**Design Decisions Reflection**

For my final 3D graphics project, I designed and programmed a campsite scene. This included a tent, a campfire, a pine tree, a backpack, and a moon in the sky. Each of these objects was chosen intentionally to demonstrate different graphics programming techniques and to create a cohesive, visually interesting environment. My goal was to build a scene that felt both immersive and technically rich, using a combination of simple and complex shapes, custom functions, texture mapping, and lighting effects.

**Scene Elements and Development Choices:**

* The campfire was designed to be the main visual and lighting focal point. It’s placed at the center of the scene and rendered using multi-layered geometry with additive blending to simulate glowing flames. This object helped demonstrate blending techniques and texture transparency.
* The tent allowed me to experiment with angled geometry and modular construction. I used multiple box and plane meshes to build its structure, and I applied texture mapping and custom lighting to give it realistic shading.
* The backpack was constructed entirely from primitive shapes (boxes, cylinders, and spheres), showcasing how detailed objects can be built procedurally. I added straps, a flap, a front pocket, and even a buckle using individual mesh components, each carefully scaled and positioned.
* The pine tree was composed of stacked cone meshes, and it added natural context to the scene. I used texture and scale variation to give the tree a layered, organic feel.
* Finally, the moon serves both a visual and functional purpose. It appears in the sky to help define the scene’s time of day, and it also acts as a directional light source, casting subtle illumination across the environment.

These choices were all based on a combination of design goals and technical goals: to demonstrate object construction, material usage, lighting, and modular code structure.

**Navigation and Camera Control:**

To interact with the scene, I implemented a custom camera system controlled by both the keyboard and mouse. The user can navigate the environment in a first-person or free-fly style using:

* W and S to move forward and backward
* A and D to strafe left and right
* Q and E to move vertically
* Mouse movement to rotate the camera view
* Scroll wheel to zoom in and out
* O and P to toggle between perspective and orthographic projections

The camera movement is based on delta time, ensuring that movement speed remains consistent regardless of frame rate. This system is implemented in a custom ViewManager class, which processes input, updates the view matrix, and handles mode switching. The goal was to give the user smooth and intuitive control over the scene’s viewpoint.

**Custom Functions and Code Modularity:**

To keep my code clean and modular, I wrote several custom functions and used lambda functions for object rendering within the scene.

One notable utility is DrawRopeBetweenPoints(), which draws a cylinder mesh between any two points in 3D space. This function is fully reusable and helps in situations where I need to connect objects — such as tent ropes or visual lines between nodes. This allowed me to encapsulate the rendering logic for each object, keeping the code organized and easy to edit or duplicate.

**In general, my rendering code follows a modular structure:**

* Each object’s geometry is grouped in its own block
* Textures and materials are set before drawing
* Transformations are isolated per object

This approach makes it much easier to manage complex scenes and test different components individually. It also supports future extensibility, as objects or features can be moved into separate files or reused in new scenes.

**Conclusion**

This project helped me build a solid foundation in graphics programming with OpenGL. I learned how to translate a visual idea into code through geometry, transformation, texture mapping, and lighting. While I found the lighting system to be the most challenging aspect, especially understanding how different light types affect a scene, I feel much more confident now. The modularity of my code and the flexibility of my camera system were two parts I’m especially proud of.

These skills could be applied in fields like simulation, game design, or even data visualization. Whether or not I go into graphics professionally, I now have a much better understanding of how 3D engines work at a lower level, and I’m excited to build on that knowledge in future projects.