#include <stdafx.h>

#include <windows.h>

#include <stdlib.h>

#include <glut.h>

#include <string>

#include <iostream>

#include <vector>

#include <io.h>

#include <cmath>

#include <ctime>

#define Pi 3.1415927

using namespace std;

const GLdouble FRUSTDIM = 100.0f;

int option = 0, i = 0;

bool enableLight = 1;

/\* Do animation ??\*/

GLfloat angle = 0,tea\_p = -40 , tea\_face = 100 , donut\_size = 3 , seat\_pos = -60, board\_pos = -50;

GLboolean deskLight = false, air\_open = false;

clock\_t clock\_agl1 = 0;// ??

clock\_t clock\_agl2 = 90;// ??

GLUquadricObj \*quadobj;

/\* For lighting\*/

GLfloat light0pos[] = {0.0f, 1.0f, 0.0f, 0.0f};

GLfloat light0\_mat1[] = {1.0, 1.0, 1.0, 1.f};

GLfloat light0\_diff[] = {1.0, 1.0, 1.0, 0.3};

GLfloat light\_specular[] = { 1.0, 1.0, 1.0, 1.0 };// ?????

//??????

float blackMat[] = { 0.0f,0.0f,0.0f,1.0f };

float ambientMat[] = { 0.1f,0.1f,0.1f,1.0f };

float diffuseMat[] = { 0.4f,0.4f,0.4f,1.0f };

float specularMat[] = { 0.9f,0.9f,0.9f,1.0f };

GLfloat spot\_position[] = { -1.0, -1.0, 0};// ????????

void init(void) // All Setup For OpenGL Goes Here

{

/\* Light 0 Settings \*/

// OpenGL??????????????????,?????????????

// ?????GL\_LIGHT1 ???????GL\_POSITION GL\_AMBIENT GL\_DIFFUSE

glLightfv(GL\_LIGHT1, GL\_POSITION, light0pos);// ???????

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);// ???????? ???Ambient

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light0\_diff);// ??? ???Diffuse

glLightfv(GL\_LIGHT1, GL\_SPECULAR, light\_specular);// ?????

glLightf(GL\_LIGHT1, GL\_SPOT\_CUTOFF, 45);//????????

glLightfv(GL\_LIGHT1, GL\_SPOT\_DIRECTION, spot\_position);//????????

// ?????

glMaterialfv(GL\_FRONT, GL\_AMBIENT, ambientMat);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, diffuseMat);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, specularMat);

quadobj = gluNewQuadric();

// ????

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT1);

glEnable(GL\_NORMALIZE);

glEnable(GL\_DEPTH\_TEST);

glFrontFace(GL\_CCW); /\* Define Counter-Clockwise as front face \*/

glEnable(GL\_COLOR\_MATERIAL);

}

//??

void draw\_light(void)

{

// ????????params

GLfloat matBlack\_light[] = { 0.0f,0.0f,0.0f,1.0f };

GLfloat matWhite\_light[] = { 0.0f,0.0f,0.0f,0.0f };

GLfloat matShininess\_light[] = { 0.0f,0.1f,0.0f,0.0f };

GLfloat matYellow\_light[] = { 0.0f,0.0f,0.5f,0.0f };

bool aCeilingLight = true;

glPushMatrix(); //Celling Light

glColor3f(1, 1, 0);

glMaterialfv(GL\_FRONT, GL\_AMBIENT, matYellow\_light);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, matYellow\_light);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, matWhite\_light);

glMaterialfv(GL\_FRONT, GL\_SHININESS, matShininess\_light);

glMaterialfv(GL\_FRONT, GL\_EMISSION, matBlack\_light);

if(aCeilingLight) glMaterialfv(GL\_FRONT, GL\_EMISSION, matYellow\_light);

glTranslatef(0,8.99,-10);

glRotatef(90, 1, 0, 0);

glutSolidCone(1,1,16,16);

glPopMatrix();

}

//?

void draw\_bed(void)

{

GLfloat x = 6;

GLfloat y1 = -7.5;

GLfloat y2 = -5;

GLfloat z1 = -19.9;

GLfloat z2 = -5;

//??

glBegin(GL\_QUADS);

glColor3f(0.f,0.54f,0.5f);

glVertex3f(x, y2, z1);

glVertex3f(-x, y2,z1);

glVertex3f(-x, y1,z1);

glVertex3f(x, y1, z1);

glEnd();

//??

glBegin(GL\_QUADS);

glColor3f(1.f,0.54f,0.5f);

glVertex3f(x, y1, z1);

glVertex3f(-x, y1,z1);

glVertex3f(-x, y1,z2);

glVertex3f(x, y1, z2);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.f,0.54f,0.5f);

glVertex3f(-x, y1, z1);

glVertex3f(-x, -8.99,z1);

glVertex3f(-x, -8.99,z2);

glVertex3f(-x, y1, z2);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.f,0.54f,0.5f);

glVertex3f(x, y1, z1);

glVertex3f(x, -8.99,z1);

glVertex3f(x, -8.99,z2);

glVertex3f(x, y1, z2);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.f,0.54f,0.5f);

glVertex3f(-x, y1,z2);

glVertex3f(x, y1, z2);

glVertex3f(x, -8.99, z2);

glVertex3f(-x, -8.99,z2);

glEnd();

//??

glBegin(GL\_QUADS);

glColor3f(1.0f,0.84f,0.5f);

glVertex3f(x-0.5, -7, z1+0.5);

glVertex3f(0.5, -7,z1+0.5);

glVertex3f(0.5, -7,-17);

glVertex3f(x-0.5, -7,-17);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.0f,0.84f,0.5f);

glVertex3f(0.5, -7,-17);

glVertex3f(x-0.5, -7,-17);

glVertex3f(x-0.5, y1,-17);

glVertex3f(0.5, y1,-17);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.0f,0.84f,0.5f);

glVertex3f(-x+0.5, -7, z1+0.5);

glVertex3f(-0.5, -7,z1+0.5);

glVertex3f(-0.5, -7,-17);

glVertex3f(-x+0.5, -7,-17);

glEnd();

glBegin(GL\_QUADS);

glColor3f(1.0f,0.84f,0.5f);

glVertex3f(-0.5, -7,-17);

glVertex3f(-x+0.5, -7,-17);

glVertex3f(-x+0.5, y1,-17);

glVertex3f(-0.5, y1,-17);

glEnd();

//??

glPushMatrix();

glTranslatef(0, -7.49, -10.5);

glScalef(10 \* x, 100.1, 100);

glutSolidCube(1);

glPopMatrix();

}

// ??

void draw\_air\_conditioning(void)

{

// bool tv\_open = false;

glPushMatrix();

glTranslatef(-100, 50, 100);

// ??

if (air\_open){// ???

glPushMatrix();

glTranslatef(0, 10, -2);

glScalef( 8.3, 20, 86);

glColor3f(1.0, 1.0, 1.0);// ??

glutSolidCube(1);

glLineWidth(4);

glColor3f(1, 1, 1);

glutWireCube(1);

glLineWidth(1);

glPopMatrix();

glPushMatrix();

glTranslatef(0, -1, 0);

glScalef( 0.001, 0.001, 83);

glColor3f(0.0, 0.0, 0.0);// ??

glutSolidCube(1);

glLineWidth(3);

glColor3f(0.0, 0.0, 0.0);

glutWireCube(1);

glLineWidth(1);

glPopMatrix();

glPushMatrix();

glTranslatef(3.0, -0.5, 0);

glScalef( 0.001, 0.001, 83);

glColor3f(0.0, 0.0, 0.0);// ??

glutSolidCube(1);

glLineWidth(3);

glColor3f(0.0, 0.0, 0.0);

glutWireCube(1);

glLineWidth(1);

glPopMatrix();

// glPushMatrix();

// glTranslatef(5.5, 0, 0);

// glScalef( 0.001, 0.001, 83);

// glColor3f(0.0, 0.0, 0.0);// ??

// glutSolidCube(1);

// glLineWidth(3);

// glColor3f(0.0, 0.0, 0.0);

// glutWireCube(1);

// glLineWidth(1);

// glPopMatrix();

}else{// ???

glPushMatrix();

glTranslatef(0, 10, -2);

glScalef( 8.3, 20, 86);

glColor3f(1.0, 1.0, 1.0);// ??

glutSolidCube(1);

glLineWidth(4);

glColor3f(1, 1, 1);

glutWireCube(1);

glLineWidth(1);

glPopMatrix();

}

glPopMatrix();

}

void display(void) // Here's Where We Do All The Drawing

{

/\* Clear the buffer \*/

glClearColor(0.0, 0.0, 0.0, 1);

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

glLoadIdentity();

glPushMatrix();

glTranslatef(0, 0, -550);

/\* Initialize material properties \*/

GLfloat no\_mat[] = {0.0,0.0,0.0,1.0};

GLfloat mat\_diffuse[] = {0.9,0.9,0.9,1.0};

GLfloat mat\_specular[] = {0.3,0.3,0.3,1.0};

GLfloat high\_shininess[] = {20.0};

GLfloat high\_mat[] = {1.0,1.0,1.0,1.0};

glMaterialfv(GL\_FRONT, GL\_AMBIENT, no\_mat);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, mat\_diffuse);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular);

glMaterialfv(GL\_FRONT, GL\_SHININESS, high\_shininess);

glMaterialfv(GL\_FRONT, GL\_EMISSION, no\_mat);

/\* ?????? \*/

draw\_air\_conditioning();

// ??

/\* Draw a Bed START \*/

glPushMatrix();

glTranslatef(-50.f, -20.f, 195.f);

// ??????

glPushMatrix();

glColor4f(0.2f, 0.2f, 0.2f,0.1f);

glTranslatef(0.f, -45.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(70.f, 100.0f, 40.f);//x:? y:? z:?

glutSolidCube(0.7f);// ????

glPopMatrix();

/\* Rotate with angle \*/

glTranslatef(0.f, -40.f, 130.f);

glRotatef(angle, 0.f, 1.f, 0.f);

glPopMatrix();

/\* draw a ?? END \*/

// ?

/\* Draw a Bed START \*/

glPushMatrix();

// glTranslatef(-100.f, 0.f,100.f);

glTranslatef(-50.f, -10.f, 200.f);

/\* Draw a support for table ??\*/

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(0.f, -90.f, 130.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 10.f, 5.f, 40.f, 20.f, 20.f);

glPopMatrix();

// ??

glPushMatrix();

glColor3f(0.2f, 0.2f, 0.2f);

glTranslatef(0.f, -45.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(25.f, 100.0f, 400.f);//x:? y:? z:?

glutSolidCube(0.7f);// ????

glPopMatrix();

//??

glPushMatrix();

GLfloat x = 6;

GLfloat z1 = -19.9;

glBegin(GL\_QUADS);

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

glTranslatef(0.f, -45.f, 65.f);

glVertex3f(10, -45, 65);

glVertex3f(0.5, -45,70);

glVertex3f(0.5, -40,70);

glVertex3f(0.5, -25,70);

glEnd();

glPopMatrix();

/\* Rotate with angle \*/

glTranslatef(0.f, -40.f, 130.f);

glRotatef(angle, 0.f, 1.f, 0.f);

glPopMatrix();

/\* draw a Bed END \*/

/\* Draw a Table START \*/

glPushMatrix();

glTranslatef(100.f, 0.f,-100.f);

/\* form the table (cyclinder part) ??\*/

glPushMatrix();

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

glTranslatef(0.f, -45.f, 130.f);

// glRotatef(-90, 1.f, 0.f, 0.f);

glScalef(130.f, 15.0f, 120.f);//x:?? y:?? z:??

glutSolidCube(0.7f);// ??

glPopMatrix();

/\* Rotate with angle \*/

glTranslatef(0.f, -40.f, 130.f);

glRotatef(angle, 0.f, 1.f, 0.f);

// ??

GLfloat matBlack[] = { 0.0f,0.0f,0.0f,1.0f };

GLfloat matWhite[] = { 0.0f,0.0f,0.0f,0.0f };

GLfloat matShininess[] = { 0.0f,0.1f,0.0f,0.0f };

GLfloat matYellow[] = { 0.0f,0.0f,0.5f,0.0f };

/\* ?? \*/

glPushMatrix();

// ????????params

GLfloat matBlack\_light[] = { 0.0f,0.0f,0.0f,1.0f };

GLfloat matWhite\_light[] = { 0.0f,0.0f,0.0f,0.0f };

GLfloat matShininess\_light[] = { 0.0f,0.1f,0.0f,0.0f };

GLfloat matYellow\_light[] = { 0.0f,0.0f,0.5f,0.0f };

// bool aCeilingLight = false;

glMaterialfv(GL\_FRONT, GL\_AMBIENT, matYellow\_light);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, matYellow\_light);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, matWhite\_light);

glMaterialfv(GL\_FRONT, GL\_SHININESS, matShininess\_light);

glMaterialfv(GL\_FRONT, GL\_EMISSION, matBlack\_light);

if(deskLight) glMaterialfv(GL\_FRONT, GL\_EMISSION, matYellow\_light);

// glTranslatef(20.f, 20.f, 0.f);

/\* ???? \*/

glPushMatrix(); //Celling Light

glColor4f(0.5f, 0.0f, 0.0f,0.5f);

glTranslatef(20.f, 20.f, 0.f);

glRotatef(90, 1, 0, 0);

glScalef(20.f, 20.0f, -20.f);

glutSolidCone(1,1,16,16);// Cone:??

glPopMatrix();

/\* ???\*/

glPushMatrix();

glColor3f(0.7f, 0.7f, 0.7f);

glTranslatef(20.f, 0.f, 0.f);

// glTranslatef(-30.f, 10.f, 0.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 3.f, 3.f, 20.f, 30.f, 30.f);

glPopMatrix();

/\* ????\*/

glPushMatrix();

glColor3f(0.0f, 0.7f, 0.0f);

glTranslatef(20.f, 0.f, 0.f);

// glTranslatef(-30.f, 10.f, 0.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 10.f, 10.f, 5.f, 30.f, 30.f);

glPopMatrix();

glPopMatrix();

glPopMatrix();

/\* draw a table END \*/

// ??

/\* pull or push the seat ??\*/

glTranslatef(seat\_pos, -100.f, 98.f);

/\*Draw seat START\*/

glPushMatrix();

glTranslatef(10.f, 0.f,-80.f);

// ????

glPushMatrix();

// ???????line???

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

glTranslatef(70.f,0.f,0.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 77, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(70.f,0.f,80.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 77, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(100.f,0.f,0.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 38, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(100.f,0.f,80.f);

glRotatef(-90, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 38, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(70.f,20.f,0.f);

glRotatef(90, 0.f, 1.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 30, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(70.f,20.f,80.f);

glRotatef(90, 0.f, 1.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 30, 20, 20);

glPopMatrix();

// ????

glPushMatrix();

glTranslatef(70.f,60.f,0.f);

glRotatef(0, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 2.f, 2.f, 80, 20, 20);

glPopMatrix();

// ?????

glPushMatrix();

glTranslatef(100.f,37.f,80.f);

glRotatef(-90, 0.f, 1.f, 0.f);

gluCylinder(quadobj, 3.f, 3.f, 30, 20, 20);

glPopMatrix();

// ?????

glPushMatrix();

glTranslatef(100.f,37.f,0.f);

glRotatef(-90, 0.f, 1.f, 0.f);

gluCylinder(quadobj, 3.f, 3.f, 30, 20, 20);

glPopMatrix();

// ?????

glPushMatrix();

glTranslatef(100.f,37.f,-2.f);

glRotatef(0, 1.f, 0.f, 0.f);

gluCylinder(quadobj, 3.f, 3.f, 84, 20, 20);

glPopMatrix();

//seat pan ??

glPushMatrix();

glTranslatef(85.f, 37.f, 40.f);

glScalef(30.f, 5.0f, 82.f);

glutSolidCube(1.f);

glPopMatrix();

glPopMatrix();

glTranslatef(-seat\_pos, 100.f, -98.f);

/\* Draw seat END\*/

// ???????

glPushMatrix();

glColor4f(0.0f, 0.0f, 0.0f,0.1f);

glTranslatef(-50.f, -47.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(7.f, 70.0f, 40.f);//x:? y:?

glutSolidCube(1.f);

glPopMatrix();

// ????

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.3f,0.1f);

glTranslatef(-50.f, -25.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(40.f, 70.0f, 40.f);//x:? y:?

glutSolidCube(1.f);

glPopMatrix();

// ?????

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.3f,0.1f);

glTranslatef(-50.f, 20.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(50.f, 70.0f, 40.f);//x:? y:? z:?

glutSolidCube(1.f);

glPopMatrix();

// ?????

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.3f,0.1f);

// glTranslatef(-50.f, -70.f, 65.f);

glTranslatef(board\_pos, -70.f, 65.f);

glRotatef(90.f, 0.f, 0.f, 30.f);

glScalef(40.f, 70.0f, 40.f);//x:? y:?

glutSolidCube(1.f);

glPopMatrix();

/\*Draw Light bulb START\*/

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.7f,0.1f);

glTranslatef(0.f, 90.f, 200);

glRotatef(-90.f, 1.f, 0.f, 0.f);

glutSolidTorus(2 , 2, 110.f, 110.f);

glPopMatrix();

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.7f,0.1f);

glTranslatef(0.f, 87.f, 200);

glRotatef(-90.f, 1.f, 0.f, 0.f);

glutSolidTorus(2 , 2, 110.f, 110.f);

glPopMatrix();

glPushMatrix();

glColor4f(0.8f, 0.8f, 0.7f,0.1f);

glTranslatef(0.f, 84.f, 200);

glRotatef(-90.f, 1.f, 0.f, 0.f);

glutSolidTorus(2 , 2, 110.f, 110.f);

glPopMatrix();

glPushMatrix();

glMaterialfv(GL\_FRONT, GL\_EMISSION, light0\_mat1);

glColor4f(0.8f, 0.8f, 0.7f,0.1f);

glTranslatef(0.f, 76.f, 200.f);

glutSolidSphere(10.f, 50.f, 50.f);

glPopMatrix();

glMaterialfv(GL\_FRONT, GL\_EMISSION, no\_mat);

/\*Draw Light bulb END\*/

/\* ?? \*/

glPushMatrix();

// ??

// clock\_t clock\_agl1 = 0;// ??

// clock\_t clock\_agl2 = 180;// ??

// Pi = 3.14;

glTranslatef(50, 40, -19.0);

glScalef(10.f, 10.f, 10.f);

//??

gluDisk(quadobj, 2.2, 2.5, 20, 4);

//??

glLineWidth(2);

glBegin(GL\_LINES);

glVertex3f(2.0 \* cos(clock\_agl1 \* Pi / 180), 2.0 \* sin(clock\_agl1 \* Pi / 180), 0);

glVertex3f(0, 0, 0);

glVertex3f(0, 0, 0);

glVertex3f(1, 0, 0);

glVertex3f(0, 0, 0);

// glLineWidth(10);

glVertex3f(1 \* cos(clock\_agl2 \* Pi / 180), 1.5 \* sin(clock\_agl2 \* Pi / 180), 0);

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

glVertex3f(0, 1.9, 0);

glVertex3f(0, 2.2, 0);

glVertex3f(1.9, 0, 0);

glVertex3f(2.2, 0, 0);

glVertex3f(0, -1.9, 0);

glVertex3f(0, -2.2, 0);

glVertex3f(-1.9, 0, 0);

glVertex3f(-2.2, 0, 0);

glEnd();

glLineWidth(1);

glColor3f(0.25, 0.25, 0.25);

gluDisk(quadobj, 0, 2.2, 20, 4);

glFlush();// ??

glPopMatrix();

/\* Draw the wall START \*/

/\* Back wall \*/

glBegin(GL\_TRIANGLE\_FAN);

glColor3f(1.f,0.54f,0.f);

glVertex3f(-110.0f, 110.0f, -200.0f);

glVertex3f(-110.0f, -110.0f, -200.0f);

glVertex3f(110.0f, -110.0f, -200.0f);

glVertex3f(110.0f, 110.0f,-200.0f);

glEnd();

/\* Top wall \*/

glBegin(GL\_POLYGON);

glColor3f(1.f,0.84f,0.3f);

glVertex3f(-80.0f, 80.0f, 0.0f);

glVertex3f(80.0f, 80.0f, 0.0f);

glVertex3f(190.0f, 190.0f,0.0f);

glVertex3f(-190.0f, 190.0f, 0.0f);

glEnd();

glBegin(GL\_QUADS);

/\* Bottom wall \*/

glColor3f(0.8f,0.44f,0.0f);

glVertex3f(80.f, -80.f, 0.f);

glVertex3f(-80.f, -80.f, 0.f);

glColor3f(1.f,0.54f,0.5f);

glVertex3f(-190.f, -190.f, 0.f);

glVertex3f(190.f, -190.f, 0.f);

/\* Right wall \*/

glColor3f(0.9f,0.87f,0.9f);

glVertex3f(190.0f, 190.0f, 0.0f);

glColor3f(1.f,0.54f,0.4f);

glVertex3f(80.0f, 80.0f, 0.0f);

glColor3f(1.f,0.54f,0.f);

glVertex3f(80.0f, -80.0f, 0.0f);

glColor3f(1.f,0.54f,0.3f);

glVertex3f(190.0f, -190.0f, 0.0f);

/\* Left wall \*/

glColor3f(0.9f,0.87f,0.9f);

glVertex3f(-190.0f, 190.0f, 0.0f);

glColor3f(1.f,0.54f,0.4f);

glVertex3f(-80.0f, 80.0f, 0.0f);

glColor3f(1.f,0.54f,0.f);

glVertex3f(-80.0f, -80.0f, 0.0f);

glColor3f(1.f,0.54f,0.3f);

glVertex3f(-190.0f, -190.0f, 0.0f);

glEnd();

/\* Draw the wall END \*/

glPopMatrix();

glutSwapBuffers();

glFlush();

/\* Refresh the frame \*/

glutPostRedisplay();

}

void reshape(int w, int h) // Resize the GL Window. w=width, h=height

{

// winWidth = w; winHeight = h;

glViewport(0, 0, (GLsizei) w, (GLsizei) h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

// Frustum Setting

glFrustum(-FRUSTDIM, FRUSTDIM, -FRUSTDIM, FRUSTDIM, 300., 800.);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

}

void keyboard(unsigned char key, int x, int y) // Handle the keyboard events here

{

switch (key)// ?????

{

case '\033':

exit(0);

break;

case 'p': /\*pause\*/

option = 1;

break;

case '1': /\*decrease light\*/

option = 11;

break;

case '2': /\*increase light\*/

option = 12;

break;

case 's': /\*switch on/off the light\*/

option = 4;

break;

case '3':// ???? ?

option = 2;// ?option==2

break;

case '5':// ??

option = 3;

break;

case '6':

option = 5;

break;

case '7':// ???

option = 13;

break;

case '8':// ???

option = 14;

break;

case 'q':// ??????

option = 7;

break;

case 'w':// ??????

option = 8;

break;

case 'e':// ????

option = 9;

break;

case 'r':// ????

option = 10;

break;

}

}

void idle()

{

if(option == 1)

{

}

else if(option == 2) /\* ?????? \*/

{

if(enableLight)// ?????,??? ????? ????

{

enableLight = 0;

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

light0\_mat1[i] = 0.0;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glDisable(GL\_LIGHT1);

option = 1;

}

else// ?????,???,???????????

{

enableLight = 1;

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

light0\_diff[i] = 0.5;

light0\_mat1[i] = 0.5;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light0\_diff);

glEnable(GL\_LIGHT1);

option = 1;

}

}

else if(option == 3) /\* Clockwise \*/

{

// clock\_agl2 = 180;

int a = 1;

clock\_agl1 += 90;

if (clock\_agl1 % 360 == 0) {

clock\_agl2 += 30;

}

glutPostRedisplay(); // ??

Sleep(a);

}

else if(option == 4)

{

if(enableLight)

{

enableLight = 0;

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

light0\_mat1[i] = 0.0;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glDisable(GL\_LIGHT1);

option = 1;

}

else

{

enableLight = 1;

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

light0\_diff[i] = 1.0;

light0\_mat1[i] = 1.0;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light0\_diff);

glEnable(GL\_LIGHT1);

option = 1;

}

}

else if(option == 5)

{

if(donut\_size <= 5)

donut\_size += 0.2;

}

else if(option == 6)

{

if(donut\_size > 3)

donut\_size -= 0.2;

}

else if(option == 7)

{

if(seat\_pos >= -80)

seat\_pos -= 1;

}

else if(option == 8)

{

if(seat\_pos < -60)

seat\_pos += 1;

}

else if(option == 9)// ????

{

// cout << "board\_pos:" << board\_pos << endl;

if(board\_pos < -25){

// cout << "board\_pos:" << board\_pos << endl;

board\_pos += 5;

}

}

else if(option == 10)

{

if(board\_pos > -50)

board\_pos -= 5;

}

else if(option == 11)// ????

{

if (enableLight)

if(light0\_mat1[0] >= 0){

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

light0\_mat1[i] -= 0.01;

light0\_diff[i] -= 0.01;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light0\_diff);

glEnable(GL\_LIGHT1);

}

}

else if(option == 12)// ????

{

if (enableLight)

if(light0\_mat1[0] <= 1){

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

light0\_mat1[i] += 0.01;

light0\_diff[i] += 0.01;

}

glLightfv(GL\_LIGHT1, GL\_AMBIENT, light0\_mat1);

glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light0\_diff);

glEnable(GL\_LIGHT1);

}

}

else if(option == 13)

{

if (!air\_open)

air\_open = true;

}

else if(option == 14)

{

if (air\_open)

air\_open = false;

}

else{}

}

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

{

/\* Initialization of GLUT Library \*/

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_RGBA|GLUT\_DOUBLE|GLUT\_DEPTH);

/\* Create a window with title specified \*/

glutInitWindowSize(600, 600);

glutInitWindowPosition(100, 100);

glutCreateWindow("3D Room");

init(); /\* not GLUT call, initialize several parameters \*/

/\* Register different CALLBACK function for GLUT to response

\* with different events, e.g. window sizing, mouse click or

\* keyboard stroke\*/

glutReshapeFunc(reshape);

glutDisplayFunc(display);

// glutDisplayFunc(draw\_light);

glutKeyboardFunc(keyboard);

glutIdleFunc(idle);

// draw\_light();

/\*Enter the GLUT event processing loop which never returns.

it will call different registered CALLBACK according

to different events. \*/

glutMainLoop();

return 0;

}