Heuristic Analysis

First of all, I was not expecting that coming up a heuristic would be as difficult as it proved to be. My approach to solving was first looking at sample_players.py and examining how those heurisitics were implemented. After that I had a somewhat good understanding about how to approach the heuristic.

Analysis 1:

For my first heuristic, my function would calculate the difference between the current player's and opponent's position from the center of the grid. From playing the game as you are closer to the center you have more possibilities to be around legal moves. As you approach the edges you remove possibilities from the number of legal moves you have. Calculating the distance was merely applying the distance formula for each player's current location coordinates. After playing around with this heuristic I also added the difference of available moves of the player and opponent to the heuristic which resulted in better performance. My thinking behind this, is if you have more available moves than your opponent then you should have a better chance of winning.

Analysis 2:

My idea behind this heuristic was quite simple. If the current player has moves that the opposing opponent has then this would work in their favor by having the possibility to make that move and thus eliminating a possible move from the opponent. Calculating this would consist of running a for loop and every time an opponent move appears in the current player's possible moves add one to a counter. At the end of the function just return the difference of possible moves between the player and opponent plus the number of moves they share.

Analysis 3:

My last heuristic did not consist of having a clear reason as to why I chose this, instead this heuristic came from trial and error. Playing around with the possible values and changing numbers gave a mix range of results and ultimately due to time constraint I settled on the one I implemented. This heuristic adds the both the number of possible values from the current player and opponent and then subtracts this from the difference of the number of spaces

available and the number of moves that have occurred in the game. This function on average resulted in a result of a little over 50% win rate.