```
************************
    Playing Matches
************************
```

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	9	1	10	0	10	0	10	0
2	MM_Open	6	4	7	3	8	2	9	1
3	MM_Center	6	4	10	0	9	1	9	1
4	MM_Improved	6	4	7	3	6	4	7	3
5	AB_Open	6	4	5	5	2	8	6	4
6	AB_Center	5	5	7	3	7	3	3	7
7	AB_Improved	6	4	4	6	6	4	4	6
	Win Rate:	62.9%		71.4%		68.6%		68.6%	

AB_Custom:

```
own_moves = len(game.get_legal_moves(player))
opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
return float(own_moves - 1.5*opp_moves)
```

AB_Custom_2:

```
own_moves = len(game.get_legal_moves(player))
opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
return float(own_moves - 0.5*opp_moves)
```

AB_Custom_3:

```
own_moves = len(game.get_legal_moves(player))
opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
return float(own_moves - 2*opp_moves)
```

My heuristics are exploring variations of the ***`improved_score`*** heuristic, such that in the function: ***`a*own_moves - b*opp_moves`***, different values are assigned to the weight factors ***`a`*** and ***`b`***

Since these 3 heuristics are just variations on *`improved_score`*, they are expected to search to the same depth and be of the same level of 'difficulty' (i.e. computation complexity). With these factors (depth, difficulty) being equal, I tested the different values of *`b`* and found *`1.5`* to perform the best. This suggests it is a good idea to give extra weight to the other player's number of moves remaining, but up to a limit, since *`own_moves - 2*opp_moves`* does not perform as well