#include<GL/freeglut.h>

#include<GL/gl.h>

#include<math.h>

#include<stdio.h>

#include<iostream>

int mat1[8][3];

float trans[3][3];

int ans[8][3];

void initialco() {

int tempMat1[8][3] = {

{150, 150, 1},

{400, 150, 1},

{150, 300, 1},

{400, 300, 1},

{150, 150, 1},

{150, 300, 1},

{400, 150, 1},

{400, 300, 1}

};

for (int i = 0; i < 8; i++) {

for (int j = 0; j < 3; j++) {

mat1[i][j] = tempMat1[i][j];

}

}

}

void getPixel(int x, int y, float interiorColor[3]){

glReadPixels(x,y,1,1,GL\_RGB, GL\_FLOAT, interiorColor);

}

void setPixel(int x, int y, float fillColor[3]){

glBegin(GL\_POINTS);

glColor3fv(fillColor);

glVertex2i(x,y);

glEnd();

glFlush();

}

void boundaryfill(int x, int y, float fillColor[3], float boundaryColor[3]){

float interiorColor[3];

getPixel(x,y,interiorColor);

if((interiorColor[0]!=boundaryColor[0] || interiorColor[1]!=boundaryColor[1] || interiorColor[2]!=boundaryColor[2])&&(interiorColor[0]!=fillColor[0] || interiorColor[1]!=fillColor[1] || interiorColor[2]!=fillColor[2])){

setPixel(x,y,fillColor);

boundaryfill(x + 1,y,fillColor, boundaryColor);

boundaryfill(x - 1, y, fillColor, boundaryColor);

boundaryfill(x, y + 1, fillColor, boundaryColor);

boundaryfill(x, y - 1, fillColor, boundaryColor);

}

}

void before(){

initialco();

glColor3f(0,0,0);

glBegin(GL\_LINES);

for(int i=0; i<8; i+=2){

glVertex2i(mat1[i][0], mat1[i][1]);

glVertex2i(mat1[i+1][0], mat1[i+1][1]);

}

glEnd();

glFlush();

}

void afterrotation(int theta){

initialco();

float thetarad = theta\*3.14/180;

/////translation//////

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

for(int j=0; j<3;j++){

if(i==j){

trans[i][j]=1;

}else{

trans[i][j]=0;

}

}

}

trans[2][0]=trans[2][1]=-150;

for(int i=0;i<8;i++){

for(int j=0; j<3; j++){

ans[i][j]=0;

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

ans[i][j]+=mat1[i][k]\*trans[k][j];

}

}

}

/////rotation ////

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

for(int j=0; j<3;j++){

if(i==j){

trans[i][j]=1;

}else{

trans[i][j]=0;

}

}

}

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

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

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

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

for(int i=0;i<8;i++){

for(int j=0; j<3; j++){

mat1[i][j]=0;

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

mat1[i][j]+=ans[i][k]\*trans[k][j];

}

}

}

////translation back

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

for(int j=0; j<3;j++){

if(i==j){

trans[i][j]=1;

}else{

trans[i][j]=0;

}

}

}

trans[2][0]=trans[2][1]=150;

for(int i=0;i<8;i++){

for(int j=0; j<3; j++){

ans[i][j]=0;

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

ans[i][j]+=mat1[i][k]\*trans[k][j];

}

}

}

glColor3f(0,0,0);

glBegin(GL\_LINES);

for(int i=0; i<8; i+=2){

glVertex2i(ans[i][0], ans[i][1]);

glVertex2i(ans[i+1][0], ans[i+1][1]);

}

glEnd();

glFlush();

}

void fillcolor(){

afterrotation(10);

float fillColor[3] = {1, 0, 0}; // Fill color: Red

float boundaryColor[3] = {0, 0, 0}; // Boundary color: Black

boundaryfill(175, 175, fillColor, boundaryColor);

}

void renderfunction(){

glClearColor(1,1,1,0);

glClear(GL\_COLOR\_BUFFER\_BIT);

gluOrtho2D(0,700,0,500);

glColor3f(0,0,0);

int choice=3;

if(choice==1){

before();

}else if(choice==2){

afterrotation(30);

}else if(choice==3){

fillcolor();

}

glFlush();

}

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

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(700,500);

glutInitWindowPosition(100,100);

glutCreateWindow("hey there");

glutDisplayFunc(renderfunction);

glutMainLoop();

return 0;

}