```
#include<stdlib.h>
#include<GL/glut.h>
#include<string.h>
#include<stdio.h>
void MyDisplay();
float t1=0, t2=-11, t3=0, t4=0, t5=0, t6=0, t7=0, t8=0;
int i=0;
void raster(char *s, int x, int y)
{
glRasterPos2i(x,y);
for(i=0;i<strlen(s);i++)
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, s[i]);
}

void *fonts[] =
{
        GLUT_BITMAP_9_BY_15,
        GLUT_BITMAP_TIMES_ROMAN_10,
        GLUT_BITMAP_TIMES_ROMAN_24,
        GLUT_BITMAP_HELVETICA_18,
        GLUT_BITMAP_HELVETICA_12
};

void output(int x, int y, char *string, void *font)
{
        int len, i;

        glRasterPos2f(x, y);
        len = (int)strlen(string);
```

```
	for (i = 0; i < len; i++) {
		glutBitmapCharacter(font, string[i]);
	}
}

void start_screen()
{
glColor3f(0.0, 0.0, 1.0);
	output(230, 470, "HKBK COLLEGE OF ENGINEERING", fonts[2]);

	glColor3f(0.5, 0.5, 0.2);
	output(210, 420, "Department of Computer Science And Engineering", fonts[2]);

	glColor3f(0.5, 0.5, 0.2);
	output(196, 370, "Computer Graphics & Visualization Laboratory-15CSL68",
fonts[2]);

	glColor3f(1.0, 0.4, 0.0);
	output(223, 320, "Elevator: A Vertical Transport Equipment", fonts[2]);

	glColor3f(0.0, 0.2, 0.0);
	output(290, 270, "Submitted By", fonts[3]);

		glColor3f(0.5, 0.1, 0.2);
	output(350, 220, "Vishruth N D - 1HK15CS113", fonts[2]);
	output(150, 220, "Sushma V B - 1HK15CS101", fonts[2]);

	glColor3f(0.1, 0.2, 0.1);
	output(260, 170, "Under the Guidance of", fonts[2]);

	glColor3f(1.5, 0.1, 0.2);
```

```
        output(275, 120, "Prof. Kavya D.S.", fonts[2]);

        glColor3f(0.5, 0.5, 0.2);
        output(500, 80, "[ Press S to Continue ]", fonts[3]);
        glutSwapBuffers();
}

void key_to_start(unsigned char key, int x, int y)
{
if(key=='s'||key=='S')
glutDisplayFunc(MyDisplay);
glutPostRedisplay();
if(t4==0)
{
if(key=='g'||key=='G')
t2=-10;
else if(key=='1')
t2=100;
else if(key=='2')
t2=210;
else if(key=='3')
t2=320;
t4=1;
}
}

void mymenu(int index)
{
switch(index)
{
```

```
case(1): t2=320.1;
break;
case(2): t2=210.0;
break;
case(3): t2=100;
break;
case(4):t2=-10;
break;
default:exit(0);
}
glutDisplayFunc(MyDisplay);
glutPostRedisplay();
}

void mymouse(int btn, int state, int x, int y)
{
if(btn==GLUT_LEFT_BUTTON && state ==GLUT_DOWN)
t3=50;
else if(btn==GLUT_LEFT_BUTTON && state ==GLUT_UP)
t3=100;
}
void lin(int e, int f, int g, int h, int n)
{
for(i=1;i<=n;i++)
{
glBegin(GL_LINES);
glColor3f(0,0,0);
glVertex2i(e+1*20, f);
glVertex2i(g+i*20, h);
glEnd();
```

```
}
}
void lin2(int x, int y)
{
for(i=0;i<5;i++)
{
glBegin(GL_LINES);
glColor3f(0,0,0);
glVertex2i(x+10, y+(i)*20);
glVertex2i(x+10, y+((i+0.5)*20));
glVertex2i(x, y+(i)*10);
glVertex2i(x+20, y+(i)*10);
glEnd();
}
for(i=0;i<10;i++)
{
glBegin(GL_LINES);
glColor3f(0,0,0);
glVertex2i(x, y+i*10);
glVertex2i(x+20,y+i*10);
glEnd();
}
}

void floor(int a, int b, int c, int d)
{
glColor3ub(139,69,19);
glBegin(GL_POLYGON);
glVertex2i(a,b);
glColor3ub(0,0,0);
```

```
glVertex2i(a+c,b);
glColor3ub(139,69,19);
glVertex2i(a+c,b+d);
glColor3ub(0,0,0);
glVertex2i(a,b+d);
glEnd();
}

void back(int a, int b, int c, int d)
{
if(t1==t2&& (b-t2)==40)
{
glColor3ub(30,30,30);
glBegin(GL_POLYGON);
glVertex2i(a,b);
glVertex2i(a+c,b);
glColor3ub(170,170,170);
glVertex2i(a+c,b+d);
glVertex2i(a,b+d);
glEnd();
}

else
{
glColor3ub(30,30,30);
glBegin(GL_POLYGON);
glVertex2i(a,b);
glColor3ub(120,120,120);
glVertex2i(a+c,b);
glColor3ub(50,50,50);
```

```
glVertex2i(a+c,b+d);
glColor3ub(30,30,30);
glVertex2i(a,b+d);
glEnd();
}
}

void liftback(int a, int b, int c, int d)
{
glColor3ub(60,60,60);
glBegin(GL_POLYGON);
glVertex2i(a,b);
glColor3ub(110,110,110);
glVertex2i(a+c,b);
glColor3ub(60,60,60);
glVertex2i(a+c,b+d);
glColor3ub(110,110,110);
glVertex2i(a,b+d);
glEnd();
}

void lift1()
{
if(t1<t2)
{
glBegin(GL_POLYGON);
glColor3ub(128,0,0);
glVertex2i(264,30);
glColor3ub(64,0,0);
glVertex2i(314,30);
```

```
glColor3ub(128,0,0);
glVertex2i(314,140);
glColor3ub(64,0,0);
glVertex2i(264,140);
glEnd();
}

else if(t1>t2)
{
glBegin(GL_POLYGON);
glColor3ub(128,0,0);
glVertex2i(314,30);
glColor3ub(64,0,0);
glVertex2i(314,140);
glColor3ub(128,0,0);
glVertex2i(264,140);
glColor3ub(64,0,0);
glVertex2i(264,30);
glEnd();
}
else
{
glBegin(GL_POLYGON);
glColor3ub(64,0,0);
glVertex2i(264,30);
glColor3ub(128,0,0);
glVertex2i(314,30);
glColor3ub(64,0,0);
glVertex2i(314,140);
glVertex2i(264,140);
```

```
glEnd();
}

glBegin(GL_POLYGON);
glColor3ub(0,0,0);
glVertex2i(314,30);
glVertex2i(314,140);
glVertex2i(315,140);
glVertex2i(315,30);
glEnd();
}
void lift2()
{
glBegin(GL_POLYGON);
glColor3ub(0,0,0);
glVertex2i(316,30);
glVertex2i(316,140);
glVertex2i(317,140);
glVertex2i(317,30);
glEnd();
if(t1<t2)
{
glBegin(GL_POLYGON);
glColor3ub(128,0,0);
glVertex2i(366,30);
glColor3ub(64,0,0);
glVertex2i(366,140);
glColor3ub(128,0,0);
glVertex2i(317,140);
glColor3ub(64,0,0);
```

```
glVertex2i(317,30);
glEnd();
}

else if(t1>t2)
{
glBegin(GL_POLYGON);
glColor3ub(128,0,0);
glVertex2i(366,140);
glColor3ub(64,0,0);
glVertex2i(317,140);
glColor3ub(128,0,0);
glVertex2i(317,30);
glColor3ub(64,0,0);
glVertex2i(366,30);
glEnd();
}
else
{
glBegin(GL_POLYGON);
glColor3ub(64,0,0);
glVertex2i(366,30);
glVertex2i(366,140);
glVertex2i(317,140);
glColor3ub(128,0,0);
glVertex2i(317,30);
glEnd();
}
}
```

```
void door(int a, int b, int c, int d)
{
glColor3ub(210,210,210);
glColor3ub(155,55,10);
glBegin(GL_POLYGON);
glVertex2i(a,b);
glColor3ub(250,220,130);
glVertex2i(a+c,b);
glColor3ub(155,55,10);
glVertex2i(a+c,b+d);
glColor3ub(250,220,130);
glVertex2i(a,b+d);
glEnd();
back(a+12,b+50,35,20);
int e=a+c+50;
int f=b+d-30;
if(a==120)
{
glColor3ub(30,30,30);
glBegin(GL_POLYGON);
glVertex2i(e,f);
glColor3ub(170,170,170);
glVertex2i(e,f+30);
glColor3ub(30,30,30);
glVertex2i(e+20,f+30);
glColor3ub(170,170,170);
glVertex2i(e+20,f);
glEnd();
glColor3ub(192,192,192);
glRasterPos2i(a+c+53,b+d-23);
```

```
if(t1<t2)
{

glBegin(GL_LINES);
glVertex2i(e+10,f+5);
glVertex2i(e+10,f+25);
glVertex2i(e+11,f+5);
glVertex2i(e+11,f+25);
glVertex2i(e+10,f+25);
glVertex2i(e+15,f+15);
glVertex2i(e+10,f+25);
glVertex2i(e+5,f+15);
glEnd();
}

else if(t1>t2)
{
glBegin(GL_LINES);
glVertex2i(e+10,f+5);
glVertex2i(e+10,f+25);
glVertex2i(e+11,f+5);
glVertex2i(e+11,f+25);
glVertex2i(e+10,f+5);
glVertex2i(e+15,f+15);
glVertex2i(e+10,f+5);
glVertex2i(e+5,f+15);
glEnd();
}

else if(t1==100)
```

```
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
else if(t1==210)
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
else if(t1==320)
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'3');
else
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'g');
}
}

void inside_lift()
{
glBegin(GL_POLYGON);
glColor3ub(128,128,128);// inside lift color
glVertex2i(265,30);
glVertex2i(365,30);
glVertex2i(365,140);
glVertex2i(265,140);
glEnd();
glBegin(GL_POLYGON);
glColor3ub(30,30,30);//color grey
glVertex2i(280,40);
glVertex2i(350,40);
glColor3ub(192,192,192);//lift door color
glVertex2i(350,125);
glVertex2i(280,125);
glEnd();
glBegin(GL_LINES);
glColor3ub(0,0,0);//color black
glVertex2i(280,40);
```

```
glVertex2i(280,125);
glVertex2i(350,40);
glVertex2i(350,125);
glVertex2i(280,125);
glVertex2i(350,125);
glVertex2i(365,140);
glVertex2i(350,125);
glVertex2i(265,140);
glVertex2i(280,125);
glEnd();
}

void MyDisplay()
{
glClear(GL_COLOR_BUFFER_BIT);
liftback(260,30,110,430);
glBegin(GL_POLYGON);
glColor3ub(128,64,0);
glVertex2i(314,460);
glVertex2i(314,t1+100);
glVertex2i(316,t1+100);
glVertex2i(316,460);
glEnd();
glPushMatrix();
if(t1<t2)
{
glTranslatef(0,t1+=0.5,0);
inside_lift();
lift1();
lift2();
```

```
}
if(t1==t2)
{
t4=0;
if(t3<50)
{
glTranslatef(0,t2,0);
inside_lift();
lift1();
lift2();
}
if(t3==50 && t5<50)
{
glPushMatrix();
glTranslatef(0,t2,0);
inside_lift();
glPopMatrix();
glPushMatrix();
glTranslatef(t6=t6-0.5,t2,0);
lift1();
glPopMatrix();
glPushMatrix();
glTranslatef(t5=t5+0.5,t2,0);
lift2();
glPopMatrix();
t4=1;
}

if(t3==50 && t5==50)
{
```

```
glPushMatrix();
glTranslatef(0,t2,0);
inside_lift();
glPopMatrix();
glPushMatrix();
glTranslatef(-50,t2,0);
lift1();
glPopMatrix();
glPushMatrix();
glTranslatef(50,t2,0);
lift2();
glPopMatrix();
t4=1;
}

if(t3==100 && t5>0)
{
glPushMatrix();
glTranslatef(0,t2,0);
inside_lift();
glPopMatrix();
glPushMatrix();
glTranslatef(t6=t6+0.5,t2,0);
lift1();
glPopMatrix();
glPushMatrix();
glTranslatef(t5=t5-0.5,t2,0);
lift2();
glPopMatrix();
t4=1;
```

```
}
if(t3==1 && t5==0)
t3=0;
}
if(t1>t2)
{
glTranslatef(0,t1-=0.5,0);
inside_lift();
lift1();
lift2();
}

glPopMatrix();
floor(10,10,620,20);
floor(10,460,620,20);
floor(10,30,20,430);
floor(610,30,20,430);
back(30,30,230,100);
back(30,140,230,100);
back(30,250,230,100);
back(30,360,230,100);
back(370,30,240,100);
back(370,140,240,100);
back(370,250,240,100);
back(370,360,240,100);
floor(30,130,580,10);
floor(30,240,580,10);
floor(30,350,580,10);
lin(0,10,0,20,31);
lin(10,20,10,30,31);
```

```
lin(-10,10,610,10,1);
lin(-10,20,610,20,1);
lin(-10,30,610,30,1);
lin(0,460,0,470,31);
lin(10,470,10,480,31);
lin(-10,460,610,460,1);
lin(-10,470,610,470,1);
lin(-10,480,610,480,1);
lin(0,130,0,135,31);
lin(10,135,10,140,31);
lin(-10,130,610,130,1);
lin(-10,135,610,135,1);
lin(-10,140,610,140,1);
lin(0,240,0,245,31);
lin(10,245,10,250,31);
lin(-10,240,610,240,1);
lin(-10,245,610,245,1);
lin(-10,250,610,250,1);
lin(10,350,10,355,31);
lin(0,355,0,360,31);
lin(-10,350,610,350,1);
lin(-10,355,610,355,1);
lin(-10,360,610,360,1);
lin2(10,30);
lin2(10,140);
lin2(10,250);
lin2(10,360);
lin2(610,30);
lin2(610,140);
lin2(610,250);
```

```
lin2(610,360);
door(120,31,60,80);
door(120,141,60,80);
door(120,251,60,80);
door(120,361,60,80);
door(460,31,60,80);
door(460,141,60,80);
door(460,251,60,80);
door(460,361,60,80);
floor(210,482,210,20);
glPushMatrix();
glTranslatef(t7-=0.5,0,0);
glColor3ub(255,255,255);
if(t8==0)
{
raster("press g for ground floor",470,484);
if(t7<-480)
{
t7=0;
t8++;
}
}
if(t8==1)
{
raster("press 1 first floor",470,484);
if(t7<-480)
{
t7=0;
t8++;
}
```

```
}
if(t8==2)
{
raster("press 2 second floor", 470,484);
if(t7<-480)
{
t7=0;
t8++;
}
}
if(t8==3)
{
raster("press 3 third floor", 470,484);
if(t7<-480)
{
t7=0;
t8++;
}
}
if(t8==4)
{
raster("left click open",470,484);
if(t7<-480)
{
t7=0;
t8=0;
}
}
glPopMatrix();
glColor3ub(192,192,192);
```

```
glRasterPos2i(132.5,82.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'G');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
//glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glRasterPos2i(472.5,82.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'G');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
//glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glRasterPos2i(132.5,192.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glRasterPos2i(472.5,192.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glRasterPos2i(132.5,302.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glRasterPos2i(472.5,302.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glRasterPos2i(132.5,412.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'3');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'1');
glRasterPos2i(472.5,412.5);
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'3');
```

```
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'0');
glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24,'2');
glFlush();
glutSwapBuffers();
glutPostRedisplay();
}

void myInit(void)
{
glClearColor(1.0,1.0,1.0,0.0);
glMatrixMode(GL_PROJECTION);
glShadeModel(GL_SMOOTH);
glBlendFunc(GL_SRC_COLOR, GL_ONE_MINUS_SRC_ALPHA);
glEnable(GL_BLEND);
glLoadIdentity();
glMatrixMode(GL_MODELVIEW);
gluOrtho2D(0.0,640.0,0.0,510.0);
}
int main(int argc, char **argv)
{
glutInit(&argc,argv);
glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB|GLUT_DEPTH);
glutInitWindowSize(1500,1000);
glutInitWindowPosition(0,0);
glutCreateWindow("elevator");
glutDisplayFunc(start_screen);
glutKeyboardFunc(key_to_start);
glutMouseFunc(mymouse);
glutCreateMenu(mymenu);
glutAddMenuEntry("third",1);
```

```
glutAddMenuEntry("second",2);
glutAddMenuEntry("first",3);
glutAddMenuEntry("ground",4);
glutAddMenuEntry("exit",5);
glutAttachMenu(GLUT_RIGHT_BUTTON);
myInit();
glutMainLoop();
}
```