**BÀI TẬP LAB 6**

I.Tên thành viên nhóm:

-Phan Văn Hà

-Hồ Nhật Huy

-Đỗ Quang Pháp

-Nguyễn Cảnh Thông

II.Bài làm

1. **LAP 5:**
2. **Code:**

#code

#include <dos.h>

#include <math.h>

#include <stdio.h>

#include <conio.h>

#include <iostream>

#include <graphics.h>

using namespace std;

// Khai bao bien toan cuc

int phepchieu; // = 1 : phoi canh; = 0 : song song

float r = 50, phi = 10,teta = 20,D = 20, tlx = 100, tly = 100;

int xo = 300, yo = 200; // (xo, yo) vi tri cai dat goc toa do tren man hinh

// Xay dung bo cong cu 3D

void chuyenHQS(float x, float y, float z, float &x1, float &y1, float &z1)

{

x1 = - x \* sin(teta) + y \* cos(teta);

y1 = - x \* cos(teta) \* sin(phi) - y \* sin(teta) \* sin(phi) + z \* cos(phi);

z1 = - x \* sin(teta) \* cos(phi) - y \* cos(teta) \* sin(phi) - z \* sin(phi) + r;

}

void chieu3D\_2D(float x, float y, float z, float &xp, float &yp)

{

if (phepchieu == 1)

{

xp = D \* x / z;

yp = D \* y / z;

}

else

{

xp = x;

yp = y;

}

}

void chuyenMH(float x, float y, int &xm, int &ym)

{

xm = (int) (tlx \* x + xo);

ym = (int) (tly \* y + yo);

}

void chuyenDen(float x, float y, float z)

{

float x1, y1, z1, xp, yp;

int xm, ym;

chuyenHQS(x, y, z, x1, y1, z1);

chieu3D\_2D(x1, y1, z1, xp, yp);

chuyenMH(xp, yp, xm, ym);

moveto(xm, ym);

}

void veDen(float x, float y, float z)

{

float x1, y1, z1, xp, yp;

int xm, ym;

chuyenHQS(x, y, z, x1, y1, z1);

chieu3D\_2D(x1, y1, z1, xp, yp);

chuyenMH(xp, yp, xm, ym);

lineto(xm, ym);

}

// Ve truc toa do

void veTruc()

{

// In ky tu

chuyenDen(-0.2, 0, 0.1);

outtext("O");

chuyenDen(4, 0, 0.3);

outtext("x");

chuyenDen(-0.1, 3, 0.3);

outtext("y");

chuyenDen(-0.2, 0, 2.1);

outtext("z");

// Ve truc

// Ox

chuyenDen(0, 0, 0);

veDen(4, 0, 0);

// Oy

chuyenDen(0, 0, 0);

veDen(0, 3, 0);

// Oz

chuyenDen(0, 0, 0);

veDen(0, 0, 2);

}

// Ve lap phuong

void veLapPhuong(float x, float y, float z, float d)

{

chuyenDen(x, y, z); // > A

veDen(x + d, y, z); // > B

veDen(x + d, y + d, z); // > C

veDen(x, y + d, z); // > D

veDen(x, y, z); // > A

veDen(x, y, z + d); // > E

veDen(x + d, y, z + d); // > F

veDen(x + d, y + d, z + d); // > G

veDen(x, y + d, z + d); // > H

veDen(x, y, z + d); // > E

chuyenDen(x + d, y, z); // > B

veDen(x + d, y, z + d); // > F

chuyenDen(x + d, y + d, z); // > C

veDen(x + d, y + d, z + d); // > G

chuyenDen(x, y + d, z); // > D

veDen(x, y + d, z + d); // > F

}

void biendoitile(float x, float y, float z, float d)

{

chuyenDen(x\*2, y\*2, z\*2); // > A

veDen((x + d)\*2, y\*2, z); // > B

veDen((x + d)\*2, (y + d)\*2, z\*2); // > C

veDen(x\*2, (y + d)\*2, z\*2); // > D

veDen(x\*2, y\*2, z\*2); // > A

veDen(x\*2, y\*2, (z + d)\*2); // > E

veDen((x + d)\*2, y\*2, (z + d)\*2); // > F

veDen((x + d)\*2, (y + d)\*2, (z + d)\*2); // > G

veDen(x\*2, (y + d)\*2, (z + d)\*2); // > H

veDen(x\*2, y\*2, (z + d)\*2); // > E

chuyenDen((x + d)\*2, y\*2, z\*2); // > B

veDen((x + d)\*2, y\*2, (z + d)\*2); // > F

chuyenDen((x + d)\*2, (y + d)\*2, z\*2); // > C

veDen((x + d)\*2, (y + d)\*2, (z + d)\*2); // > G

chuyenDen(x\*2, (y + d)\*2, z\*2); // > D

veDen(x\*2, (y + d)\*2, (z + d)\*2); // > F

}

void tinhtienLapPhuong(float x,float y,float z,float d)

{

setcolor(BLACK);

setcolor(WHITE);

veTruc();

setcolor(4);

chuyenDen(x+0.5, y+0.5, z+0.5); // > A

veDen(x + d+0.5, y+0.5, z+0.5); // > B

veDen(x + d+0.5, y + d+0.5, z+0.5); // > C

veDen(x+0.5, y + d+0.5, z+0.5); // > D

veDen(x+0.5, y+0.5, z+0.5); // > A

veDen(x+0.5, y+0.5, z + d+0.5); // > E

veDen(x + d+0.5, y+0.5, z + d+0.5); // > F

veDen(x + d+0.5, y + d+0.5, z + d+0.5); // > G

veDen(x+0.5, y + d+0.5, z + d+0.5); // > H

veDen(x+0.5, y+0.5, z + d+0.5); // > E

chuyenDen(x + d+0.5, y+0.5, z+0.5); // > B

veDen(x + d+0.5, y+0.5, z + d+0.5); // > F

chuyenDen(x + d+0.5, y + d+0.5, z+0.5); // > C

veDen(x + d+0.5, y + d+0.5, z + d+0.5); // > G

chuyenDen(x+0.5, y + d+0.5, z+0.5); // > D

veDen(x+0.5, y + d+0.5, z + d+0.5);

}

void xoayquanhtrucOz(float x,float y,float z,float d){

setcolor(BLACK);

float goc;

cout<<"nhap goc quay";cin>>goc;

float a=(goc\*3.14152)/180;

setcolor(WHITE);

veTruc();

setcolor(BLUE);

chuyenDen(x\*cos(a) - y\*sin(a), x\*sin(a) + y\*cos(a), z); // > A

veDen((x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) + y\*cos(a), z); // > B

veDen((x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) + (y+d)\*cos(a), z); // > C

veDen(x\*cos(a) - (y + d)\*sin(a),x\*sin(a) + (y+ d)\*cos(a), z); // > D

veDen(x\*cos(a) - y\*sin(a), x\*sin(a) + y\*cos(a), z); // > A

veDen(x\*cos(a) - y\*sin(a), x\*sin(a) + y\*cos(a), z + d); // > E

veDen((x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) + y\*cos(a), z + d); // > F

veDen((x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) + (y+d)\*cos(a), z + d); // >

veDen(x\*cos(a) - (y + d)\*sin(a),x\*sin(a) + (y+ d)\*cos(a), z + d); // > H

veDen(x\*cos(a) - y\*sin(a), x\*sin(a) + y\*cos(a), z + d); // > E

chuyenDen((x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) + y\*cos(a), z); // > B

veDen((x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) + y\*cos(a), z + d); // > F

chuyenDen((x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) + (y+d)\*cos(a), z); // >

veDen((x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) + (y+d)\*cos(a), z + d); // >

chuyenDen(x\*cos(a) - (y + d)\*sin(a),x\*sin(a) + (y+ d)\*cos(a), z); // > D

veDen(x\*cos(a) - (y + d)\*sin(a),x\*sin(a) + (y+ d)\*cos(a), z + d); // > F

}

void xoayquanhtrucOy(float x, float y, float z, float d)

{

setcolor(BLACK);

float goc;

cout<<"nhap goc quay";cin>>goc;

float a=(goc\*3.14152)/180;

setcolor(WHITE);

veTruc();

setcolor(BLUE);

chuyenDen(x\*cos(a) + x\*sin(a), y , x\*cos(a) - x\*cos(a)); // > A

veDen((x+d)\*cos(a) + x\*sin(a), y , (x+d)\*cos(a) - x\*cos(a)); // > B

veDen((x+d)\*cos(a) + (x+d)\*sin(a), y , (x+d)\*cos(a) - (x+d)\*cos(a)); // > C

veDen(x\*cos(a) + (x+d)\*sin(a), y , x\*cos(a) - (x+d)\*cos(a)); // > D

veDen(x\*cos(a) + x\*sin(a), y , x\*cos(a) - x\*cos(a)); // > A

veDen(x\*cos(a) + x\*sin(a), y + d , x\*cos(a) - x\*cos(a)); // > E

veDen((x+d)\*cos(a) + x\*sin(a), y + d , (x+d)\*cos(a) - x\*cos(a)); // > F

veDen((x+d)\*cos(a) + (x+d)\*sin(a), y + d , (x+d)\*cos(a) - (x+d)\*cos(a)); // >

veDen(x\*cos(a) + (x+d)\*sin(a), y + d , x\*cos(a) - (x+d)\*cos(a)); // > H

veDen(x\*cos(a) + x\*sin(a), y + d , x\*cos(a) - x\*cos(a)); // > E

chuyenDen((x+d)\*cos(a) + x\*sin(a), y , (x+d)\*cos(a) - x\*cos(a)); // > B

veDen((x+d)\*cos(a) + x\*sin(a), y + d , (x+d)\*cos(a) - x\*cos(a)); // > F

chuyenDen((x+d)\*cos(a) + (x+d)\*sin(a), y , (x+d)\*cos(a) - (x+d)\*cos(a)); // >

veDen((x+d)\*cos(a) + (x+d)\*sin(a), y + d , (x+d)\*cos(a) - (x+d)\*cos(a)); // >

chuyenDen(x\*cos(a) + (x+d)\*sin(a), y , x\*cos(a) - (x+d)\*cos(a)); // > D

veDen(x\*cos(a) + (x+d)\*sin(a), y + d , x\*cos(a) - (x+d)\*cos(a)); // > F

}

void xoayquanhtrucOx(float x, float y , float z, float d)

{

setcolor(BLACK);

float goc;

cout<<"nhap goc quay";cin>>goc;

float a=(goc\*3.14152)/180;

setcolor(WHITE);

veTruc();

setcolor(BLUE);

chuyenDen(x ,x\*cos(a) - y\*sin(a), x\*sin(a) - y\*cos(a)); // > A

veDen(x ,(x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) - y\*cos(a)); // > B

veDen(x ,(x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) - (y+d)\*cos(a)); // > C

veDen(x ,x\*cos(a) - (y + d)\*sin(a),x\*sin(a) - (y+ d)\*cos(a)); // > D

veDen(x ,x\*cos(a) - y\*sin(a), x\*sin(a) - y\*cos(a)); // > A

veDen(x + d,x\*cos(a) - y\*sin(a), x\*sin(a) - y\*cos(a)); // > E

veDen(x + d ,(x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) - y\*cos(a)); // > F

veDen(x + d , (x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) - (y+d)\*cos(a)); // >

veDen(x + d , x\*cos(a) - (y + d)\*sin(a),x\*sin(a) - (y+ d)\*cos(a)); // > H

veDen(x + d, x\*cos(a) - y\*sin(a), x\*sin(a) - y\*cos(a)); // > E

chuyenDen(x , (x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) - y\*cos(a)); // > B

veDen(x + d, (x + d)\*cos(a) - y\*sin(a),(x + d)\*sin(a) - y\*cos(a)); // > F

chuyenDen(x, (x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) - (y+d)\*cos(a)); // >

veDen(x + d, (x + d)\*cos(a) - (y+d)\*sin(a) , (x + d)\*sin(a) - (y+d)\*cos(a)); // >

chuyenDen(x, x\*cos(a) - (y + d)\*sin(a),x\*sin(a) - (y+ d)\*cos(a)); // > D

veDen(x + d, x\*cos(a) - (y + d)\*sin(a),x\*sin(a) - (y+ d)\*cos(a)); // > F

}

void doixungLapPhuong(float x, float y, float z, float d)

{

chuyenDen(x, -y, z); // > A

veDen(x + d, -y, z); // > B

veDen(x + d, -(y + d),z); // > C

veDen(x, -(y + d), z); // > D

veDen(x, -y, z); // > A

veDen(x, -y, (z + d)); // > E

veDen(x + d, -y, (z + d)); // > F

veDen(x + d, -(y + d), (z + d)); // > G

veDen(x,-(y + d), (z + d)); // > H

veDen(x, -y, (z + d)); // > E

chuyenDen(x + d, -y, z); // > B

veDen(x + d, -y, (z + d)); // > F

chuyenDen(x + d, -(y + d), z); // > C

veDen(x + d, -(y + d), (z + d)); // > G

chuyenDen(x, -(y + d), z); // > D

veDen(x, -(y + d), (z + d)); // > F

}

void chontrucquay()

{

setcolor(BLACK);

float truc;

cout<<"chon truc quay: 1 -> Ox || 2 -> Oy || 3 -> Oz";

cout<<"Nhap vao: ";cin>>truc;

if(truc == 1)

{

xoayquanhtrucOx(1.5,1.5,0,1);

}else if(truc == 2)

{

xoayquanhtrucOy(1.5,1.5,0,1);

}else if(truc == 3)

{

xoayquanhtrucOz(1.5,1.5,0,1);

}

}

int main()

{

int driver = DETECT, mode = 0;

initgraph(&driver, &mode,"C:\\Dev-Cpp\\include");

// Ve truc toa do

setcolor(WHITE);

veTruc();

// Ve lap phuong

setcolor(WHITE);

veLapPhuong(0.5, 0.5, 0, 1);

float k;

cout<<"\nnhap yeu cau: ";

cout<<"\n 1 -> bien doi ti le: ";

cout<<"\n 2 -> quay quanh truc: ";

cout<<"\n 3 -> tinh tien: ";

cout<<"\n 4 -> doi xung: ";

cout<<"\nnhap: ";cin>>k;

if(k == 1)

{

biendoitile(1.5,1.5,0,1);

}else if(k == 2)

{

chontrucquay();

}else if(k == 3)

{

tinhtienLapPhuong(1.5,1.5,0,1);

}else if(k == 4)

{

doixungLapPhuong(1.5,1.5,0,1);

}else{

cout<<"\nmoi nhap lai! ";

}

getch();

return 0;

}

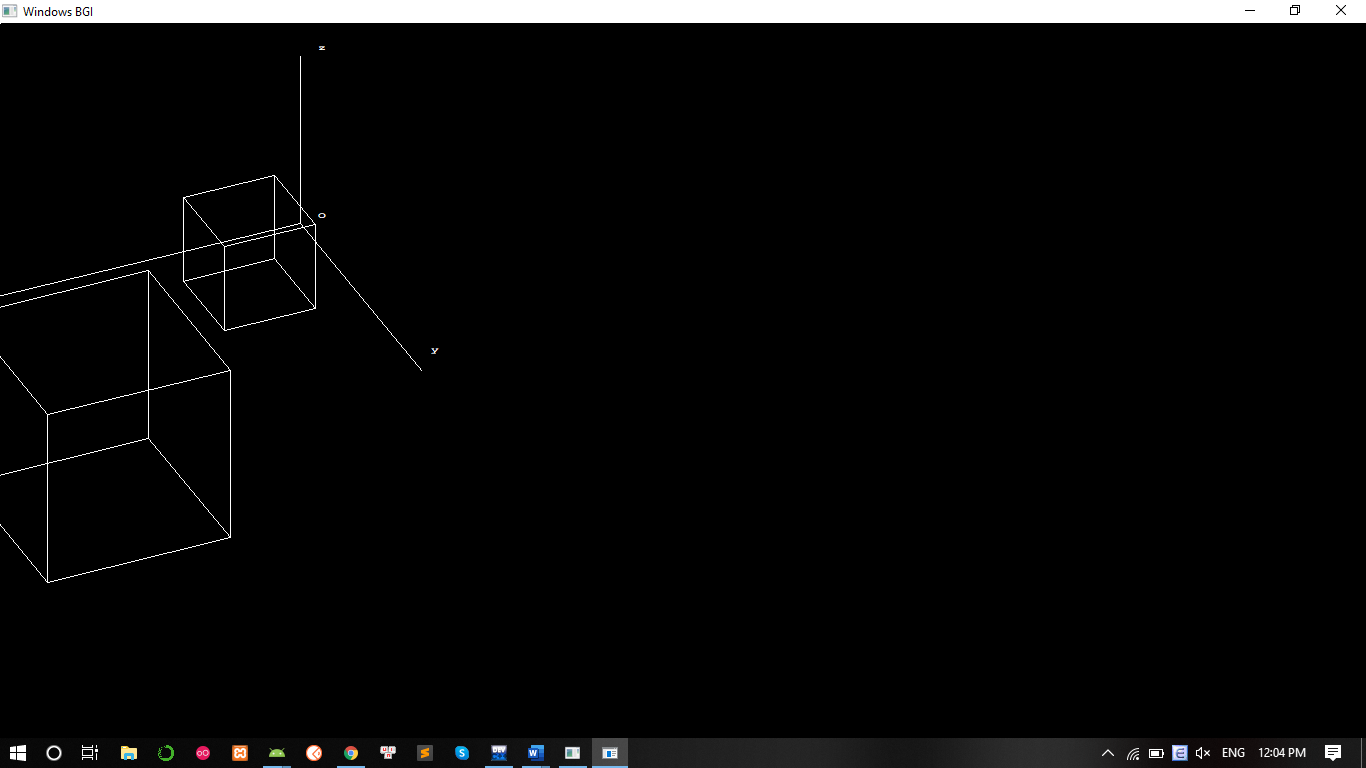
#code

1. **Dữ liệu thử:**

* Vehinhlapphuong: x = 0.5, y = 0.5, z = 0, d = 1;
* biendoitile: x = 1.5, y = 1.5, z = 0, d = 1;
* quayquanhtruc: x = 1.5, y = 1.5, z = 0, d = 1; goc quay a = 30 độ;
* tinhtien: x = 0.5, y = 0.5, z = 0, d = 1;
* doixung: x = 0.5, y = 0.5, z = 0, d = 1;

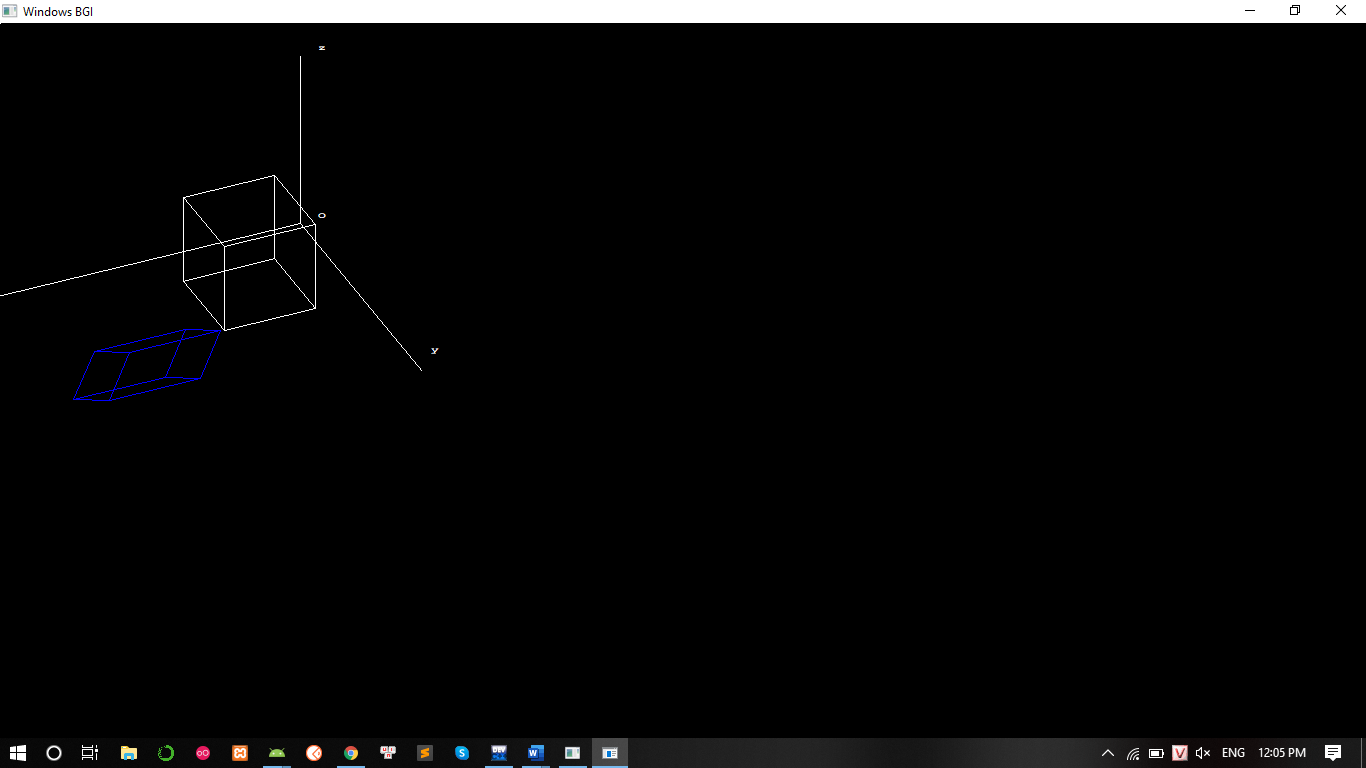
1. **Màn Hình kết quả:**

* Biến dổi tỉ lệ:

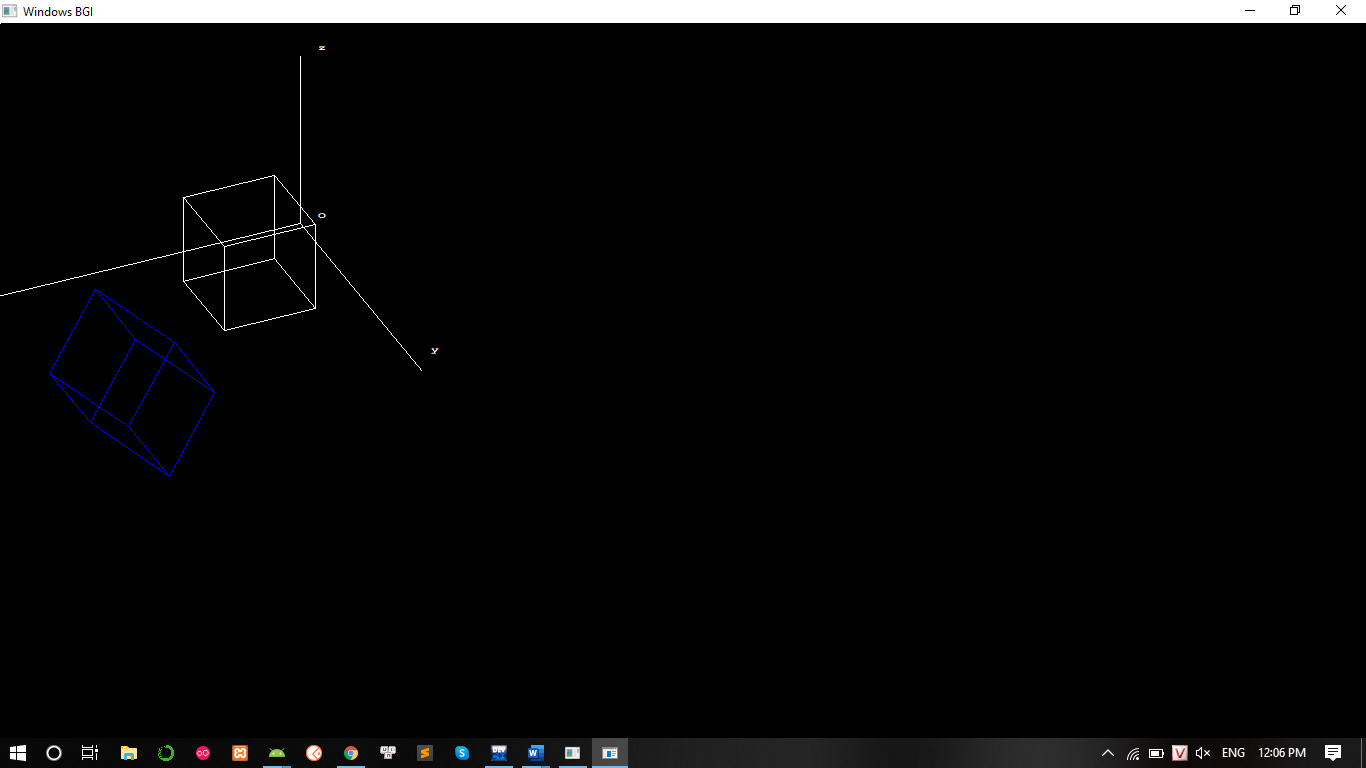


* Quay quanh trục:

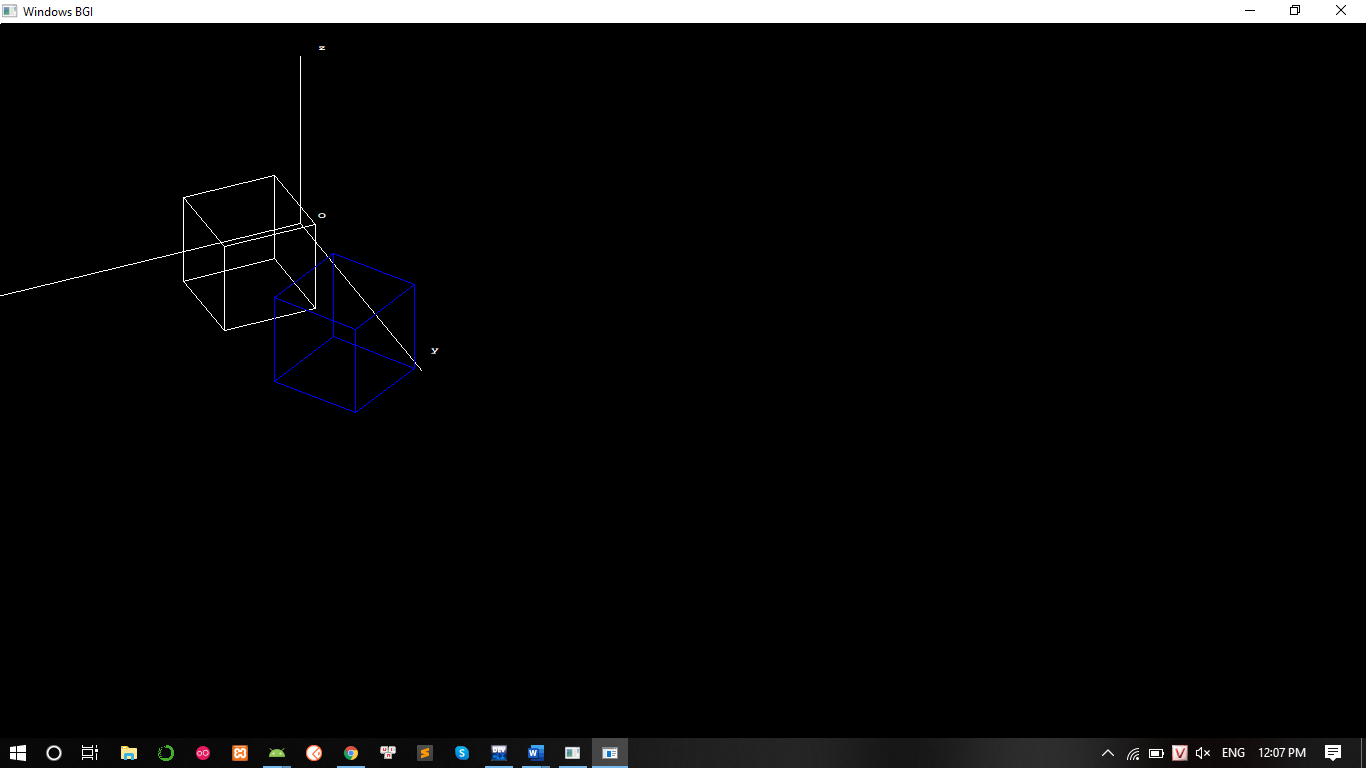
+ Ox:



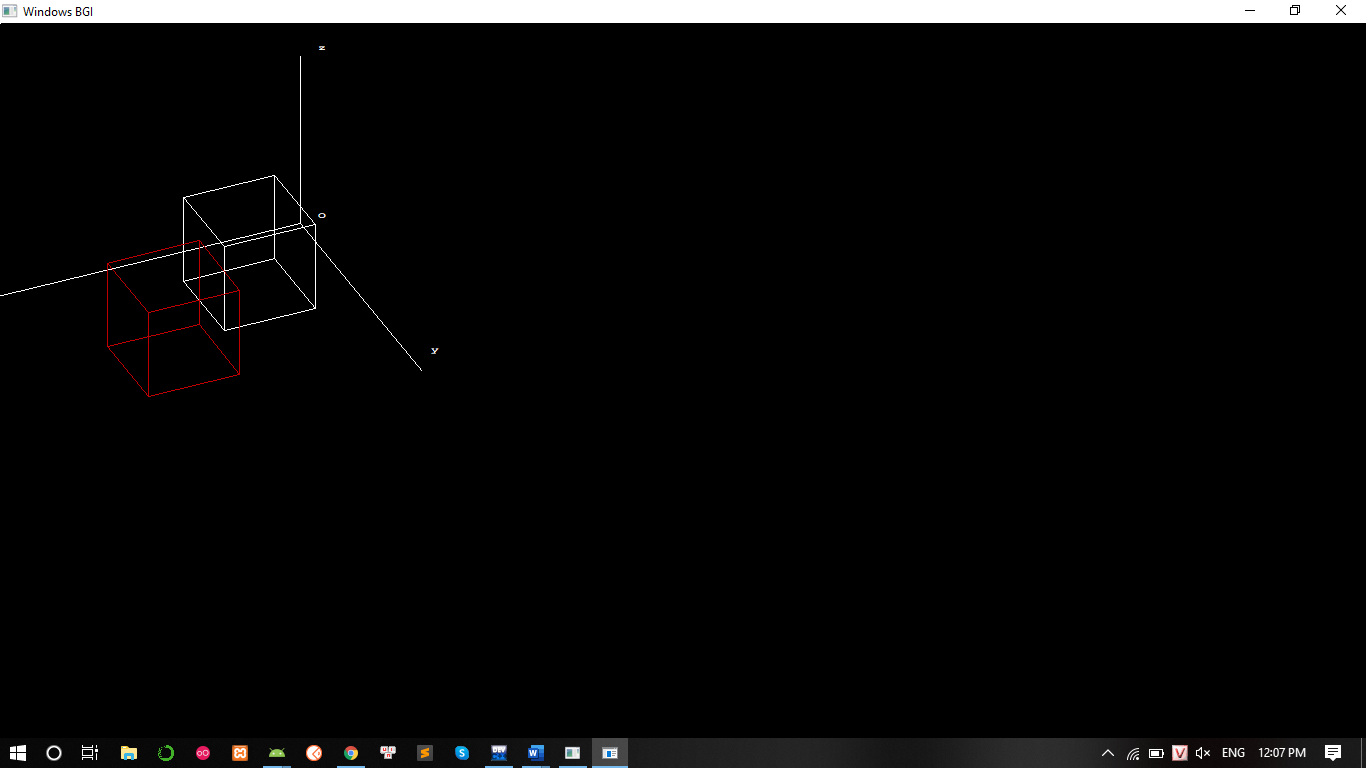
+ Oy:



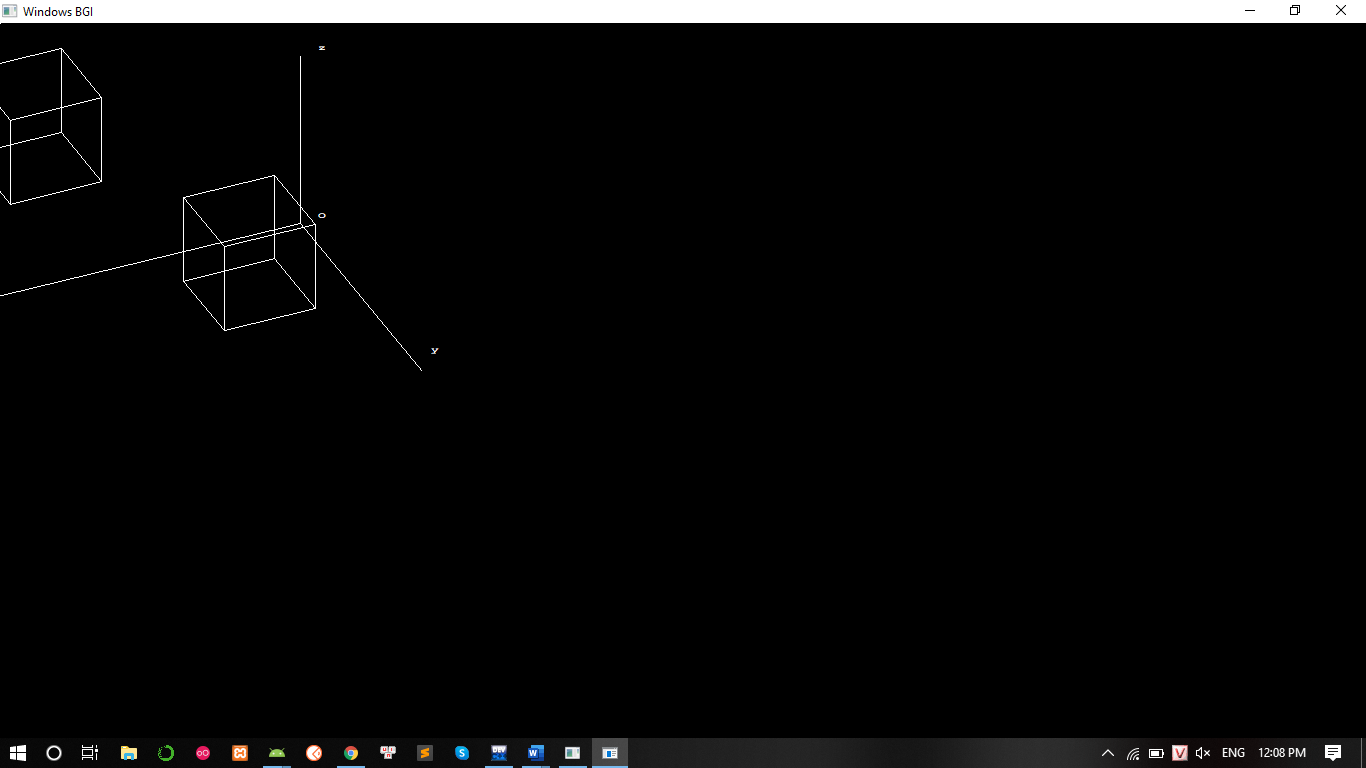
+ Oz:



* Tinh tien:



* Đối xứng:



1. **LAP 6:**

**1.Vật thể 3D theo mô hình khung kết nối**

- Thông số ban đầu:

(int [5][3],int [8][2],int [5][5])

- Mã nguồn:

# include <iostream>

# include <graphics.h>

# include <conio.h>

# include <math.h>

# define Xc -0.3

# define Yc 0.3

# define Zc -1.0

void show\_screen( );

void draw\_3d\_polygon(int [5][3],int [8][2],int [5][5]);

void get\_projected\_point(int&,int&,int&);

void Line(const int,const int,const int,const int);

int main( )

{

int driver=VGA;

int mode=VGAHI;

initgraph(&driver,&mode,"..\\Bgi");

show\_screen( );

int vertices[5][3]={

{245,325,75}, // base front left

{395,325,75}, // base front right

{395,325,-75}, // base back right

{245,325,-75}, // base back left

{320,125,0} // top

};

int edges[8][2]={

// first is starting point vertex number

// second is ending point vertex number

{0,3} , {3,2} , {2,1} , {1,0} , {0,4} ,

{3,4} , {2,4} , {1,4}

};

int surfaces[5][5]={

// First entry is number of edges

// Rest entries are edge numbers

{4,0,1,2,3}, {3,0,4,5} , {3,1,5,6} ,

{3,2,6,7} , {3,3,4,7}

};

for(int i=0;i<5;i++)

get\_projected\_point(vertices[i][0],vertices[i][1],vertices[i][2]);

setcolor(15);

draw\_3d\_polygon(vertices,edges,surfaces);

settextstyle(2,0,4);

setcolor(11);

outtextxy(300,355,"S1");

outtextxy(260,270,"S2");

outtextxy(310,250,"S3");

outtextxy(370,280,"S4");

outtextxy(310,317,"S5");

setcolor(10);

outtextxy(255,315,"E1");

outtextxy(320,291,"E2");

outtextxy(403,315,"E3");

outtextxy(310,337,"E4");

outtextxy(250,250,"E5");

outtextxy(288,240,"E6");

outtextxy(390,240,"E7");

outtextxy(356,260,"E8");

setcolor(14);

outtextxy(212,344,"V1");

outtextxy(255,293,"V2");

outtextxy(420,293,"V3");

outtextxy(370,350,"V4");

outtextxy(320,115,"V5");

getch( );

closegraph( );

return 0;

}

void draw\_3d\_polygon(int vertices[5][3],int edges[8][2],int surfaces[5][5])

{

for(int i=0;i<5;i++)

{

for(int j=1;j<=surfaces[i][0];j++)

{

int x\_1=vertices[edges[surfaces[i][j]][0]][0];

int y\_1=vertices[edges[surfaces[i][j]][0]][1];

int x\_2=vertices[edges[surfaces[i][j]][1]][0];

int y\_2=vertices[edges[surfaces[i][j]][1]][1];

Line(x\_1,y\_1,x\_2,y\_2);

}

}

}

void get\_projected\_point(int& x,int& y,int& z)

{

float projection\_matrix[4][4]={

{ -Zc,0,0,0 },

{ 0,-Zc,0,0 },

{ Xc,Yc,0,1 },

{ 0,0,0,-Zc }

};

float matrix3d[4]={x,y,z,1};

float matrix2d[4]={0,0,0,0};

for(int count\_1=0;count\_1<4;count\_1++)

{

for(int count\_2=0;count\_2<4;count\_2++)

matrix2d[count\_1]+=

(matrix3d[count\_2]\*projection\_matrix[count\_2][count\_1]);

}

x=(int)(matrix2d[0]+0.5);

y=(int)(matrix2d[1]+0.5);

z=(int)(matrix2d[2]+0.5);

}

void Line(const int x\_1,const int y\_1,const int x\_2,const int y\_2)

{

int color=getcolor( );

int x1=x\_1;

int y1=y\_1;

int x2=x\_2;

int y2=y\_2;

if(x\_1>x\_2)

{

x1=x\_2;

y1=y\_2;

x2=x\_1;

y2=y\_1;

}

int dx=abs(x2-x1);

int dy=abs(y2-y1);

int inc\_dec=((y2>=y1)?1:-1);

if(dx>dy)

{

int two\_dy=(2\*dy);

int two\_dy\_dx=(2\*(dy-dx));

int p=((2\*dy)-dx);

int x=x1;

int y=y1;

putpixel(x,y,color);

while(x<x2)

{

x++;

if(p<0)

p+=two\_dy;

else

{

y+=inc\_dec;

p+=two\_dy\_dx;

}

putpixel(x,y,color);

}

}

else

{

int two\_dx=(2\*dx);

int two\_dx\_dy=(2\*(dx-dy));

int p=((2\*dx)-dy);

int x=x1;

int y=y1;

putpixel(x,y,color);

while(y!=y2)

{

y+=inc\_dec;

if(p<0)

p+=two\_dx;

else

{

x++;

p+=two\_dx\_dy;

}

putpixel(x,y,color);

}

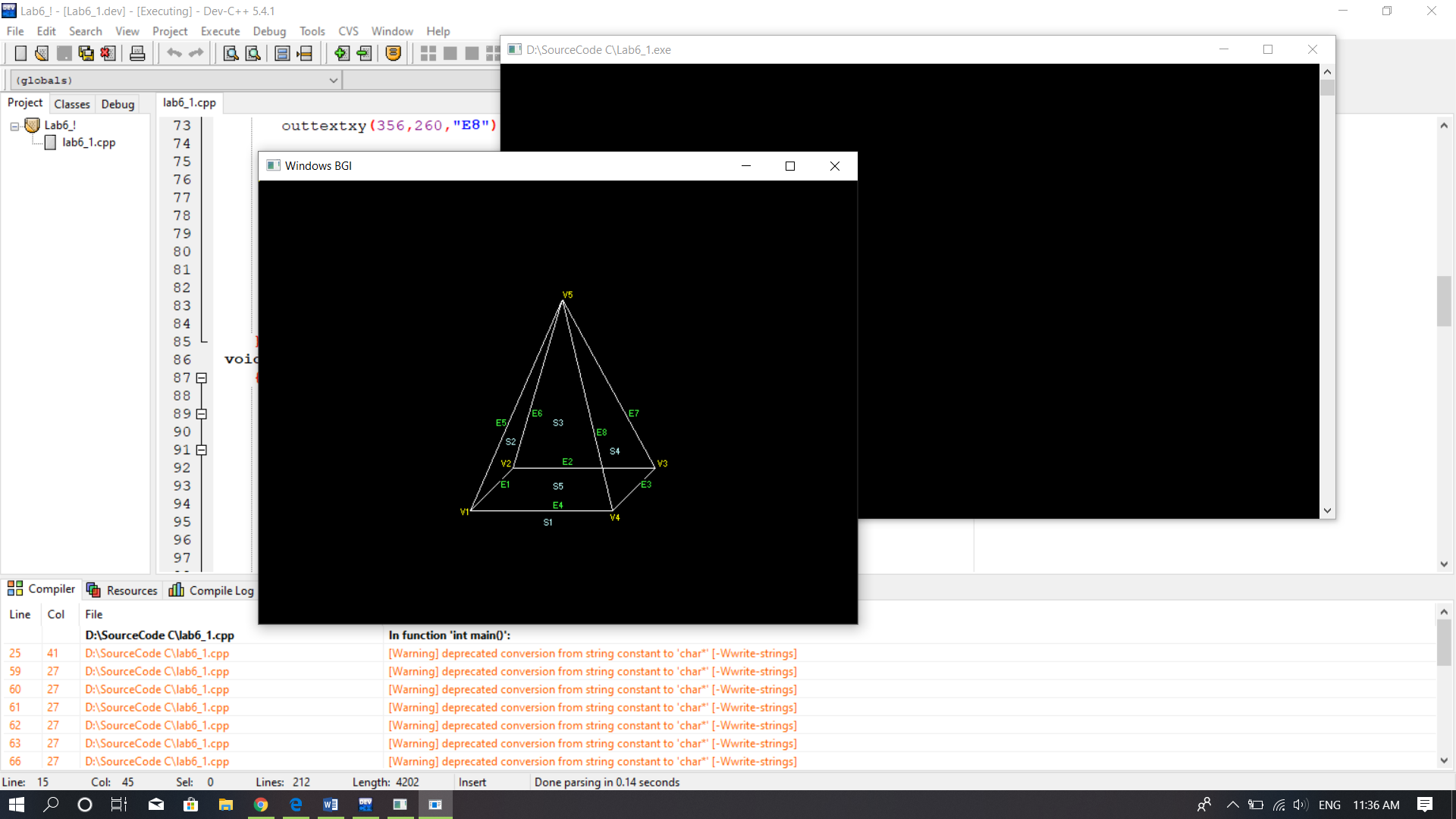
}

}

void show\_screen( )

{

}



2.Vẽ mặt đa giác

* Mã nguồn:

#include<windows.h>

#include<GL/glut.h>

char title[] = "3D Shape";

void initGL()

{

glClearColor(0.0f,0.0f,0.0f,1.0f);

glClearDepth(1.0f);

glEnable(GL\_DEPTH\_TEST);

glDepthFunc(GL\_LEQUAL);

glShadeModel(GL\_SMOOTH);

glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT, GL\_NICEST);

}

void display()

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(1.5f,0.0f,-7.0f);

glBegin(GL\_QUADS);

//green

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

glVertex3f(1.0f,1.0f,-1.0f);

glVertex3f(-1.0f,1.0f,-1.0f);

glVertex3f(-1.0f,1.0f,1.0f);

glVertex3f(1.0f,1.0f,1.0f);

//Orange

glColor3f(1.0f,0.5f,0.0f);

glVertex3f(1.0f,-1.0f,1.0f);

glVertex3f(-1.0f,-1.0f,1.0f);

glVertex3f(-1.0f,-1.0f,-1.0f);

glVertex3f(1.0f,-1.0f,-1.0f);

//Red

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

glVertex3f(1.0f,1.0f,1.0f);

glVertex3f(-1.0f,1.0f,1.0f);

glVertex3f(-1.0f,-1.0f,1.0f);

glVertex3f(1.0f,-1.0f,1.0f);

//Yellow

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

glVertex3f(1.0f,-1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,-1.0f);

glVertex3f(-1.0f,1.0f,-1.0f);

glVertex3f(1.0f,1.0f,-1.0f);

//Blue

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

glVertex3f(-1.0f,1.0f,1.0f);

glVertex3f(-1.0f,1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,-1.0f);

glVertex3f(-1.0f,-1.0f,1.0f);

//Magenta

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

glVertex3f(1.0f,1.0f,-1.0f);

glVertex3f(1.0f,1.0f,1.0f);

glVertex3f(1.0f,-1.0f,1.0f);

glVertex3f(1.0f,-1.0f,-1.0f);

glEnd();

glLoadIdentity();

glTranslatef(-1.5f,0.0f,-6.0f);

glBegin(GL\_TRIANGLES);

//front

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

glVertex3f(0.0f,1.0f,0.0f);

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

glVertex3f(-1.0f,-1.0f,1.0f);

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

glVertex3f(1.0f,-1.0f,1.0f);

//right

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

glVertex3f(0.0f,1.0f,0.0f);

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

glVertex3f(-1.0f,-1.0f,1.0f);

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

glVertex3f(1.0f,-1.0f,-1.0f);

//back

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

glVertex3f(0.0f,1.0f,0.0f);

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

glVertex3f(-1.0f,-1.0f,-1.0f);

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

glVertex3f(-1.0f,-1.0f,-1.0f);

//left

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

glVertex3f( 0.0f, 1.0f, 0.0f);

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

glVertex3f(-1.0f,-1.0f,-1.0f);

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

glVertex3f(-1.0f,-1.0f, 1.0f);

glEnd();

glutSwapBuffers();

}

void reshape(GLsizei width, GLsizei height)

{

if(height == 0) height = 1;

GLfloat aspect = (GLfloat)width / (GLfloat)height;

glViewport(0,0, width,height);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(45.0f,aspect,0.1f,100.0f);

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE);

glutInitWindowSize(640,480);

glutInitWindowPosition(50,50);

glutCreateWindow(title);

glutDisplayFunc(display);

glutReshapeFunc(reshape);

initGL();

glutMainLoop();

return 0;

}

