DATE: 26/10/2023

GROUP C: PRATICAL.6

NAME: SAYALI TANAJI PAWAR
ROLL NO: S213002
CLASS: SE
DIV: C
BATCH: C1

PROBLEM STATEMENT:

Write OpenGL program to draw Sun Rise and Sunset.

CODE:

```
#include&lt;GL/glut.h&gt;
#include&lt;math.h&gt;
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)                    //function to draw the ball
{
glColor3f(colR,colG,colB);

glTranslatef(ballX,ballY,ballZ);
glutSolidSphere (0.05, 30, 30);
}
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 initRendering()                    //function to initialize rendering
{
glEnable(GL_DEPTH_TEST);
glEnable(GL_COLOR_MATERIAL);
glEnable(GL_LIGHTING);
glEnable(GL_LIGHT0);
glEnable(GL_LIGHT1);
glEnable(GL_NORMALIZE);
}
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,1.0,200.0);
}
void drawScene()                        //func to draw scene
{
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
glClearColor(bgColR,bgColG,bgColB,0.0);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
GLfloat ambientColor[] = {0.2f, 0.2f, 0.2f, 1.0f};
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, ambientColor);
GLfloat lightColor0[] = {0.5f, 0.5f, 0.5f, 1.0f};
GLfloat lightPos0[] = {4.0f, 0.0f, 8.0f, 1.0f};
glLightfv(GL_LIGHT0, GL_DIFFUSE, lightColor0);
glLightfv(GL_LIGHT0, GL_POSITION, lightPos0);
GLfloat lightColor1[] = {0.5f, 0.2f, 0.2f, 1.0f};
GLfloat lightPos1[] = {-1.0f, 0.5f, 0.5f, 0.0f};
glLightfv(GL_LIGHT1, GL_DIFFUSE, lightColor1);
glLightfv(GL_LIGHT1, GL_POSITION, lightPos1);
```

```
glPushMatrix();
drawBall();
glPopMatrix();

glPushMatrix();
drawAv();
glPopMatrix();
glPushMatrix();
glPopMatrix();
glutSwapBuffers();
}
void update(int value)
{
if(ballX&gt;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;
colB+=0.005;
bgColB+=0.001;

if(ballX&gt;0.01)
{
flag=0;
}
}
if (!flag)
{
ballX += 0.001f;
ballY -=0.0007f;
colR+=0.001;
colB-=0.01;
bgColB-=0.001;
if(ballX&lt;-0.3)
```

```
{
flag=1;
}
}
glutPostRedisplay();
glutTimerFunc(25, update, 0);
}
int main(int argc,char** argv)
{
glutInit(&amp;argc,argv);
glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB|GLUT_DEPTH);
glutInitWindowSize(400,400);
glutCreateWindow(&quot;Sun&quot;);
initRendering();

glutDisplayFunc(drawScene);
glutFullScreen();
glutReshapeFunc(handleResize);
glutTimerFunc(25, update, 0);
glutMainLoop();
return(0);
}
```

OUTPUT: