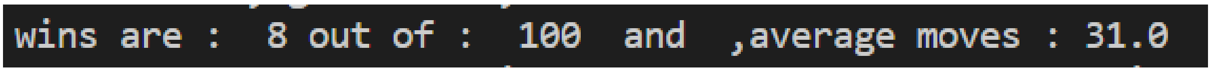
Four-Connect Game Tree Implementation

1) Stats for Game-Tree Depth = 3

The evaluation functions used for the game are as follows :

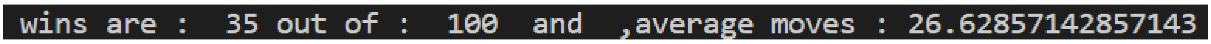
1) Initially just random values were assigned to the utility functions to get a base win rate based on random values.



```
wins are :  8 out of :  100  and  ,average moves : 31.0
```

Total Games played : 100
Total Games won by Game Tree Player : 8
Average moves played when player 2 won : 31.0
Win rate : 8 %

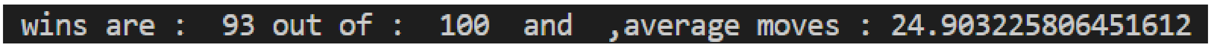
2) Each set of 4 for max player was given a Utility value of +10000 while the same for the min player was given a utility value of -10000.



```
wins are :  35 out of :  100  and  ,average moves : 26.62857142857143
```

Total Games played : 100
Total Games won by Game Tree Player : 35
Average moves played when player 2 won : 26.63
Win rate : 35 %

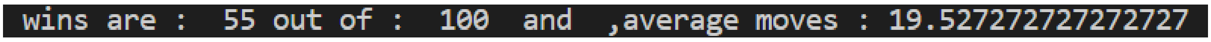
3) A set of two was given a Utility value of +10, a set of 3 was given a Utility value of +100 while a set of 4 was given a utility of +10000 and same for the min player in negative .



```
wins are :  93 out of :  100  and  ,average moves : 24.903225806451612
```

Total Games played : 100
Total Games won by Game Tree Player : 93
Average moves played when player 2 won : 24.90
Win rate : 93 %

4) Taking many factors into consideration such as current available sets, proximity to the centre and number of available winning positions for the opponent.

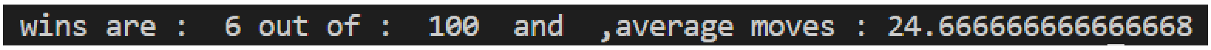


```
wins are :  55 out of :  100  and  ,average moves : 19.527272727272727
```

Total Games played : 100
Total Games won by Game Tree Player : 55
Average moves played when player 2 won : 19.53
Win rate : 73 %

2) Implementing Alpha-Beta Pruning

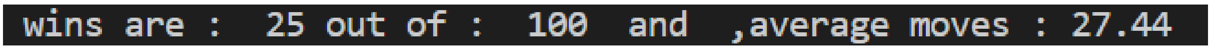
1) Initially just random values were assigned to the utility functions to get a base win rate based on random values.



```
wins are :  6 out of :  100  and  ,average moves : 24.666666666666668
```

Total Games played : 100
Total Games won by Game Tree Player : 6
Average moves played when player 2 won : 24.66
Win rate : 6%

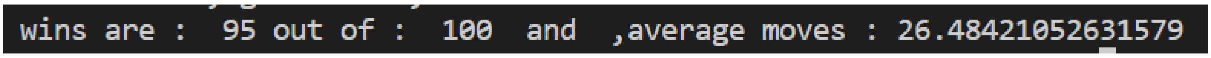
2) Each set of 4 for max player was given a Utility value of +500 while the same for the min player was given a utility value of -500.



```
wins are :  25 out of :  100  and  ,average moves : 27.44
```

Total Games played : 100
Total Games won by Game Tree Player : 25
Average moves played when player 2 won : 27.44
Win rate : 25 %

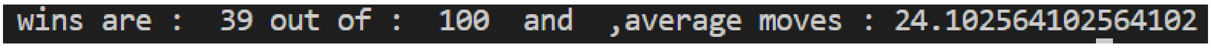
3) A set of two was given a Utility value of +10, a set of 3 was given a Utility value of +100 while a set of 4 was given a utility of +1000 and same for the min player in negative.



```
wins are :  95 out of :  100  and  ,average moves : 26.48421052631579
```

Total Games played : 100
Total Games won by Game Tree Player : 95
Average moves played when player 2 won : 26.48
Win rate :

4) Taking many factors into consideration such as current available sets, proximity to the centre and number of available winning positions for the opponent.



```
wins are :  39 out of :  100  and  ,average moves : 24.102564102564102
```

Total Games played : 100
Total Games won by Game Tree Player : 39
Average moves played when player 2 won : 24.10
Win rate : 39 %

3) Increasing Game Tree Depth to 5

1) Initially just random values were assigned to the utility functions to get a base win rate based on random values.

   Total Games played : 100
   Total Games won by Game Tree Player : 7
   Average moves played when player 2 won : 26.27
   Win rate : 7%

2) Each set of 4 for max player was given a Utility value of +500 while the same for the min player was given a utility value of -500.

   Total Games played : 100
   Total Games won by Game Tree Player : 35
   Average moves played when player 2 won : 24.4
   Win rate : 35%

3) A set of two was given a Utility value of +10, a set of 3 was given a Utility value of +100 while a set of 4 was given a utility of +1000 and same for the min player in negative.

   Total Games played : 100
   Total Games won by Game Tree Player : 98
   Average moves played when player 2 won : 22.8
   Win rate : 98%

4) Taking many factors into consideration such as current available sets, proximity to the centre and number of available winning positions for the opponent.

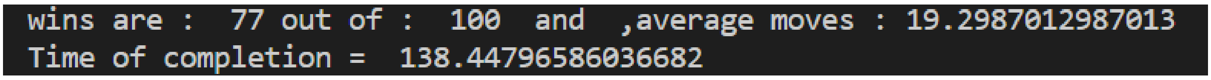
   Total Games played : 100
   Total Games won by Game Tree Player : 48
   Average moves played when player 2 won : 22.62
   Win rate : 48 %

4) Implementing Move Ordering (Assuming Game-Tree Depth = 3)

The move ordering strategy used here was starting from the middle column (Column 3), as the player having the middle of the game board had a higher probability of winning the game faster. Similarly from there we move onto Column 2 and 4, then Column 1 and 5, and Finally Column 0 and 6. This would get us the higher utility functions beforehand hence helping us to prune out the weaker results.

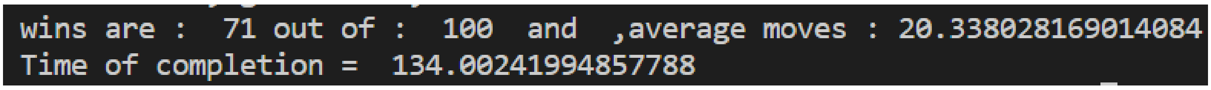
Total Games played : 100

Total game time play for 100 games before move ordering : 138.4479



```
wins are :  77 out of :  100  and  ,average moves : 19.2987012987013
Time of completion =  138.44796586036682
```

Total game time play for 100 games after move ordering : 134.0021 seconds



```
wins are :  71 out of :  100  and  ,average moves : 20.338028169014084
Time of completion =  134.00241994857788
```

Does increasing the cut-off depth to 5 increase the frequency of winning? Does the average number of moves before a win decrease?

Ans.

Increasing the cutoff depth to 5 does increase the winning rate as shown earlier in this report, this is because the game tree player knows the game board 5 moves ahead in time, which helps it choose a better action out of a greater variety of actions. Hence, this justifies the increase in winning rate for player 2.

Average moves before a win should ideally reduce in case we increase cutoff depth, as the Game Tree player can look ahead more moves ahead in time, so it could choose moves which could end the game early, in the above report, except for one evaluation function, the rest of the functions have the average number of moves reduced, hence confirming this fact.