Introduction

The basic purpose of this project, Power Chess, is to allow two human players to play a game of Power Chess. This requires them to be able to perform the basic moves of the original chess game as well as the enhanced moves developed for powerups.

Requirement Artifacts

- The project is intended to run through a JavaScript user interface within any modern web browser.
- The user must be able to launch the program and see and interact with the game board and pieces.
- When the program is launched, it should initialize the chess board with the correct pieces in the correct positions and display the initial board to the players.
- During initialization, the program should also randomly distribute ‘power-up tokens’ across the board.
- Board positions that are not assigned power-up tokens should be initialized with a randomized counter to be used for future assignment of power-up tokens.
- The program should allow the gameplay to alternate between the two players, allowing each player to move their own pieces, and only their pieces, on their turn.
- On their turn, the program should allow the current player to select one of their pieces.
- When a piece is selected, the program should display all valid moves that that piece can make given the current board state.
- Each piece should determine its valid moves based on its current location and the rules created for that piece type.
- The valid moves determination should include the restriction that no move can result in the moving player being in check. If the player is in check before their move, the only valid moves are those moves that take the player out of check. If the player is not in check before their move, the only valid moves are those moves that do not put the player into check.
- While a piece is selected, the player should be able to unselect the currently selected piece, to allow them to select a different piece.
- While a piece is selected, the player should be able to select one of the valid moves, which triggers the currently selected piece to move to the targeted position.

- When a piece is moved, the program should handle the effects of that move, such as capturing an opponent's piece, and determining if the opponent has been placed in check or checkmate.
- If a piece is moved onto a position that contains a power-up token, that token should be removed from the position and assigned to the piece.
- When a token is assigned to a piece, the position should be assigned a new randomized counter.
- If a piece is in possession of a power-up token, its valid moves list should be determined based on the expanded 'powered-up' rule set created for that piece type.
- When a piece that is in possession of a power-up token is moved, its power-up token should be removed, regardless of where it is moved.
- After a piece is moved, all positions that do not have a power-up token currently assigned should decrement their counter value to track the number of turns remaining until they acquire a token.
- When a counter reaches zero, the position that counter belongs to should be assigned a power-up token.
- If a counter reaches zero on an occupied position, the token should be assigned to the position, but not to the occupying piece.
- Once a piece is moved, the game should switch to the other player and allow them to conduct their turn.
- When a player is determined to be in checkmate, the program should end the game and indicate which player is the winner.
- Once a game is ended, the program should offer the players the ability to play another game or end the program.