Validating Multiple Chess Moves on a Chessboard

Overview

For this assignment you will replace your Javascript code from HW 2 with one that uses arrays and loops for validating multiple chess moves. As before, chess pieces can be a King (K), Queen (Q), Bishop (B), Knight (N), Rook (R), or a Pawn (P). Your program will keep asking the user to enter the type of the chess piece, as well as its starting (current) position and ending (intended) position. It will then validate whether this piece can change its position with a single valid move (given that all of the valid moves entered so far have already taken place). When the user is done entering all moves, your program will then display a chessboard with all of the valid/invalid moves.

Program Specification

Specifically, your program should do the following (in the same order):

- Ask the user if he/she wants to enter another move. If yes, use the 5 prompts below to ask for the move. If not, display the final chess-board (as described below).

  - Prompt the user to enter the type of the (white) chess piece. The valid options are: K, Q, B, N, R, P.
  - Prompt the user to enter the starting column (valid options are: a, b, c, d, e, f, g, h).
  - Prompt the user to enter the starting row (valid options are: 1, 2, 3, 4, 5, 6, 7, 8).
  - Prompt the user to enter the ending column (valid options are: a, b, c, d, e, f, g, h).
  - Prompt the user for ending row (valid options are: 1, 2, 3, 4, 5, 6, 7, 8).
  - If the piece can move from the starting position to the ending position in a single move, then:
    - The move is valid and the piece is assigned to the ending position.
    - Displays a message (display on the HTML page, not in an alert box) that lists the 5 input fields and a message as shown below:

        N d 4 f 5 : This is a valid move.
- If the piece can *not* move from the starting position to the ending position in a single move, then:
  - The move is invalid and the piece is retained at the starting position.
  - Your program should display a message (display on the HTML page, not in an alert box) that lists the 5 input fields and a message as shown below:

  Q g 6 e 3 : This is an INVALID move.

- Once the user is done entering all moves (answers “no” to the first question above), your program should display a chessboard, in which:
  - For each valid move, the starting and ending cells are highlighted in green (#00FF00), and the piece is displayed in the *ending* cell.
  - For each invalid move, the starting and ending cells are highlighted in red (#FF0000), and the piece is displayed in the *starting* cell, and a bold “?!” (question & exclamation mark) is displayed in the ending cell.

- Align your chess-board to the center of the page.

**NOTE:**
- Consider (and display) only white pieces.
- If *any* of the user input is invalid, simply display an appropriate error message, and do nothing else (do not display the chess-board).
- When a user enters a move, if the starting/ending positions overlap with *any* of the starting/ending positions he has entered for non-overlapping moves so far, display an appropriate error message, and ignore the move (do not display it on the final chess-board). But for such an input, your program should also display a message (in HTML, not in an alert box) that lists the 5 input fields and a message as shown below:

  Q g 6 f 5 : This is an OVERLAPPING move.

- Use 2-D arrays to remember/store for each square of the chess-board (based on user input):
  - What color should be the background
  - What image/text should be displayed (if at all)
- Use (nested) loops for displaying the chess-board.
- Use the chess rules specified below to figure out if a move is valid or invalid. (Note: below, we assume that 64 cells of the chess-board can be specified using (X,Y) coordinates, where the cell in the lower left-hand corner has a coordinate of (1,1), and the cell on the upper right-hand corner has a coordinate of (8,8). Similarly, the cell on the upper left-hand corner will have a position of (1,8) and the cell on the lower right-hand corner will have a position of (8,1).)
  - Valid King (K) move:
    - If the piece moves from \((X_1, Y_1)\) to \((X_2, Y_2)\), the move is valid if and only if \(|X_2 - X_1| \leq 1\), and \(|Y_2 - Y_1| \leq 1\).
  - Valid Bishop (B) move:
    - If the piece moves from \((X_1, Y_1)\) to \((X_2, Y_2)\), the move is valid if and only if \(|X_2 - X_1| = |Y_2 - Y_1|\).
Valid Rook (R) move:
- If the piece moves from \((X_1, Y_1)\) to \((X_2, Y_2)\), the move is valid if and only if \(X_2 = X_1\) or \(Y_2 = Y_1\).

Valid Queen (Q) move:
- A queen's move is valid if it is either a valid move for either a bishop or a rook.

Valid Knight (N) move:
- If the piece moves from \((X_1, Y_1)\) to \((X_2, Y_2)\), the move is valid if and only if \(|X_2 - X_1| = 1\) and \(|Y_2 - Y_1| = 2\) or \(|X_2 - X_1| = 2\) and \(|Y_2 - Y_1| = 1\).

Valid Pawn (P) move:
- If the piece moves from \((X_1, Y_1)\) to \((X_2, Y_2)\), the move is valid if and only if \(X_2 = X_1\) and \(Y_2 - Y_1 = 1\).

Sample Output:

Below, you’ll find screenshots of a couple of sample outputs (Firefox browser).

**Sample Input 1:**
OK N d 4 f 5 OK Q g 6 e 3 Cancel

**Sample Output 1:**
- N d 4 f 5 : This is a valid move.
- Q g 6 e 3 : This is an INVALID move.
Sample Input 2:
OK N d 4 f 5 OK Q g 9

Sample Output 2:
N d 4 f 5: This is a valid move.

Sample Input 3:
OK N d 4 f 5 OK Q d 4 g 7

Sample Output 3:
N d 4 f 5: This is a valid move.
Q d 4 e 3: This is an OVERLAPPING move.

The starting position overlaps with a previous move. This move will be ignored.
Programming Style/Guidelines

About 10% of the grade is reserved for a good programming style. So do remember to indent your code properly and do remember to add sufficient comments.

Tips

This homework is designed to give you practice on using arrays and loops. These are the only additional constructs you need to complete this homework.

If you use arrays effectively to remember the color and image for each chessboard square, you can do away with most selection (if) statements from the part of your code that is actually displaying/drawing the final chessboard. If you use your arrays effectively, you will also easily be able to check whether a move is overlapping with a previous move.

Note that the use of loops will reduce the total lines of Javascript code you end up writing (compared to Assignment # 2); however, the code will be more complex. So do start early (today), else you will not be able to complete the assignment on time!

Your homework will be public and available to the whole world including web search engines such as Google. So don't put any information on the pages that you don't want the whole world to see. Do not create a link to this page from any other page.

Submission Instructions

- Your HTML should include the following comments at the top of the page

  Your name
  COMP 110
  Assignment number 3
  Total time taken to complete this assignment
  Pledge: I have neither given nor received unauthorized aid on this assignment

  (signed) ____________________

  Sign the pledge by typing your full name.

- Give a secret (and cryptic) name to your HTML file (e.g., xx4yzrt.html). Please do not use this example name — use a different one. This will help ensure that others cannot copy your submission. Do not share the name of your file with anyone (other than the instructor or the TAs). Do not create a link to this file from any other web-page!

- Turn in a single sheet that contains on paper your pledge for this homework (the pledge should have your name, assignment number, your pledge, and signature). Please type-set (do not hand-write). Hand this to either the instructor or the TAs by the submission deadline.
• Submit on blackboard the full URL and source code for your web page n blackboard. e.g., if your ONYEN login id is “onyen”, and your assignment file is called xx4yzrt.html, then under “Assignment # 3”:
  
  o Submit the following text in the “Comments” box under “Assignment # 3”:
    
    http://www.unc.edu/~onyen/xx4yzrt.html
  
  o Also submit your HTML file as an attachment, by using the “attach local file” under “Assignment # 3” (and choosing xx4yzrt.html).

• You do not need to turn in a paper copy of your HTML file.

• Do NOT modify your HTML files after submitting the above URL on blackboard! So make sure your homework is ready for grading before you submit it. If your file is modified after the submission deadline, your assignment will be considered (correspondingly) late.

A Reminder on the Honor Code

Please review the honor code for this course. It is ok to:

• Discussing the assigned homework problem to understand its meaning.

• Discuss about specific HTML and Javascript features

In all cases you must explicitly acknowledge any and all substantive help received from other individuals. That is, if you collaborate in any of the above ways with other individuals then you must include an explicit acknowledgment in your homework solution of the persons from whom you received aid. Acknowledging others, if done properly, will not adversely affect your grade.

Unacceptable collaboration on written homework includes:

• Copying HTML or Javascript code (or any portion of your homework solution) from other students, or from the web.

Should questions arise during the course of working on a problem please feel free to contact the instructor. In principle, if you work with others in good faith and are honest and generous with your attributions of credit you will have no problems.