UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG
School of Computer Science & Applied Mathematics

COMS3005A Assignment
Move Execution

1 Introduction

In the previous part, we generated the valid moves at a given state. We will now need to implement functionality so that one of these moves can actually be played. This means we must take in the initial position (as a FEN string), and the move to be executed. The program will then output the resulting board position once that move has been played (again as a FEN string).

2 Move Representations

For every submission, we will represent a move as a string `<start_square><end_square>` specifying the starting location of the piece to move and then square the piece ends up on. For example, the move `e3e4` represents a piece moving from `e3` to `e4`.

3 Executing Moves

Each move specifies the square that a piece moves from and the square it moves to. To update the board, we must simply remove the piece from its starting square and place it at the target square (if there is an opposing piece at that square, that piece is "captured" and so is removed from the board). There are three other considerations:

1. After a move has been played, the side to move switches. If Black moved, it is then White's turn (and vice versa).
2. The move counter in the FEN string counts the number of moves made, but is only incremented after Black plays their turn.
3. One other consideration is that of drowning pieces. If a piece of the moving player begins in the river, and after the move is executed it remains in the river, then it too must be removed from the board.

4 Input Hint

Hint: a reminder not to be careful when mixing `cin` with `getline`. An example of doing so is below:

```
int N;
cin >> N;
cin.ignore(); //NB!
for (int i = 0; i < N; ++i) {
    string fen;
    getline(cin, fen);
}
```

Submission: Execute Moves

Write a C++ program that accepts a FEN string and a move to be executed and stores the piece location information in appropriate data structures. It should then output the FEN string of the position that results when the move is executed, as well as whether the game has been won by either side.

Input

The first line of input is N, the number of input positions given as FEN strings. $2N$ lines follow consisting of FEN strings and the move to be executed. You may assume that each FEN string is a valid position, and that the move to play is a valid one.

Output

For each FEN string and move, output two lines. The first line of output should be the resulting position as a FEN string. The second line should specify whether the game is over. If the game is not over, print `Continue`. If the move was made by White and resulted in it winning the game, output `White wins`. Otherwise if Black has just won, output `Black wins`.

Example Input-Output

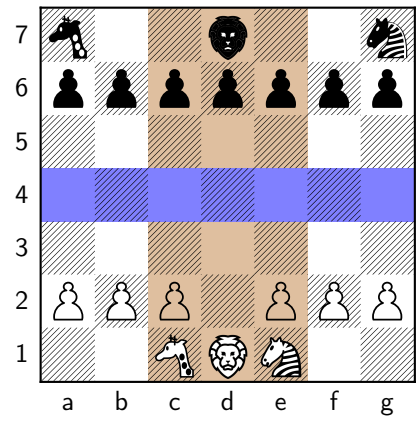
Sample Input

```
3
g2l2z/ppppppp/7/7/7/PPP1PPP/2GLZ2 w 4
d1d2
1z5/pPp1lP1/5gp/4P1p/4L1p/2p2pP/7 b 35
f5f7
1z5/pPp1lP1/6p/4P1g/4L1p/2p2pP/7 b 12
g4e4
```

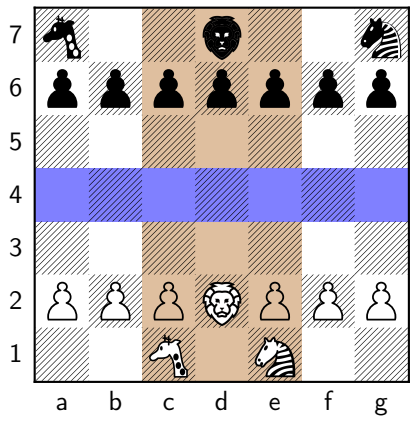
Sample Output

```
g2l2z/ppppppp/7/7/7/PPPLPPP/2G1Z2 b 4
Continue
1z3g1/pPp1lP1/6p/4P2/4L1p/2p2pP/7 w 36
Continue
1z5/pPp1lP1/6p/7/4L1p/2p2pP/7 w 13
Continue
```

Visualisation of Above Test Cases

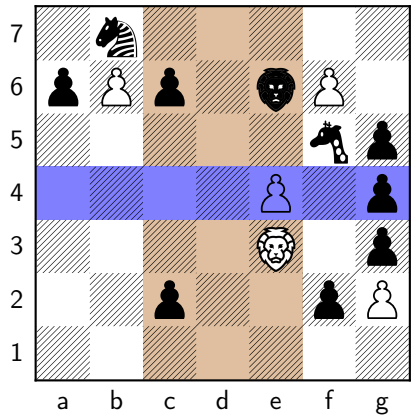


(a) `g2l2z/ppppppp/7/7/7/PPP1PPP/2GLZ2 w 4`

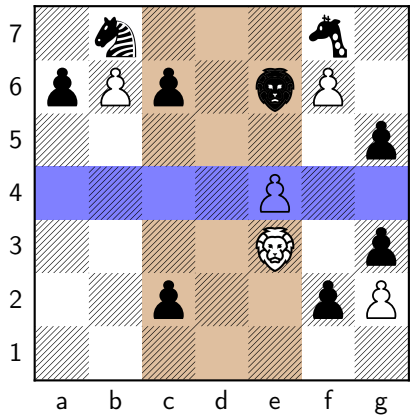


(b) `g2l2z/ppppppp/7/7/7/PPPLPPP/2G1Z2 b 4`

Figure 1: Initial and next positions after the move `d1d2`. The White lion moves one square forward.

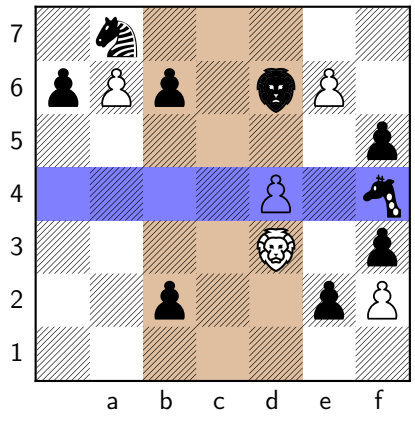


(a) `1z5/pPp1lP1/5gp/4P1p/4L1p/2p2pP/7 b 35`

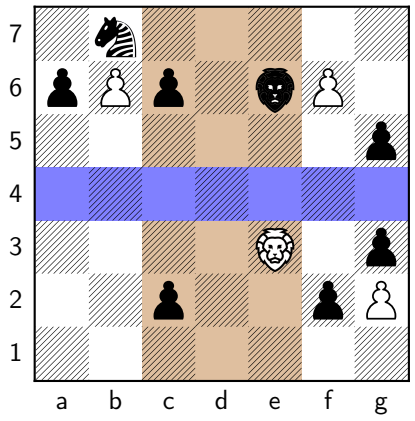


(b) `1z3g1/pPp1lP1/6p/4P2/4L1p/2p2pP/7 w 36`

Figure 2: Initial and next positions after the move `f5f7`. The Black giraffe moves to f7, but the pawn on g4 drowns at the end of the move and is removed. The move count is incremented.



(a) `1z5/pPp1lP1/6p/4P1g/4L1p/2p2pP/7 b 12`



(b) `1z5/pPp1lP1/6p/7/4L1p/2p2pP/7 w 13`

Figure 3: Initial and next positions after the move `g4e4`. The Black giraffe on g4 captures the White pawn on e4. However, it started and ended in the river (although on different squares) and so after moving, it too drowns and is removed from the board.