#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<GL/gl.h>

#include<GL/glu.h>

#include<GL/glut.h>

int count=0;

float mat[3][20]={{350.0,350.0,380.0,380.0,410.0,410.0,440.0,440.0,470.0,470.0,350.0,470.0,350.0,470.0,350.0,470.0,350.0,470.0,350.0,470.0},

{350.0,470.0,350.0,470.0,350.0,470.0,350.0,470.0,350.0,470.0,350.0,350.0,380.0,380.0,410.0,410.0,440.0,440.0,470.0,470.0},

{1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,}},trans[3][3],res[3][20];

float x;

typedef struct pixel

{

GLubyte red,green,blue;

}pixel;

pixel c, boundary,fill;

void Init()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glClearColor(1,1,1,0);

//glColor3f(1,1,0);

glClear(GL\_COLOR\_BUFFER\_BIT );

glColor3ub(0,0,0);

gluOrtho2D(0 , 800 ,0 , 600);

}

void mult()

{

int i,j,k;

for(i=0;i<3;i++)

{

for(j=0;j<20;j++)

{

res[i][j]=0;

for(k=0;k<3;k++)

{

res[i][j]+=trans[i][k]\*mat[k][j];

}

}

}

}

int sign(int a)

{

if(a>=0)

return 1;

else

return -1;

}

void breshnam(int X1,int Y1,int X2,int Y2)

{

float x,y,dx,dy,length;

int i;

dx=abs(X2-X1);

dy=abs(Y2-Y1);

if(dx>=dy)

length=dx;

else

length=dy;

dx=(X2-X1)/length;

dy=(Y2-Y1)/length;

x=X1 + 0.5\*sign(X1);

y=Y1 + 0.5\*sign(Y1);

i=1;

while(i<=length)

{

glBegin (GL\_POINTS);

glVertex2i (x, y);

glEnd ();

glFlush ();

x=x+dx;

y=y+dy;

i=i+1;

}

}

void initmat()

{

trans[0][0]=cos(x);

trans[0][1]=sin(x);

trans[1][0]=-sin(x);

trans[1][1]=cos(x);

trans[2][1]=trans[2][0]=trans[0][2]=trans[1][2]=0;

trans[2][2]=1;

}

void drawcb(float mat[3][20])

{

int i;

for(i=0;i<20;i=i+2)

{

breshnam(mat[0][i],mat[1][i],mat[0][i+1],mat[1][i+1]);

}

}

void boundary\_fill(int x,int y,pixel b,pixel f)

{

glReadPixels(x,y,1,1,GL\_RGB,GL\_UNSIGNED\_BYTE,&c);

if(c.red!=b.red && c.green!=b.green && c.blue!=b.blue && c.red!=f.red && c.green!=f.green && c.blue!=f.blue)

{

glColor3ub(f.red,f.green,f.blue);

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

boundary\_fill(x+1,y,b,f);

boundary\_fill(x-1,y,b,f);

boundary\_fill(x,y+1,b,f);

boundary\_fill(x,y-1,b,f);

glFlush();

}

}

void mouse(int btn,int state,int x,int y)

{

int ymax=glutGet(GLUT\_WINDOW\_HEIGHT);

if(btn==GLUT\_LEFT\_BUTTON && state==GLUT\_DOWN)

{

boundary\_fill(x,ymax-y,boundary,fill);

}

}

void draw()

{

initmat();

//drawcb(mat);

mult();

drawcb(res);

}

int main(int argc,char \*\* argv)

{

int p;

fill.red = 0.0;

fill.green = 0.0;

fill.blue = 0.0;

boundary.red = 100;

boundary.green = 100;

boundary.blue = 100;

printf("ENTER THE ANGLE FOR ROTATION : ");

scanf("%d",&p);

x=(p\*3.14)/180;

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowPosition(0,0);

glutInitWindowSize(800,600);

glutCreateWindow("chess board");

Init();

glutDisplayFunc(draw);

glutMouseFunc(mouse);

glutMainLoop();

return 0;

}