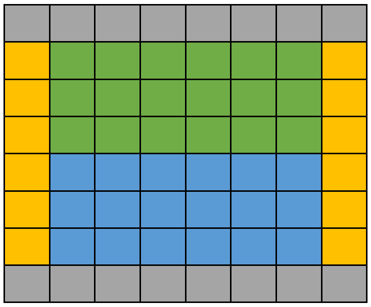
CSC384 - Assignment 2: Game Tree Search Additional Heuristic

Minimax with alpha-beta pruning, state caching, and advanced heuristic is implemented in the code.

State caching is implemented by creating a dictionary. The dictionary stores a state as the key and the node value (utility) alpha, and beta values as the value to the key. When a state that have been explored before (state is in the dictionary) is reached and the alpha and beta values matched, then the node value is returned.

The advanced heuristic is implemented based on a research paper: The game method of checkers based on alpha-beta search strategy with iterative deepening by Zhao, Wu, Liang, Lv, Yu. The heuristic takes the board piece locations into account. To evaluate the board, the board is divided into six regions, namely: two halves (our side (blue) and opposite side (green)), left edge and right edge (yellow), upper boundary and lower boundary (grey), as shown in the figure below.



The utility value is a total of four evaluation factors:

$$Utility = P_1 + P_2 + P_3 + P_4$$

1. Number of pieces left on the board
   King is worth more than a pawn. Each pawn is given 5 pts, while each king is given 17.5 pts.
   $$P_1 = \left(17.5 * (\# \, of \, red \, pawns) + 5 * (\# \, of \, red \, kings)\right) - \left(17.5 * (\# \, of \, black \, pawns) + 5 * (\# \, of \, black \, kings)\right)$$
2. Location value of each piece
   Since king piece is worth more, we want to push the pieces to be kinged. So, some (2 in this implementation) points will be given when our pieces are in the opposite area e.g., red pieces in green area. Edge pieces are pieces that are in the yellow area. They are important too since they cannot be captured. 2 points will also be given for each edge piece. Pieces in their respective home row (grey area) are notable since it will avoid opposite player pieces to be kinged. 0.5 points is given for each home piece.
   $$P_2 = \left(2 * (\#of \, opposite \, area \, reds) + 2 * (\#of \, edge \, reds) + 0.5 * (\# \, of \, home \, reds)\right) - \left(2 * (\#of \, opposite \, area \, blacks) + 2 * (\#of \, edge \, blacks) + 0.5 * (\# \, of \, home \, blacks)\right)$$

3. Value of possibility of capturing a neighboring piece
   This factors the possibility of capturing a neighboring opposite player's piece. If it is the red player's turn, from this state configuration we will check if any black pieces can be captured. And vice versa for black pieces capturing red pieces. If a pawn can be captured, 2.5 points is given. If a king capture is the next move, 8.75 points is given. This will help in choosing which child state to go to next.
   $$If \, the \, player \, color \, is \, red, \quad P_3 = 2.5(\#of \, pawns \, can \, be \, captured \, at \, least) + 8.75(\# \, of \, kings \, can \, be \, captured \, at \, least)$$
   $$If \, the \, player \, color \, is \, black, \quad P_3 = -(2.5(\#of \, pawns \, can \, be \, captured \, at \, least) + 8.75(\# \, of \, kings \, can \, be \, captured \, at \, least))$$

4. Layout value
   Neighboring pieces of the same color is a strong layout. This ensures that the pieces cannot be captured. 0.3 points is given for each piece that has a same color neighbor piece.
   $$P_4 = 0.3 * (\# \, of \, neighboring \, reds) - 0.3 * (\# \, of \, neighboring \, blacks)$$