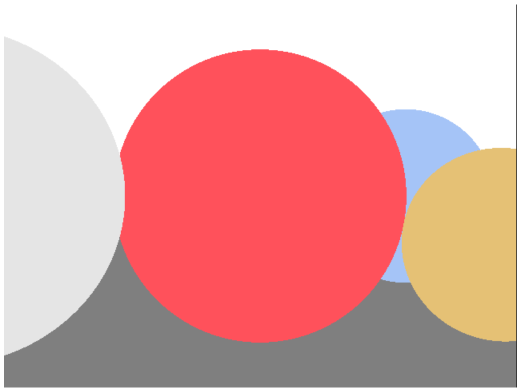
Ray Tracing

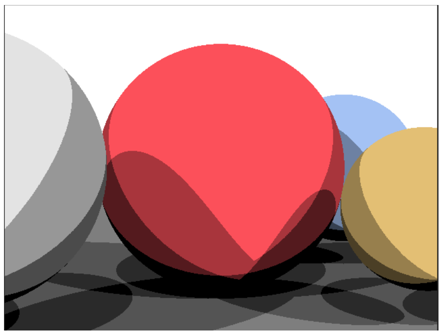
Ray Casting

For ray casting I made sure a sphere is in front of the camera by comparing the t received back from the intersect function. I then check the specific t, crossing the sphere, is not behind the camera. The pixels the spheres that can be seen by the camera will get drawn.
Before implementing the rest of the sections this resulted in the following image:



Shadow Rays

In the loop over the spheres I go over the 3 given lights, in which I again loop over the spheres to check how they intersect. Each spheres pixel starts with a black color, which I then add the spheres surface color multiplied by 0.33 (as there are 3 lights), depending on the of lights number of lights reaching each pixel. In other words, brightening said pixel. If a pixel of a sphere is blocked by another, then I break out of the loop. Which, before implementing the Phong Reflection model resulted in:

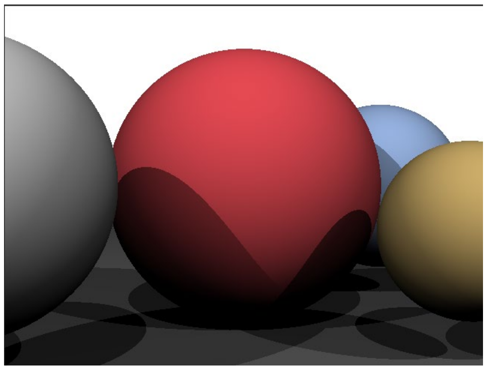


Illumination Models

For the diffuse function I used the formula given by the professor:

*0.333 * kd * max(L.dot(N), 0) * diffuseColor*

Which before the implementation of the Phong method resulted in:



Which is then used with the Phong Reflection Model, implementing the following function:

$$I_p = k_a i_a + \sum_{m \in \text{lights}} (k_d (\hat{L}_m \cdot \hat{N}) i_{m,d} + k_s (\hat{R}_m \cdot \hat{V})^{\alpha} i_{m,s})$$

Source: https://en.wikipedia.org/wiki/Phong_reflection_model

And where the reflection vector, Rm, is calculated by the formula given to us by the professor as well:

$$\hat{R}_m = 2(\hat{L}_m \cdot \hat{N})\hat{N} - \hat{L}_m$$

Lm being the vector pointing from the surface to the light source, and N is the normal of the pixel on the surface of the sphere.

Implementing the final Phong Reflection Model formula resulted in this final Image: