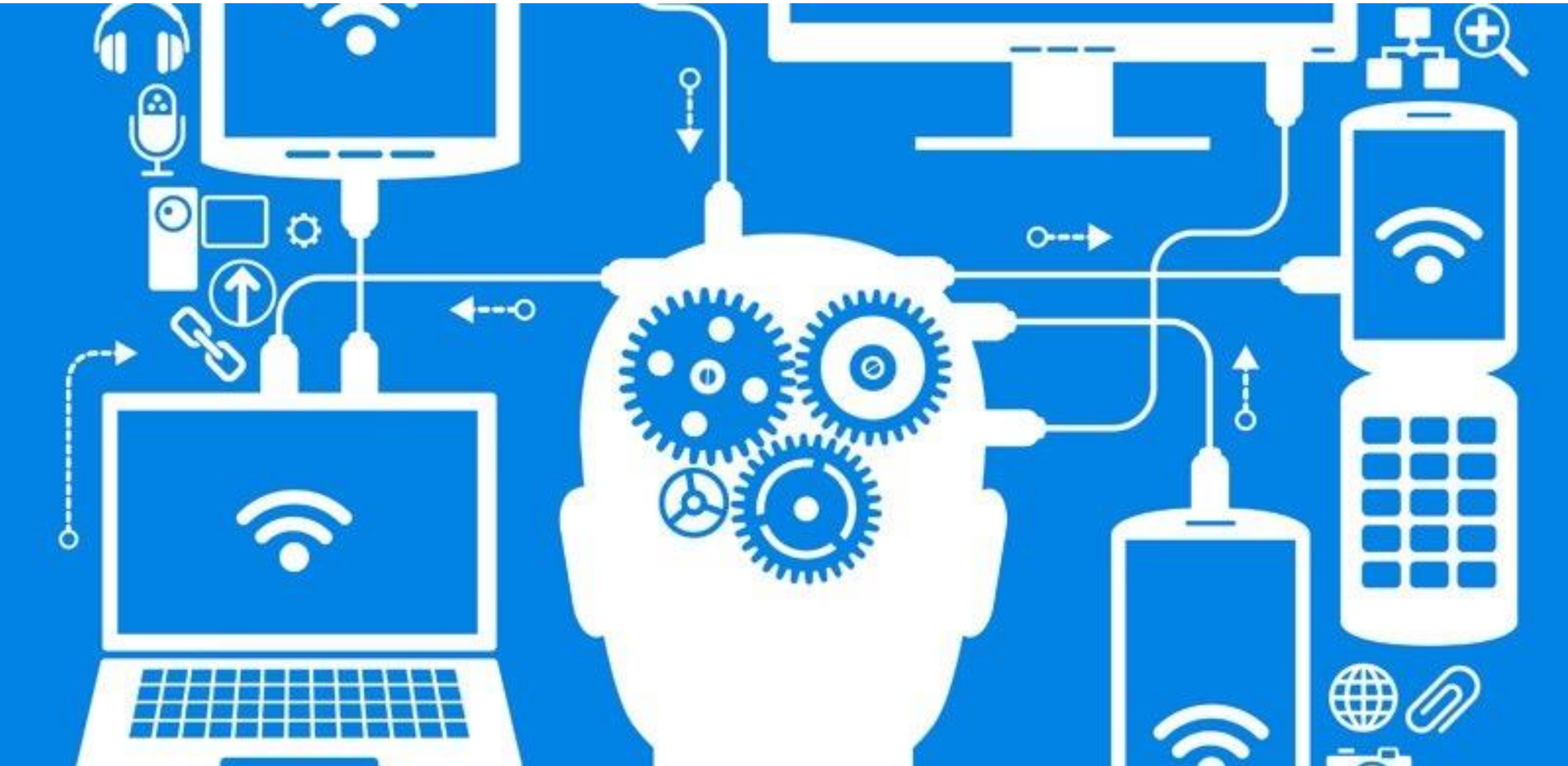
Introduction to Software Engineering

Dr. Elishai Ezra Tsur
Neuro & Biomorphic Engineering LAB

JERUSALEM COLLEGE OF TECHNOLOGY
LEV ACADEMIC CENTER



Recitations

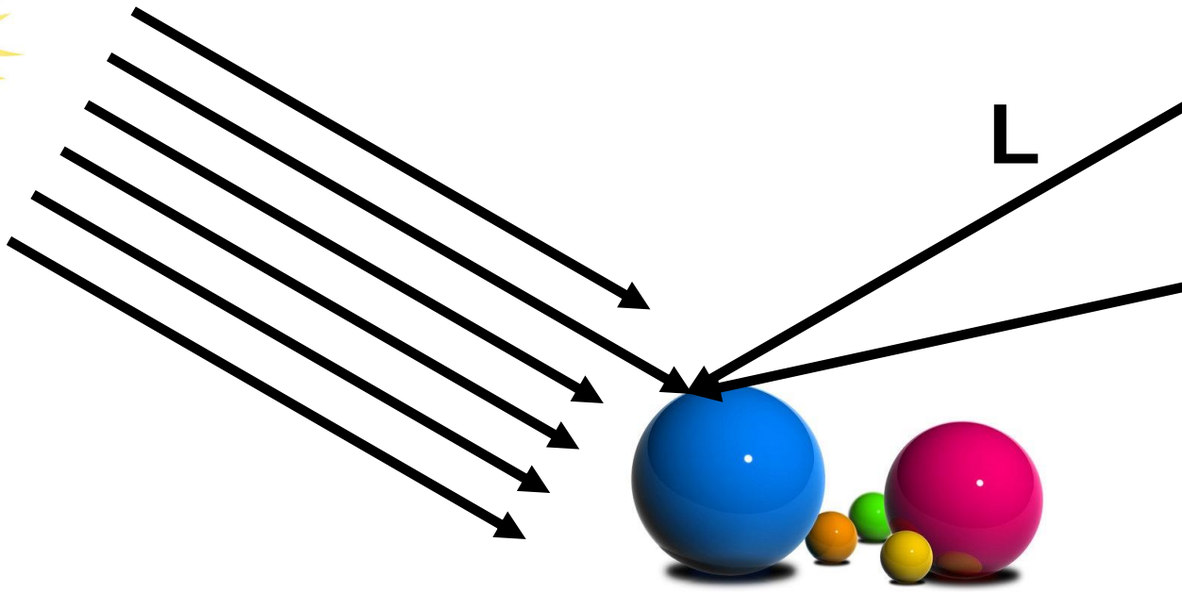
- **Week 1**: **Home assignment 0**: going through an integrated example (employees) **(not graded).** Introduction to OOP with Java, Junits.
- **Week 2: Home assignment 1**: implementation of primitives and geometries.
- **Week 3: Home assignment 2**: implementation of vector operations, camera class, and the ray construction function. Implementation of primitives and camera unit testing.
- **Week 4: Home assignment 3**: refactoring the geometry package, implementation of ray-geometry intersections and implementation of geometries unit testing.
- **Week 5: Home assignment 4**: implementation of ambient light, scene, render and image-writer classes with their appropriate test units.
- **Week 6: Home assignment 5**: implementation of naive Phong model, with directional, point and spot lighting. Adding material support.
- **Week 7: Home assignment 6**: refactoring color calculation according to Phong's model, adding support for multiple light sources
- **Week 8: Home assignment 7**: implementation of shadow rays.
- **Week 9: Home assignment 8**: implementation of ray tracing algorithm.
- **Week 10-11: Mini-project 1**: implementation of one of the algorithms described in class.
- **Week 12-13: Mini-project 2**: implementation of one of the algorithms described in class.

Directional Light

Spot Light

L

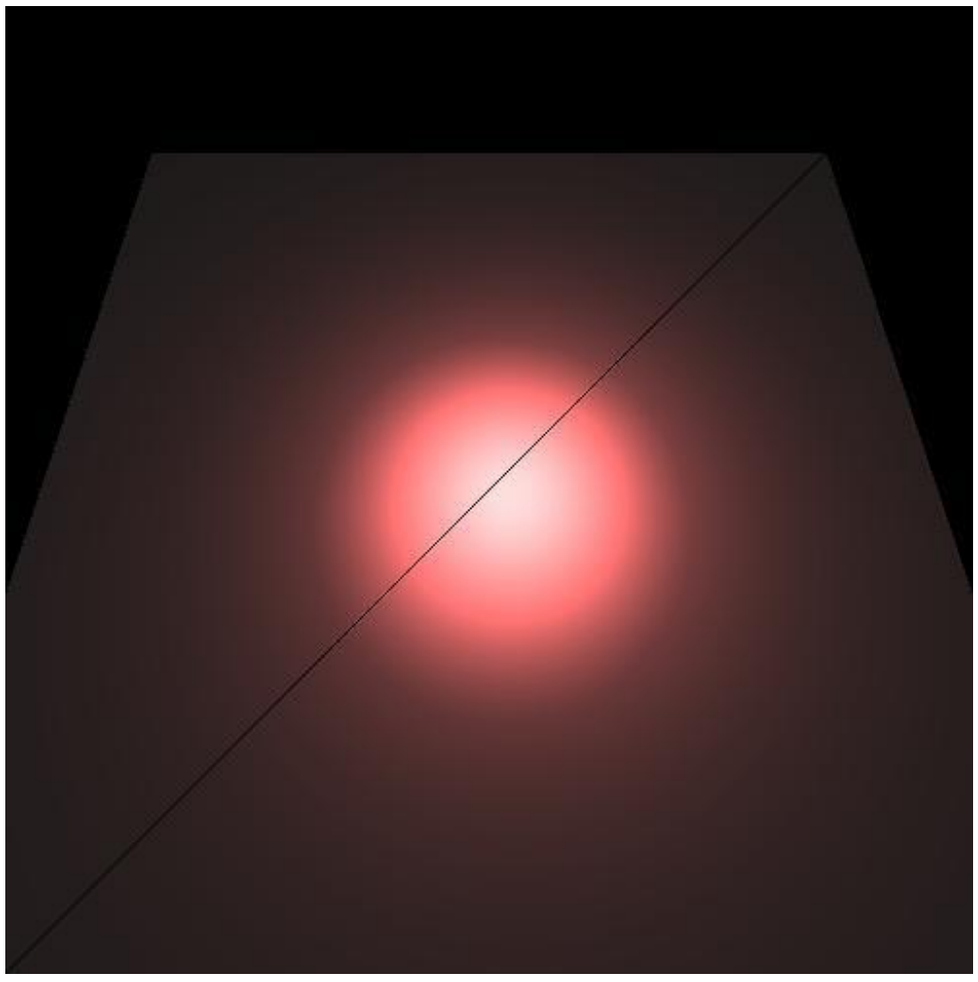
D



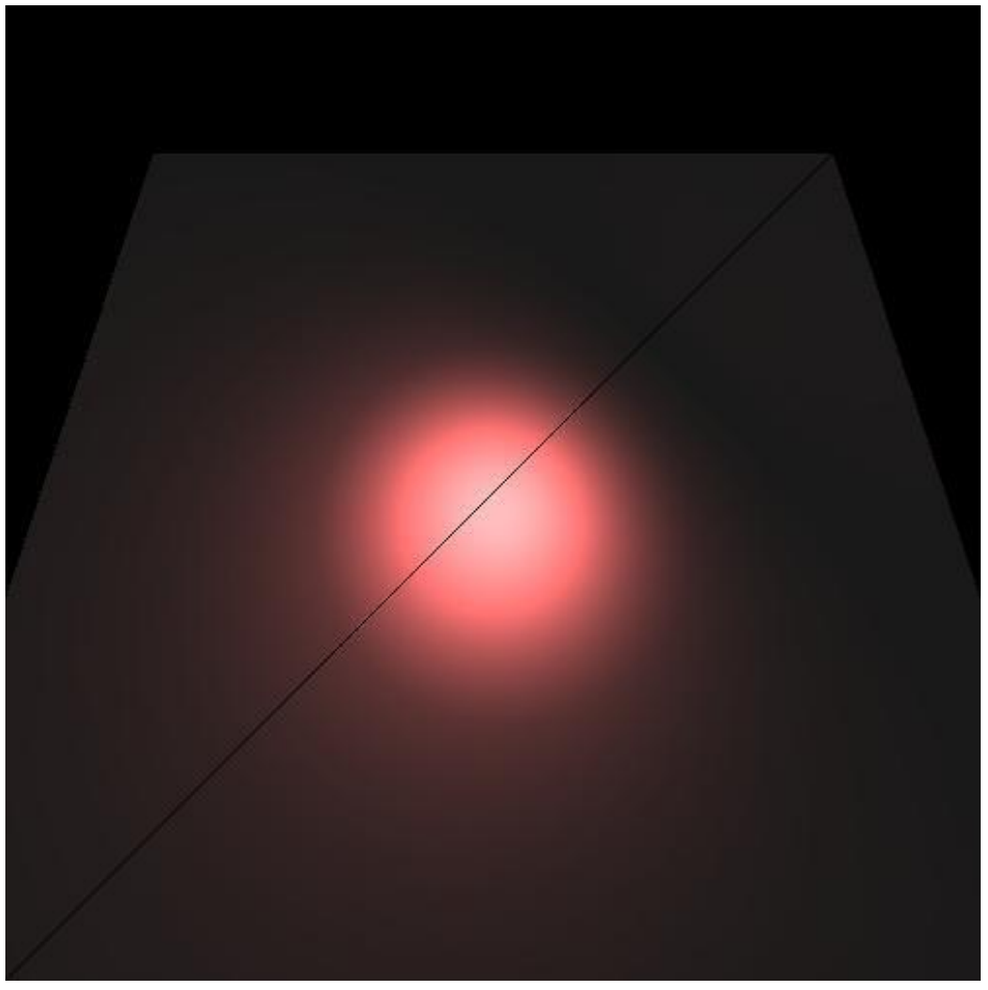
Point Light

Test Case

Point light test

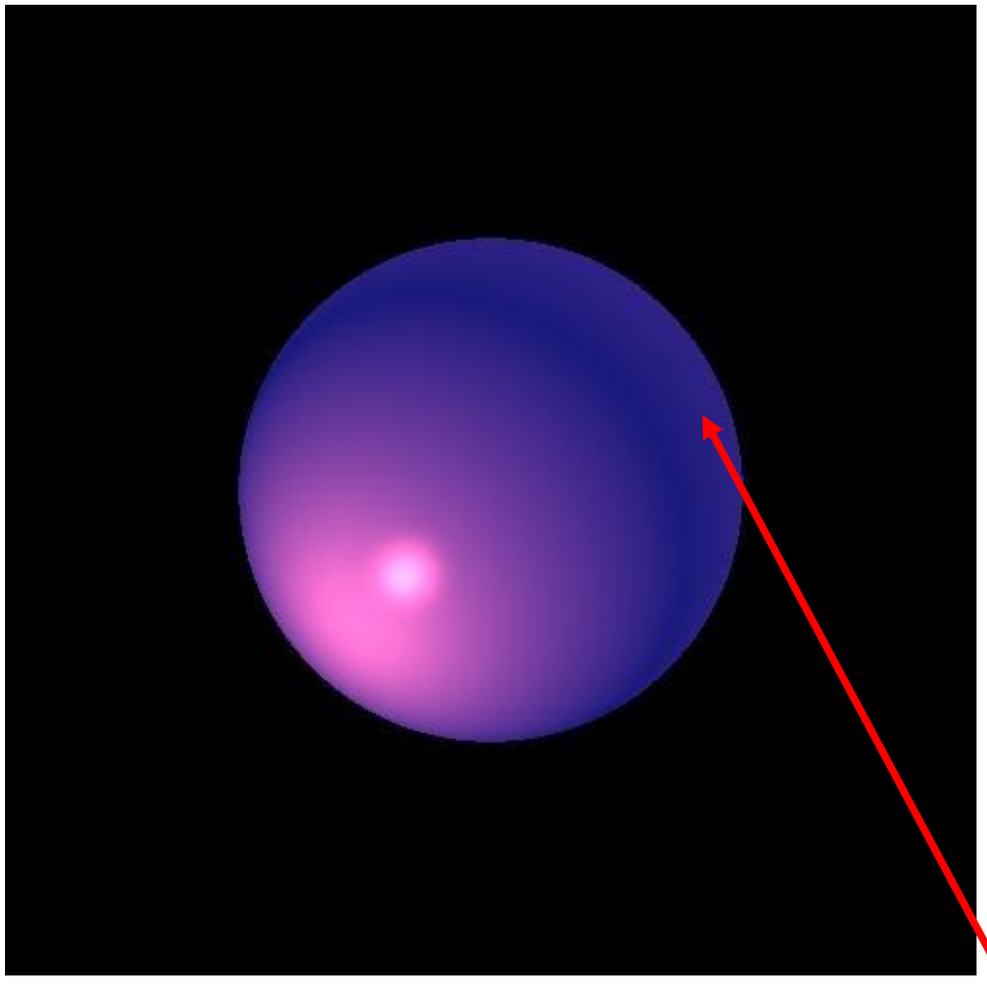


Spot light test

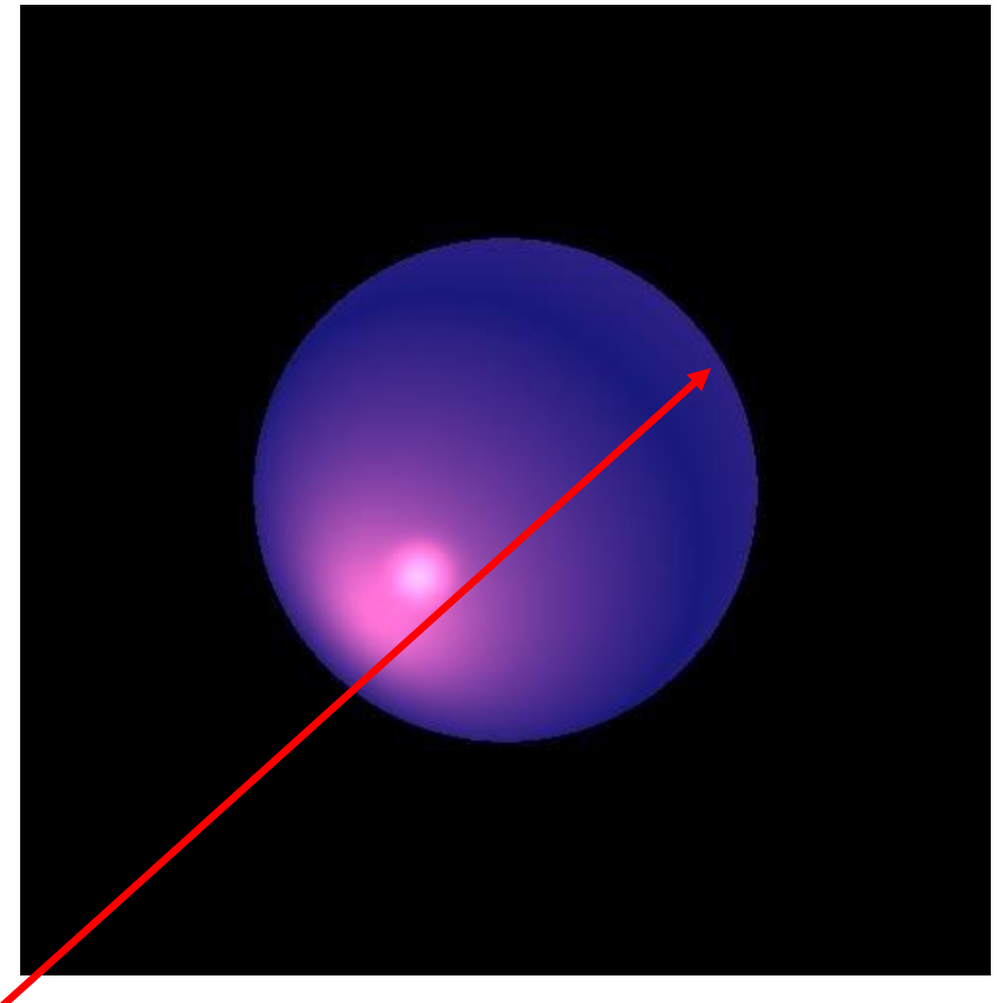


Test Case

Point light test

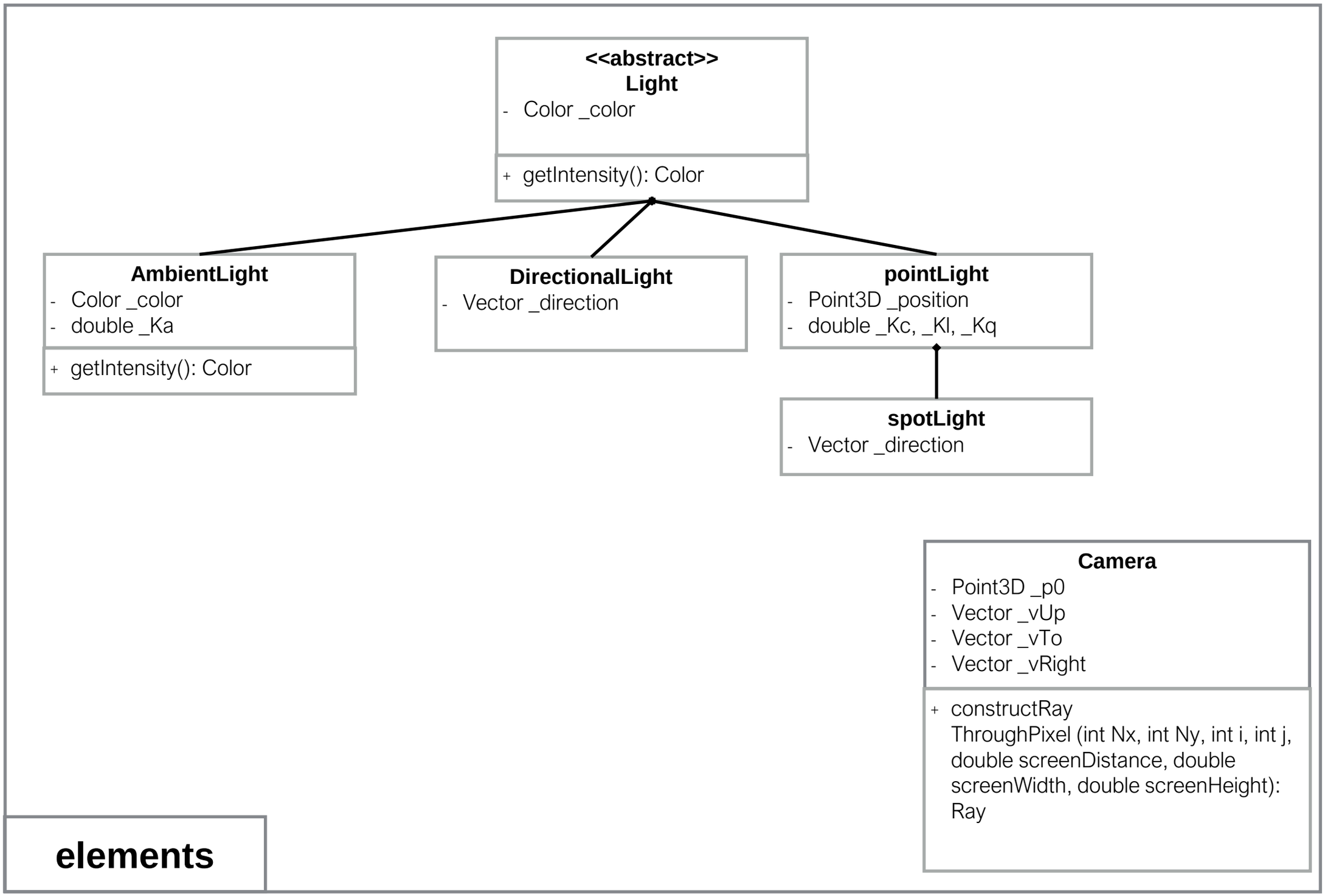


Spot light test

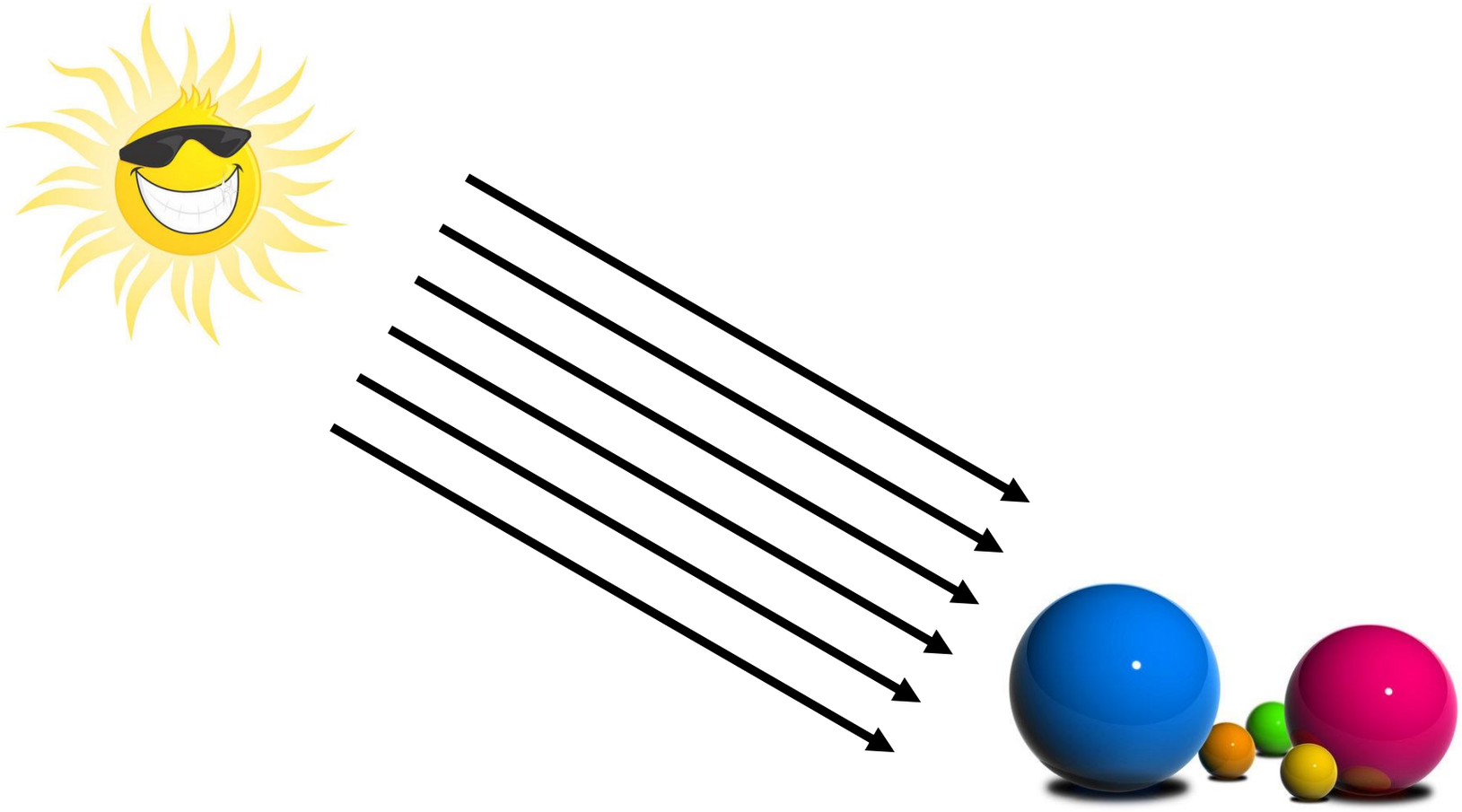


Pay attention to a bug (it should not appear in your tests…)
The issue here is that the view ray and the light source reach the sphere surface from opposite sides and it should not be lighted.

Our architectural design



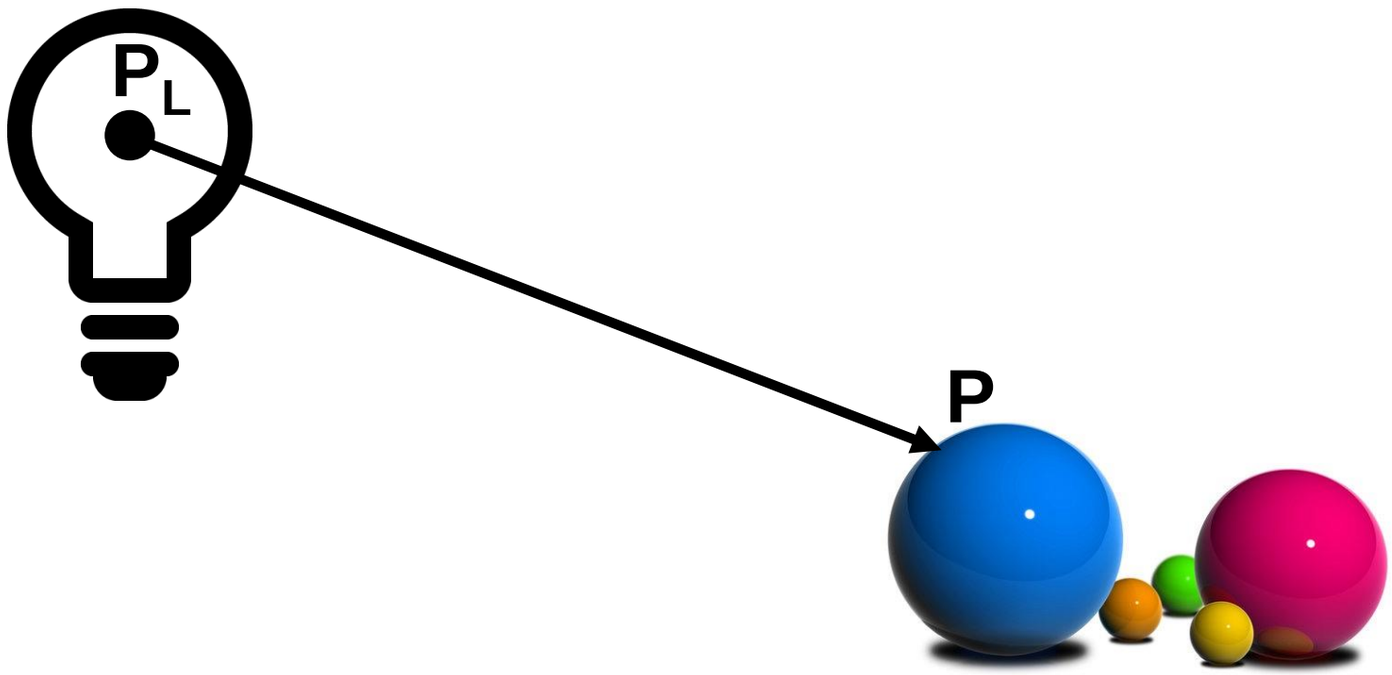
Directional Light Source



- Light source is far away (at infinity?) - like a sun)
- Intensity (I_0)
- Direction (Vector)
- No attenuation with distance

$$\boldsymbol{I_L = I_0}$$

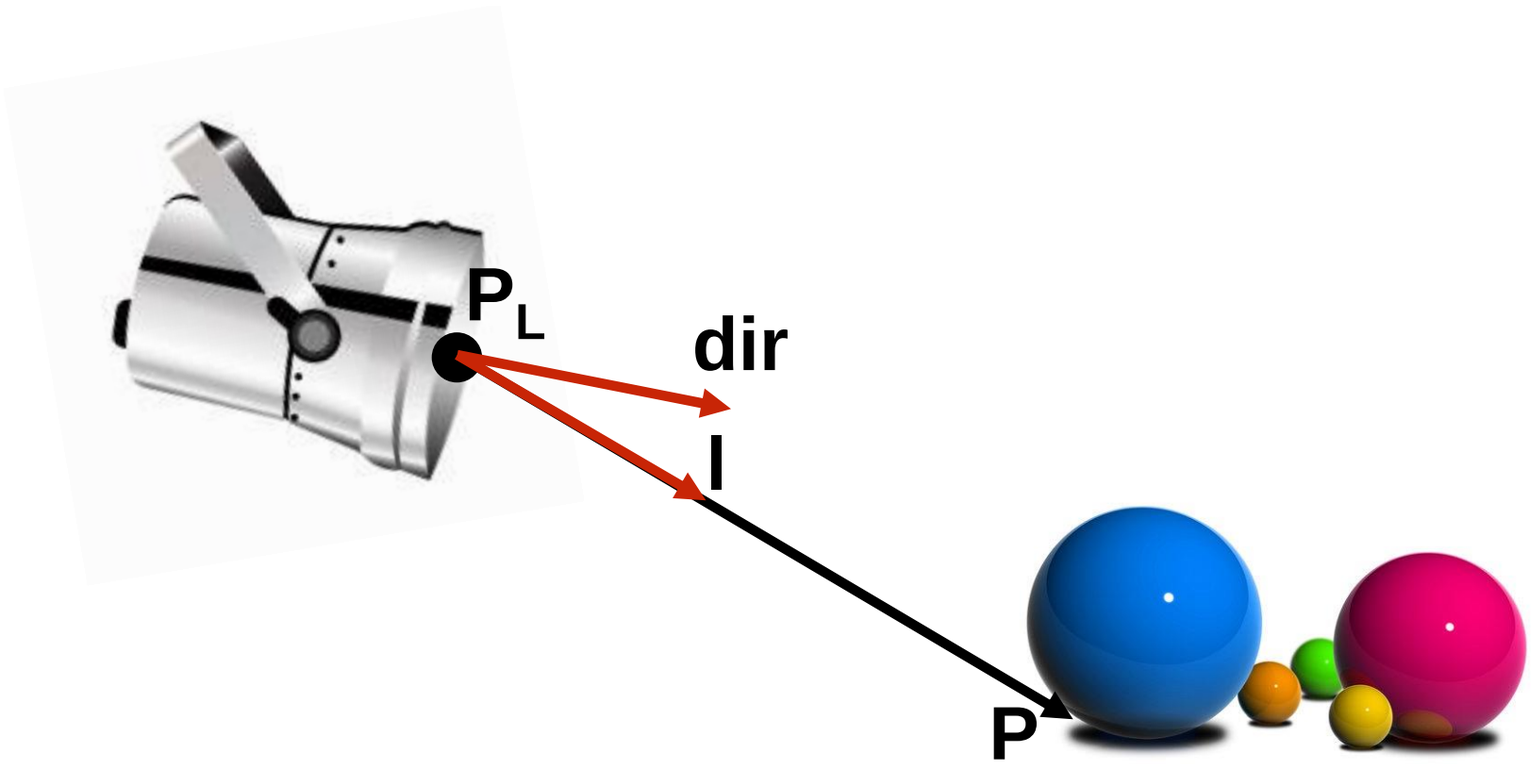
Point Light Source



- Models omni-directional point source (such as a bulb)
- Intensity (I_0)
- Position (P_L)
- Factors (k_c, k_l, k_q) for attenuation with distance (d)

$$I_L = \frac{I_0}{k_c + k_l \cdot d + k_q \cdot d^2}$$

Spot Light Source



- Models point light source with direction (such as a luxo lamp)
- Intensity (I_0)
- Position (P_L)
- Direction dir (Vector) - normalized
- Attenuation factors

dot product ($\cos\theta$)

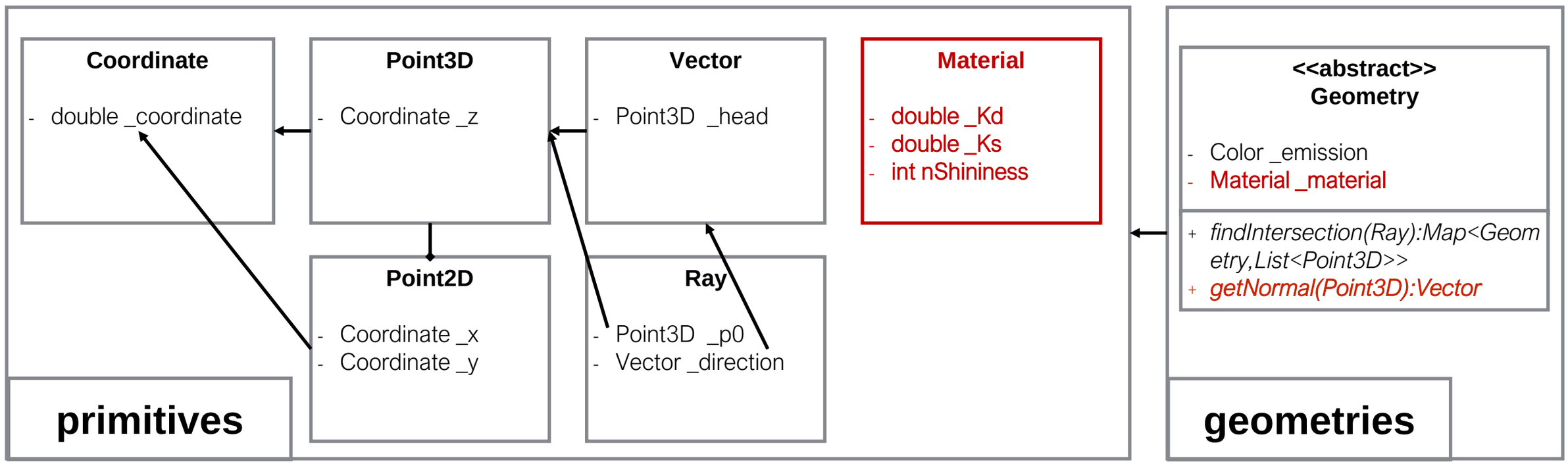
$$I_L = \frac{I_0 \cdot (dir \cdot l)}{k_c + k_l \cdot d + k_q \cdot d^2}$$

Light-geometry interaction

The Phong Reflectance Model

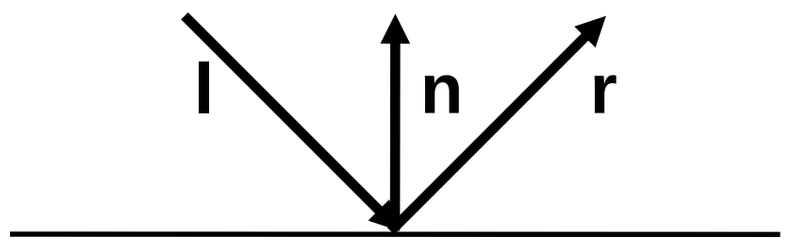
$$\boldsymbol{I_P = k_A \cdot I_A + I_E + (k_D \cdot (l \cdot n) + k_S \cdot (-v \cdot r)^{n_{sh}}) \cdot I_L}$$

Finding normals and supporting materials



Light-geometry interaction
The Phong Reflectance Model

Calculating reflectance vector:



$$\boldsymbol{r = l - 2 \cdot (l \cdot n) \cdot n}$$

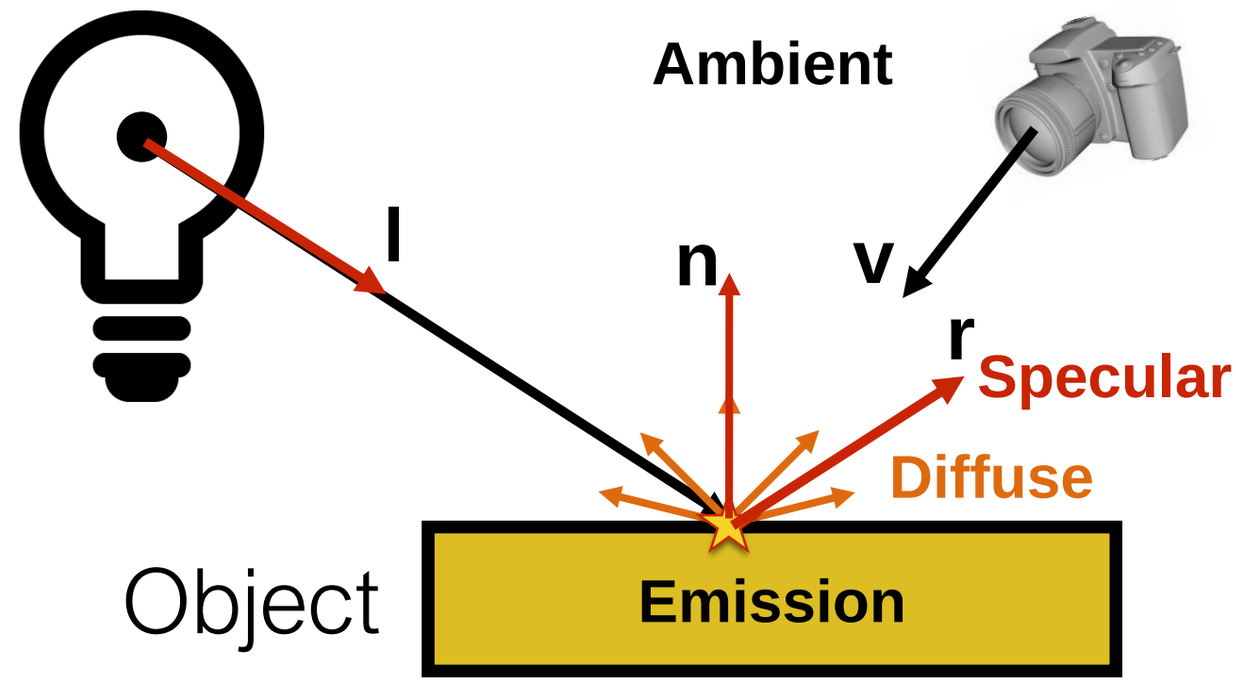
Diffusion and Specular components are seen only in the same side of the tangent surface as the light source. That is, only if the point of view is in the same side:

$$\boldsymbol{l \cdot n = v \cdot n}$$

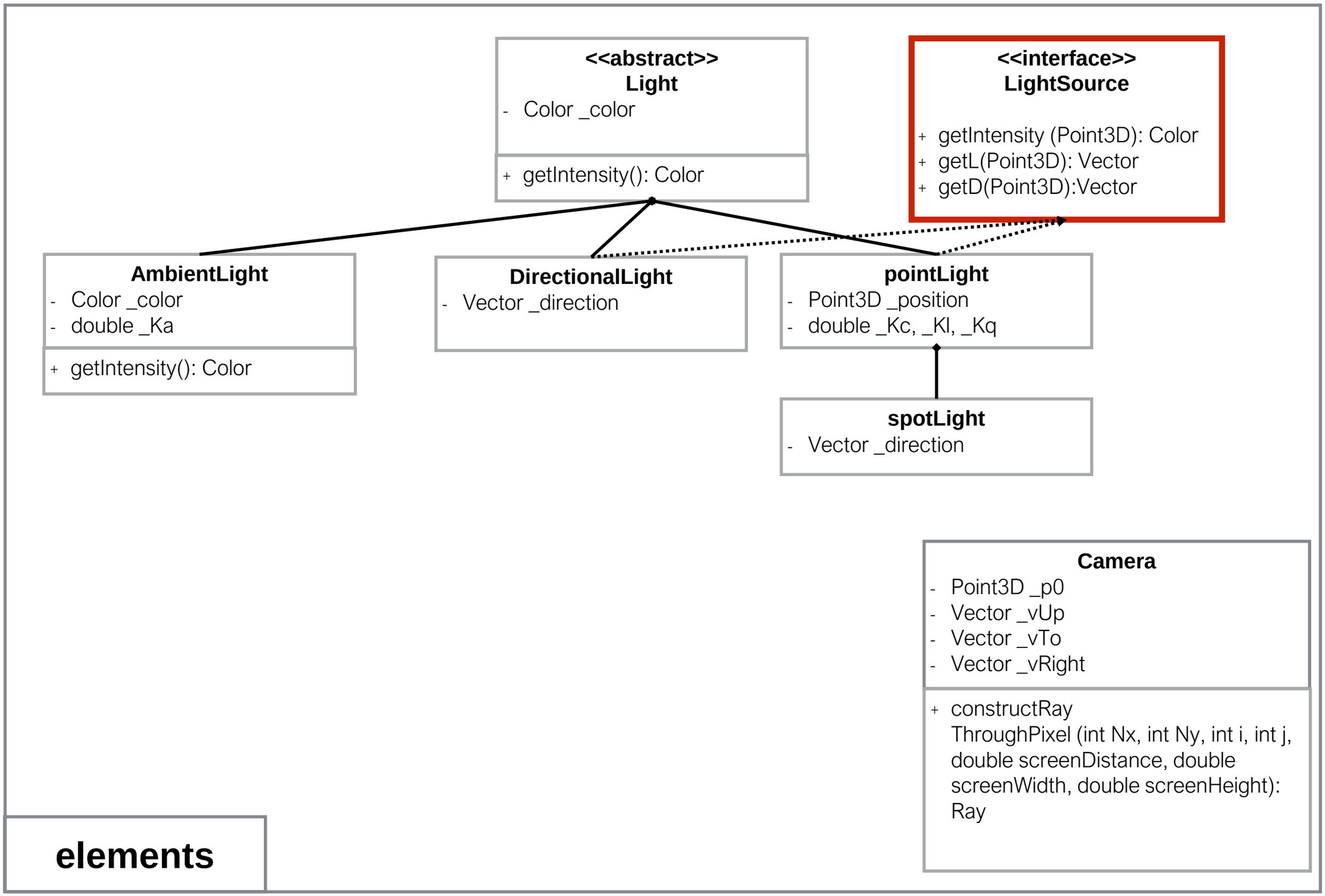
(or their signs "±" are same)

$$I_P = k_A \cdot I_A + I_E + \sum_i (k_D \cdot (l_i \cdot n) + k_S \cdot (-v \cdot r_i)^{n_{sh}}) \cdot I_{L_i}$$

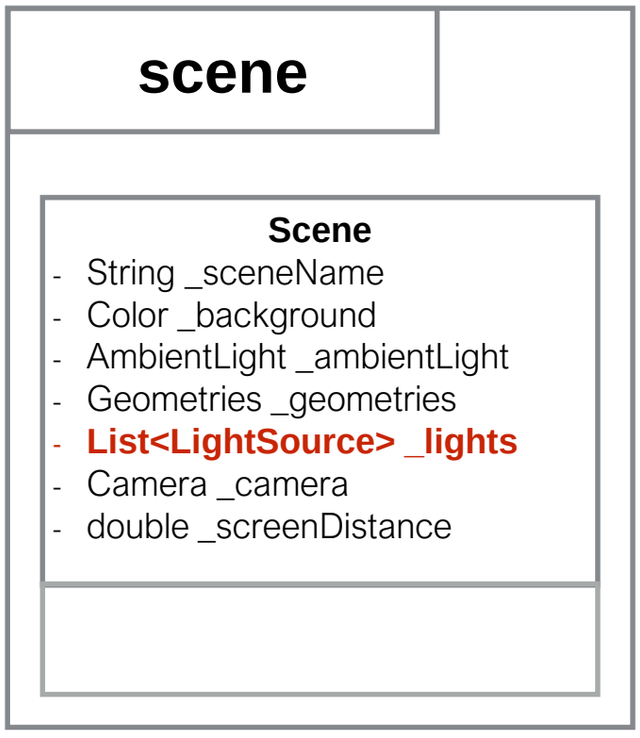
or for the general case of multiple light sources?



Our architectural design



$$I_P = k_A \cdot I_A + I_E + \sum_i (k_D \cdot (l_i \cdot n) + k_S \cdot (-v \cdot r_i)^{n_{sh}}) \cdot I_{L_i}$$



Adding diffusion/specular calculation

calcColor

```
private Color calcColor(Geometry geometry, Point3D point) {
   Color color = scene.ambientLight.getIntensity();
   color = color.add(geometry.getEmission());

   Vector n = geometry.getNormal();
   int nShininess = geometry.getShininess();
   double kd = geometry.material.getKd();
   double ks = geometry.material.getKs();

   for (LightSource lightSource : scene.getLights) {
      Color lightIntensity = lightSource.getIntensity(point);
      Vector l = lightSource.getL(point);
      Vector v = point.subtract(_scene.getCamera().getP0());
      color.add(calcDiffusive(kd, l, n, lightIntensity),
                  calcSpecular(ks, l, n, v, nShininess, lightIntensity));
   }
   return color;
}
```

Home assignment 6 Part I

Add support for Directional/Point/Spot lights.

Home assignment 6 Part II

Add Multiple Light Sources support

Home assignment 6 Part III

Add Phong model to Color calculation

Write Test cases as you see fit