**Design Proposal: Bughouse Chess**

**Project Description:** This project will simulate Bughouse Chess, which is a variant of the classic game of chess. In this variant, there are two teams and two games. Each player of a team plays the opposite color; for instance, teammate #1 will play as white, and teammate #2 will play as black. When one teammate captures the opposing player’s piece, he passes it to his teammate (in the other game), who can then use it as his own piece. For instance, if a player captures a black piece, he passes it to his teammate, who is playing as black and can use the piece in his own game. The overall game ends when one of the two games ends conventionally (checkmate).

**Competitive Analysis:** Chess is a common idea for a game, especially since it has a recursive backtracking element to determining the possible moves a piece can make (when exploring a possible chess solution). My game will differ since it will not only add an extra chess game, but also add the element of being able to reuse a captured piece in the other game. This will add a layer of complexity and prolong the overall game.

**Structural Plan:** I will organize the chess game into objects and classes. Specifically, I will create a class for each chess piece, as well as the board. This will make my code cleaner and more robust, allowing me to deal with the more complicated elements such as transferring reusable pieces and determining game solutions.

**Algorithmic Plan:** The hardest part will be determining a valid chess game solution using a game AI. To do this, I will create a separate game mode that will have the computer solve the game using both sides. I will first try to solve each game separately, and once I get this working, I will add in the reusable pieces between games.

**Timeline Plan:** For TP1, I will do my best to get as much of a working chess game. I will create classes for the pieces and the board, and be able to control the pieces using my arrow keys. I will also create rules for piece interaction (bishop can diagonally attack another piece, etc.).

**Version Control Plan:** Whenever I make a big change, I will save it as a Google Drive File.

**[TP2 UPDATE]:** After finishing a basic working game of chess, I have a couple of ideas on where to go next. I am open to the idea of pursuing Bughouse chess, but if not, I am also interested in being able to display the possible next moves of a piece after making a move.

**[TP3 UPDATE]:** Core game of chess finished. All the pieces can move in their legal way, attack, and put the king in check. The king is also subject to a check, and if it cannot move anywhere without resulting in a check, the game ends in a checkmate. While the king is in check, no friendly pieces may be moved (until the king is out of danger). The king cannot attack his way out of a check, nor can he move into a check. Additionally, I added the special aspects of castling (king and rook may switch places once at the very beginning of a game), and also allowing a pawn to move two spaces forward on its very first move of the game. I will try to implement some sort of method of determining all the legal moves a piece can move at any time. Finally, I added in an option where the key “m” is pressed; this shows all the legal possible moves the selected piece can make.