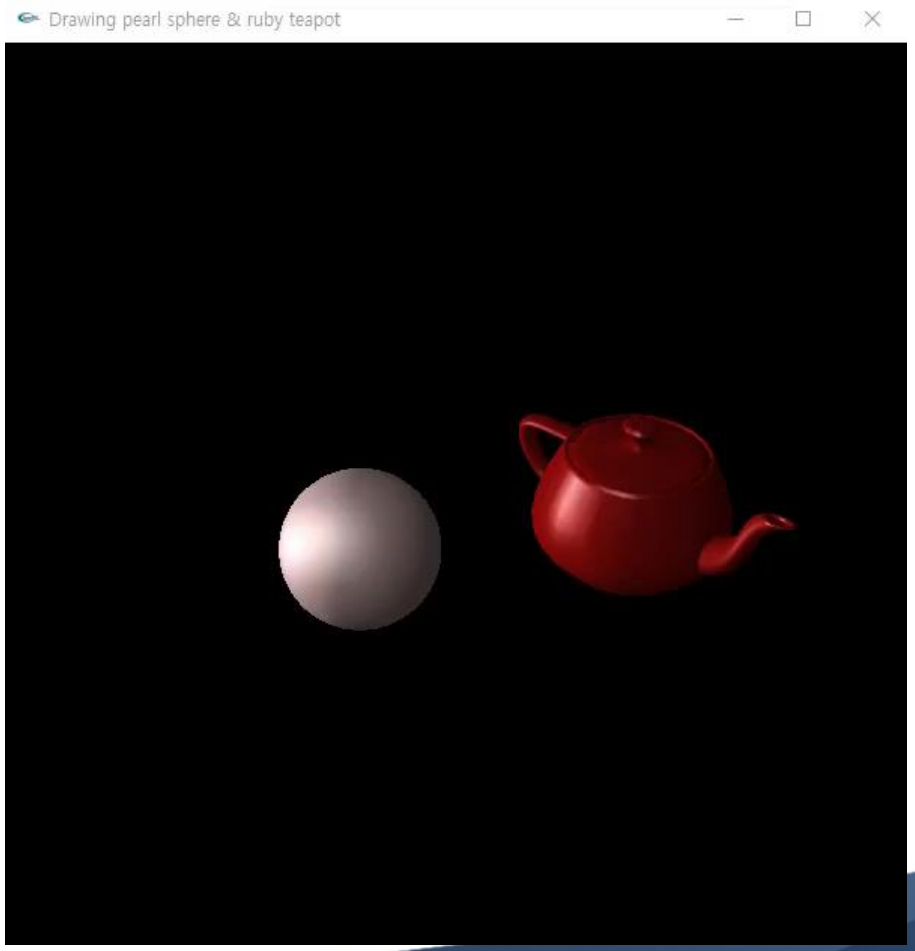
LAB I
Week 08

Seoul National University
Graphics & Media Lab
HyeonSeung Shin

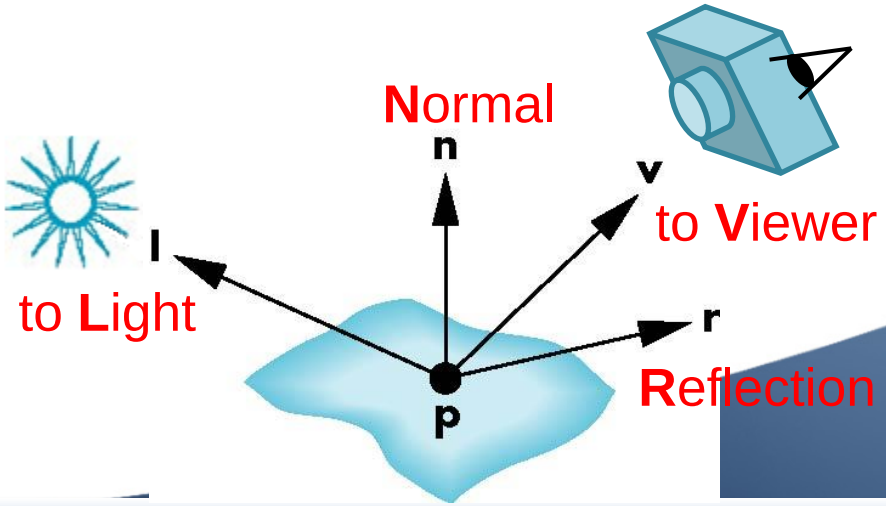
Today's Mission

- Shading
  - Draw a pearl sphere and a ruby teapot with two directional lights



How to Shade in OpenGL?

1. Set the light sources
   - Position
   - Type: directional light, point light
   - Other Properties: ambient, diffuse, specular
2. Set the materials
   - Properties: ambient, diffuse, specular, emission, shininess
3. Set the shading method
   - Flat or Gouraud



Enabling Lighting and Lights

- Lighting in general must be enabled
  - glEnable(GL_LIGHTING);
- Each individual light must be enabled
  - glEnable(GL_LIGHT0);
- OpenGL supports at least 8 light sources

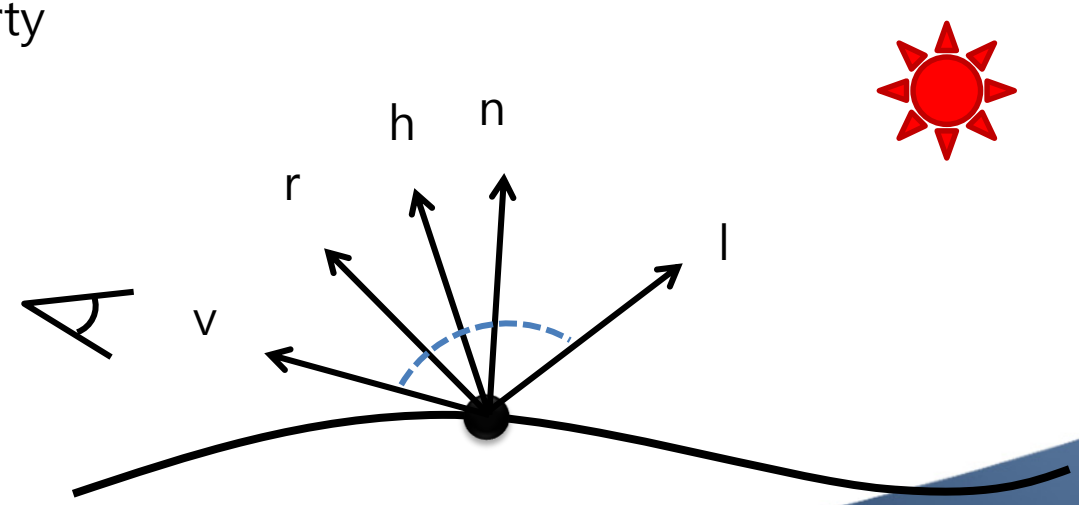
OpenGL Lighting Model

- Modified Phong Model

$$I = I_{emissive} + I_{ambient} + I_{diffuse} + I_{specular}$$

$$= K_e + K_a L_{ga} + \sum_{lights} (Spot_i)(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n} \cdot \mathbf{l}) + K_s L_{i,s}(\mathbf{n} \cdot \mathbf{h}_i)^{\alpha_i})$$

K: Material property
L: Light property



Global Ambient Light

- Set ambient intensity for entire scene

```
GLfloat al[] = {0.2, 0.2, 0.2, 1.0};
glLightModelfv(GL_LIGHT_MODEL_AMBIENT, al);
```

$$= K_e + K_a \boxed{L_{ga}} + \sum_{lights} (Spot_i)(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n} \cdot \mathbf{l}) + K_s L_{i,s}(\mathbf{n} \cdot \mathbf{h}_i)^{\alpha_i})$$

Defining Each Light Source

- Set individual attribute for a single light source

```
GLfloat light_ambient[] = {0.2, 0.2, 0.2, 1.0};
GLfloat light_diffuse[] = {1.0, 1.0, 1.0, 1.0};
GLfloat light_specular[] = {1.0, 1.0, 1.0, 1.0};
GLfloat light_position[] = {-1.0, 1.0, -1.0, 0.0};
glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

$$= K_e + K_a L_{ga} + \sum_{lights} (Spot_i)(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n}\cdot\mathbf{l}) + K_s L_{i,s}(\mathbf{n}\cdot\mathbf{h}_i)^{\alpha_i})$$

Point vs. Directional Sources

- Point light source

```
GLfloat light_position[] = {-1.0, 1.0, -1.0, 1.0};
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

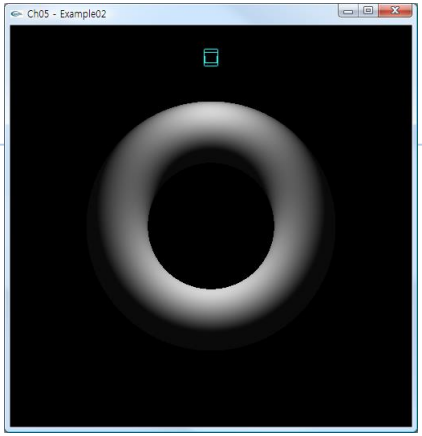
- Directional light source

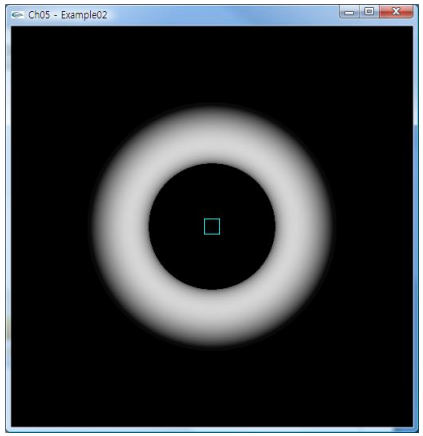
```
GLfloat light_position[] = {-1.0, 1.0, -1.0, 0.0};
glLightfv(GL_LIGHT0, GL_POSITION, light_position);
```

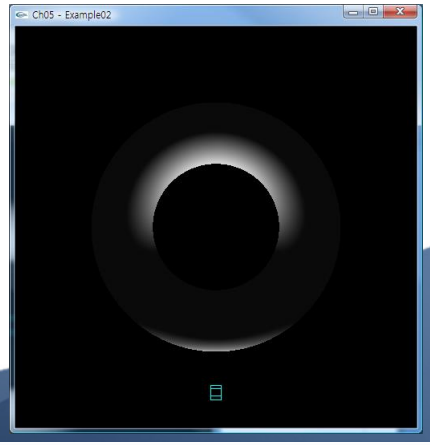
Code Example 1

```
void display() {
   GLfloat position[] = { 0.0, 0.0, 1.5, 1.0 };

   glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glPushMatrix ();
   glTranslatef (0.0, 0.0, -5.0);

   glPushMatrix ();
   glRotated ((GLdouble) spin, 1.0, 0.0, 0.0);
   glLightfv (GL_LIGHT0, GL_POSITION, position);

   glTranslated (0.0, 0.0, 1.5);
   glDisable (GL_LIGHTING);
   glColor3f (0.0, 1.0, 1.0);
   glutWireCube (0.1);
   glEnable (GL_LIGHTING);
   glPopMatrix ();

   glutSolidTorus (0.275, 0.85, 100, 100);
   glPopMatrix ();
   glFlush ();
}
```

The pos and dir of the OpenGL light source are subject to the modelview transform.





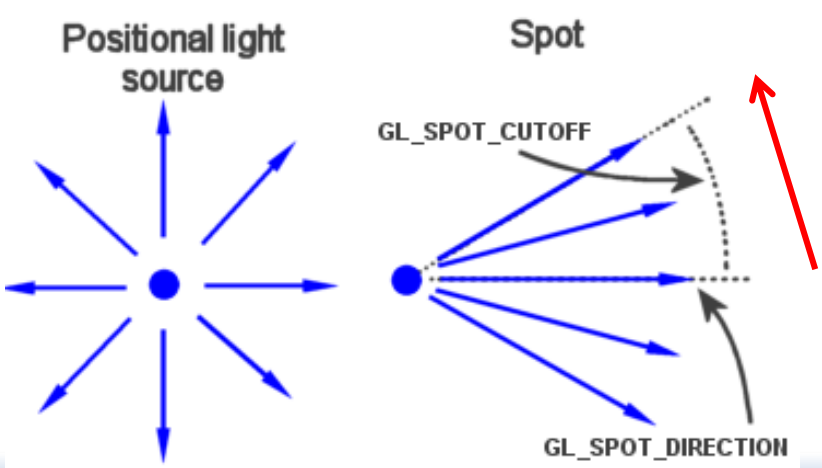


Spotlights

- Create point source as before
- Specify additional properties to create spotlight

```
GLfloat sd[] = {-1.0, -1.0, 0.0};
glLightfv(GL_LIGHT0, GL_SPOT_DIRECTION, sd);
glLightf(GL_LIGHT0, GL_SPOT_CUTOFF, 45.0);
glLightf(GL_LIGHT0, GL_SPOT_EXPONENT, 2.0);
```

$$= K_e + K_a L_{ga} + \sum_{lights} \boxed{(Spot_i)}(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n}\cdot\mathbf{l}) + K_s L_{i,s}(\mathbf{n}\cdot\mathbf{h}_i)^{\alpha_i})$$



GL_SPOT_EXPONENT: attenuation occurs based on the off-center angle

Attenuation

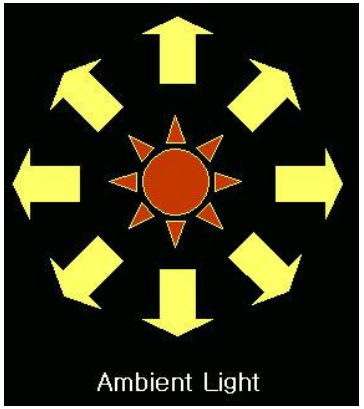
- Intensity of light decreases as distance from the light increases
  - No effect in the case of directional lights

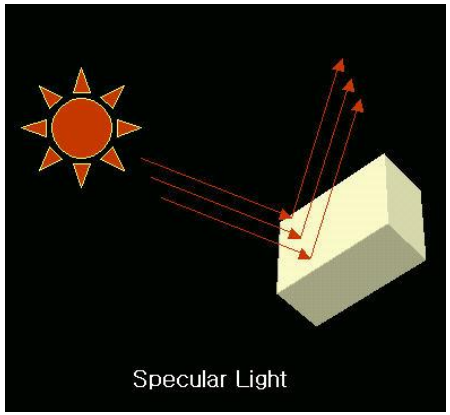
$$= K_e + K_a L_{ga} + \sum_{lights} (Spot_i)(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n} \cdot \mathbf{l}) + K_s L_{i,s}(\mathbf{n} \cdot \mathbf{h}_i)^{\alpha_i})$$

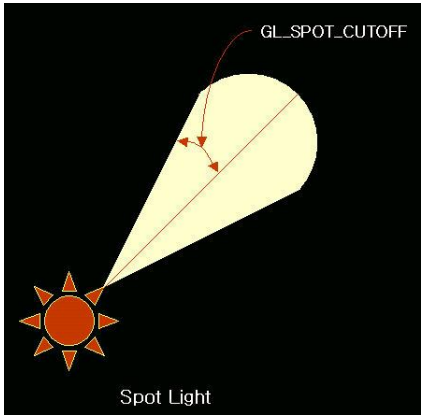
$$Att_i = \frac{1}{k_c + k_l d_i + k_q d_i^2}$$

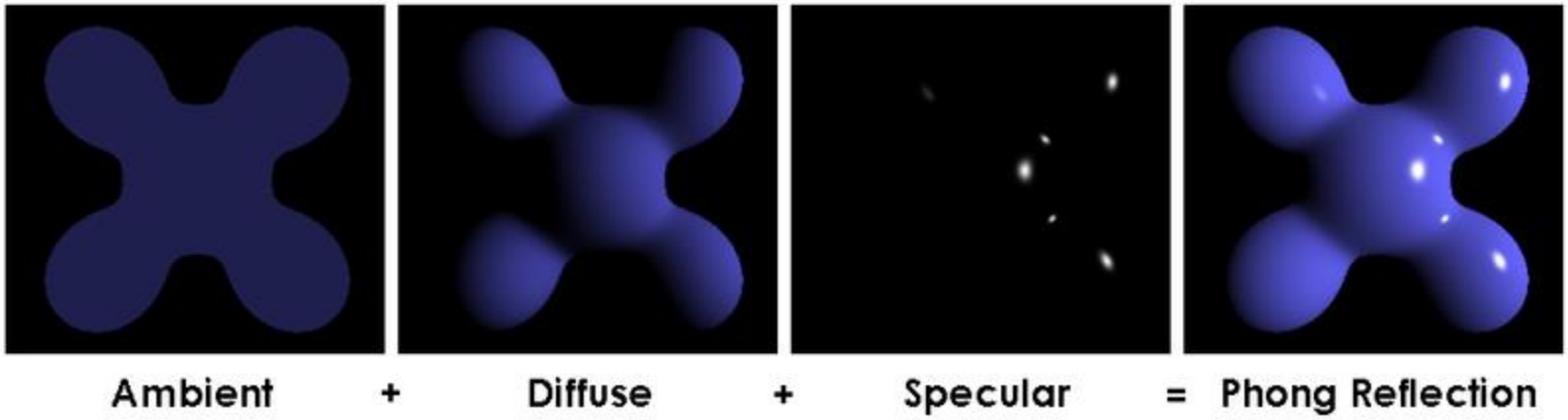
```
glLightf(GL_LIGHT0, GL_CONSTANT_ATTENUATION, 2.0);
glLightf(GL_LIGHT0, GL_LINEAR_ATTENUATION, 1.0);
glLightf(GL_LIGHT0, GL_QUADRATIC_ATTENUATION, 0.5);
```

OpenGL Lighting Model



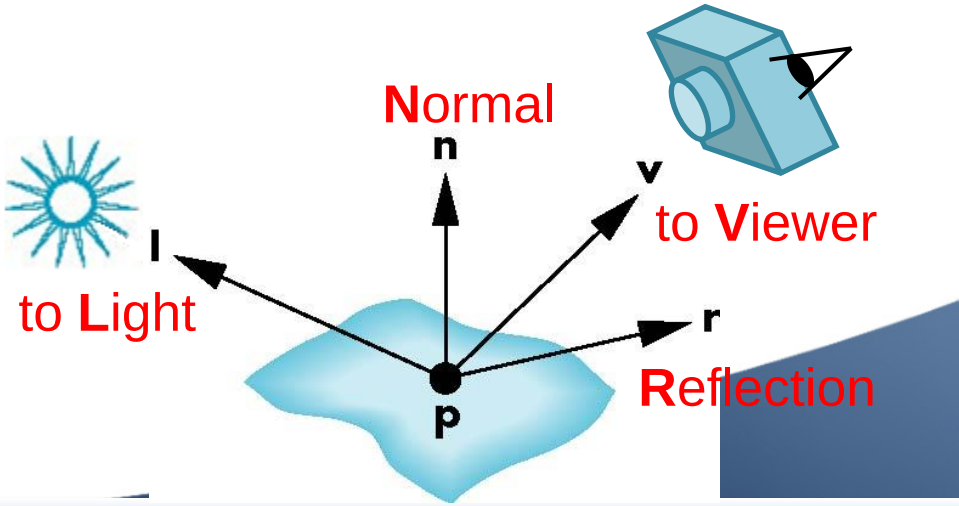






How to Shade in OpenGL?

1. Set the light sources
   - Position
   - Type: directional light, point light
   - Other Properties: ambient, diffuse, specular
2. Set the materials
   - Properties: ambient, diffuse, specular, emission, shininess
3. Set the shading method
   - Flat or Gouraud



Defining Material Properties

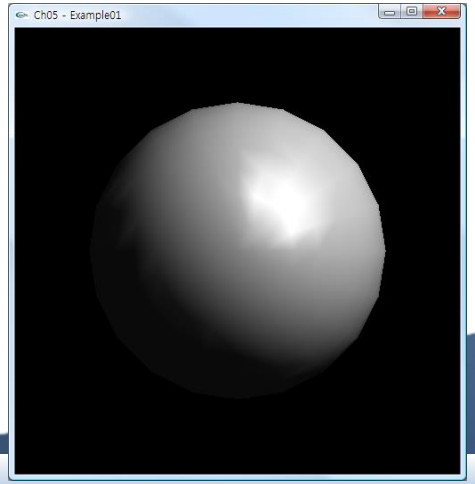
- Set both specular coefficients and shininess

```
GLfloat mat_e[] = {0.1, 0.0, 0.0, 1.0};
GLfloat mat_a[] = {0.1, 0.1, 0.1, 1.0};
GLfloat mat_d[] = {0.1, 0.5, 0.8, 1.0};
GLfloat mat_s[] = {1.0, 1.0, 1.0, 1.0};
GLfloat low_sh[] = {5.0};
glMaterialfv(GL_FRONT, GL_EMISSION, mat_e);
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_a);
glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_d);
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_s);
glMaterialfv(GL_FRONT, GL_SHININESS, low_sh);
```

$$= K_e + K_a I_{ga} + \sum_{lights} (Spot_i)(Att_i)(K_a L_{i,a} + K_d L_{i,d}(\mathbf{n}\cdot\mathbf{l}) + (F_i) K_s L_{i,s} (\mathbf{n}\cdot\mathbf{h}_i)^{\alpha_i})$$

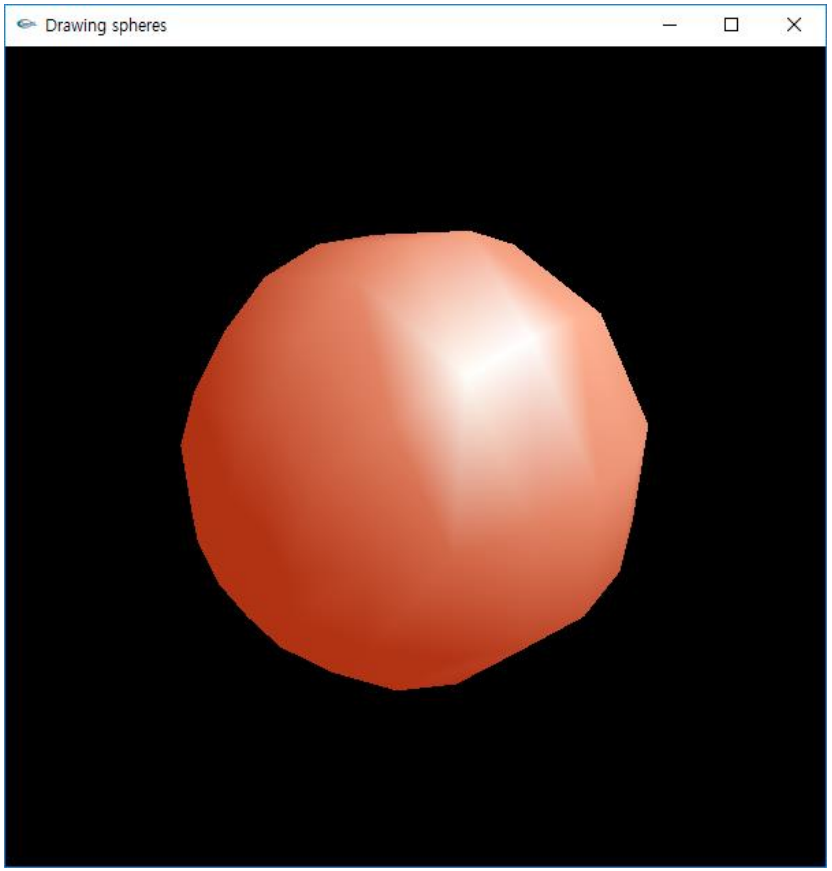
Code Example 2

```
void init() {
   GLfloat mat_specular[] = { 1.0, 1.0, 1.0, 1.0 };
   GLfloat mat_shininess[] = { 50.0 };
   GLfloat light_position[] = { 1.0, 1.0, 1.0, 0.0 };
   glClearColor (0.0, 0.0, 0.0, 0.0);
   glShadeModel (GL_SMOOTH);

   glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
   glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);
   glLightfv(GL_LIGHT0, GL_POSITION, light_position);

   glEnable(GL_LIGHTING);
   glEnable(GL_LIGHT0);
   glEnable(GL_DEPTH_TEST);
}
```

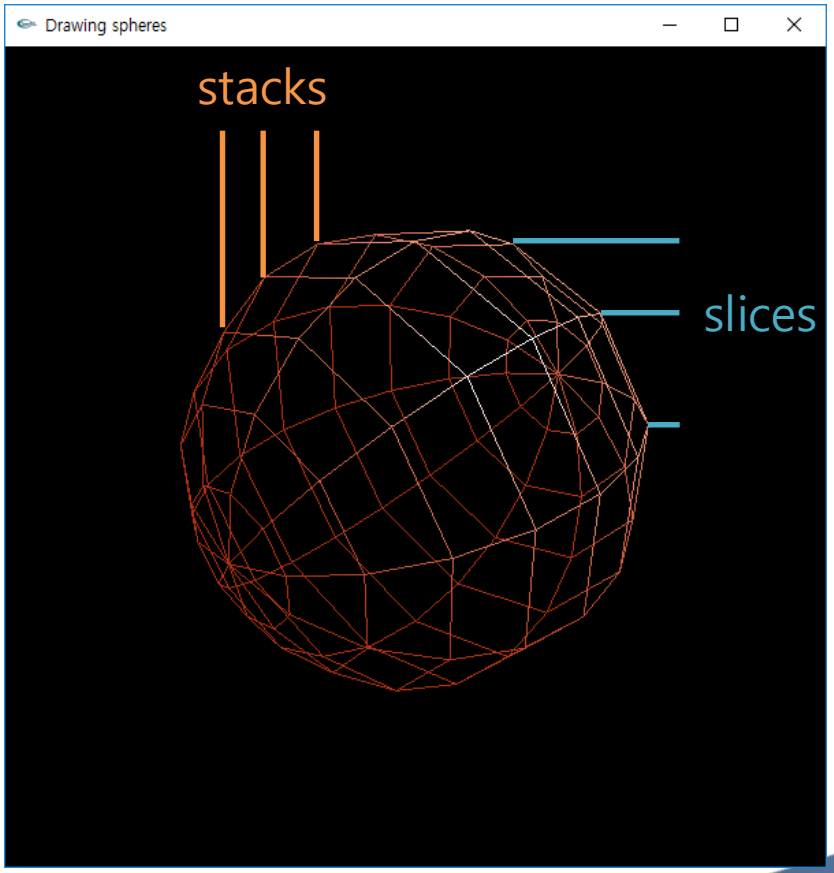


glutSolidSphere, glutWireSphere

- void glutSolidSphere(GLdouble radius, GLint slices, GLint stacks);



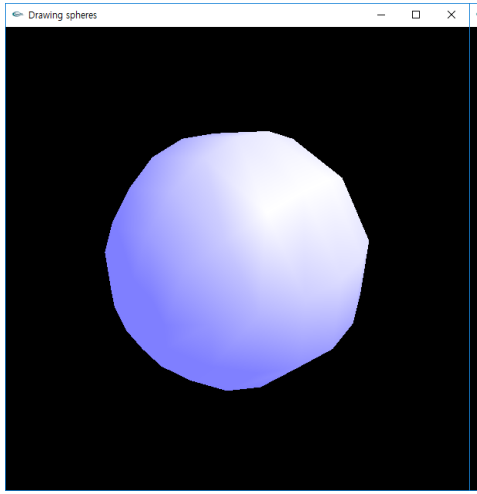
glutSolidSphere(2, 10, 10);



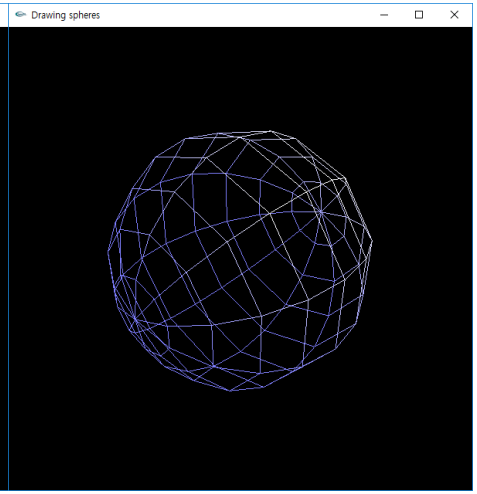
glutWireSphere(2, 10, 10);

Other shapes

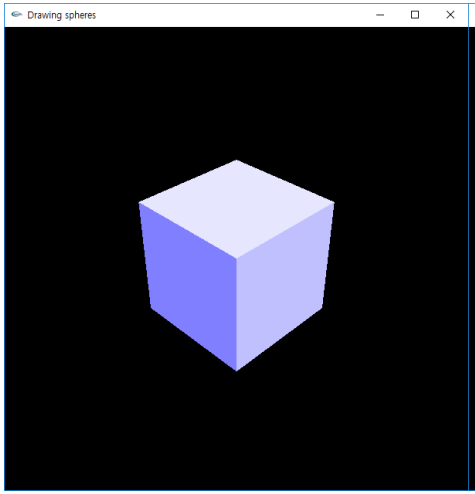
glutSolidSphere(2, 10, 10);



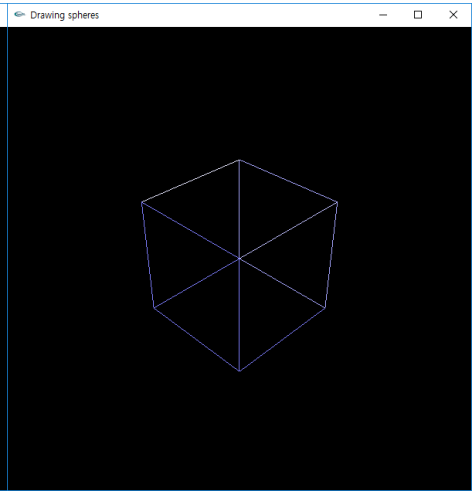
glutWireSphere(2, 10, 10);



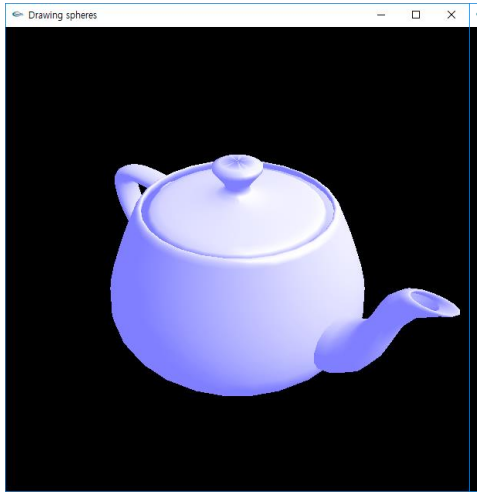
glutSolidCube(2);



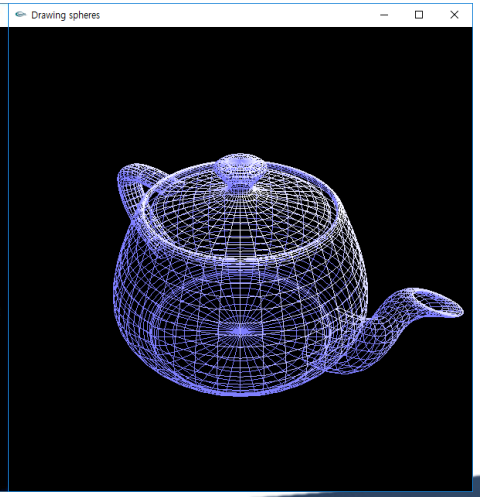
glutWireCube(2);



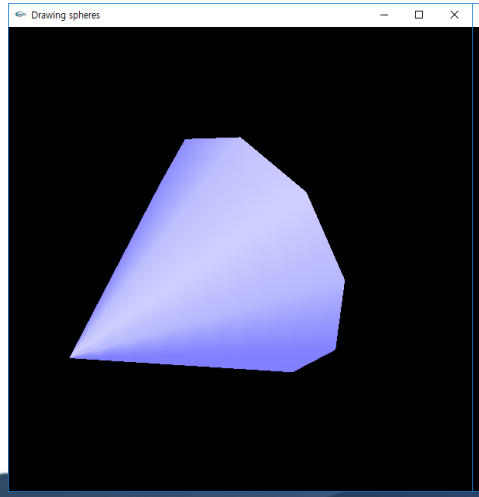
glutSolidTeapot(2);



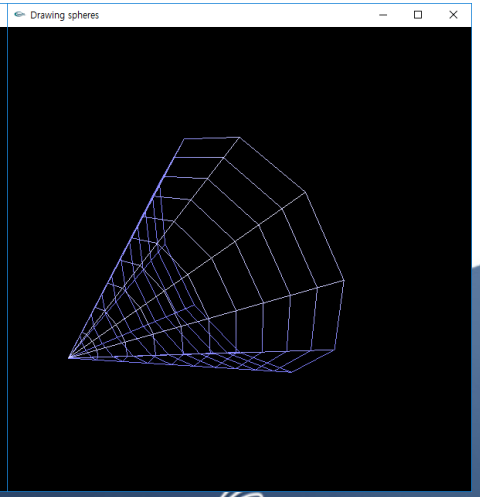
glutWireTeapot(2);



glutSolidCone(2, 3, 10, 10);

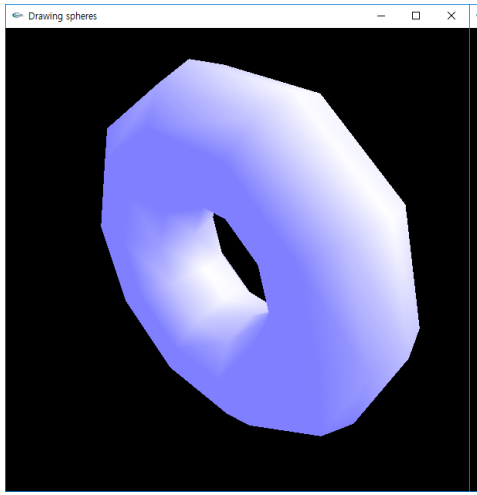


glutWireCone(2, 3, 10, 10);

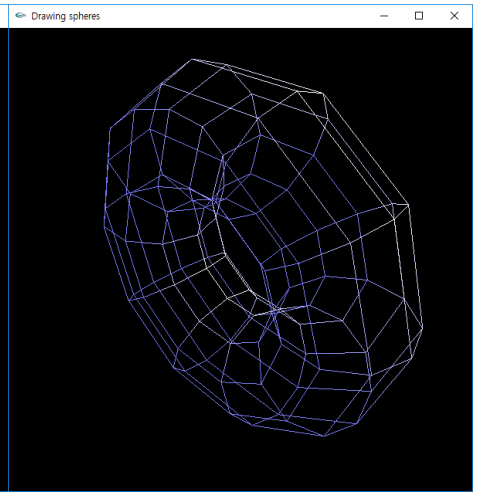


Other shapes

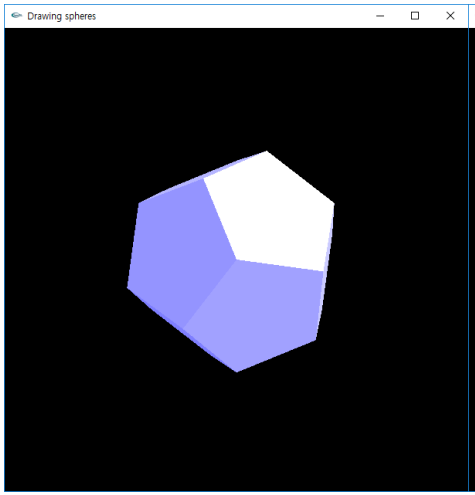
glutSolidTorus(1, 2, 10, 10);



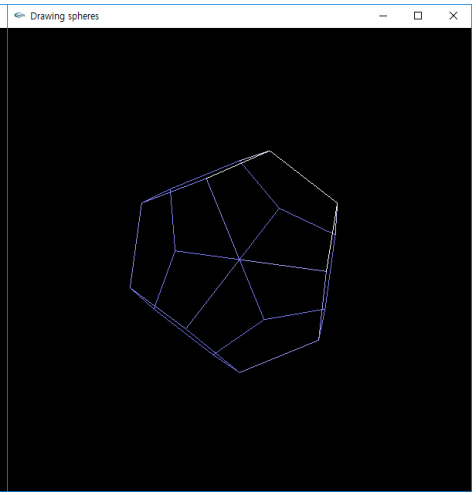
glutWireTorus(1, 2, 10, 10);



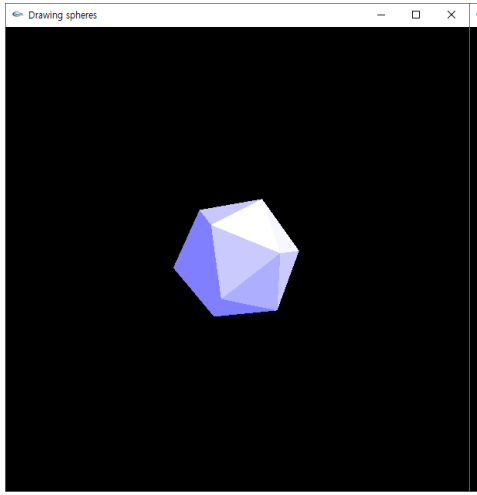
glutSolidDodecahedron();



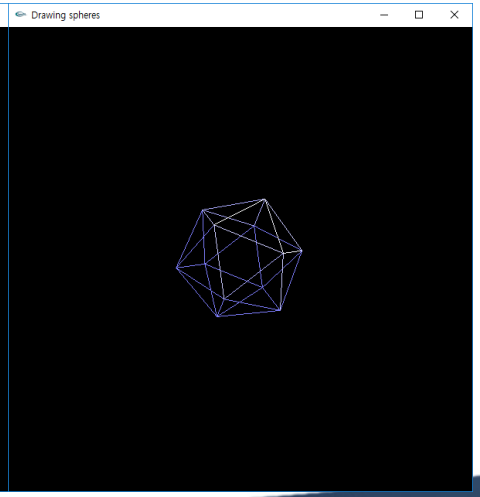
glutWireDodecahedron();



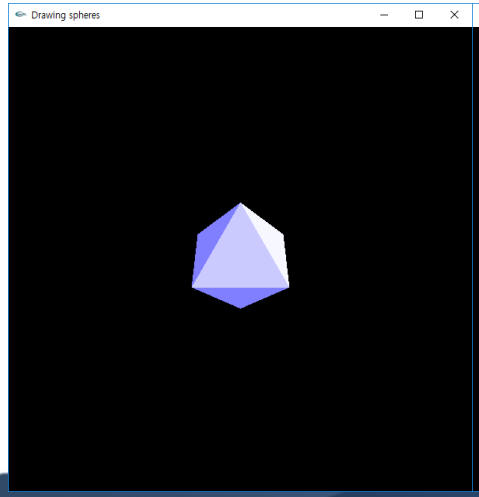
glutSolidIcosahedron();



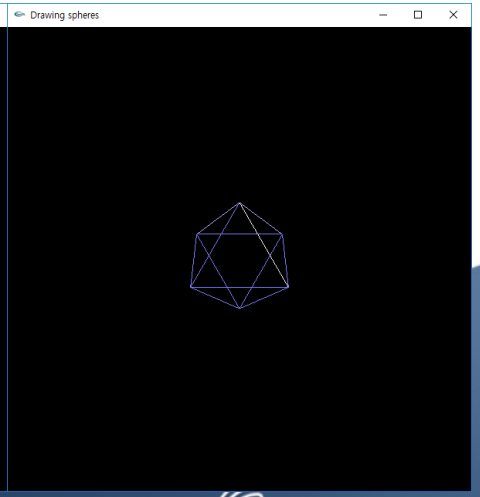
glutWireIcosahedron();



glutSolidOctahedron();



glutWireOctahedron();



Material properties

Material	Ambient	Diffuse	Specular	Shininess
Brass	0.329412 0.223529 0.027451 1.0	0.780392 0.568627 0.113725 1.0	0.992157 0.941176 0.807843 1.0	27.8974361 6
Bronze	0.2125 0.1275 0.054 1.0	0.714 0.4284 0.18144 1.0	0.393548 0.271906 0.166721 1.0	25.6
Polished bronze	0.25 0.148 0.06475 1.0	0.4 0.2368 0.1036 1.0	0.774597 0.458561 0.200621 1.0	76.8
Chrome	0.25 0.25 0.25 1.0	0.4 0.4 0.4 1.0	0.774597 0.774597 0.774597 1.0	76.8
Copper	0.19125 0.0735 0.0225 1.0	0.7038 0.27048 0.0828 1.0	0.256777 0.137622 0.086014 1.0	12.8
Polished copper	0.2295 0.08825 0.0275 1.0	0.5508 0.2118 0.066 1.0	0.580594 0.223257 0.0695701 1.0	51.2
Emerald	0.0215 0.1745 0.0215 0.55 (or 1.0)	0.07568 0.61424 0.07568 0.55 (or 1.0)	0.633 0.727811 0.633 0.55 (or1.0)	76.8
Gold	0.24725 0.1995 0.0745 1.0	0.75164 0.60648 0.22648 1.0	0.628281 0.555802 0.366065 1.0	51.2
Polished gold	0.24725 0.2245 0.0645 1.0	0.34615 0.3143 0.0903 1.0	0.797357 0.723991 0.208006 1.0	83.2

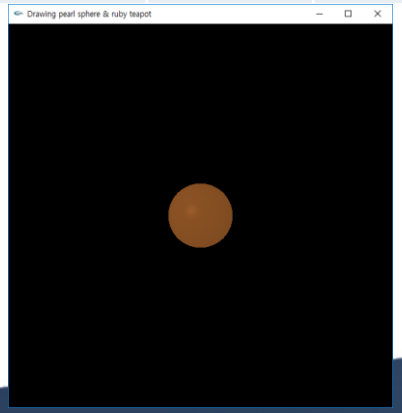
Material	Ambient	Diffuse	Specular	Shininess
Jade	0.135 0.2225 0.1575 0.95 (or 1.0)	0.54 0.89 0.63 0.95 (or 1.0)	0.316228 0.316228 0.316228 0.95 (or 1.0)	12.8
Obsidian	0.05375 0.05 0.06625 0.82 (or 1.0)	0.18275 0.17 0.22525 0.82 (or 1.0)	0.332741 0.328634 0.346435 0.82 (or 1.0)	38.4
Pearl	0.25 0.20725 0.20725 0.922 (or 1.0)	1.0 0.829 0.829 0.922 (or 1.0)	0.296648 0.296648 0.296648 0.922 (or 1.0)	11.264
Pewter	0.105882 0.058824 0.113725 1.0	0.427451 0.470588 0.541176 1.0	0.333333 0.333333 0.521569 1.0	9.84615
Ruby	0.1745 0.01175 0.01175 0.55 (or 1.0)	0.61424 0.04136 0.04136 0.55 (or 1.0)	0.727811 0.626959 0.626959 0.55 (or 1.0)	76.8
Silver	0.19225 0.19225 0.19225 1.0	0.50754 0.50754 0.50754 1.0	0.508273 0.508273 0.508273 1.0	51.2
Polished silver	0.23125 0.23125 0.23125 1.0	0.2775 0.2775 0.2775 1.0	0.773911 0.773911 0.773911 1.0	89.6
Turquoise	0.1 0.18725 0.01745 0.8 (or 1.0)	0.396 0.74151 0.69102 0.8 (or 1.0)	0.297254 0.30829 0.306678 0.8 (or 1.0)	12.8

Material properties

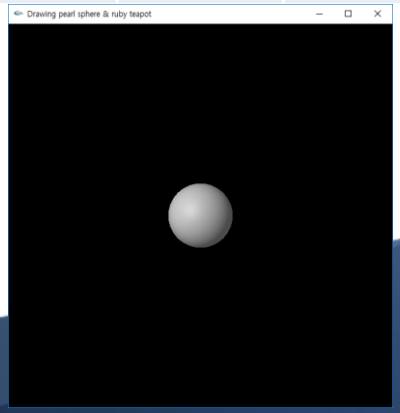
Material	Ambient	Diffuse	Specular	Shininess
Plastic (black)	0.0 0.0 0.0 1.0	0.01 0.01 0.01 1.0	0.5 0.5 0.5 1.0	32
Plastic (cyan)	0.0 0.1 0.06 1.0	0.0 0.50980392 0.50980392 1.0	0.50196078 0.50196078 0.50196078 1.0	32
Plastic (green)	0.0 0.0 0.0 1.0	0.1 0.35 0.1 1.0	0.45 0.55 0.45 1.0	32
Plastic (red)	0.0 0.0 0.0 1.0	0.5 0.0 0.0 1.0	0.7 0.6 0.6 1.0	32
Plastic (white)	0.0 0.0 0.0 1.0	0.55 0.55 0.55 1.0	0.7 0.7 0.7 1.0	32
Plastic (yellow)	0.0 0.0 0.0 1.0	0.5 0.5 0.0 1.0	0.6 0.6 0.5 1.0	32

Material	Ambient	Diffuse	Specular	Shininess
Rubber (black)	0.02 0.02 0.02 1.0	0.01 0.01 0.01 1.0	0.4 0.4 0.4 1.0	10
Rubber (cyan)	0.0 0.05 0.05 1.0	0.4 0.5 0.5 1.0	0.04 0.7 0.7 1.0	10
Rubber (green)	0.0 0.05 0.0 1.0	0.4 0.5 0.4 1.0	0.04 0.7 0.04 1.0	10
Rubber (red)	0.05 0.0 0.0 1.0	0.5 0.4 0.4 1.0	0.7 0.04 0.04 1.0	10
Rubber (white)	0.05 0.05 0.05 1.0	0.5 0.5 0.5 1.0	0.7 0.7 0.7 1.0	10
Rubber (yellow)	0.05 0.05 0.0 1.0	0.5 0.5 0.4 1.0	0.7 0.7 0.04 1.0	10

polished copper

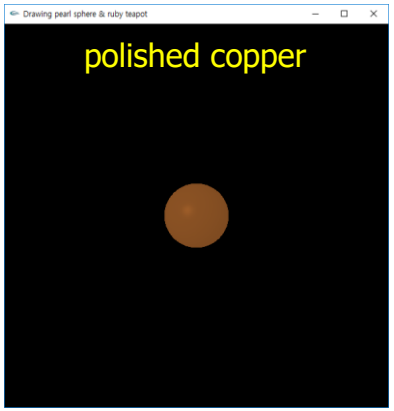


white rubber



Practice

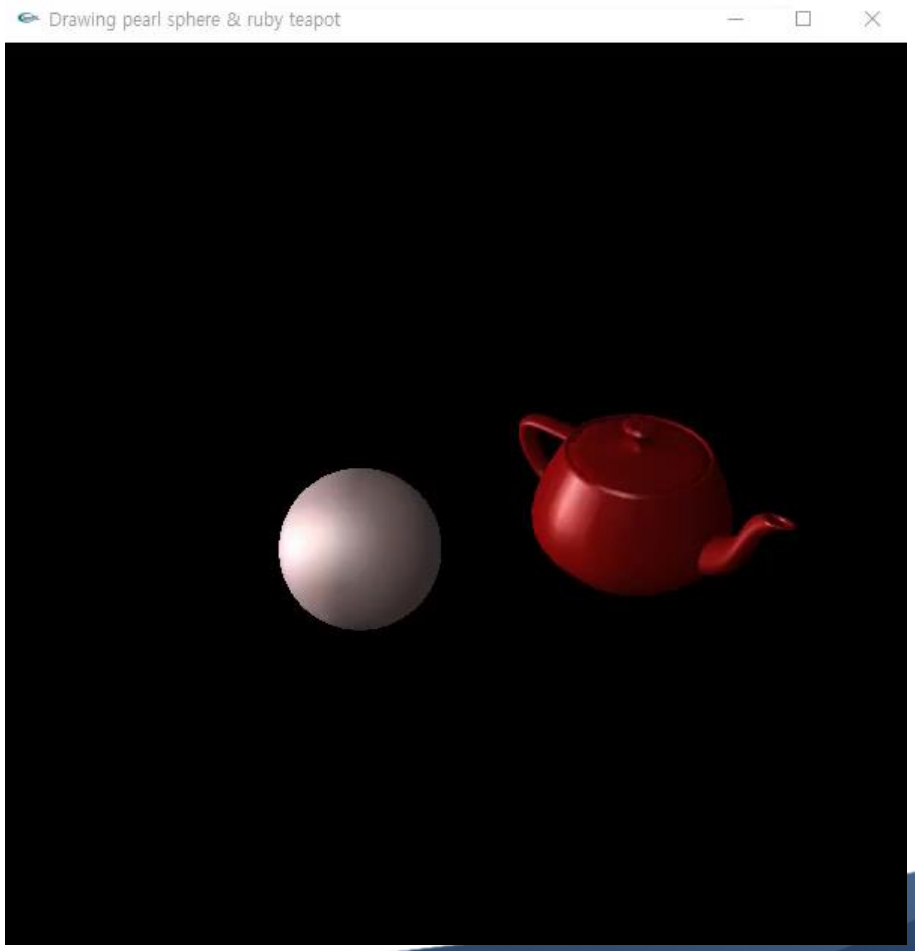
```
void init() {
        glEnable(GL_LIGHTING);
        glEnable(GL_DEPTH_TEST);

        // Material setting (polished copper)
        float mat_emission[] = { 0.0, 0.0, 0.0, 1.0 };
        float mat_ambient[] = { 0.2295, 0.08825, 0.0275, 1.0 };
        float mat_diffuse[] = { 0.5508, 0.2118, 0.066, 1.0 };
        float mat_specular[] = { 0.580594, 0.223257, 0.0695701, 1.0 };
        float mat_shininess[] = { 51.2 };

        glShadeModel(GL_SMOOTH);
        glMaterialfv(GL_FRONT, GL_EMISSION, mat_emission);
        glMaterialfv(GL_FRONT, GL_AMBIENT, mat_ambient);
        glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_diffuse);
        glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
        glMaterialfv(GL_FRONT, GL_SHININESS, mat_shininess);

        // Light setting
        glEnable(GL_LIGHT0);

        float light_ambient[] = { 1.0, 1.0, 1.0, 1.0 };
        float light_diffuse[] = { 0.7, 0.7, 0.7, 1.0 };
        float light_specular[] = { 0.5, 0.5, 0.5, 1.0 };
        float light_position[] = { 10, 0.0, 0.0, 1.0 };

        glLightfv(GL_LIGHT0, GL_AMBIENT, light_ambient);
        glLightfv(GL_LIGHT0, GL_DIFFUSE, light_diffuse);
        glLightfv(GL_LIGHT0, GL_SPECULAR, light_specular);
        glLightfv(GL_LIGHT0, GL_POSITION, light_position);
}
```




```
void renderScene() {
        // Clear Color and Depth Buffers
        glClear(GL_COLOR_BUFFER_BIT |
        GL_DEPTH_BUFFER_BIT);

        // Use the Projection Matrix
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        // Set the correct perspective.
        gluPerspective(45.0f, 1.0f, 0.1f, 100.0f);

        // Reset transformations
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        // Set the camera
        gluLookAt(25.0f, 25.0f, 25.0f,
        0.0f, 0.0f, 0.0f,
        0.0f, 1.0f, 0.0f);

        glutSolidSphere(2, 100, 100);

        glutSwapBuffers();
}
```

Today's Mission

- Shading
  - Draw a pearl sphere and a ruby teapot with two directional lights



Given & To Do

- Given
  - Global variable
    - sphere, teapot, lights
    - Rotation angle

```
Sphere sphere(0, 0, 5, 3);  // x, y, z, radius
Teapot teapot(5, 0, -5, 4); // x, y, z, size
vector<Light> lights;
float angle = 0;
```

- To Do
  - Define sphere, teapot and light class
  - Set material properties and light properties
  - Rotate only one light

To Do (details)

- To Do
  - sphere
    - center: (0, 0, 5)
    - radius: 3
    - material property: pearl
  - teapot
    - center: (5, 0, -5)
    - size: 4
    - material property : ruby

To Do (details)

- To Do
  - light1
    - cannot move
    - center: (0, 100, 100)
    - ambient: (0.1, 0.1, 0.1, 1.0)
    - diffuse: (0.3, 0.3, 0.3, 1.0)
    - specular: (1.0, 1.0, 1.0, 1.0)

    you can change
  - light2
    - rotate around y-axis
    - center: (200, 0, 0)
    - ambient: (0.5, 0.5, 0.5, 1.0)
    - diffuse: (0.5, 0.5, 0.5, 1.0)
    - specular: (1.0, 1.0, 1.0, 1.0)

    you can change

To Do (details)

- To Do
  - Set material properties and light properties

```
class Light {
public:
        Light();
        Light(float x, float y, float z, int L_ID);
        Light(const Light& Lt);
        ~Light();

        void setAmbient(float r, float g, float b, float a);
        void setDiffuse(float r, float g, float b, float a);
        void setSpecular(float r, float g, float b, float a);

        void draw() const;

private:
        int lightID;
        float *center_pos;
        float *ambient;
        float *diffuse;
        float *specular;
};
```

To Do (details)

- To Do
  - Set material properties and light properties

```
float light_position[] = { center_pos[0], center_pos[1], center_pos[2], 1.0 };
glLightfv(GL_LIGHT0 + lightID, GL_POSITION, light_position);
```

GL_LIGHT0
GL_LIGHT1 +1
⋮
GL_LIGHT7 +1