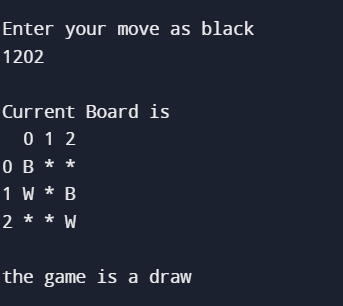
**Pawnscape**

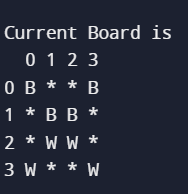
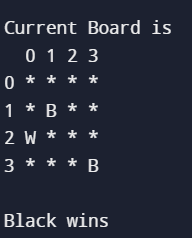
For a 3x3 setup, I manually played optimally for both, and the game always ended in a draw. In the 3x3 setup, the pawns always tend to confront each other after sequentially capturing, allowing no space to move, eventually ending in a draw.



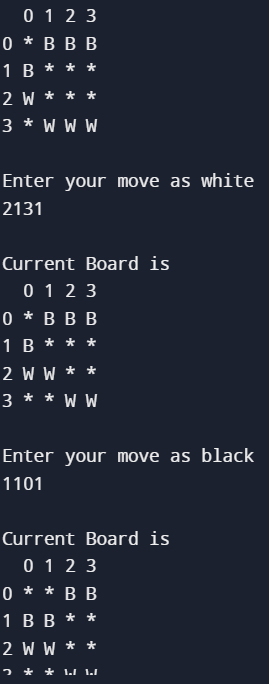
For a 4x4 board, it was difficult to decipher any patterns effectively.

Playing with white, I could not find a strategy ensuring white would win every time.

However, for Black, if the initial moves of white are copied, before White captures a piece, and then an optimal logic is applied, then Black goes on to win the game. Therefore, if the black player copies the initial moves of white, then it can lead to a winning strategy.

The above figure shows the state of the game, if black initially copies the moves of White (before any capturing is done), and subsequently wins.



However, if the extreme pieces are moved, and the moves are copied, then Black may have to change strategies to conclude a Draw or Win.

Thus, I was not able to find a concrete strategy for either colour, and thus the optimal algorithm could not be crafted.

As a coding exercise, I re-created the playing environment on Python as a two-player game similar to the helper code (without failsafes, which can easily be added later).

