**Project 3 Write-Up**

For Project 3 I sought to implement a 3D chess game. Each of the pieces and the base board were created in blender, and are reused in the actual display of the game based on piece ID. Each of the thirty two pieces is assigned an ID at initialization that aligns with its position in the vector container used to store references to all game pieces. The camera shifts according to the current player’s turn, showing the game board from their perspective. Piece movements are executed by user input from the command prompt.

On runtime a new game is initialized, which creates a new board and piece objects and sets them in their respective place. Movement commands are given in a four-character string of integers. The first two integers correspond to the piece to be moved, and the second pair of integers identify the target square to be moved to. Row/Column locations are indexed from 1 to 8, and invalid movement commands (i.e. selecting an empty square, invalid piece movements, letter input) are rejected and request new input. Pieces will automatically capture if directed to move to a square occupied by the other player’s piece.

The piece ID is used to mark each piece’s position on the board, which is represented by a two-dimensional 8x8-integer array with open spaces marked by 0s. Implementing the game pieces required leveraging inheritance with each unique piece as a separate subclass, but then allowed for shared values and methods, as well as storing every piece in the same vector container. This in turn allows the piece ID to be used as the index into the vector container, simplifying the process of interacting with different pieces on the actual board. Lighting is set so that an infinitely distant light sits above the board, and a local light sits behind the board. The intent was such that based on the board’s position according to the current player, the side facing the camera would be lighter than the side facing away. Texturing of the actual board was not fully implemented, though the import object class was enhanced to process textures from .obj files.

Unlike previous projects, I started working on this project much earlier. While this helped with implementing a number of features, my timeline derailed on the actual lighting and texturing implementations in the game world because I failed to identify those problems and seek help. Inheritance and successfully developing sub and super classes for the game pieces also proved to be an unexpected time sink, however the understanding of how to work with those concepts in C++ will be extremely valuable moving forward and was well worth the one-time cost of learning the nuances for the language. Along this line, planning and organization of separate parts of the project, such as the pieces, the board, and their integration in the main function, was done in a way that enabled me to easily understand the flow of my subcomponents and assisted with identifying various bugs. Major takeaways for future products are the importance of coherent organization, leveraging inheritance where possible to facilitate said organization, and identifying issues that require external assistance sooner and actually seeking the necessary help.