

```
/*NAME:-  
ROLL_NO:-  
DIV:-  
AIM:- Write Opengl program to draw Sun Rise and Sun Set.  
*/
```

```
#include<iostream>  
#include<stdlib.h>
```

```
#ifdef __APPLE__  
#include<opengl/opengl.h>  
#include<GLUT/glut.h>  
#else  
#include<GL/glut.h>  
#endif
```

```
using namespace std;
```

```
float ballX = -0.8f;  
float ballY = -0.3f;  
float ballZ = -1.2f;  
float colR=3.0;  
float colG=1.5;  
float colB=1.0;  
float bgColR=0.0;  
float bgColG=0.0;  
float bgColB=0.0;
```

```
static int flag=1;
```

```
void drawBall(void) {
```

```
    glColor3f(colR,colG,colB); //set ball colour  
    glTranslatef(ballX,ballY,ballZ); //moving it toward the screen a bit on creation  
    glutSolidSphere (0.05, 30, 30); //create ball.
```

```
}
```

```
void drawAv(void) {
```

```
    glBegin(GL_POLYGON);  
  
    glColor3f(1.0,1.0,1.0);
```

```

    glVertex3f(-0.9,-0.7,-1.0);

    glVertex3f(-0.5,-0.1,-1.0);

    glVertex3f(-0.2,-1.0,-1.0);

    glVertex3f(0.5,0.0,-1.0);

    glVertex3f(0.6,-0.2,-1.0);

    glVertex3f(0.9,-0.7,-1.0);

    glEnd();

}

void drawClouds(){}
void keyPress(int key, int x, int y)
{
    if(key==GLUT_KEY_RIGHT)
        ballX -= 0.05f;
    if(key==GLUT_KEY_LEFT)
        ballX += 0.05f;

    glutPostRedisplay();
}

void initRendering() {
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_COLOR_MATERIAL);
    glEnable(GL_LIGHTING); //Enable lighting
    glEnable(GL_LIGHT0); //Enable light #0
    glEnable(GL_LIGHT1); //Enable light #1
    glEnable(GL_NORMALIZE); //Automatically normalize normals
    //glShadeModel(GL_SMOOTH); //Enable smooth shading
}

//Called when the window is resized
void handleResize(int w, int h) {
    //Tell OpenGL how to convert from coordinates to pixel values
    glViewport(0, 0, w, h);

```

```

glMatrixMode(GL_PROJECTION); //Switch to setting the camera perspective

//Set the camera perspective
glLoadIdentity(); //Reset the camera
gluPerspective(45.0,          //The camera angle
               (double)w / (double)h, //The width-to-height ratio
               1.0,           //The near z clipping coordinate
               200.0);        //The far z clipping coordinate
}

void drawScene()
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
    glClearColor(bgColR,bgColG,bgColB,0.0);
    glMatrixMode(GL_MODELVIEW);

    glLoadIdentity();

    //Add ambient light
    GLfloat ambientColor[] = {0.2f, 0.2f, 0.2f, 1.0f}; //Color (0.2, 0.2, 0.2)
    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, ambientColor);

    //Add positioned light
    GLfloat lightColor0[] = {0.5f, 0.5f, 0.5f, 1.0f}; //Color (0.5, 0.5, 0.5)
    GLfloat lightPos0[] = {4.0f, 0.0f, 8.0f, 1.0f}; //Positioned at (4, 0, 8)
    glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor0);
    glLightfv(GL_LIGHT0, GL_POSITION, lightPos0);

    //Add directed light
    GLfloat lightColor1[] = {0.5f, 0.2f, 0.2f, 1.0f}; //Color (0.5, 0.2, 0.2)
    //Coming from the direction (-1, 0.5, 0.5)
    GLfloat lightPos1[] = {-1.0f, 0.5f, 0.5f, 0.0f};
    glLightfv(GL_LIGHT1, GL_DIFFUSE, lightColor1);
    glLightfv(GL_LIGHT1, GL_POSITION, lightPos1);

    //drawing the SUN
    glPushMatrix();
        drawBall();
    glPopMatrix();
    //drawing the Mount Avarest
    glPushMatrix();
        drawAv();
    glPopMatrix();
}

```

```
//drawing the Clouds
glPushMatrix();
    drawClouds();
glPopMatrix();

glutSwapBuffers();
}
```

```
//float _angle = 30.0f;
void update(int value) {
```

```
    if(ballX>0.9f)
    {
        ballX = -0.8f;
        ballY = -0.3f;
        flag=1;
        colR=2.0;
        colG=1.50;
        colB=1.0;
```

```
        bgColB=0.0;
    }
```

```
    if(flag)
    {
        ballX += 0.001f;
        ballY +=0.0007f;
        colR-=0.001;
        //colG+=0.002;
        colB+=0.005;
```

```
    bgColB+=0.001;
```

```
    if(ballX>0.01)
    {
        flag=0;

    }
```

```
    }
    if (!flag)
    {
        ballX += 0.001f;
```

```

    ballY -=0.0007f;
    colR+=0.001;
    colB-=0.01;

    bgColB-=0.001;

    if(ballX<-0.3)
    {
        flag=1;
    }
}

glutPostRedisplay(); //Tell GLUT that the display has changed

//Tell GLUT to call update again in 25 milliseconds
glutTimerFunc(25, update, 0);
}

int main(int argc,char** argv)
{
    glutInit(&argc,argv);

    glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB|GLUT_DEPTH);

    glutInitWindowSize(400,400);

    glutCreateWindow("Sun");

    initRendering();

    glutDisplayFunc(drawScene);

    glutFullScreen();

    glutSpecialFunc(keyPress);
    glutReshapeFunc(handleResize);

    glutTimerFunc(1, update, 0);

    glutMainLoop();

    return(0);

```

```
}
```

```
/* *****OUTPUT*****
```

```
user@user:~$ g++ EXP6C.cpp -lglut -lGLU -lGL
```

```
user@user:~$ ./a.out
```