Computer Science Department

Sam Houston State University

COSC 4332 Computer Graphics

OpenGL Lab
Basic Lightening

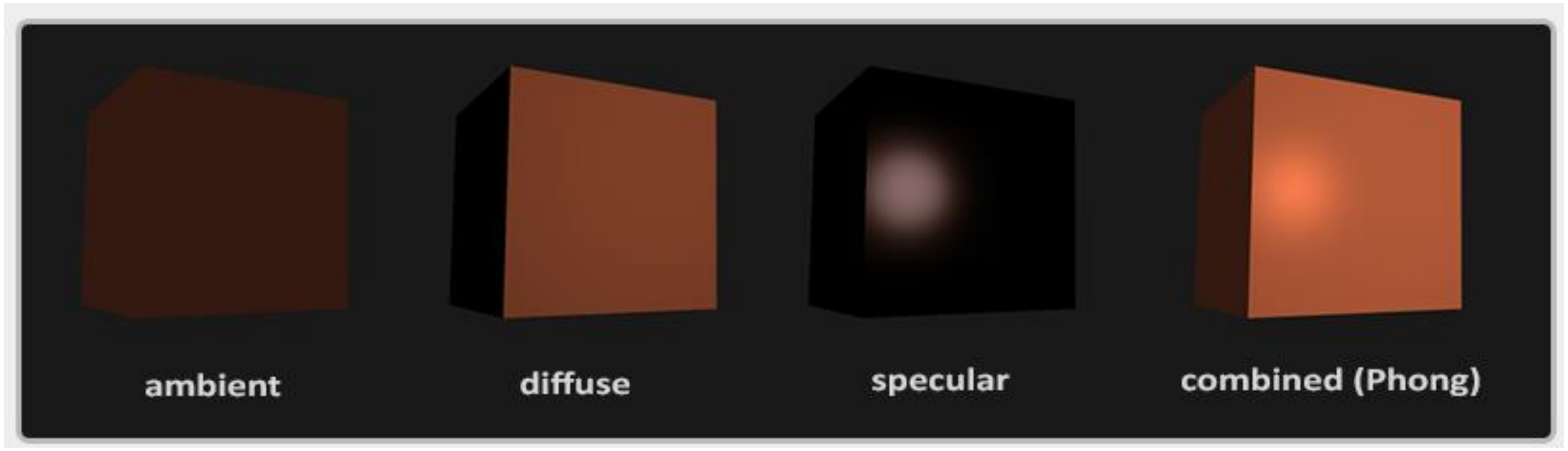
Dr. Khaled Rabieh

Basic Lighting

- Lighting in the real world is extremely complicated and depends on too many factors, something we can't afford to calculate on the limited processing power we have.
- Lighting in OpenGL is based on approximations of reality using simplified models.
  - Based on the physics of light as we understand it such as Phong lighting model.
  - The major building blocks of the Phong model consist of 3 components: ambient, diffuse and specular lighting.

https://learnopengl.com/#!Lighting/Basic-Lighting

Basic Lighting

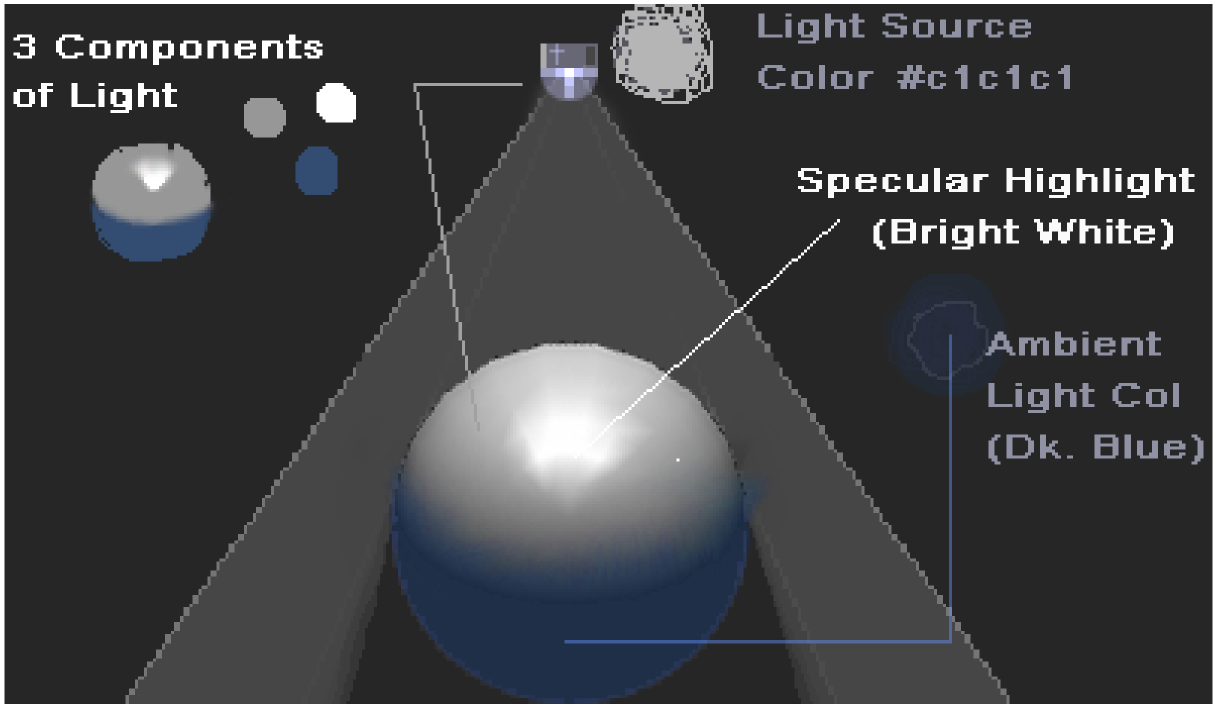


- Ambient lighting: Even when it is dark there is usually still some light somewhere in the world (the moon, a distant light) so objects are almost never completely dark. To simulate this we use an ambient lighting constant that always gives the object some color.

- Diffuse lighting: color of the light. This is the most visually significant component of the lighting model

- Specular lighting: the shiny, glossy reflection spot

Standard light model in OpenGL


3 Components of Light
Light Source Color #c1c1c1
Specular Highlight (Bright White)
Ambient Light Col (Dk. Blue)

You are seeing the standard **OpenGL Light** model on this screenshot diagram consisting of 3 abstract components: Ambient Light, Diffuse Light (color of the light source) and specular highlight (the shiny, glossy reflection spot).

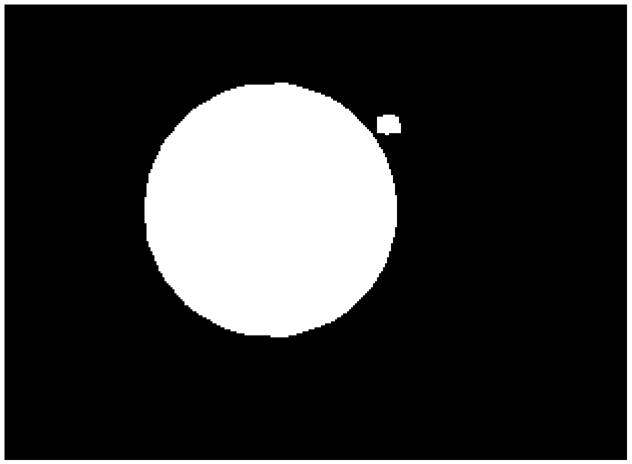
Global Illumination Algorithms

- Light usually does not come from a single light source
  - But from many light sources scattered all around us
  - Light can scatter and bounce in many directions reaching spots that aren't in its direct vicinity
  - Light can thus *reflect* on other surfaces and have an indirect impact on the lighting of an object.
  - Algorithms that take this into consideration are called global illumination algorithms, but these are expensive and/or complicated.

Basic Lighting

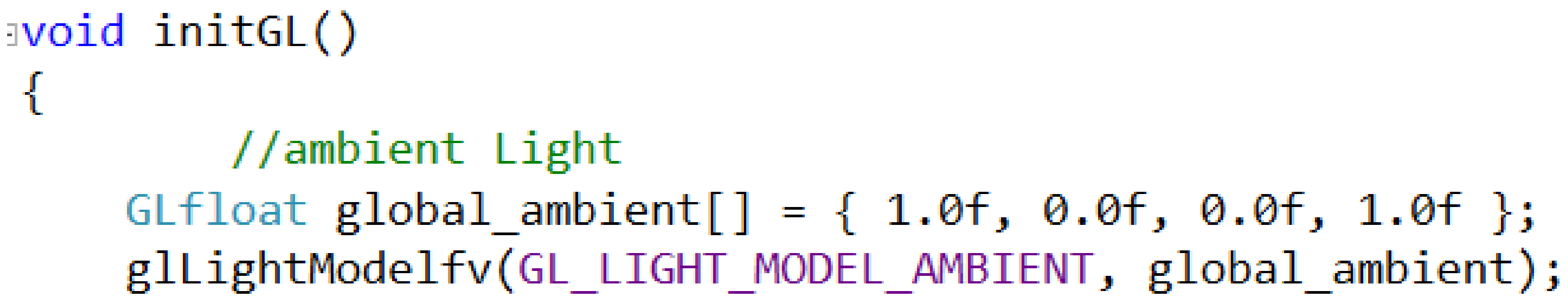
- OpenGL supports a fixed number of lights.
- By default in OpenGL there are maximum of 8 light sources identified by ID handles GL_LIGHT0 through GL_LIGHT7 in a single scene.
- By default, the light in OpenGL is white
- To Enable lighting in general

```
glEnable(GL_LIGHTING);
```



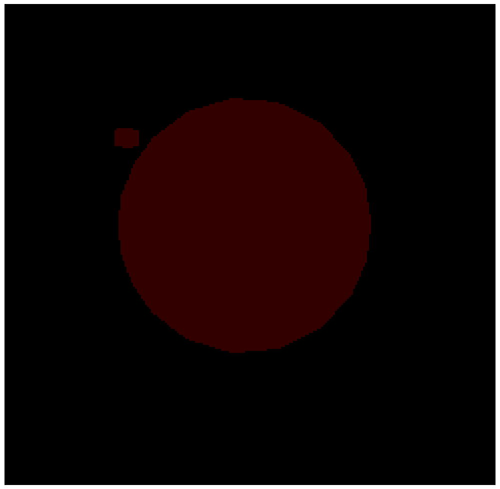
Setting up Ambient light

- It is possible to set the GLOBAL amount of the ambient light which will be cast on all rendered objects.
- In init function – specify the color of the ambient light



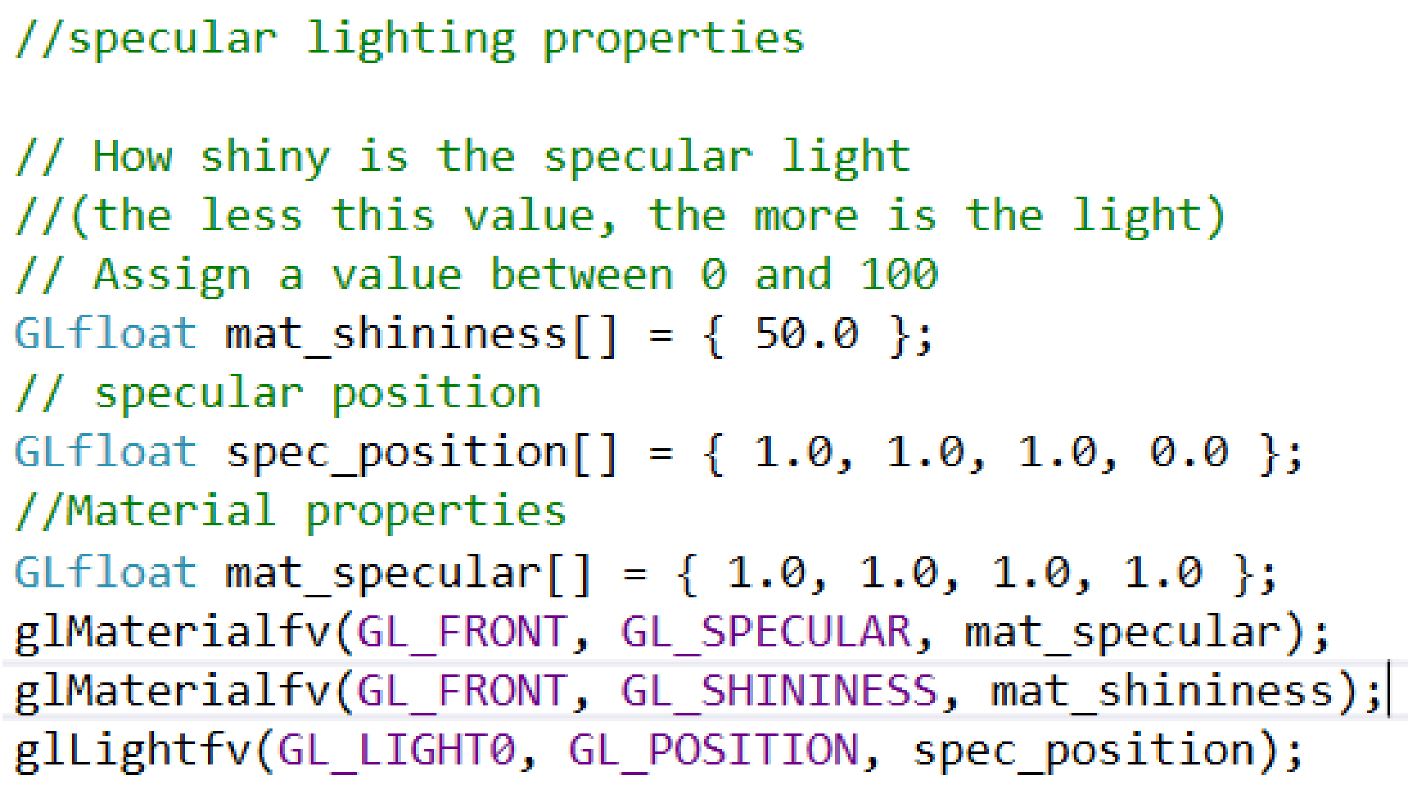
```
void initGL()
{
        //ambient Light
    GLfloat global_ambient[] = { 1.0f, 0.0f, 0.0f, 1.0f };
    glLightModelfv(GL_LIGHT_MODEL_AMBIENT, global_ambient);
```

Red ambient light



Defining a specular light source properties

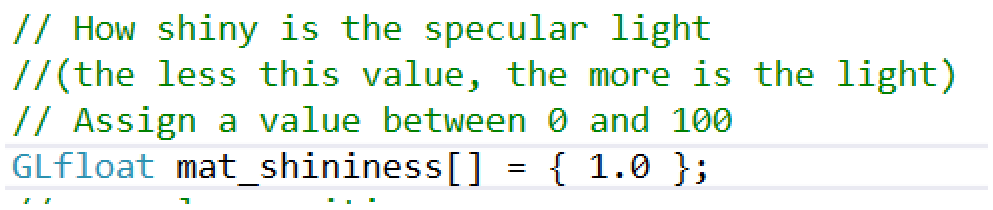
- Define the position of specular light
- Define the intensity of specular light
- Define the material properties



```
//specular lighting properties

// How shiny is the specular light
//(the less this value, the more is the light)
// Assign a value between 0 and 100
GLfloat mat_shininess[] = { 50.0 };
// specular position
GLfloat spec_position[] = { 1.0, 1.0, 1.0, 0.0 };
//Material properties
GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
glLightfv(GL_LIGHT0, GL_POSITION, spec_position);
```

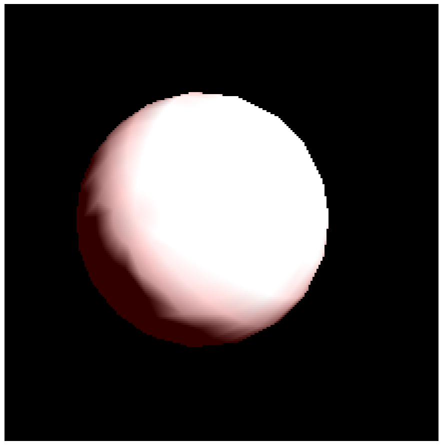
Specular Light Intensity

The intensity of specular light with respect to different values as shown below

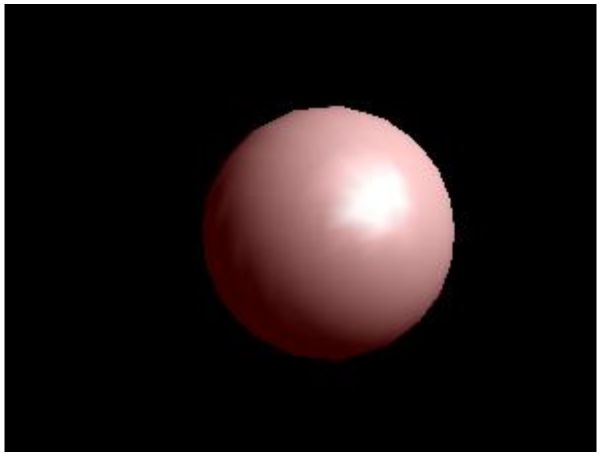


```
// How shiny is the specular light
//(the less this value, the more is the light)
// Assign a value between 0 and 100
GLfloat mat_shininess[] = { 1.0 };
```

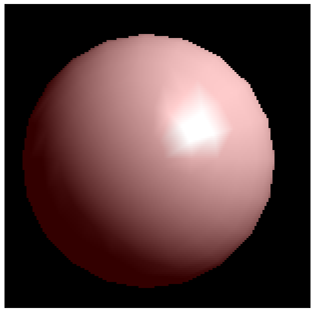
1



50



100

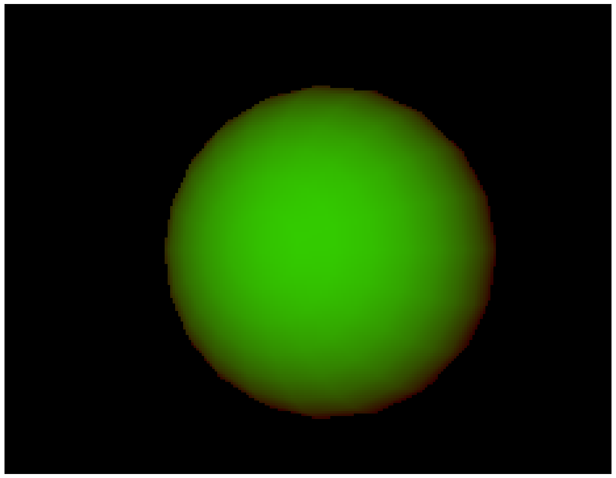


Defining the main light source (Diffuse light) properties

Define a green diffuse light

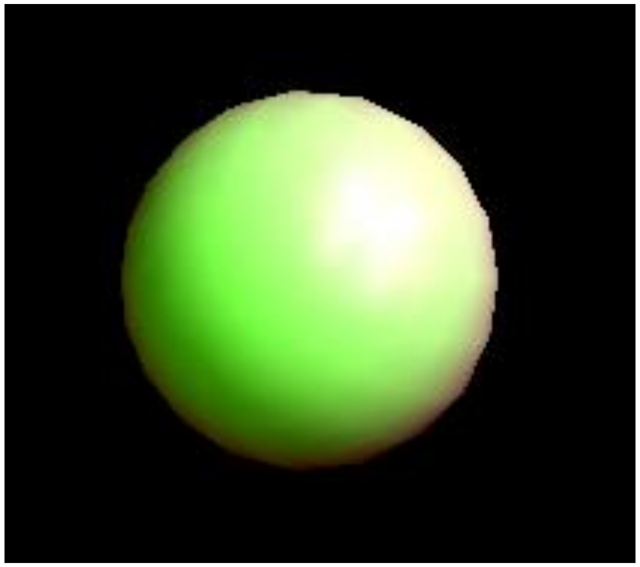
```
//Diffuse light the  main light position
float diffuse_position[] = { -1.0f, 1.0f, 1.0f, 1.0f };
//Diffuse light Color
GLfloat diffuseLight[] = { 0.0f, 1.0f, 0.0, 1.0f };
glLightfv(GL_LIGHT1, GL_DIFFUSE, diffuseLight);
glLightfv(GL_LIGHT1, GL_POSITION, diffuse_position);
```

Notice the red ambient color that appears in dark positions

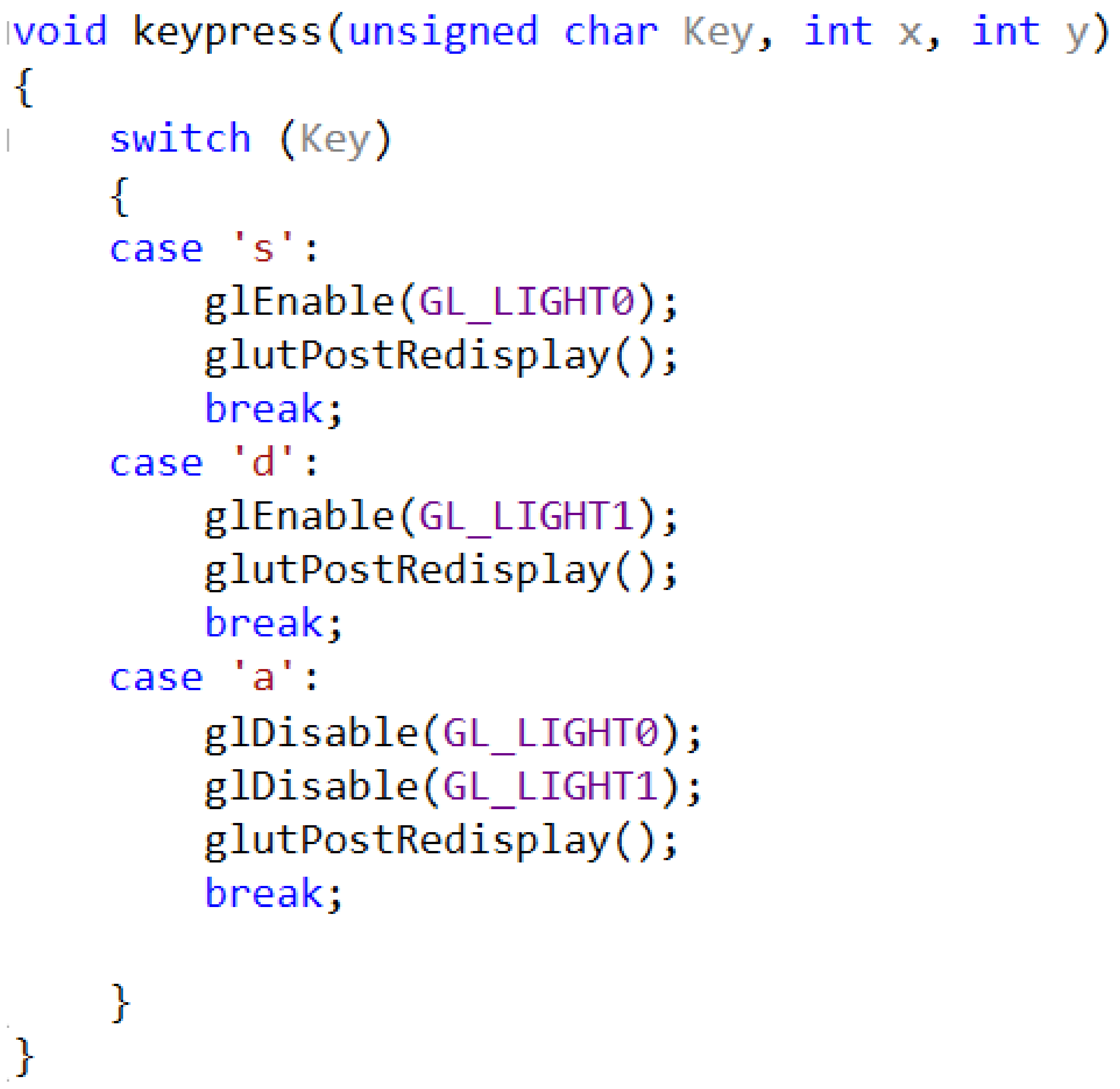


Lightening Tips

- You should enable every light by its own
  - glEnable(GL_LIGHT0)
  - glEnable(GL_LIGHT1)
- glEnable(GL_LIGHTING) alone is not enough
- An object that receives lightening from two sources can be as follows

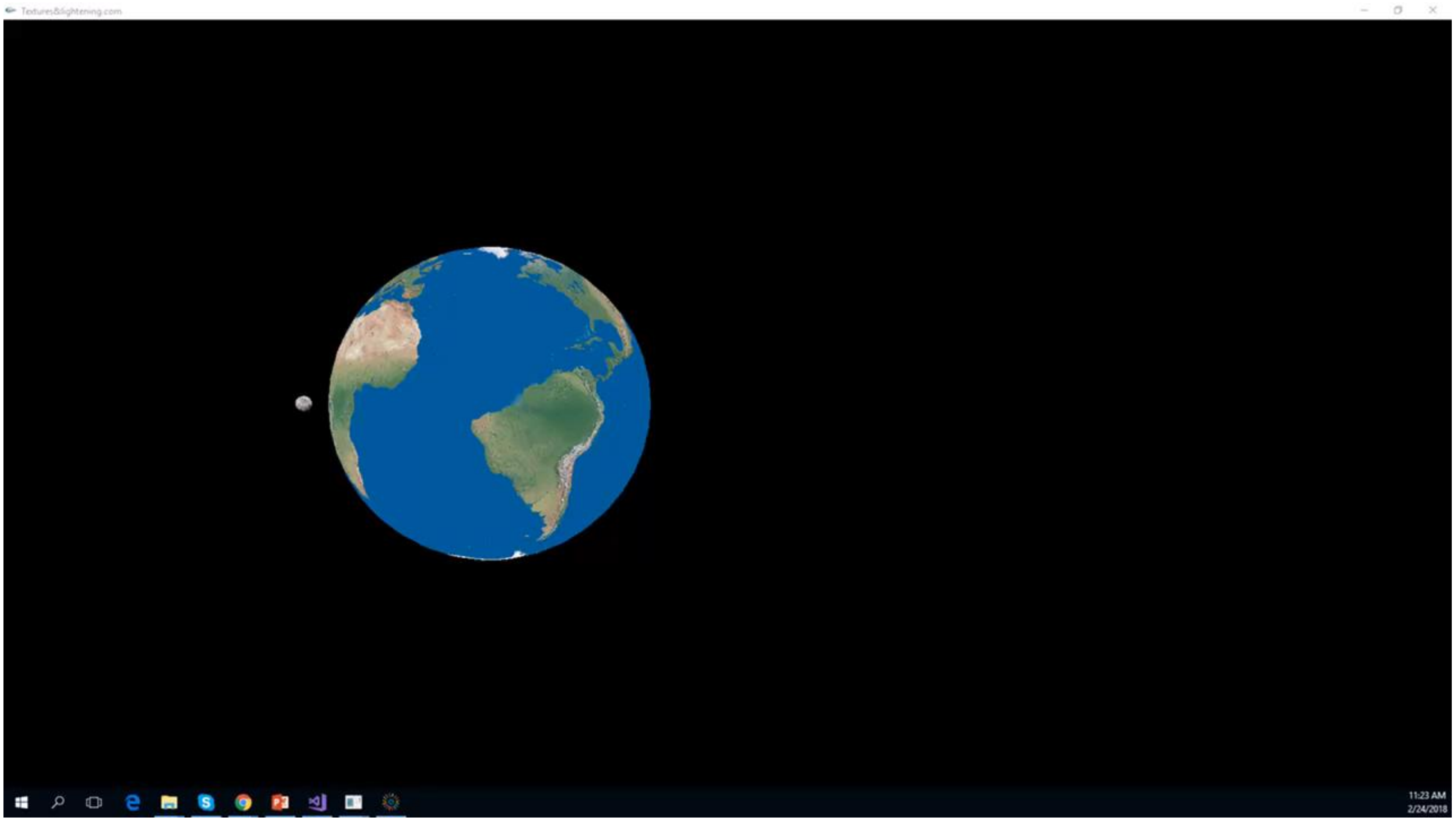


Enabling and disabling Lighting using Keyboard keys



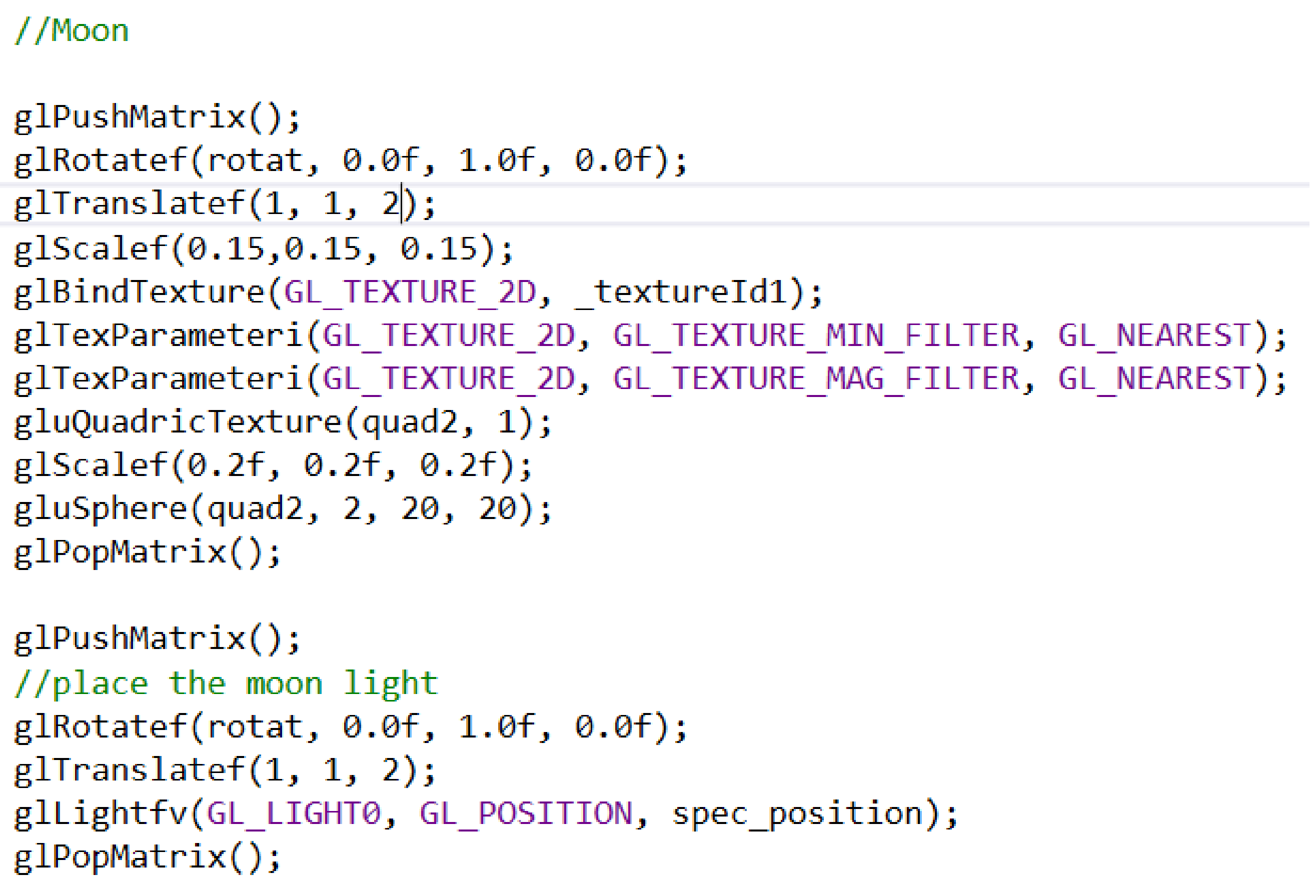
```
void keypress(unsigned char Key, int x, int y)
{
    switch (Key)
    {
    case 's':
        glEnable(GL_LIGHT0);
        glutPostRedisplay();
        break;
    case 'd':
        glEnable(GL_LIGHT1);
        glutPostRedisplay();
        break;
    case 'a':
        glDisable(GL_LIGHT0);
        glDisable(GL_LIGHT1);
        glutPostRedisplay();
        break;

    }
}
```

Lightening the earth and the moon

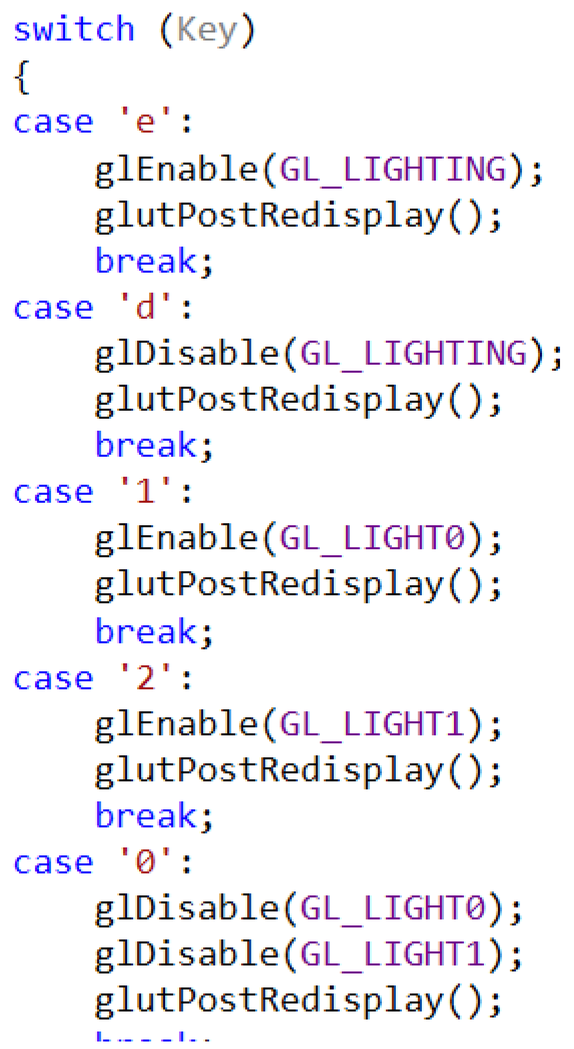


Important Tips

You should move the light with the moon



```
//Moon

glPushMatrix();
glRotatef(rotat, 0.0f, 1.0f, 0.0f);
glTranslatef(1, 1, 2);
glScalef(0.15,0.15, 0.15);
glBindTexture(GL_TEXTURE_2D, _textureId1);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
gluQuadricTexture(quad2, 1);
glScalef(0.2f, 0.2f, 0.2f);
gluSphere(quad2, 2, 20, 20);
glPopMatrix();

glPushMatrix();
//place the moon light
glRotatef(rotat, 0.0f, 1.0f, 0.0f);
glTranslatef(1, 1, 2);
glLightfv(GL_LIGHT0, GL_POSITION, spec_position);
glPopMatrix();
```



```
switch (Key)
{
case 'e':
    glEnable(GL_LIGHTING);
    glutPostRedisplay();
    break;
case 'd':
    glDisable(GL_LIGHTING);
    glutPostRedisplay();
    break;
case '1':
    glEnable(GL_LIGHT0);
    glutPostRedisplay();
    break;
case '2':
    glEnable(GL_LIGHT1);
    glutPostRedisplay();
    break;
case '0':
    glDisable(GL_LIGHT0);
    glDisable(GL_LIGHT1);
    glutPostRedisplay();
    break;
```

Thank You

Questions

Khaled Rabieh