(300,400,350,487) ;  
DDA(350,487,450,487) ;  
DDA(450,487,500,400) ;  
  
DDA(520,400,460,296);  
  
DDA(460,296,340,296);  
DDA(340,296,280,400);  
DDA(280,400,340,504);  
DDA(340,504,460,504);  
DDA(460,504,520,400);  
  
DDA(550,400,475,270) ;  
DDA(475,270,325,270) ;  
DDA(325,270,250,400) ;  
DDA(250,400,325,530) ;  
DDA(325,530,475,530) ;  
DDA(475,530,550,400) ;  
  
DDA\_dot(580,400,490,244);  
DDA\_dot(490,244,310,244);  
DDA\_dot(310,244,220,400);  
DDA\_dot(220,400,310,556);  
DDA\_dot(310,556,490,556);  
DDA\_dot(490,556,580,400);  
  
glFlush ();  
}  
  
  
void Init()  
  
{   
  
glClearColor(1.0,1.0,1.0,0);   
  
glMatrixMode(GL\_PROJECTION);  
  
glLoadIdentity();  
  
gluOrtho2D(0 , 800 , 0 , 800);  
  
}  
  
int main(int argc, char \*\*argv)  
  
{  
  
glutInit(&argc,argv);   
  
glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);   
  
glutInitWindowSize(800,800);   
  
glutInitWindowPosition(80,80);   
  
glutCreateWindow("Primitives");  
  
Init();   
  
glutDisplayFunc(Primitives);   
  
glutMainLoop();   
  
return 0;  
  
}

**Bresenham Algorithm**

#include "stdafx.h"  
#include<GL/glut.h>  
#include <math.h>  
int sign(float x)  
{  
    if(x<0.0)  
        return -1 ;  
    else if (x>0.0)  
        return 1;  
    else  
        return 0 ;  
}  
  
void Bresenham(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, 0.0, 1.0);  
          
            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 Bresenham\_dot(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++)  
    {  
        if(i%11==1)  
        {  
            glPointSize(size) ;  
            glColor3f(1.0, 0.0, 0.0);  
            glBegin(GL\_POINTS);  
            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 Bresenham\_dash(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++)  
    {  
        if(i%29<=29/2)  
        {  
            glPointSize(size) ;  
            glColor3f(0.0, 1.0, 0.0);  
            glBegin(GL\_POINTS);  
            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 Primitives(void)  
{  
//clear all pixels  
glClear (GL\_COLOR\_BUFFER\_BIT);  
  
  
Bresenham(500,400,450,313,2.0) ;  
Bresenham (450,313,350,313,2.0) ;  
Bresenham (350,313,300,400,2.0) ;

Bresenham (300,400,350,487,2.0) ;  
Bresenham (350,487,450,487,2.0) ;  
Bresenham (450,487,500,400,2.0) ;  
  
Bresenham (520,400,460,296,4.0);  
  
Bresenham (460,296,340,296,4.0);  
Bresenham (340,296,280,400,4.0);  
Bresenham (280,400,340,504,4.0);  
Bresenham (340,504,460,504,4.0);  
Bresenham (460,504,520,400,4.0);  
  
Bresenham \_dash(550,400,475,270,4.0) ;  
Bresenham \_dash(475,270,325,270,4.0) ;  
Bresenham \_dash(325,270,250,400,4.0) ;  
Bresenham \_dash(250,400,325,530,4.0) ;  
Bresenham \_dash(325,530,475,530,4.0) ;  
Bresenham \_dash(475,530,550,400,4.0) ;  
  
Bresenham \_dot(580,400,490,244,8.0);  
Bresenham \_dot(490,244,310,244,8.0);  
Bresenham \_dot(310,244,220,400,8.0);  
Bresenham \_dot(220,400,310,556,8.0);  
Bresenham \_dot(310,556,490,556,8.0);  
Bresenham \_dot(490,556,580,400,8.0);  
  
glFlush ();  
}  
  
  
void Init()  
  
{  
  
glClearColor(1.0,1.0,1.0,0);  
glMatrixMode(GL\_PROJECTION);  
glLoadIdentity();  
gluOrtho2D(0 , 800 , 0 , 800);  
  
}  
  
int main(int argc, char \*\*argv)  
  
{  
  
glutInit(&argc,argv);  
glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  
glutInitWindowSize(800,800);  
glutInitWindowPosition(80,80);  
glutCreateWindow("Primitives");  
Init();  
glutDisplayFunc(Primitives);  
glutMainLoop();  
return 0;  
  
}

