Here's a summary of the building of the model for our raytracer, so far.

1. There will be a **main.c** module (Unit 7 Notes), which will call the **model_init( )** function, which is in the **model.c** module.

   A model looks like the following:

   ```c
   model_t
   *cam
   *objs
   *lfts
   *mats
   ```

2. The **model_init( )** function in the **model.c** module will:
   a. declare and malloc space for a model
   b. initialize that space to 0
   c. initialize 3 list structures for the **objs**, **lfts**, and **mats** (each list will be empty at this point)
   d. call the **model_load_entities( )** function

3. The **model_load_entities( )** function in the same **model.c** module will:
   a. call the **camera_init( )** function, which is in the **camera.c** module
   b. call the **material_init( )** function, which is in the **material.c** module
   c. call the **plane_init( )** function, which is in the **plane.c** module
   d. call the **sphere_init( )** function, which will be in the **sphere.c** module

   Each time a **material** is initialized, it’s attributes are loaded and it is added to the material list.

   Each time a **plane**, or **sphere** (or other object) is initialized, the **object_init( )** function is called, which is in the **object.c** module, which gets the material from the material list; then the **plane_init( )** function or the **sphere_init( )** function continues where it left off, and that’s where the rest of the attributes are loaded, and also where the entity is added to the proper list.

After the model is built, the raytracing begins.

1. In the **main.c** module, after the model is built using the the **model_init( )** function from the **model.c** module, the **image_create( )** function is called, which is in the **image.c** module.

   ```c
   image_create( ) calls make_row( ) in image.c
   make_row( ) calls make_pixel( ) in image.c
   make_pixel( ) calls camera_getdir( ) in camera.c
   make_pixel( ) calls ray_trace( ) in image.c
   ray_trace( ) calls find_closest_object( ) in raytrace.c
   make_pixel( ) calls camera_store_pixel( ) in camera.c
   ```

2. An algorithm for the **find_closest_object( )** function in **raytrace.c**:

   ```c
   declare a temp object
   loop while not end of list {
      get next object from object list and assign it to temp
      calculate the distance by calling the polymorphic distance function
      (example on how the polymorphic hits function is called is on page 8 from the Unit 6 Notes)
      keep track of the minimum distance as you go through each object
      advance list pointer to the next object
   }
   ```