CSE 259 - Logic in Computer Science Fall 2024

Recitation-8

Project 2: Chess - Part 2 (Task 1)

Waqar Hassan Khan



Project 2

- Implement a Chess program
- 3 Tasks
 - Visualize the chess board
 - Write codes for playerA so that it can move on its own. PlayerB codes are already there!
 - 3. Use PlayerA's code to play against PlayerB

- We will call main. from the console
- If the template is ran, the following output is seen: It asks for whites move and the black moves on it's own

```
| ?- main.

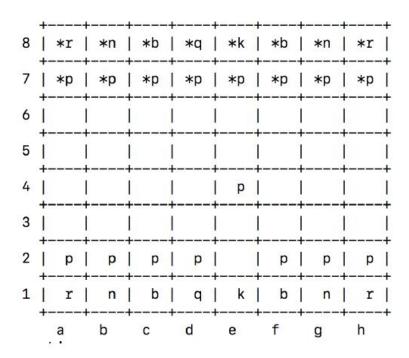
white move -> e2e4.

Working...

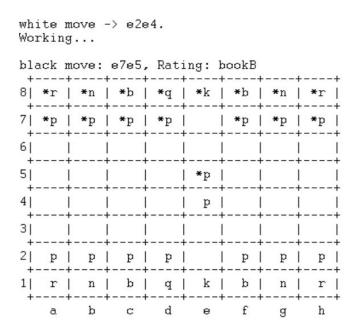
black move: e7e5, Rating: bookB
[state(white,_94,_95,_96),state(black,_99,_100,_101),piece(a-8,black,rook),piece(b-8,black,night),piece(c-8,black,bishop),piece(d-8,black,que)

white move ->
```

Write codes so that the chase board is drawn visually



Print once after both players have completed their moves



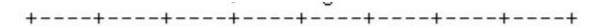
Write your visualization codes here

Use the same old drawSymbol code!

```
drawSymbol(Symbol, 0).
drawSymbol(Symbol, N) :- N > 0, write(Symbol), N1 is N - 1, drawSymbol(Symbol, N1).
```

 Use this code to draw the border lines in each row. We saw this code in last class!

```
drawBorderLine(0) :- drawSymbol('+', 1), nl.
drawBorderLine(Col) :-
   Col > 0,
   drawSymbol('+', 1), drawSymbol('-', 4),
   NewCol is Col - 1,
   drawBorderLine(NewCol).
```



Slight modification in the code to draw empty cells

```
drawContentCell(BoardStates, Row, 0) :- drawSymbol('|', 1), nl.
drawContentCell(BoardStates, Row, Col) :-
   Col > 0,
   drawSymbol('|', 1), drawCell(BoardStates, Row, Col),
   NewCol is Col - 1,
   drawContentCell(BoardStates, Row, NewCol).
```

The code to draw the cell numbers

```
drawPair :-
  drawSymbol(' ', 4), drawSymbol('a', 1), drawSymbol(' ', 4), drawSymbol('b', 1),
  drawSymbol(' ', 4), drawSymbol('c', 1), drawSymbol(' ', 4), drawSymbol('d', 1),
  drawSymbol(' ', 4), drawSymbol('e', 1), drawSymbol(' ', 4), drawSymbol('f', 1),
  drawSymbol(' ', 4), drawSymbol('g', 1), drawSymbol(' ', 4), drawSymbol('h', 1).
```

- The printing function is called in the template. No need to worry about where and how to call it.
- Call drawBoard from it which will draw the board

```
print_board(Board) :-
    drawBoard(Board, 8, 8), nl.
```

drawBoard is same as before!

```
drawBoard(BoardStates, 0, Col) :- drawSymbol('-', 1), drawBorderLine(Col), drawPair.
drawBoard(BoardStates, Row, Col) :-
   Row > 0,
   drawSymbol('-', 1),
   drawBorderLine(Col),
   drawSymbol(Row, 1),
   drawContentCell(BoardStates, Row, Col),
   NewRow is Row - 1,
   drawBoard(BoardStates, NewRow, Col).
You, 2 days ago * task-1 added
```

```
% finds whether the current cell has any white or black piece in it
drawCell(BoardStates, Row, Col) :-
 pair(Name, Col),
  myMember(piece(Name-Row, Color, Piece), BoardStates),
 drawSymbol(' ', 1),
    (Color == black, drawSymbol('*', 1));
    (Color == white, drawSymbol(' ', 1))
  ),
  pair(Piece, PieceAbbreviation),
  drawSymbol(PieceAbbreviation, 1),
  drawSymbol(' ', 1).
% deals with white space
drawCell(BoardStates, Row, Col) :-
  pair(Name, Col),
  \+ (myMember(piece(Name-Row, Color, Piece), BoardStates)),
  drawSymbol(' ', 4).
```

```
pair(a, 8).
pair(b, 7).
pair(c, 6).
pair(d, 5).
pair(e, 4).
pair(f, 3).
pair(g, 2).
pair(h, 1).
pair(rook, r).
pair(bishop, b).
pair(king, k).
pair(pawn, p).
pair(queen, q).
pair(knight, n).
```

- We started with (8, 8) in the board. Row started with
 8.
- Same goes for the column. We start at 8, 7, 6... So, we pair (a, 8), (b, 7), etc.
- The other pair facts are for abbreviation of chess pieces,

- At first, depending on the the column, we identify the corresponding letter (a, b, c, etc.) using pair(Name, Col). Now the letter is in Name
- Then we use myMember, we pass the current row, col and the board state. If the cell is occupied, the Color and Piece variable will be populated with actual value

```
pair(Name, Col), You, 2 days ago task-1 added myMember(piece(Name-Row, Color, Piece), BoardStates),
```