COP-4338 Programming III, Summer 2018

Programming Assignment 5:
I/O System Calls in C

Due Date: July 27 at 4:00 PM

In this assignment, you are asked to write a program that reads the initial position of chess pieces on the chessboard from the file "board.csv" and make necessary changes on the same file after each move or capture.

1 File Format

A chessboard can be represented by an 8×8 table of integers in which every field is corresponding to one of the 64 square cells and determines the occupant of the cell. As seen in

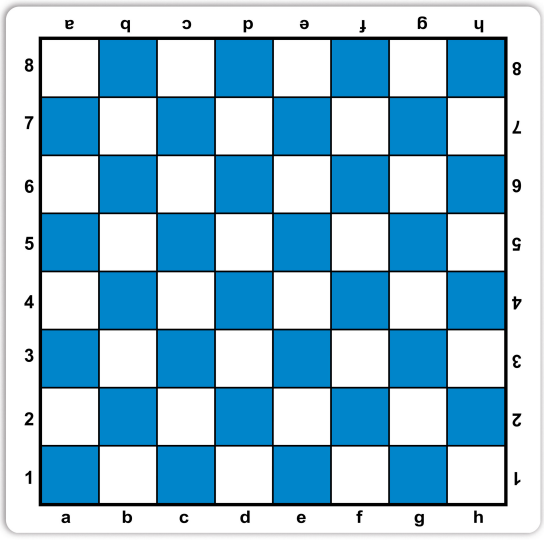

Figure 1: Row and column labels in a chessboard.

Figure 1, the table row with index 0 is labeled with "8" and the table row with index 7 is labeled with "1". Also, the table column with index 0 is labeled with "a", while the table column with index 7 is labeled with "h".

If the value of a field in the table is zero, it means that the cell corresponding to the field is empty. If the value is positive, it means that a white piece occupies the corresponding

cell; while a negative value means that the piece color is black. The absolute value of the field specifies which piece occupies the corresponding cell. Here is the general rule:

$$|table[i][j]| = \begin{cases} 0 & \text{cell is empty} \\ 1 & \text{cell is occupied by a pawn} \\ 2 & \text{cell is occupied by a knight} \\ 3 & \text{cell is occupied by a bishop} \\ 4 & \text{cell is occupied by a rook} \\ 5 & \text{cell is occupied by a queen} \\ 6 & \text{cell is occupied by a king} \end{cases} \quad (1)$$

The file "board.csv" stores the 8×8 table representing the chessboard and chess pieces. Its format follows the CSV format mentioned in Assignment 4.

2 Program Commands

Your program must perform the following commands entered by user through console. After each command, the file "board.csv" must be updated using **low-level write system calls**. After each command, **at-most two characters in the file can be overwritten**. Refreshing the whole chessboard after each command is costly and inefficient.

- mv $\alpha_0\beta_0$ $\alpha_1\beta_1$: moves the piece in cell with label $\alpha_0\beta_0$ to cell with label $\alpha_1\beta_1$ where α_0 and α_1 can be letters from a to h; while β_0 and β_1 can be integers from 1 to 8.
- cp $\alpha_0\beta_0$ $\alpha_1\beta_1$: captures the piece in cell with label $\alpha_1\beta_1$ with the piece in cell $\alpha_0\beta_0$ where α_0 and α_1 can be letters from a to h; while β_0 and β_1 can be integers from 1 to 8.
- show: prints out the content of file "board.csv" on the standard output (screen) in the .tl5 format.

For the sake of simplicity, assume that there are only *pawns* and *knights* on the board. Figure 2 shows how knight can move on the board and capture pieces with opposite color on the board. Also, it shows the direction in which white pawn moves and captures. The black pawn moves and captures in the opposite direction. In summary, a pawn can move only one cell forward at a time; however, if it is white and is in the second row (or is black and in the seventh row), it has the option of moving two cells forward if there is no piece on its way.

If a 'cp' or 'mv' command does not follow the above rules, the program must make no change on the file "board.csv" and prints out an error message.

3 50% Bonus Part

As the bonus point, your program must support all type of pieces on the board.

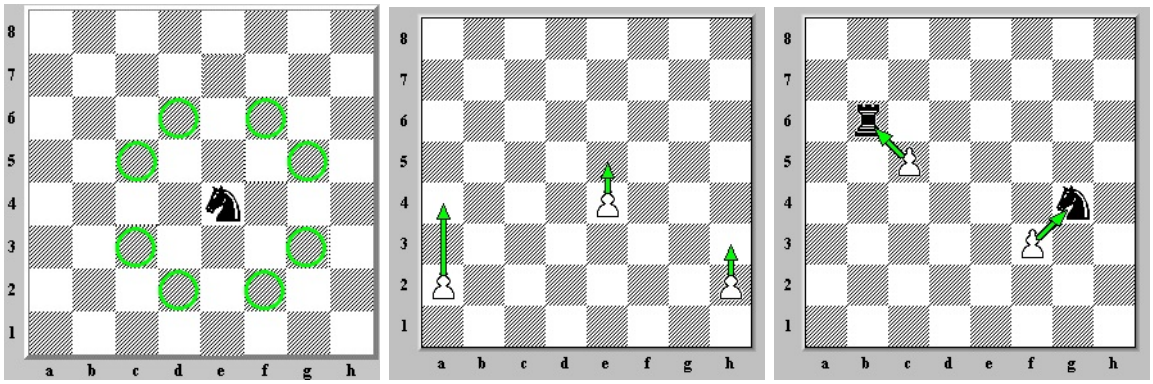

Figure 2: Movement and capture of knight (left), movement (middle) and capture of pawn (right) on chessboard.

4 Submissions

You need to submit a *.zip* file compressing the C source file(s) related to the assignment (*.c* files).