#include<gl/glu.h>

#include<gl/glut.h>

struct point{

float x, y;

}w[4],oVer[4];

int Nout;

void drawPoly(point p[], int n) {

glBegin(GL\_POLYGON);

for(int i=0 ; i<n ; i++)

glVertex2f(p[i].x, p[i].y);

glEnd();

}

bool insideVer(point p) {

if((p.x >= w[0].x) && (p.x <= w[2].x))

if((p.y >= w[0].y) && (p.y <= w[2].y))

return true;

return false;

}

void addVer(point p) {

oVer[Nout] = p;

Nout = Nout + 1;

}

point getIntersect(point s, point p, int edge) {

point in;

float m;

if(w[edge].x == w[(edge+1)%4].x) {

m = (p.y - s.y) / (p.x - s.x);

in.x = w[edge].x;

in.y = in.x \* m + s.y;

} else {

m = (p.y - s.y) / (p.x - s.x);

in.y = w[edge].y;

in.x = (in.y - s.y) / m;

}

return in;

}

void clipAndDraw(point inVer[], int Nin) {

point s, p, intersec;

for(int i=0 ; i<4 ; i++) {

Nout = 0;

s = inVer[Nin-1];

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

p = inVer[j];

if(insideVer(p)==true){

if(insideVer(s)==true) {

addVer(p);

} else {

intersec = getIntersect(s,p,i);

addVer(intersec);

addVer(p);

}

} else {

if(insideVer(s) == true) {

intersec = getIntersect(s, p, i);

addVer(intersec);

}

}

s = p;

}

inVer = oVer;

Nin = Nout;

}

drawPoly(oVer, 4);

}

void init() {

glClearColor(0.0f, 0.0f, 0.0f, 0.0f);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(0.0, 100.0, 0.0, 100.0, 0.0, 100.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

w[0].x = 20, w[0].y = 10;

w[1].x = 20, w[1].y = 80;

w[2].x = 80, w[2].y = 80;

w[3].x = 80, w[3].y = 10;

}

void display(void) {

point inVer[4];

init();

glColor3f(1.0f, 1.0f, 0.0f);

drawPoly(w, 4);

glColor3f(0.0f, 1.0f, 0.0f);

inVer[0].x = 10, inVer[0].y = 40;

inVer[1].x = 10, inVer[1].y = 60;

inVer[2].x = 60, inVer[2].y = 60;

inVer[3].x = 60, inVer[3].y = 40;

drawPoly(inVer, 4);

glColor3f(0.0f, 0.0f, 1.0f);

clipAndDraw(inVer, 4);

glFlush();

}

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

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(400, 400);

glutInitWindowPosition(100, 100);

glutCreateWindow("Polygon Clipping");

glutDisplayFunc(display);

glutMainLoop();

return 0;

}