Heuristics Analysis:

Totally, I have written 3 heuristics, and all three heuristics have below 2 parts in common :

1. If the Non-Opposite player is the winner, return an infinite reward (+inf)
2. If the Opposite player is the winner, return an infinite penalty (-inf)

However, if the game hasn't been ended yet, the 3 heuristics differ as follows:

1. The first heuristic, custom_score_1, assigns a relatively High Reward for the number of moves the player has, and low penalty for the number of moves opponent has.
2. The second heuristic, custom_score_2, assigns a relatively low Reward for the number of moves the player has, and High penalty for the number of moves opponent has.
3. The third heuristic, custom_score_3, assigns Equal Reward for the number of moves the player has, and for the number of moves opponent has.

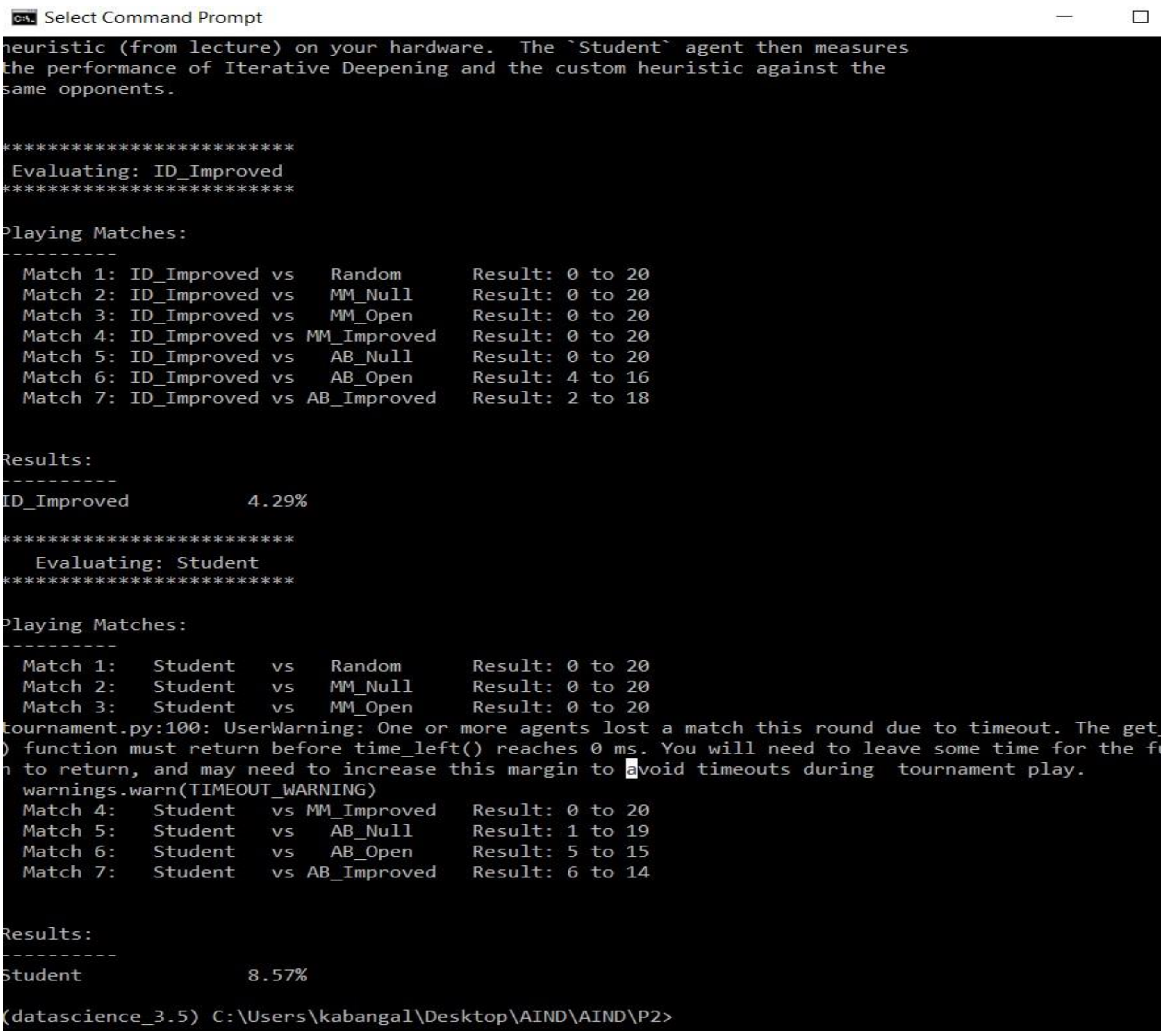
The reward and penalties are scaled according the variables reward_coffiecient and penalty_coefficient respectively.

Out of the 3 heuristics above, I chose the second heuristic custom_score_2, where in reward_coffiecient is **2** and penalty_coefficient is **4**.

The numbers 2 and 4 were arrived by empirically verifiying the results, assuming to be a sort of a hyperparameter that I have to tune.

The time complexity of all the 3 heuristics remain the same, as they are almost identical, but only differ by the coefficients.

The screenshots of the results of best heuristic is below:


Select Command Prompt

```
heuristic (from lecture) on your hardware.  The `Student` agent then measures
the performance of Iterative Deepening and the custom heuristic against the
same opponents.


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

Playing Matches:
----------
  Match 1: ID_Improved vs   Random      Result: 0 to 20
  Match 2: ID_Improved vs   MM_Null     Result: 0 to 20
  Match 3: ID_Improved vs   MM_Open     Result: 0 to 20
  Match 4: ID_Improved vs MM_Improved   Result: 0 to 20
  Match 5: ID_Improved vs   AB_Null     Result: 0 to 20
  Match 6: ID_Improved vs   AB_Open     Result: 4 to 16
  Match 7: ID_Improved vs AB_Improved   Result: 2 to 18


Results:
----------
ID_Improved          4.29%

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

Playing Matches:
----------
  Match 1:   Student   vs   Random      Result: 0 to 20
  Match 2:   Student   vs   MM_Null     Result: 0 to 20
  Match 3:   Student   vs   MM_Open     Result: 0 to 20
tournament.py:100: UserWarning: One or more agents lost a match this round due to timeout. The get_
) function must return before time_left() reaches 0 ms. You will need to leave some time for the f
n to return, and may need to increase this margin to avoid timeouts during  tournament play.
  warnings.warn(TIMEOUT_WARNING)
  Match 4:   Student   vs MM_Improved   Result: 0 to 20
  Match 5:   Student   vs   AB_Null     Result: 1 to 19
  Match 6:   Student   vs   AB_Open     Result: 5 to 15
  Match 7:   Student   vs AB_Improved   Result: 6 to 14


Results:
----------
Student              8.57%

(datascience_3.5) C:\Users\kabangal\Desktop\AIND\AIND\P2>
```