Lab 5

1. The representation of a state includes information about:
   - the moves of each player
   - whose turn to move it is

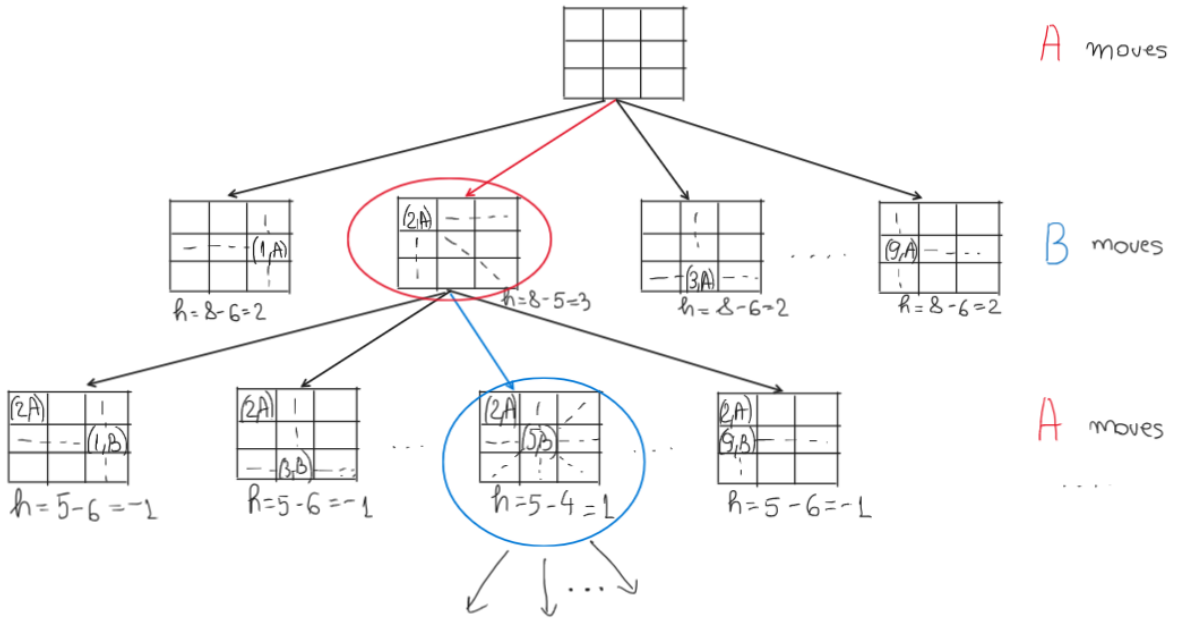
   Be careful to alternate player turns when moving from one state to another via the transitions. The final state should indicate the winner or the fact that it was a draw.

```
while(not is_final(state)):
     if it's my turn:
         choose a move (to bring opponent in the worst possible state)
     else:
         get the opponent's move

     new_state = transition(state, chosen move)
     if valid(new_state):
         state = new_state
         change turns
```

   A heuristic h can be used to help players decide their moves.
   h(state) = how good is this state for one of the players (= how worse is for the other)

   Example of an heuristic for tic-tac-toe (isomorphic to number scrabble):
   - h(state) = number of directions of 3 squares open for you – number of directions of 3 squares open for the opponent



2. MINIMAX

```
def minimax(state, depth, is_max_player):


     if (depth == 0 || is_final(state)):

            return h(state)

     if (is_max_player):

           value = - INF
           for each valid child of state:
                 value = max(value, minimax(child, depth-1, False))
      else:
           value = + INF
           for each valid child of state:
                 value = min(value, minimax(child, depth-1, True))
      return value
```

Example of heuristic (A maximizes, B minimizes):

- h(state) = number of directions of 3 squares open for A – number of directions of 3 squares open for B
    - Obs: For final states, you can modify h to guide the search even more for each player towards their winning state(s). For e.g., if in a final state MAX wins, then h could get a very big value (INF), if MIN wins, then h could get a very small value (-INF).