CS247 Chess Project Plan of Attack

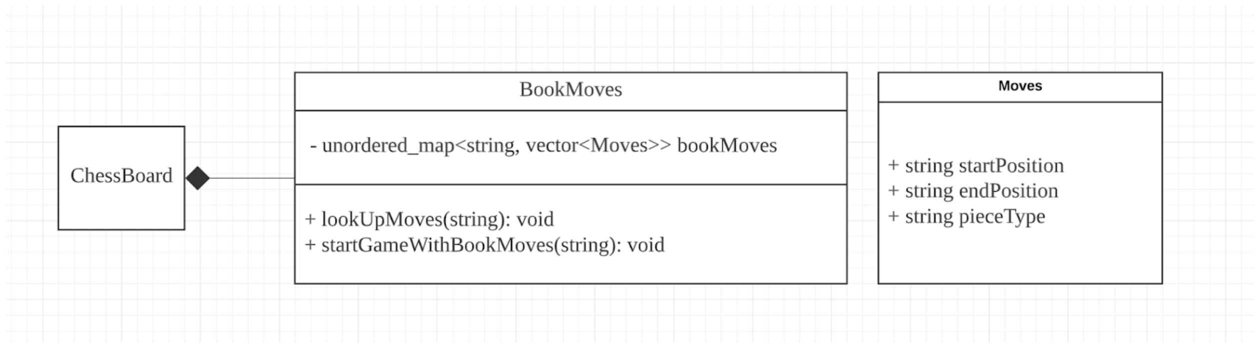
Team members: Kevin Gao, Sean Song, Ke Xu

Expected timeline for implementation

Tasks	assignee	July 16th-17th	July 18th-19th	July 20th-22th	July 23rd-25th
Command Line Input	Kevin Gao	🟧			
Command Line Set up mode	Kevin Gao	🟧			
Player Base Class + Human Player	Sean Song	🟧			
Piece Base Class	Ke Xu	🟧			
Computer Player Level One	Sean Song		🟧		
TextDisplay	Kevin Gao		🟧		
Bishop, Knight, Queen	Ke Xu		🟧		
Scoring	Kevin Gao		🟧		
Computer Player Level Two	Sean Song			🟧	
King, Pawn	Ke Xu			🟧	
Checkmate	Ke Xu			🟧	
Graphic Interface	Kevin Gao				🟧
Stalemate	Ke Xu				🟧
Computer Player Three	Sean Song				🟧
Memory Management	Kevin Gao				🟧

Question: Chess programs usually come with a book of standard opening move sequences, which list accepted opening moves and responses to opponents' moves, for the first dozen or so moves of the game. Although you are not required to support this, discuss how you would implement a book of standard openings if required?

To implement chess standard opening book moves, we can have the chessboard component store an instance of the new class "BookMoves", and this class would contain a private field of bookMoves which lists classic opening moves as an unordered_map of string and vector. The string represents the name of the opening moves, and the vector stores the list of sequences of moves with a struct Move storing the informance of each move. Then, the user would have the choice of looking up the opening moves with the void method lookUpMoves(string), which prints the sequence of moves in the terminal. Also, the user has the choice of opening the game with one of the classical openings, which we can call startGameWithBookMoves(string) and let the chessboard play the moves from the bookMove vector.


ChessBoard
BookMoves
- unordered_map<string, vector<Moves>> bookMoves
+ lookUpMoves(string): void
+ startGameWithBookMoves(string): void
Moves
+ string startPosition
+ string endPosition
+ string pieceType

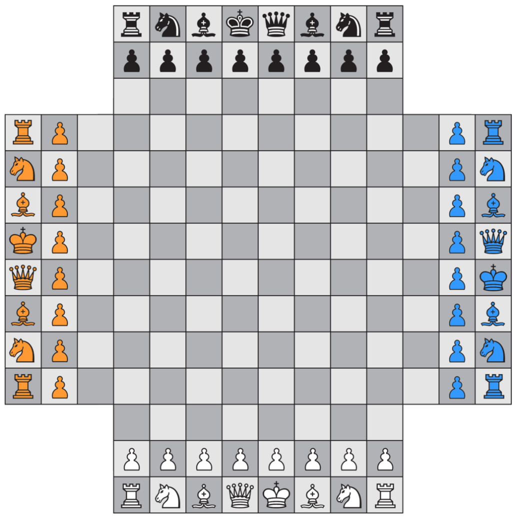
To achieve these two additional features, we would also need to set up two new commands in the input interface. Once the command is received, we can make the chessboard act as a controller and dispatch the command to the BookMove component.

Question: How would you implement a feature that would allow a player to undo his/her last move? What about an unlimited number of undos?

To achieve undo, first, we need to store a stack of moves in the chessboard class. Each element in the stack would contain a pair of moves (<Position start, Position end>), where the first element of the stack contains the start position before the move, and the second element contains the end position after the move. The struct Position simply stores two integers representing row and column respectively.

Then, after the game start, whenever the game makes a valid move, we push the move to the top of the stack as a pair. To undo the moves, we then simply pop the top pairs from the stack, and make a reverse move from the end position to the start position.

Question: Variations on chess abound. For example, four-handed chess is a variant that is played by four players (search for it!). Outline the changes that would be necessary to make your program into a four-handed chess game.



Two make the game into four-handed chess, we need to

- Resize the chessboard grid from 8x8 to 14 x 14 and set restrictions on the bound of the board so that the pieces won't move to the empty corner of the chessboard.
- Initialize four players with four different colours instead of two
- Initialize two more additional sets of pieces
- The setup command would now require checking whether all three other kings are checked
- The textdisplay would now require different names for the pieces since we can't simply use lowercase and uppercase to denote different players
- If the rule is win by most points, we need to store a “score” field in each player and update the score fields after the players made a capture or check.
- The methods for check and getNextPossibleMoves need to be refactored to accommodate the increase number of kings and resizing of board