**CS 330 Final Project Reflection**

**1. Justification for Development Choices in the 3D Scene**

Table: A table serves as the foundational object in many workspace scenes. The choice of including a table helps ground the other objects. It acts as a platform for placing other objects, creating a relatable and familiar environment for most users.

Monitor: A monitor is an essential part of any computer setup. By adding a monitor, the scene immediately becomes recognizable as a workspace or an office setting, helping users to relate and understand the purpose of the environment.

Keyboard: Like the monitor, the keyboard is a crucial component of a computer workspace. Its inclusion makes the scene more detailed and lifelike.

Computer: This is the core of the setup. By adding a computer tower, we give context to the monitor and keyboard, completing the look of a functional workspace.

These objects were chosen to offer a holistic and realistic representation of a typical computer workspace. They are relatable and can be easily identified by a broad audience, making the 3D scene more engaging and understandable.

**2. Navigation in the 3D Scene**

Navigating the 3D scene is made possible through a virtual camera. The camera is set up to respond to user input, allowing them to view the scene from different angles and distances.

Keyboard Controls: The arrow keys (or WASD keys) can be used to move the camera forward, backward, left, or right. The E and Q key serve as up and down camera movement.

Mouse Controls: Dragging the mouse rotates the camera around the scene. Scroll wheel functionality allows for manipulating the movement speed of the camera.

These control options ensure that users can explore the 3D environment seamlessly.

**3. Custom Functions for Modular and Organized Code**

processInput: This function processes user input through the GLFW library. It handles keyboard inputs for various tasks like moving the camera and toggling perspectives.

Purpose and Reusability: It abstracts away the input handling mechanism from the main rendering loop, ensuring a cleaner main loop. This function can be adapted for other projects where similar inputs need to be processed.

mouseCallback: This is a callback function for handling mouse movements. It processes the mouse's x and y positions to adjust the camera's orientation.

Purpose and Reusability: It decouples mouse movement logic from the main loop, allowing for more readable code.

scrollCallback: This is a callback function for the mouse wheel/scroll event. It processes scrolling to manipulate the movement speed of the camera.

Purpose and Reusability: It keeps zooming logic separate, making it easier to modify or replace the speed change mechanism in the future. It can be adapted for any project that requires mouse scroll-based speed manipulation functionality.

The struct definitions for TV and Computer also serve modularity. They group related objects (cubes) that make up the composite objects (Monitor and Computer), ensuring that these objects are logically grouped and can be manipulated more easily.

By creating these functions and struct definitions, the main loop of the program is kept more readable, and the code structure becomes more maintainable. Each function has a specific task, which follows the Single Responsibility Principle – a principle of software design that states that a function or module should have one, and only one, reason to change. This approach makes it easier to debug, modify, or expand upon the program in the future.