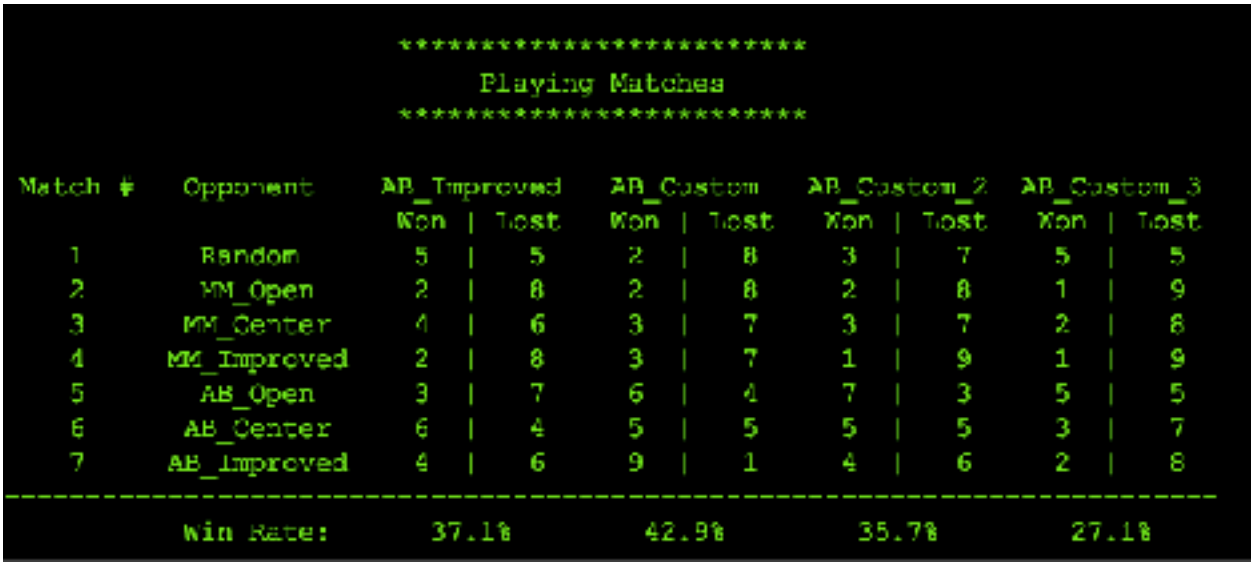
IMPLEMENTATION

For the heuristic analysis, three functions have been tried, they are:
AB_Custom = #myMoves - 2* #opponentMoves
AB_Custom_2 = #myMoves - #opponentMoves
AB_Custom_3 = #myMoves

On running the tournament.py file, the following Heuristic Analysis Results are obtained



****
Playing Matches
****

Match #	Opponent	AB_Improved		AB_Custom		AB_Custom_2		AB_Custom_3	
		Won	Lost	Won	Lost	Won	Lost	Won	Lost
1	Random	5	5	2	8	3	7	5	5
2	MM_Open	2	8	2	8	2	8	1	9
3	MM_Center	4	6	3	7	3	7	2	8
4	MM_Improved	2	8	3	7	1	9	1	9
5	AB_Open	3	7	6	4	7	3	5	5
6	AB_Center	6	4	5	5	5	5	3	7
7	AB_Improved	4	6	9	1	4	6	2	8
	Win Rate:	37.1%		42.9%		35.7%		27.1%	

Fig. 1: Heuristic analysis with the currently implemented heuristics in the code

OBSERVATIONS AND INFERENCES

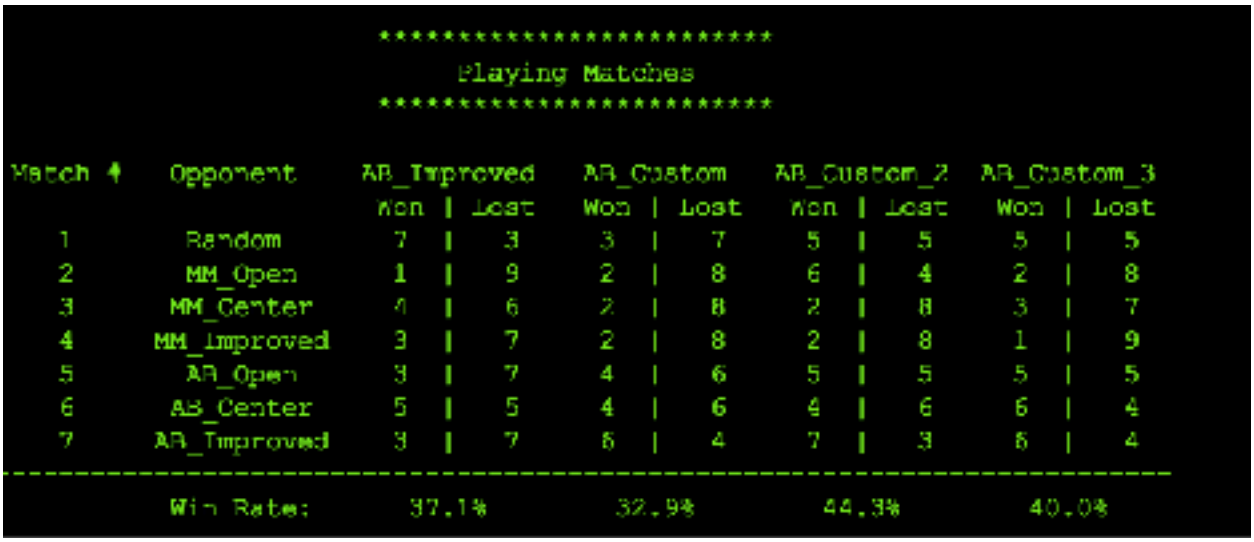
As can be seen from the figure above, AB_Custom which is the first heuristic gives a better win rate than the rest and hence I've chosen heuristic 1 as the final heuristic for the code.

The probable reason for that happening are that while #myMoves provides us with an estimate for the number of moves left for our agent, it does not save us from the evil clutches of the horizon effect. In that scenario #myMoves - #opponentMoves performs better because here, not only are we trying to win but we are also trying to thwart our opponents chances at winning. In #myMoves - 2*#opponentMoves we concentrate a bit more on the latter part resulting in this heuristic performing even better in situations of a horizon effect.

SIDE EXPERIMENTATION

On a side note: I've also tried a heuristic with a slight modulation to heuristic 1. It was basically: #myMoves - 10* #opponentMoves. Counter-intuitively, it does not give a better performance than heuristic 1. Here is a comparison for:
AB_Custom = #myMoves - 10* #opponentMoves
AB_Custom_2 = #myMoves - 2* #opponentMoves

AB_Custom_3 = #myMoves



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

Match #   Opponent      AB_Improved     AB_Custom     AB_Custom_2   AB_Custom_3
                        Won | Lost    Won | Lost    Won | Lost    Won | Lost
   1        Random       7  |  3       3  |  7       5  |  5       5  |  5
   2        MM_Open      1  |  9       2  |  8       6  |  4       2  |  8
   3       MM_Center     4  |  6       2  |  8       2  |  8       3  |  7
   4     MM_Improved     3  |  7       2  |  8       2  |  8       1  |  9
   5        AB_Open      3  |  7       4  |  6       5  |  5       5  |  5
   6       AB_Center     5  |  5       4  |  6       4  |  6       6  |  4
   7     AB_Improved     3  |  7       6  |  4       7  |  3       6  |  4
--------------------------------------------------------------------------
          Win Rate:      37.1%         32.9%         44.3%         40.0%
```

Fig. 2: Heuristic analysis for an experimental code.