Assignment 2

2-Player Chess

Posted Thu, Feb 18. Due Mon, Mar 21 by 11 PM in Bitbucket.

Worth 100 points (10% of course grade.)

For this assignment you will explore how to apply the design ideas learned in class to design and implement a board game.

You will work **in pairs** on this assignment. Read the DCS Academic Integrity Policy for Programmming Assignments - you are responsible for this. In particular, note that **"All Violations of the Academic Integrity Policy will be reported by the instructor to the appropriate Dean".**

You will implement the game of Chess for two players. Your program, when launched, should draw the board using the terminal and prompt whomever's turn it is (white or black) for a move. Once the move is executed, the move should be played and the new board drawn, and the other player queried.

Output

The board should be drawn on the screen with ascii art EXACTLY as shown in this example. Note there is a blank line above and below any prompt/message your program will print, and the board itself.

Deviations from this WILL INCUR A PENALTY!

If you have ANY questions or issues about this, CHECK with your TA.

NOTE:

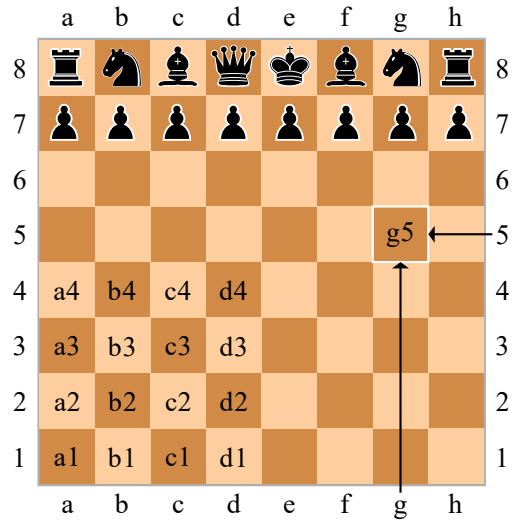
- Every piece must know what moves are allowed on it. If a player attempts an illegal move on a piece, your program should not execute the move, but instead, print "Illegal move, try again".
- When a move is made, and it puts the opponent's King under check, your program should print "Check" before asking for the opponent's move.
- If a checkmate or stalemate is detected, your program should print "Checkmate" or "Stalemate".

  Stalemate implementation is optional - 10 points extra credit if you do it correctly - see Extra Credit section below.
- The last thing before termination should be a display of "Black wins", "White wins" or "Draw".

Input

Your program needs to accept input of the form "FileRank FileRank", where the first file (column) and rank (row) are the coordinates of the piece to be moved, and the second file and rank are the coordinates of where it should end up. (See the board example shown above.)

The figure immediately below should make it clear which rank and file combinations belong to which squares. The white pieces always intially occupy ranks 1 and 2. The black pieces always initially occupy ranks 7 and 8. The queen always starts on the d file.



As an example, advancing the white king's pawn two spaces would be input as "e2 e4".

A castling move is indicated by specifying where the king begins and ends. So, white castling king's side would be "e1 g1".

A pawn promotion is indicated by putting the piece to be promoted to after the move. So, promoting a pawn to a knight might be "g7 g8 N". If no promotion is indicated, it is assumed to be a queen.

Example of black winning

Ending the game

- If checkmate or stalemate occurs, the game shall end immediately with the result reported.
- A player may resign by entering "resign".
    - Example of white resigning
    - Example of black resigning
- A player may offer a draw by appending "draw?" to the end of an otherwise regular move. The draw may be accepted by the other player submitting "draw" as the entirety of his or her next move. There will be no automatic draws (due to unchangeing positions over long periods of time, etc).
    - Example of a draw

You are NOT required to implement termination by threefold repetition, or the fifty-move rule. You are welcome to include them in your code to make it complete; however, there is no extra credit for either.

Grading

- **Correctness, 90 pts**: Implementation of all required functionality including drawing the board:
    - All legitimate basic moves for all pieces
    - Castling
    - Enpassant
    - Promotion
    - Identification of check
    - Identifcation of checkmate
    - Identification of illegal move (print "Illegal move, try again")
    - Resign
    - Draw

    - Drawing board display as specified
- **Javadoc, 5 pts**: Comment classes, fields, and methods with Javadoc tags. Also, make sure you record your name in each Java file with the @author Javadoc tag.
- **Design, 5 pts**: Appropriate application of object-oriented design ideas you have learned. Although design points are low, our experience is that if you design your solution badly, you will have a lot of trouble implementing/debugging it.
- **Extra Credit, 10 points**: Correct implementation of stalemate.

Submission - In Bitbucket

Create a new repository and give your grader read access.

Name your project `ChessXX`, where XX is the two digit group number. Use packages as necessary. There should be at least one package, called `chess`, with the main class called `Chess` in it. This is the class that we will run when testing your program.