Starter Code for Assignment 7

I am providing you with starting code for this assignment. You should edit the file `rt.cpp`, and modify the following functions:

- `setup_camera()`. This uses `eye`, `lookat`, and `vup` to initialize the VCS-to-WCS matrix.
- `rayColor()`. Create a ray, and find the first object hit. Return that object's `color` field.
- `setRay()`. Initialize the ray that begins at the given (x y) DCS coordinate. Modify the `start` and `direction` parameters.
- `first_hit()`. Test the ray against every object in the scene. If the ray hits the object, save the `Hit` object into a `vector` of hits. When done, find the closest hit, and set the `hit` parameter. Return `true`/`false` if there is/isn't a hit.

You should also modify these two files, which implement the `intersects()` method from the parent `Object` class:

- `triangle.cpp` : A colored three-dimensional triangle.
- `sphere.cpp` : A colored sphere

The `intersects()` method should return `true`/`false` if there is/isn't a hit. If there *is* a hit, it should *ALSO* set the `hit` parameter that is passed in.

Debugging

In `rt.cpp`, the function `mouse_button_callback` sets the flag `debugOn` when the user clicks on the screen. I suggest that you use this flag to trigger debug `cout <<` statements. Alternatively, if you are using an IDE, you can set a breakpoint in this function. Either way, you can trace the behavior of `rayColor` (and the functions that it triggers) for a single pixel, to better understand what your code is doing.

The Scene List

There is a `vector` called `sceneObjects` which has pointers to all the objects in the scene: 3 spheres and 2 triangles. This list is populated in the `init_scene()` function. Notice that these are pointers, so each object is created using `new`. Also, this means that testing for intersection requires a call like this: `sceneObjects[i]->intersects(...)`.