Heuristic Analysis

0. Number of my legal moves - Number of opponent's legal moves

```own_moves = len(game.get_legal_moves(player))
   opp_moves = len(game.get_legal_moves(game.get_opponent(player)))
   return float(own_moves - opp_moves)
```

- This is the base function implemented in the agents.
- The difference between my legal moves and opponent's is directly related to the final outcome of the game.
- It's fast to compute.

1. Number of common moves

```own_moves = game.get_legal_moves(player)
   opp_moves = game.get_legal_moves(game.get_opponent(player))
   return float(len(set(own_moves) & set(opp_moves)))
```

- The intention was to get into opponent's way by maximizing the joint spaces
- The result showed the lowest win rate, implies chasing after opponent is not a good strategy
- It's fast to compute.

2. Number of my legal moves

```own_moves = len(game.get_legal_moves(player))
   return float(own_moves)
```

- Now only try to maximize my legal moves
- The result win rate is low compare to the base function and ID_Improved, implies I can't totally ignore the opponent's position and moves
- It's fast to compute.

3. (Number of my legal moves - Number of opponent's legal moves) - 1 step forward

```own_moves = len(game.get_legal_moves(player))
   opp_moves = len(game.get_legal_moves(game.get_opponent(player)))

   for move in game.get_legal_moves(player):
      game_forecast = game.forecast_move(move)
      own_moves += len(game_forecast.get_legal_moves(player))
      opp_moves += len(game_forecast.get_legal_moves(game_forecast.get_opponent(player)))
```

```
    return float(own_moves - opp_moves)
```

- This is the base function implemented with one step forward, so we sum up the total legal moves for the next step.
- It further emphasize the idea of maximizing the difference between my vs opponents legal moves.
- It's slower to compute compare to others, so it can't search as deep.

4. Combina 0 and 3 depends on the stage of the game

```
if len(game.get_blank_spaces()) > 20:
        return custom_score0(game, player)
  else:
        return custom_score3(game, player)
```

- While there are more than 20 blank spaces, which is about mid-game, we use the base score, and use the 1-step-forward after the number of legal moves are more limited.
- The difference between my legal moves and opponent's is directly related to the final outcome of the game. And search one step forward in the later stage enhance the idea.
- It's faster than use 1-step-forward during the whole game.
- This is chosen as the final custom score.

	0		1		2		3		FINAL	
	ID	Student	ID	Student	ID	Student	ID	Student	ID	Student
Random	17	18	18	19	18	18	18	16	19	18
MM_Null	17	17	15	14	18	16	17	19	18	19
MM_Open	16	13	16	10	14	13	14	13	12	17
MM_Improved	15	14	14	14	15	12	14	15	15	14
AB_Null	18	16	15	12	11	14	17	15	15	16
AB_Open	15	15	14	12	15	12	11	14	15	14
AB_Improved	14	12	13	11	11	10	12	9	13	13
Total Win	112	105	105	92	102	95	103	101	107	111
% Win	80.00%	75.00%	75.00%	65.71%	72.86%	67.86%	73.57%	72.14%	76.43%	79.29%