Udacity AI Nanodegree
Project 2 – AIND-Isolation
Daniele Pestilli

Heuristic #1 – Number of moves

This heuristic takes into consideration the number of legal moves that each player has. The returned result is the difference between one's own moves and twice the moves of the opponent. The reasoning behind this is that if one's own moves are greater than twice the moves of the opponent, the chances of winning area greater. If, on the other hand, one's own moves are less than twice the opponent's moves, there's a good chance of failure. This estimation yields the following results in a tournament:

Playing Matches:

Match 1:	Student	vs	Random	Result: 17 to 3
Match 2:	Student	vs	MM_Null	Result: 16 to 4
Match 3:	Student	vs	MM_Open	Result: 10 to 10
Match 4:	Student	vs	MM_Improved	Result: 9 to 11
Match 5:	Student	vs	AB_Null	Result: 11 to 9
Match 6:	Student	vs	AB_Open	Result: 13 to 7
Match 7:	Student	vs	AB_Improved	Result: 10 to 10

Results:

Student **61.43%**

Heuristic #2 – Center proximity

This heuristic takes into consideration the player's location. It then determines the center of the game board and returns the player's distance from the center. The reasoning behind this is that a player at the center of the board has the greatest mobility, whereas players near the edges are intrinsically constrained by the board. This is true for games such as tic-tac-toe where the center of the board is the most highly valued position as it grants access to any other position on the grid. This heuristic yields the following results in a tournament:

Playing Matches:

Match 1:	Student	vs	Random	Result: 18 to 2
Match 2:	Student	vs	MM_Null	Result: 13 to 7
Match 3:	Student	vs	MM_Open	Result: 13 to 7
Match 4:	Student	vs	MM_Improved	Result: 8 to 12
Match 5:	Student	vs	AB_Null	Result: 15 to 5
Match 6:	Student	vs	AB_Open	Result: 12 to 8
Match 7:	Student	vs	AB_Improved	Result: 10 to 10

Results:

Student **63.57%**

Heuristic #3 – Mobility

This heuristic takes into consideration each player's legal moves. It then sums up the number of legal moves each player has. The returned value is the difference in each player's mobility plus the

difference between each player's number of moves. The reasoning behind this is that the mobility and number of potential moves should give a good indication as to which player has the best potential to maneuver around the game board, thus being able to win the game. This heuristic yields the following results in a tournament:

Playing Matches:

Match 1:	Student	vs	Random	Result: 18 to 2
Match 2:	Student	vs	MM_Null	Result: 16 to 4
Match 3:	Student	vs	MM_Open	Result: 10 to 10
Match 4:	Student	vs	MM_Improved	Result: 10 to 10
Match 5:	Student	vs	AB_Null	Result: 12 to 8
Match 6:	Student	vs	AB_Open	Result: 11 to 9
Match 7:	Student	vs	AB_Improved	Result: 15 to 5

Results:

Student **65.71%**

Ultimately, I selected this final heuristic as the one to use in the custom_score method, both because it yields the best result and because the idea of mobility via potential moves is probably the most logical in terms of determining a winner for this particular game.