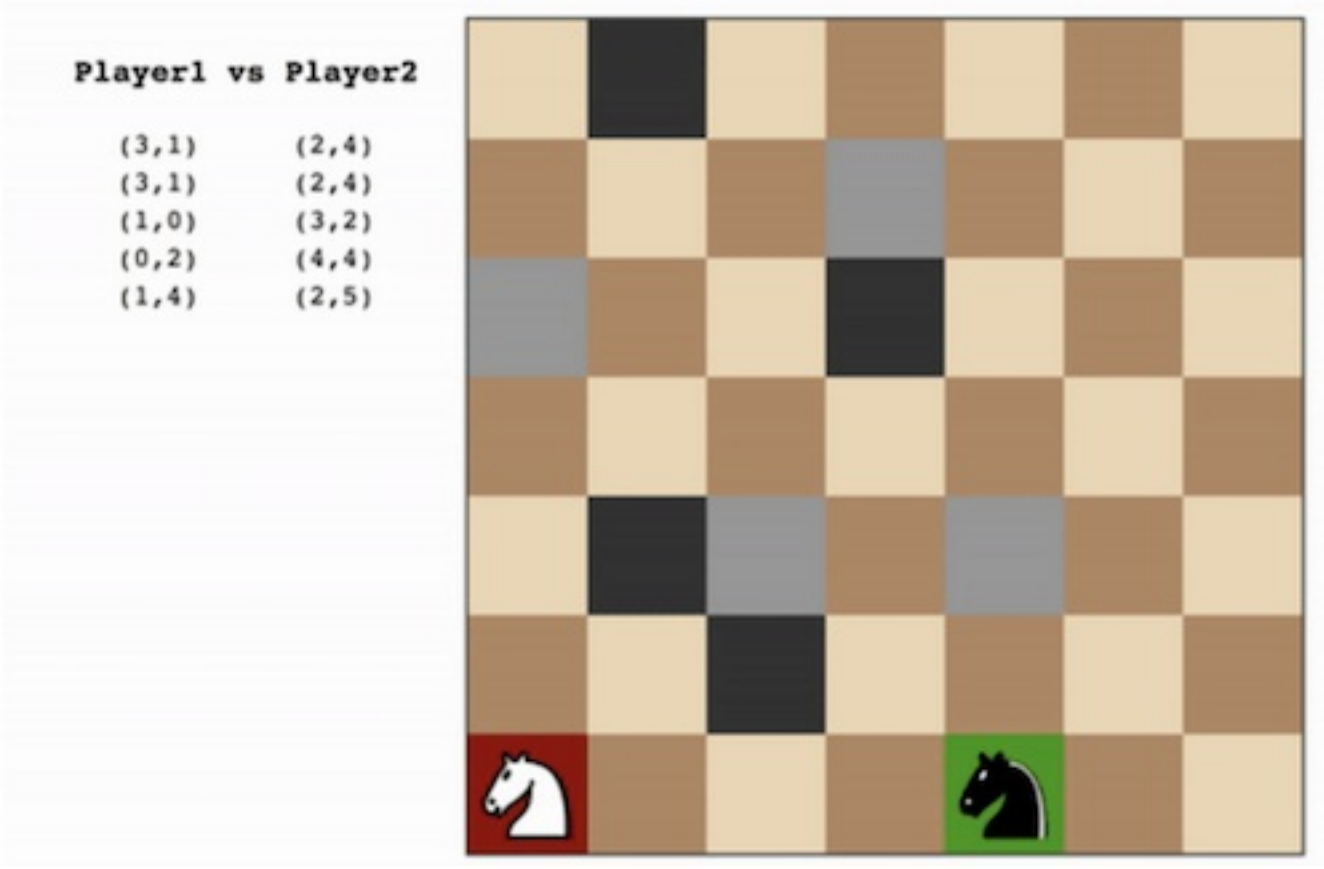
Running `python tournament.py`

```
Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                       Won | Lost   Won | Lost   Won | Lost   Won | Lost
   1       Random       7  |   3     9  |   1     8  |   2     8  |   2
   2       MM_Open      6  |   4     8  |   2     8  |   2     8  |   2
   3      MM_Center     5  |   5     7  |   3     8  |   2     7  |   3
   4     MM_Improved    6  |   4     6  |   4     5  |   5     6  |   4
   5       AB_Open      4  |   6     4  |   6     6  |   4     5  |   5
   6      AB_Center     4  |   6     7  |   3     4  |   6     5  |   5
   7     AB_Improved    4  |   6     5  |   5     6  |   4     5  |   5
--------------------------------------------------------------------------
          Win Rate:      51.4%        65.7%        64.3%        62.9%
```

My custom score methods all slightly outperformed `AB_Improved`.

Heuristic 1

Looking at the example gif, it seemed that a good strategy would be to avoid the walls and corners, because those positions limit the number of moves and as a result they'd likely be the end game scenario.



So I modified `improved_score` to subtract `percent_completed * square_distance` from it, where

`percent_completed` is (# of moves made by you and the opponent / total number of squares)t, and `square_distance` is the square of the distance from the current position to the center.

Heuristic 2

I modified `improved_score` such that initially, it tries to "chase" the opponent by aggressively reducing the number of opponent's moves. And later, it tries to care less about the opponent and more about your own survival, by trying to maximize the number of moves you have.

This is done by multiplying `percent_completed` (same as heuristic 1) with the number of moves you have, and `1 - percent_completed` (same as heuristic 2) with the number of moves opponent have.

Heuristic 3

I modified `improved_score` to favor positions that block one of the opponent's possible moves. This is done by adding to `improved_score` the `bias`, where `bias` is 2 (a small value that seems to work well) if you're in a position that blocks one of the opponent's possible moves, 0 otherwise.