Heuristic Analysis

For this project, I studied with Udacity AIND Courses and came up with three different heuristics on the isolation game. And here is the summary of the result. The chart below presents the outcome on how each heuristic performs against the 'ID_improved' which was provided by the Udacity Staff. Based on the performance, the best heuristic to choose is the third one which I used to submit for review.

	ID_improved	Student
Heuristic 1	*72.86%*	*73.57%*
Heuristic 2	*68.57%*	*75%*
Heuristic 3	***64.29%***	***74.29%***

Heuristic 1

Heuristic one simply returns the different number of moves left for each players. If the player and its opponent has the same number of moves, then the return value is zero. Whoever has the higher return value, wins the game. It is not a good heuristic, although it is easy to interpret and fast to compute. It is oblivious to the notion of positional advantage. Therefore, in the next section, I add the positional advantage implementation into the heuristic 2.

By running the tournament.py, we can analysis the result as showing below:

```
**************************
 Evaluating: ID_Improved
**************************

Playing Matches:
----------
  Match 1: ID_Improved vs   Random      Result: 18 to 2
  Match 2: ID_Improved vs   MM_Null     Result: 16 to 4
  Match 3: ID_Improved vs   MM_Open          Result: 15 to 5
  Match 4: ID_Improved vs MM_Improved       Result: 13 to 7
  Match 5: ID_Improved vs   AB_Null      Result: 13 to 7
  Match 6: ID_Improved vs   AB_Open      Result: 13 to 7
  Match 7: ID_Improved vs AB_Improved        Result: 14 to 6


Results:
----------
ID_Improved        72.86%

**************************
   Evaluating: Student
```

****

Playing Matches:

Match 1: Student vs Random Result: 18 to 2
Match 2: Student vs MM_Null Result: 14 to 6
Match 3: Student vs MM_Open Result: 15 to 5
Match 4: Student vs MM_Improved Result: 13 to 7
Match 5: Student vs AB_Null Result: 19 to 1
Match 6: Student vs AB_Open Result: 13 to 7
Match 7: Student vs AB_Improved Result: 11 to 9

Results:

Student 73.57%

Heuristic 2

With this heuristic, if a player is closer to the center of the board, it is more than likely that this player will do better than a player who has the moves near the edges of the board. This heuristic performs a better result, but not good enough because the positional advantage that does not know what to do next if the position is near the center of the board. Therefore, I take a different approach which is derived from this corners, edges, and center concept.

By running the tournament.py, we can analysis the result as showing below:

****
Evaluating: ID_Improved
****

Playing Matches:

Match 1: ID_Improved vs Random Result: 15 to 5
Match 2: ID_Improved vs MM_Null Result: 14 to 6
Match 3: ID_Improved vs MM_Open Result: 15 to 5
Match 4: ID_Improved vs MM_Improved Result: 13 to 7
Match 5: ID_Improved vs AB_Null Result: 13 to 7
Match 6: ID_Improved vs AB_Open Result: 15 to 5
Match 7: ID_Improved vs AB_Improved Result: 11 to 9

Results:

ID_Improved 68.57%

```
**************************
  Evaluating: Student
**************************

Playing Matches:
----------
 Match 1:  Student  vs  Random        Result: 19 to 1
 Match 2:  Student  vs  MM_Null       Result: 15 to 5
 Match 3:  Student  vs  MM_Open       Result: 13 to 7
 Match 4:  Student  vs MM_Improved    Result: 15 to 5
 Match 5:  Student  vs  AB_Null       Result: 16 to 4
 Match 6:  Student  vs  AB_Open       Result: 15 to 5
 Match 7:  Student  vs AB_Improved    Result: 12 to 8


Results:
----------
Student         75.00%
```

Heuristic 3

In this approach, I stills output the difference in the number of moves available to two players but I added the penalizing and rewarding moves to the maximizing player that in the corner and minimizing player in the corner, respectively. The result compares to heuristic two increases a bit little since they are almost identical in terms of the logic. The difference I used a different way to return the moves on the behavior of two players when their remaining moves are located in the corners. The penalizing and rewards method works more accurately than the heuristic two. However, there are is still a way to increase the performance by implement a heuristic more precisely based on how long the player can be survived, for example, and adding this to the positional advantage.

By running the tournament.py, we can analysis the result as showing below:

```
**************************
 Evaluating: ID_Improved
**************************

Playing Matches:
----------
 Match 1: ID_Improved vs   Random      Result: 16 to 4
 Match 2: ID_Improved vs   MM_Null     Result: 17 to 3
 Match 3: ID_Improved vs   MM_Open          Result: 11 to 9
 Match 4: ID_Improved vs MM_Improved       Result: 13 to 7
 Match 5: ID_Improved vs   AB_Null      Result: 14 to 6
```

```
  Match 6: ID_Improved vs   AB_Open     Result: 11 to 9
  Match 7: ID_Improved vs AB_Improved         Result: 8 to 12


Results:
----------
ID_Improved         64.29%

*************************
   Evaluating: Student
*************************

Playing Matches:
----------
  Match 1:   Student   vs   Random          Result: 16 to 4
  Match 2:   Student   vs   MM_Null         Result: 17 to 3
  Match 3:   Student   vs   MM_Open         Result: 13 to 7
  Match 4:   Student   vs MM_Improved   Result: 13 to 7
  Match 5:   Student   vs   AB_Null         Result: 15 to 5
  Match 6:   Student   vs   AB_Open         Result: 18 to 2
  Match 7:   Student   vs AB_Improved     Result: 12 to 8


Results:
----------
Student           74.29%
```