CSC 471: Program IV – Due Thursday 2/27

Introduction: Building on your prior programs slowly building up to your final project, your goal for this program is to add a free virtual camera to allow the user to explore the scene you have created. The user is allowed to move through the world (by moving the camera). The user is allowed to move forward, backward, and side to side (relative to their current gaze). **You are allowed the freedom of creativity in terms of creating and arranging your scene, but make sure there are models through out the scene (not all clustered together). You are encouraged to spend time to make a visually interesting world, especially one that will help you prepare for your final project.**

This is an individual assignment. There are a number of tutorials that closely follow the requirements for this program. Make sure you implement and **understand your own solution**.

The specifics include:

- Your program should allow the user to move around and explore the world. This will be controlled by moving the camera around in the scene. The camera can be translated using the following keyboard events:
    - "a" = move to the left of the current gaze (this should look like a translation not a rotation).
    - "d"= move to the right of the current gaze (this should look like a translation not a rotation).
    - "w"= move forward in the scene along the current gaze.
    - "s"=move backward in the scene along the current gaze.
- In addition the view can be rotated to allow the user to look around the world. Rotation of the view will be controlled using the mouse. A movement to the right with the mouse should allow the user to rotate their view to look to the left – likewise with a movement to the left. The user should be able to spin their view 360 degrees around. A mouse movement down should allow the user to look up while a movement up should allow the user to look down. Constrain the vertical movement of the view such that the user can only move a total of 160 degrees (80 degrees up and 80 down). Once the view has been rotated, all camera translations should be relative to this new view! This should allow the user to completely explore your world. You are allowed freedom to choose the exact speed of the camera motion for rotation and translation but it must be reasonable! That means a user must be able to feel like they can explore the world (i.e. not wait for lag or become confused by rapid motion).
- Include a skybox that fits your scene!
- **You do not need to include any kind of collision detection.** This means that a user can walk through your models. You are allowed to include collision detection if you choose to (if so document this in your readme).

- **Your program should include lighting (models should be correctly shaded).** You need to pick a lighting situation that you would like to emulate – i.e. think about your lighting design. Consider whether you should use a directional light or point light, or

spot light, or multiple lights. Place your lights and choose materials that are visually appealing! Please choose materials and lighting to show off your scene.

Please feel free to be creative and include as much detail as you'd like in this program, extra credit points will be awarded for extras, however, note that the bulk of the program points are for the basics so get those working first.

Feel free to think about including visual complexity in the form of complex models, animating models, texture mapping, fake shadows, collision detection, fancy shaders, terrain read in from DEM files, etc.

Point break down:

20 pts for correct camera rotation
20 pts for correct camera translation
20 pts for skybox
20 pts for good lighting and model placement
10 pts for general world construction (ie creative, interesting, good materials, etc.)
10 pts for general (code, readme, usability, etc.)

An example world: