Chess

Final Project 2022, SUSTech

1 Background

Chess is a board game played between two players.It is played on a square chessboard with 64 squares arranged in an eight-by-eight grid, as shown in Figure 1. At the start, each player (one controlling the white pieces, the other controlling the black pieces) controls sixteen pieces: one king, one queen, two rooks, two bishops, two knights, and eight pawns. The object of the game is to checkmate the opponent's king, whereby the king is under immediate attack (in "check") and there is no way for it to escape. There are also several ways a game can end in a draw.

In this project, we need to write a simple chess game program with a visual graphical interface and basic game rules, such as movement, win/lose decisions, etc. It may sound interesting, so let's get started.

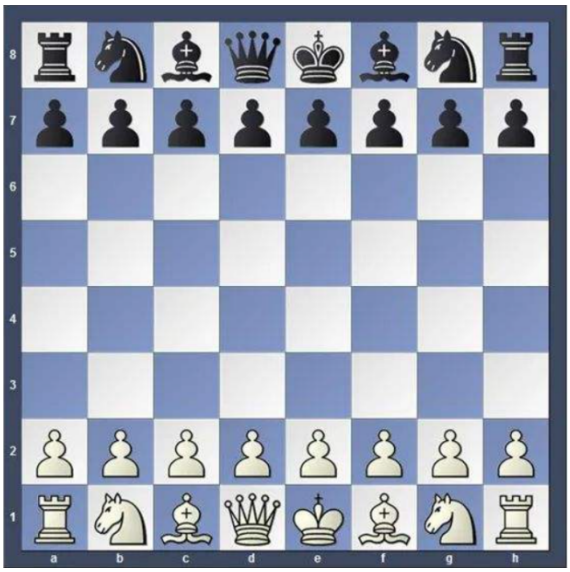

Figure 1: Checkerboard layout

2 Task Specification

The program can be roughly divided into the following tasks, and the score points for each task are given.

2.1 Initialization(10%)

- Board initialization (5%). Initialize the game, form a board with pieces in correct positions and showing the status of the black and white turn.
- Board reset (5%). Click the reset button, the pieces are restored to their initial positions and the previous game is cleaned up with no conflicts.

2.2 Load and store game(20%)

- Load a game from existing archives and then form a board with correct player turn notification function (2%).

- Can store a game with board information and the correct player turn (2%).
- Can load and store game multiple times (2%).
- Can save steps when storing a game(3%).
- Have error check on archive when loading the game (5 * 2% = 10%):
  1. The board is not 8*8.
  2. The pieces are not one of six, or not black and white.
  3. Missing next move.
  4. Wrong file format (e.g. txt is specified but json is imported).
  5. Illegal move (the stored movement is illegal).
- Save the game as a file (2%).

2.3 Game rules(40%)

1. Movement (6 * 2% = 12%). White moves first, after which players alternate turns, moving one piece per turn, except for castling, when two pieces are moved. Each piece has its own way of moving. The rules are as follows:
   (a) King: The king moves one square in any direction. There is also a special move called castling that involves moving the king and a rook.
   (b) Queen: A queen can move any number of squares along a rank, file, or diagonal, but cannot leap over other pieces.
   (c) Rook: A rook can move any number of squares along a rank or file, but cannot leap over other pieces. Along with the king, a rook is involved during the king's castling move.
   (d) Bishop: A bishop can move any number of squares diagonally, but cannot leap over other pieces.
   (e) knight: A knight moves to any of the closest squares that are not on the same rank, file, or diagonal. (Thus the move forms an "*L*"-shape: two squares vertically and one square horizontally, or two squares horizontally and one square vertically.) The knight is the only piece that can leap over other pieces.Here, we stipulate that there is no restriction on "*poor horse leg*".
   (f) Pawn: A pawn can move forward to the unoccupied square immediately in front of it on the same file, or on its first move it can advance two squares along the same file, provided both squares are unoccupied. A pawn can capture an opponent's piece on a square diagonally in front of it by moving to that square (black crosses). It cannot capture a piece while advancing along the same file.
2. Three special moves (3 * 4% = 12%).
   (a) En passant. When a pawn makes a two-step advance from its starting position and there is an opponent's pawn on a square next to the destination square on an adjacent file, then the opponent's pawn can capture it en passant ("in passing"), moving to the square the pawn passed over. This can be done only on the turn immediately following the enemy pawn's two-square advance; otherwise, the right to do so is forfeited.
   (b) Castling. Once per game, each king can make a move known as castling. Castling consists of moving the king two squares toward a rook of the same color on the same rank, and then placing the rook on the square that the king crossed.Castling is permissible if the following conditions are met(Optional):
      i. There are no pieces between the king and the rook.
      ii. The king is not in check.
      iii. The king does not pass through or land on any square attacked by an enemy piece.

iv. Neither the king nor the rook has previously moved during the game.

(c) Promotion.When a pawn advances to its eighth rank, as part of the move, it is promoted and must be exchanged for the player's choice of queen, rook, bishop, or knight of the same color.

3. End of the game (4 * 4% =16%).
   (a) Win. The king is in check and the player has no legal move.
   (b) Draw. There are several ways a game can end in a draw:
      i. Perpetual check. If a player can check the other infinite times in a row, with no reasonable chance of escape, the game is drawn.
      ii. Threefold repetition. Threefold repetition is when a game is drawn due to the position in a game being the same three times in the game.
      iii. Stalemate. If the player to move has no legal move, but is not in check, the position is a stalemate, and the game is drawn.

2.4 Graphical interface(10%)

Design user GUI using Swing or FX. 3 points will be deducted for each time the application is found to require interaction with the console, and 3 points will be deducted for each time the application is found to crash abnormally.

2.5 Bonus(30% in week15, 20% in week16)

1. Platform and aesthetics (maximum 12%).
   (a) Adding start menu for different functions.(1%)
   (b) Use *JFileChooser* to load existing archives.(1%)
   (c) Game user properties.(2%)
   (d) Ranking list.(1%)
   (e) Replace board image.(1%)
   (f) Embed background image.(1%)
   (g) Theme skin switching.(1%)
   (h) Embed chess sound effects, background music.(2%, if only one is 1%)
   (i) Load and store the game GUI module.(2%)
   (j) Load or store game archives for different users.(2%, not add points with (b))
   (k) Mouse over pieces or board squares with color change.(2%)
   (l) The board size adapts to the window size change.(2%)
   (m) All-round rewrite *JOptionPane*.(3%)
2. AI and algorithm(maximum 12%)
   (a) Show the next legal move of a piece when it is selected.(1%)
   (b) Show alarm when King is attacked.(1%)
   (c) Use random moves in PvE mode.(2%)
   (d) Use greedy strategy moves in PvE mode.(3%)
   (e) Use pruning search algorithms in PvE mode and be able to reasonably describe the evaluation function.(6%)

   (f) Multiple difficulties in PvE mode.(4%)

3. Repentance and time-consuming components (maximum 8%)
   (a) Can retract a movement.(1%)
   (b) Can retract arbitrary movements.(2%, do not add extra points with (1))
   (c) Show turn time and switch to the next player when time is up.(3%)
   (d) Game steps replay.(3%)
   (e) Show animation when pieces move or when capturing pieces.(3%)
4. Packing (maximum 2%)
   Package into *exe* executable file.(2%)
5. LAN battle (maximum 12%)
   (a) Support online battle.(2%)
   (b) Server and client side and interact well.(4%)
   (c) Can reconnect when disconnection occurs.(4%)
   (d) Watch the battle.(4%)
   (e) Real-time visibility of each other's actions, including clicks, etc.(4%)
6. Version control (maximum 2%)
   Use github or gitee for version control, and all the group members have a certain amount of commit. (2%)