CSE 259 - Logic in Computer Science

Recitation-7

Project 2: Chess - Part 2 (Task 1)

Waqar Hassan Khan

Project 2

- Implement a Chess program
- 3 Tasks
  1. Visualize the chess board
  2. 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

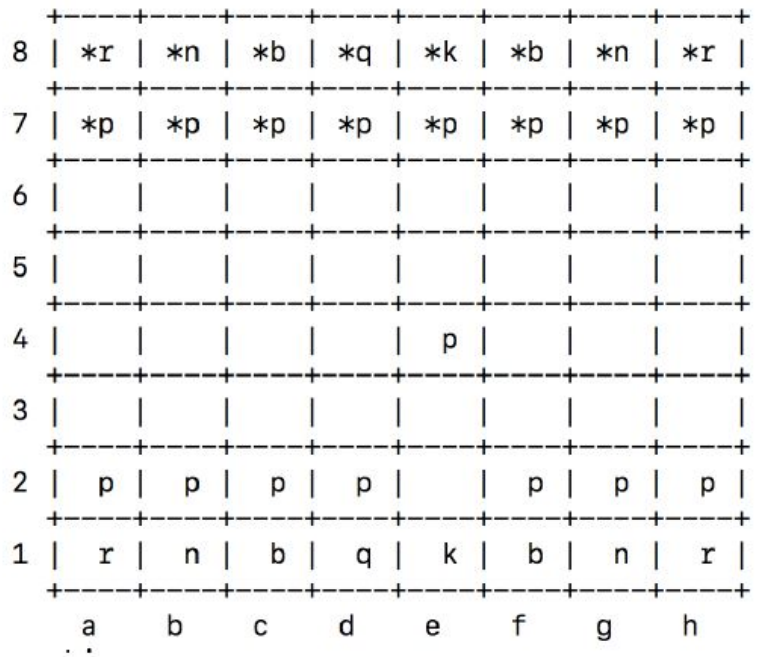
Project 2 - Task 1

- 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

```
% e:/Programming/TA/ASU-CSE-259-Prolog/Project Templates/Project-2-Chess/chess.pl compiled 0.03 sec, 169 clauses
?- main.

white move -> e2e4.
Working...

black move: e7e5, Rating: bookB
[state(white,_1304,_1306,_1308),state(black,_1320,_1322,_1324),piece(a-8,black,rook),piece(b-8,black,knight),piece(c-8,black,bishop),piece(d-8,black,queen)
,piece(e-8,black,king),piece(f-8,black,bishop),piece(g-8,black,knight),piece(h-8,black,rook),piece(a-7,black,pawn),piece(b-7,black,pawn),piece(c-7,black,pa
wn),piece(d-7,black,pawn),piece(f-7,black,pawn),piece(g-7,black,pawn),piece(h-7,black,pawn),piece(a-1,white,rook),piece(b-1,white,knight),piece(c-1,white,b
ishop),piece(d-1,white,queen),piece(e-1,white,king),piece(f-1,white,bishop),piece(g-1,white,knight),piece(h-1,white,rook),piece(a-2,white,pawn),piece(b-2,w
hite,pawn),piece(c-2,white,pawn),piece(d-2,white,pawn),piece(f-2,white,pawn),piece(g-2,white,pawn),piece(h-2,white,pawn),piece(e-4,white,pawn),piece(e-5,bl
ack,pawn)]

white move -> |:
```

Project 2 - task 1

- Write codes so that the chase board is drawn visually



Project 2 - task 1

- Print once after both players have completed their moves

```
% e:/Programming/TA/ASU-CSE-259-Prolog/Recitation-7/ch
?- main.

white move -> e2e4.
Working...

black move: e7e5, Rating: bookB
 +----+----+----+----+----+----+----+----+
8| *r | *n | *b | *q | *k | *b | *n | *r |
 +----+----+----+----+----+----+----+----+
7| *p | *p | *p | *p |    | *p | *p | *p |
 +----+----+----+----+----+----+----+----+
6|    |    |    |    |    |    |    |    |
 +----+----+----+----+----+----+----+----+
5|    |    |    |    | *p |    |    |    |
 +----+----+----+----+----+----+----+----+
4|    |    |    |    |  p |    |    |    |
 +----+----+----+----+----+----+----+----+
3|    |    |    |    |    |    |    |    |
 +----+----+----+----+----+----+----+----+
2|  p |  p |  p |  p |    |  p |  p |  p |
 +----+----+----+----+----+----+----+----+
1|  r |  n |  b |  q |  k |  b |  n |  r |
 +----+----+----+----+----+----+----+----+
    a    b    c    d    e    f    g    h

white move -> |:
```

Project 2 - task 1

- Write your visualization codes here

```
412  /*----------------------------------------------------------------------*/
413  /* WRITE YOUR CODE FOR TASK-1 HERE */
414  /* TASK 1: REPLACE THE print_board PREDICATE BELOW WITH YOUR CODE */
415  /*        KEEP THE NAME print_board, JUST CHANGE THE IMPLEMENTATION*/
416  /*----------------------------------------------------------------------*/
417  print_board(Board) :-      You, 2 weeks ago • added new slide
418      write('hello world'), nl.
419  /*----------------------------------------------------------------------*/
```

Project 2 - utilizing old codes

- Use the same old drawSymbol code!

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

Project 2 - utilizing old codes

- 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).
```

```
+----+----+----+----+----+----+----+----+
```

Project 2 - utilizing old codes

- 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).
```

Project 2 - utilizing old codes

- 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).
```

Project 2 - task 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.
/*
```

Project 2 - task 1

- 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
```

Project 2 - task 1

```
% 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).
```

Project 2 - task 1

```
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,

Project 2 - task 1

- 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),
```