# CSC418 Assignment 3 & 4

Sandro Young

I completed the assignment in the order in which tasks were listed in the handout:

- First I read all the code and tried to get a handle on how things were organized
- Then I implemented the sphere and plane intersection functions
- Then I implemented proper normal & point-of-intersection calculation
- Then I implemented the phong lighting model, one component at a time
- Then I implemented shadows
- Then I implemented reflections

At each stage, I produced rendered output and compared in against the expected output. I didn't proceed to the next step until I had debugged the previous step

Most of my assignment 3 code is straightforward. The only thing I think needs to be described is my plane intersection code. The plane intersection code that I implemented works by using the same technique as the triangle intersection code from class. However, instead of allowing points where beta + gamma < 1, I allow all points where beta < 1 and gamma < 1 (this gives a complete rectangular section of plane instead of a triangle).

For assignment 4, I implemented the following features:

- Anti-aliasing
- Area light sources. I didn't use the isLightSource property, and decided not to insert area light sources as visible objects; instead, I created a utility function insertAreaLS that take a plane, creates a grid of points on that plane, and then inserts a point light source at each grid point.
- New primitive: I added a cylinder primitive
- New primitive: I added a hemisphere primitive
- Glossy reflections: I used the algorithm presented in tutorial
- Texture mapping: I implemented texture mapping for both spheres and cylinders
- Environment mapping
- Multi-threading: I used OpenMP. This was mostly straight-forward, but drand48 is not thread-safe, so I had to use erand48 and have separate random-number-generator state for each thread.

The final scene that I implemented shows off all of these features:

- It uses environment mapping to create a chapel background
- It uses texture mapping on the marble table and on the fruits
- It uses flossy reflection on the bottom of the fruit bowl
- It uses cylinder primitives on the table legs and hemisphere primitives for the bowl
- It uses an area light source as illumination
- It uses multithreading to speed up rendering
- It was rendered using anti-aliasing

I think I definitely gained a better understanding of ray tracers through this assignment. The coolest part is that once you have a basic ray-tracing framework working, adding complex new features is often almost trivial. Ray tracing is a really powerful rendering model.