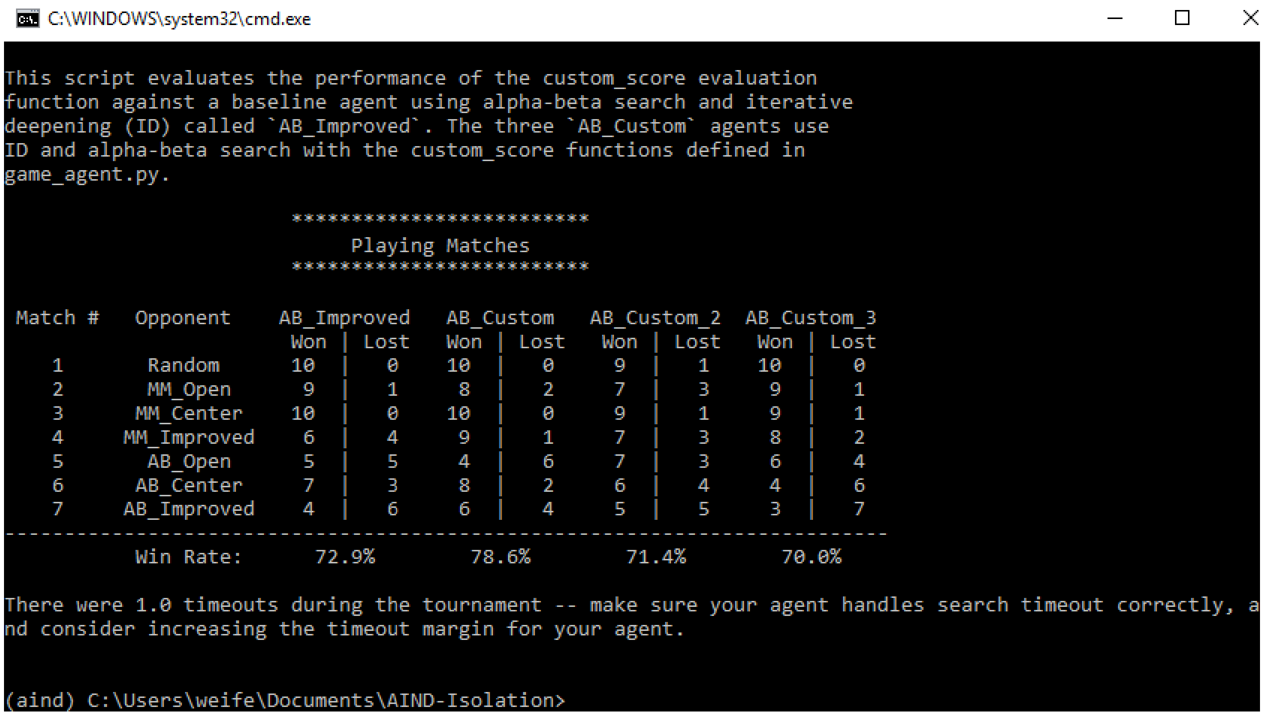
Analysis of Heuristic Evaluation Functions for Game-Playing Agent

Three heuristic evaluation functions were created to estimate the chance of winning when the agent has to cut off the search due to time limit.

- Function 1: # of open moves of the player – 1.2 * # of open moves of the opponent + distance to center
- Function 2: # of unique open moves of the player – 1.2 * # of unique open moves of the opponent
- Function 3: # of open moves of the player – 1.2 * # of open moves of the opponent

Three agents using the above three heuristic evaluation functions, together with a baseline agent using the alpha-beta search and iterative deepening algorithm AB_Improved, were played against seven opponents. Tournament results are shown in the screen shot below:



```
This script evaluates the performance of the custom_score evaluation
function against a baseline agent using alpha-beta search and iterative
deepening (ID) called `AB_Improved`. The three `AB_Custom` agents use
ID and alpha-beta search with the custom_score functions defined in
game_agent.py.

                        *************************
                             Playing Matches
                        *************************

 Match #   Opponent    AB_Improved   AB_Custom   AB_Custom_2  AB_Custom_3
                        Won | Lost   Won | Lost   Won | Lost   Won | Lost
    1       Random      10  |   0    10  |   0     9  |   1    10  |   0
    2       MM_Open      9  |   1     8  |   2     7  |   3     9  |   1
    3      MM_Center    10  |   0    10  |   0     9  |   1     9  |   1
    4     MM_Improved    6  |   4     9  |   1     7  |   3     8  |   2
    5       AB_Open      5  |   5     4  |   6     7  |   3     6  |   4
    6      AB_Center     7  |   3     8  |   2     6  |   4     4  |   6
    7     AB_Improved    4  |   6     6  |   4     5  |   5     3  |   7
--------------------------------------------------------------------------
           Win Rate:      72.9%        78.6%        71.4%        70.0%

There were 1.0 timeouts during the tournament -- make sure your agent handles search timeout correctly, a
nd consider increasing the timeout margin for your agent.


(aind) C:\Users\weife\Documents\AIND-Isolation>
```

Overall, Function 1 has the highest win rate: 78.6%, followed by the default agent AB_Improved (72.9%), Function 2 (71.40%), and Function 3 (70.0%). Therefore, I would choose Function 1. Below are three reasons I chose Function 1:

1. Function 1 has the highest winning rate in multiple tournaments
2. Although this function needs to calculate three terms, they are very easy to calculate. This means it should not affect search time at each level too much and therefore should not prevent it from going deeper levels compared to other algorithms

3. I believe Function 1 is closer to the optimal evaluation function because it takes into account three factors which are essential to game winning. Therefore, this function is expected to perform better with a wider variety of opponent algorithms