**Justify development choices for your 3D scene.**

**For my program, I went with objects that were intentionally easier. This wasn’t because I don’t think I can handle more complex objects, but it was because I was using an uncomfortable language, IDE, and library, I didn’t want to complicate things further. I think I learned quite a lot from the objects I chose. This being a 3D plane, Cylinder, Sphere, and a rectangle.**

**Explain how a user can navigate your 3D scene.**

**Users can navigate the scene through a few simple methods. There’s the typical WASD, which allows them to move around naturally. W -> forward, A -> left, S -> back, and D -> right. I threw in 2 additional key usages, such as Q and E. Q -> up, and E -> down. This allows the user to navigate easily through the world. On top of this the user can move their mouse to view the scene more easily and can also use their scroll wheel to zoom in and out.**

**Explain the custom functions in your program that you are using to make your code more modular and organized.**

**To start off, one of the major functions in the application is the “UProcessInput” function. This is what takes the users input from the keyboard or mouse and produces some specific output in the game. This function makes things very organized because anything can be passed through super easily, and only a small line needs to be added to support a new specific parameter input. The only downfall of the function in its current state is how its size is very linear and will only get larger as more input support is added. This could be improved upon.**

**Another major function in the project is the “URender” function. This is sort of where the “magic” happens. This is how everything is officially rendered, and it’s run constantly in the game loop. Specifically, this is what sort of spawns each object, this function is hosting support for all 4 objects. However, like the other function I mentioned, this one is a bit flawed as well. Not in the fact that it doesn’t work, but in the fact that it’s a bit too generic and fits too many things. A better way to do it would be to have a render function per object type, making things a lot more modular. I don’t think this is a huge deal though for this program, as it’s small enough to not really matter.**

**There are honestly too many functions to talk about all of them, but I’d say the last most important one would be the UCreateMesh function. This runs before the URender function does, and essentially initializes all the mesh objects. In this case there’s 3 mesh objects, one for each type of object.**

**Overall, I’d say I’m happy with how everything turned out. It could definitely be a lot better, but I’ll try not to let perfection be my enemy on this one.**