Networks Lab 5

1 Students

- Moustafa Esam El-Sayed Amer - 21011364
- Ahmed Mostafa Elmosrsi Amer - 21010189
- Mahmoud Abdelghany Darwish Ads Darwish - 21011275

- Github Repo: Github Repo

2 Explanation

2.1 Heuristic

Our Heuristic is based on the idea of useful features in the game and their weights (inspired by Deep Blue idea of using features and weight in chess)

Features and their weights: Getting a connected four → 10000 Getting an open (could be used to build a sequence of four) three → 5000 Getting an open two → 1000

2.2 Expectiminmax

Useful DataStructures:

Strings to store states Node to store Tree values (for tree printing) List to track available places to play in columns Hashmap to store results of states not to re-work them again if found

Optimization idea is that this problem has many similar states that will expand the same subTree unnecessarily.

2.3 Minmax

It assumes that both players play optimally. The algorithm simulates all possible game states up to a certain depth k by recursively evaluating moves.
Maximizing Player: Tries to maximize the score.
Minimizing Player: Tries to minimize the score.
The Game can be started for the Agent as a first player by setting it as a maximizer, or second player by setting it as minimizer without affecting the heuristic (for the implementation provided).
Steps Of Algorithm:
Generate the game tree up to K levels, at which either reach K or maxdepth (42) it terminates the node by evaluating the heuristic at this current state. Also, on reaching a state pre-visited (aka Transposition) it does not recompute this state and take its previouslt stored values.
Backpropagate scores, selecting the maximum for the maximizing player and minimum for the minimizing player.
Implications:
The Algorithm introduces exponential memory growth, thus not efficient for large K values and won't be realistic in runtime as it expands all possible moves with a branching factor of 7. Game Representation: The Connect Four board is represented as a 2D grid. Legal moves are columns where new pieces can be dropped.
Heuristic Function: Evaluates board states based on patterns (e.g., consecutive pieces, potential threats).

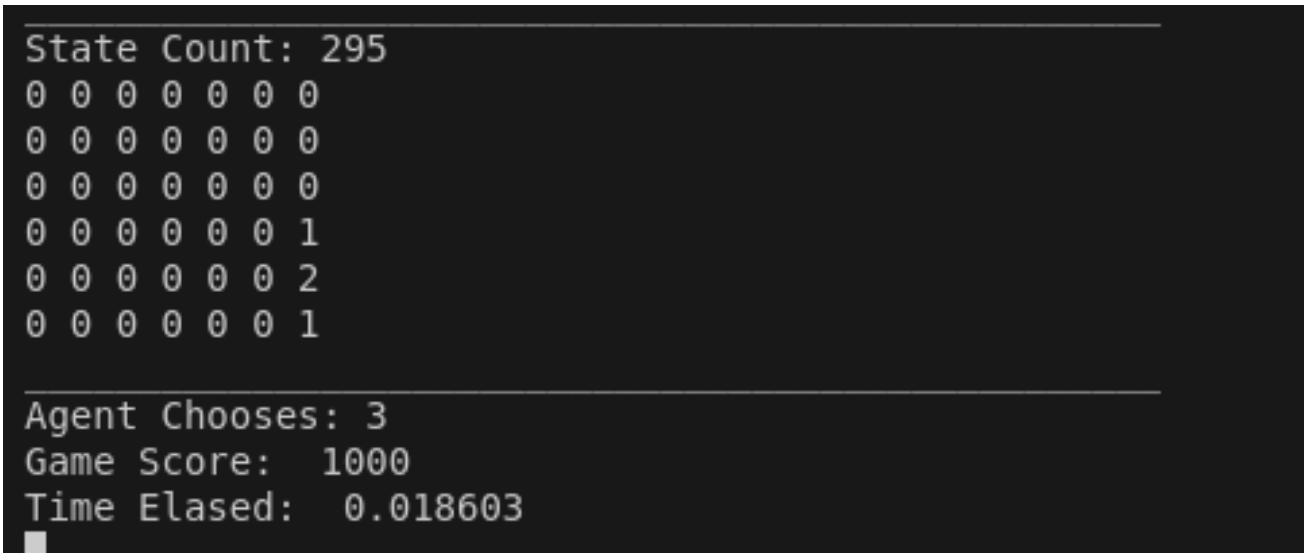
2.4 AlphaBeta

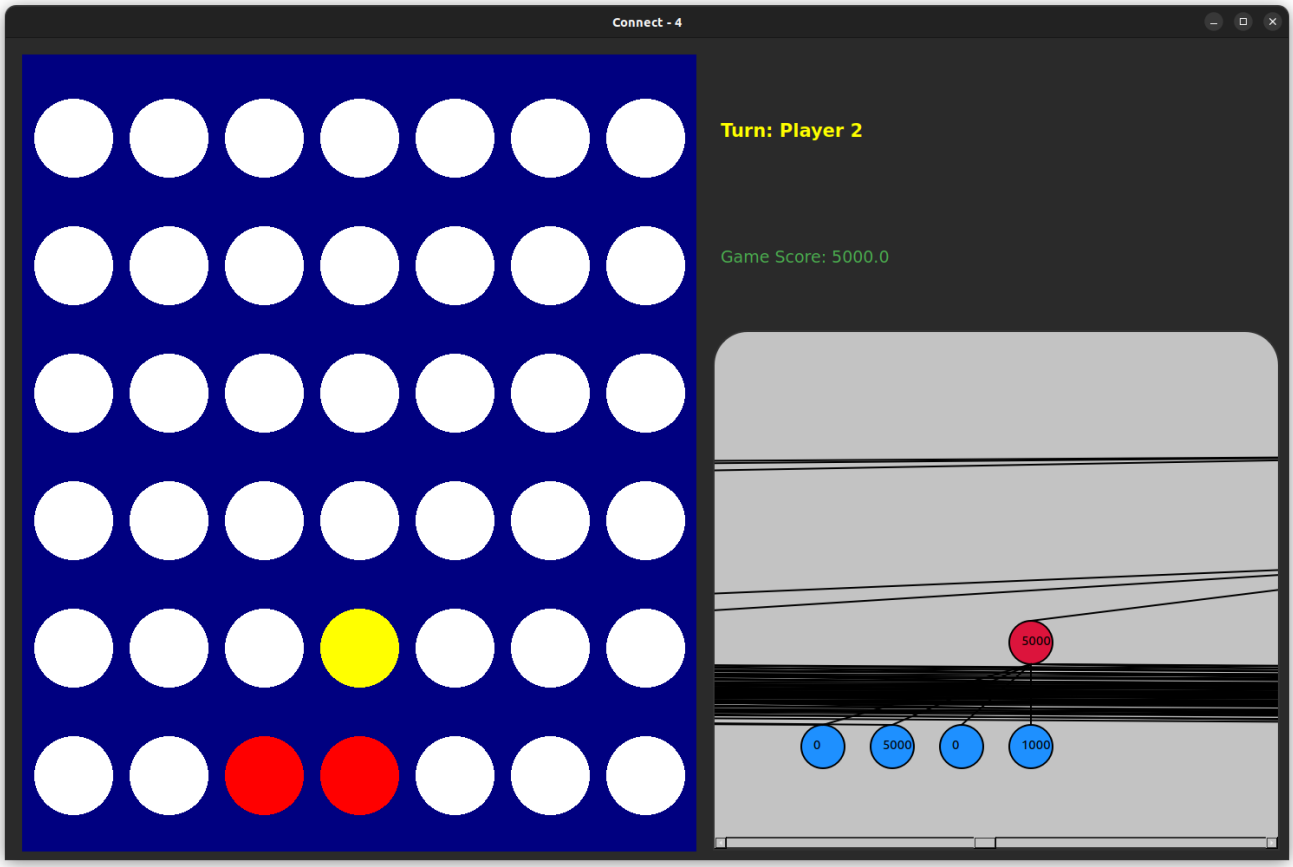
Alpha-Beta Pruning optimizes the Minimax algorithm by reducing the number of nodes evaluated in the game tree.
Pruning Logic:
While traversing the tree, if a node's value is worse than a previously examined alternative, the branch is skipped. This eliminates the need to explore branches that cannot affect the final decision.
It achieves the same result as Minimax but with significantly fewer computations. In the Implementation provided, it reduces the nodes expanded greatly, thus optimizes the solution runtime.

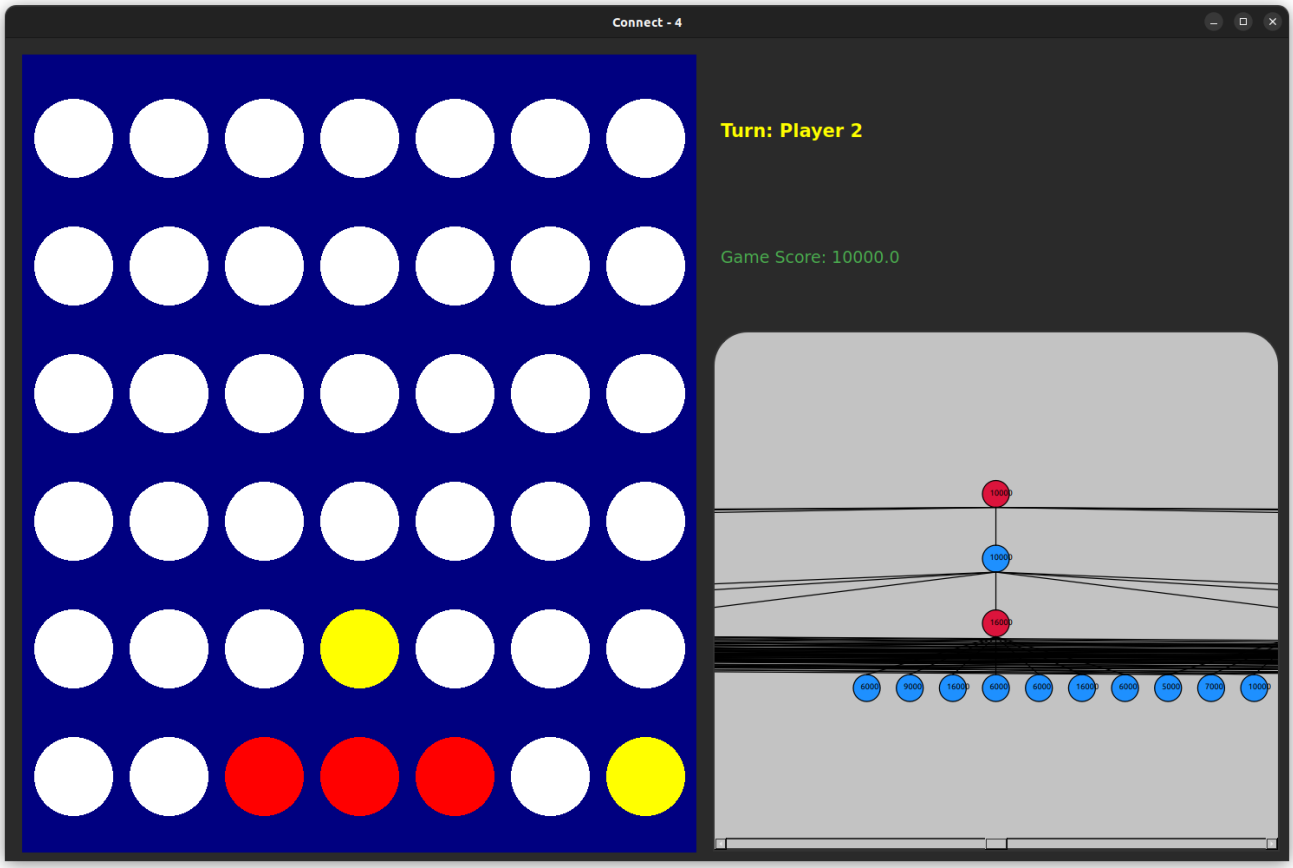
3 Tests

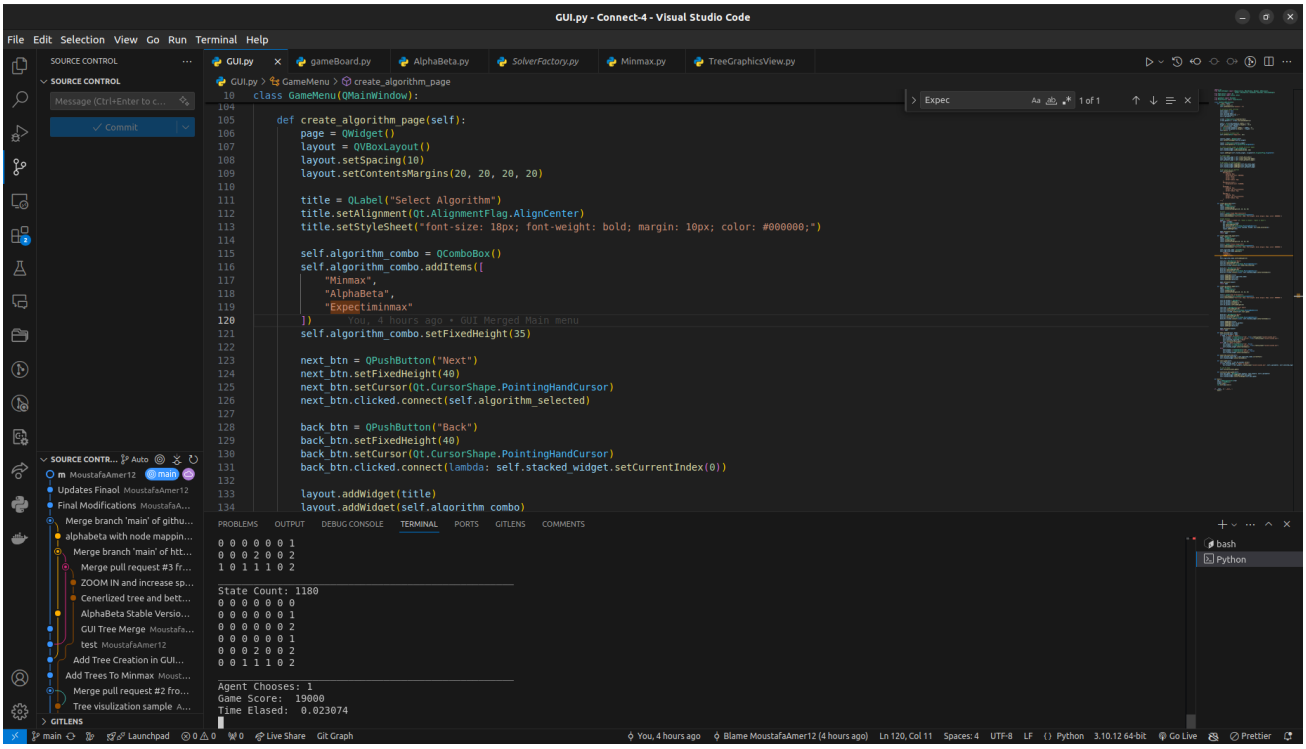
3.1 K = 3

- Minmax Algorithm

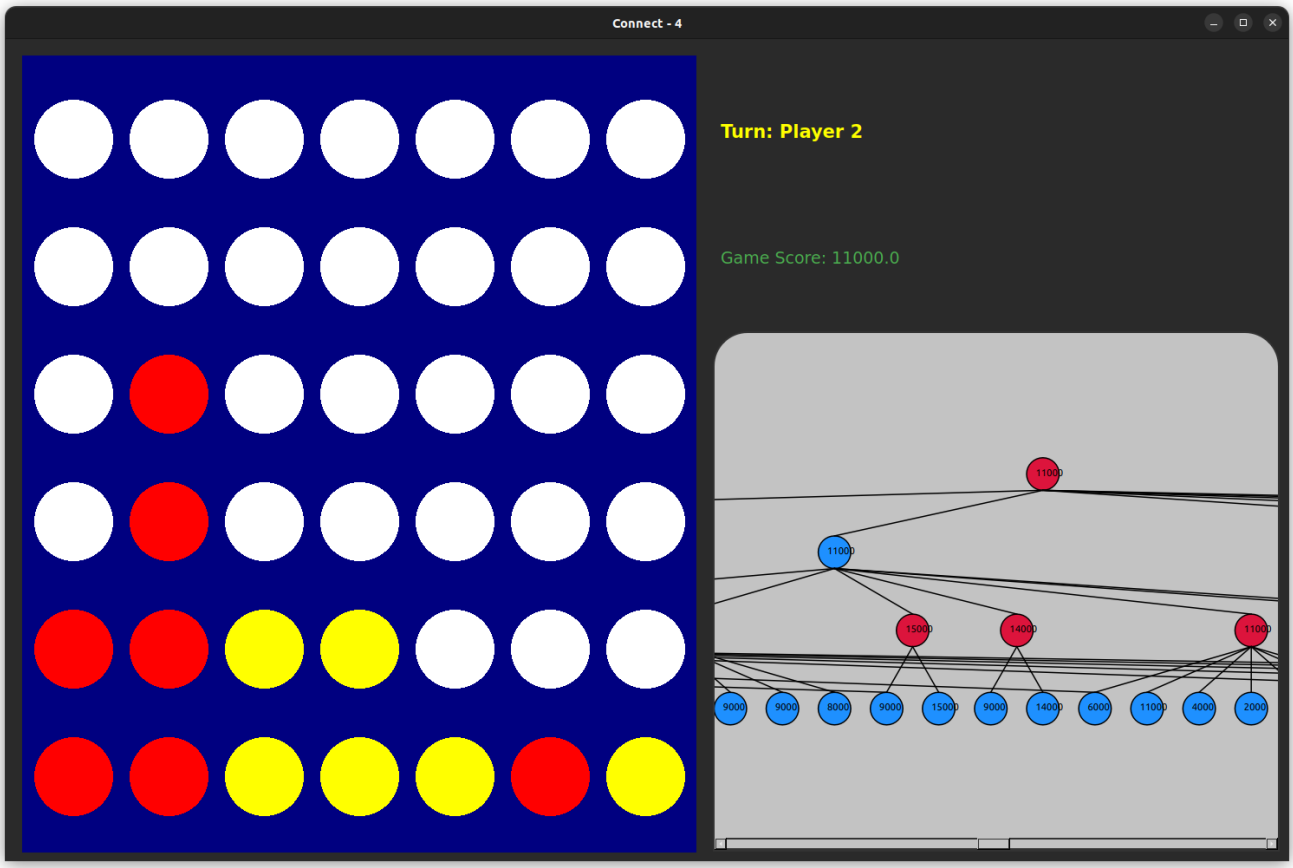



Connect - 4
Turn: Player 2
Game Score: 5000.0
5000
0
5000
0
1000

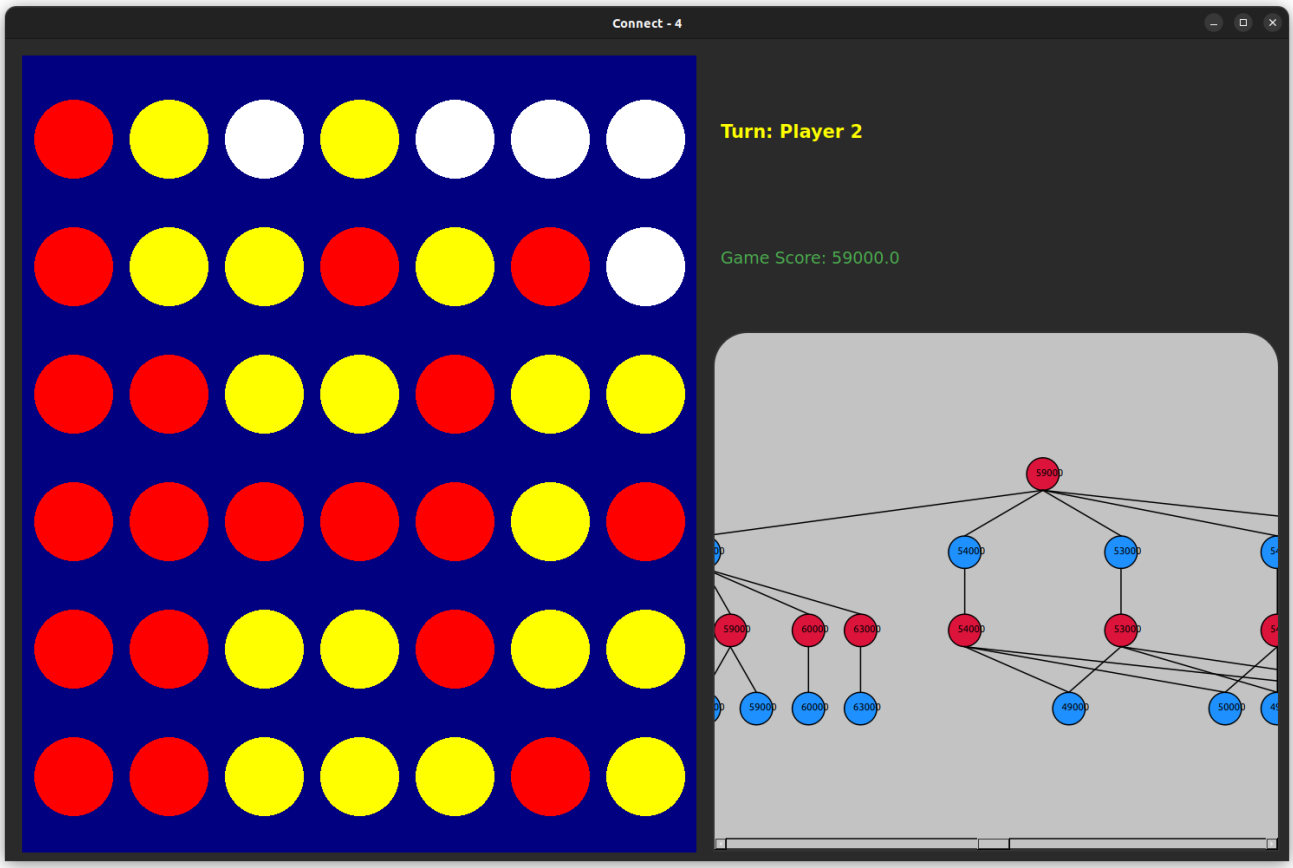

Connect - 4
Turn: Player 2
Game Score: 10000.0



- Alphabeta Pruning

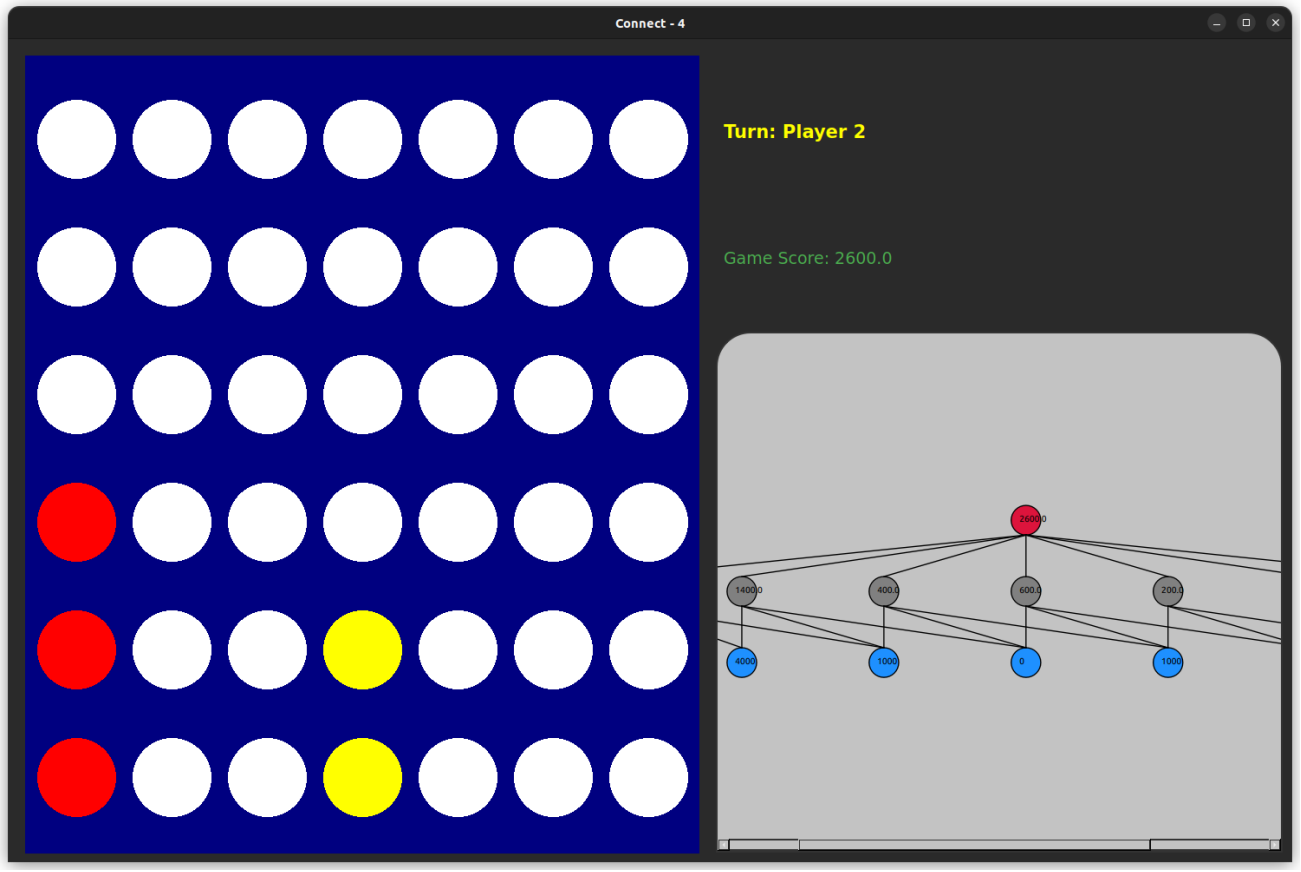


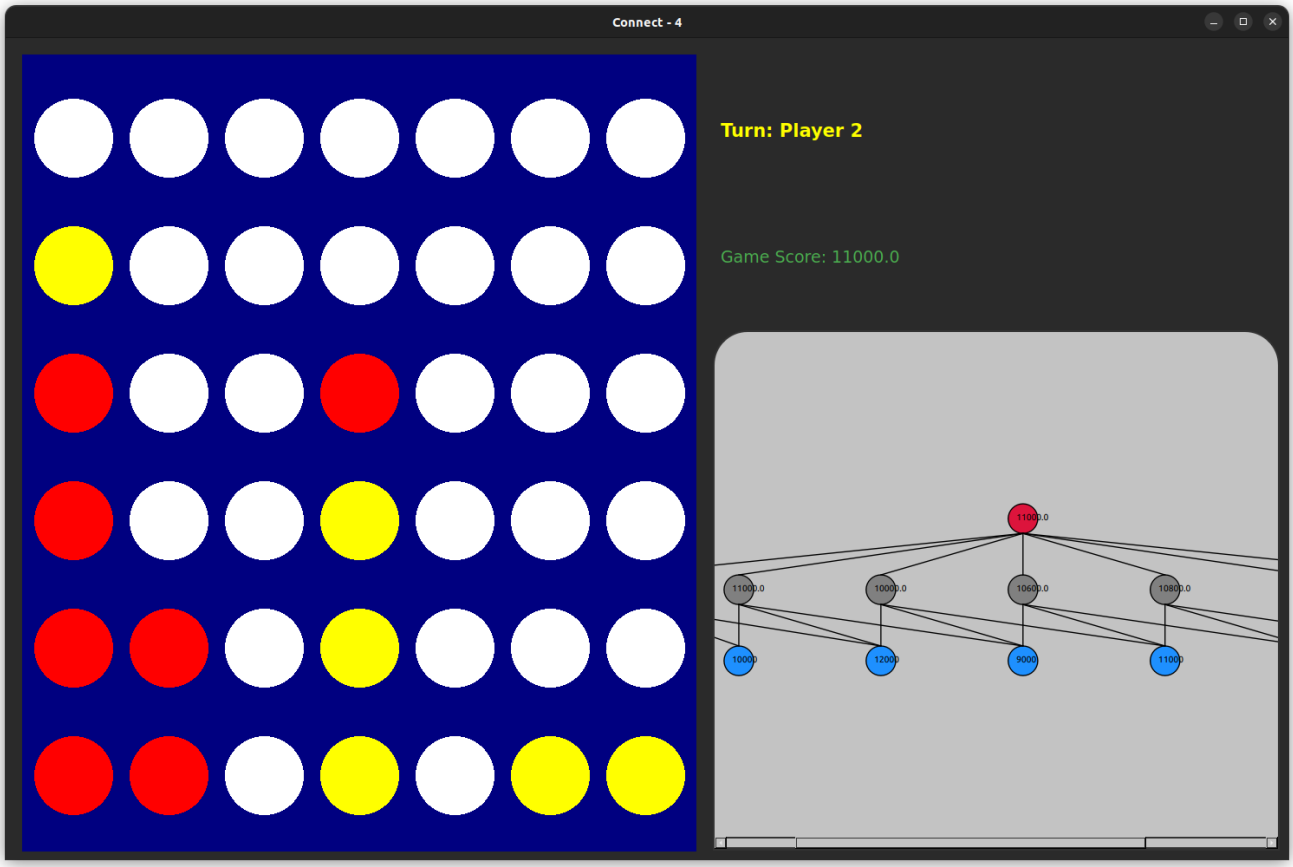
```
State Count: 719
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 0
2 1 0 0 0 0 1
1 1 2 2 0 0 1
1 1 2 2 2 1 2

Agent Chooses: 1
Game Score:  11000
Time Elased:  0.015987
```

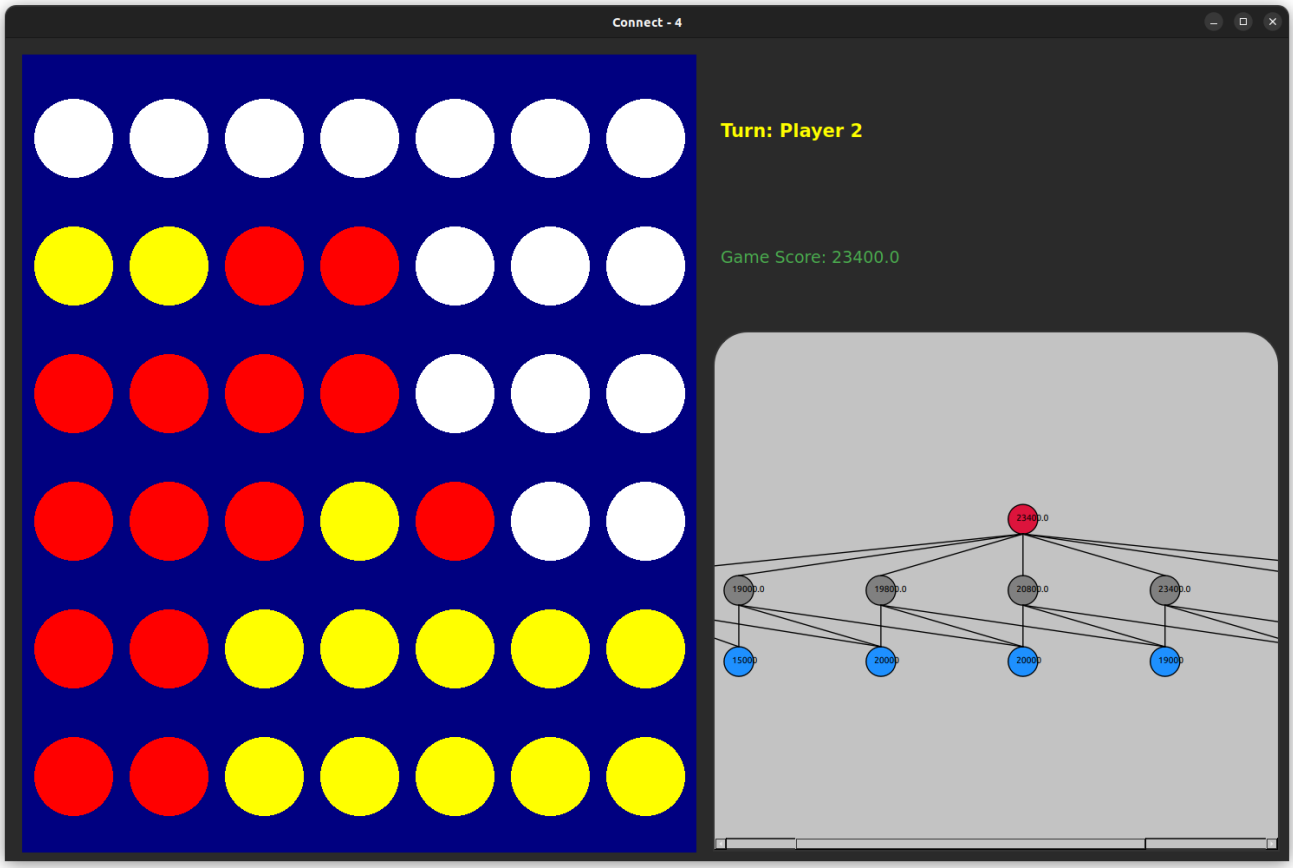


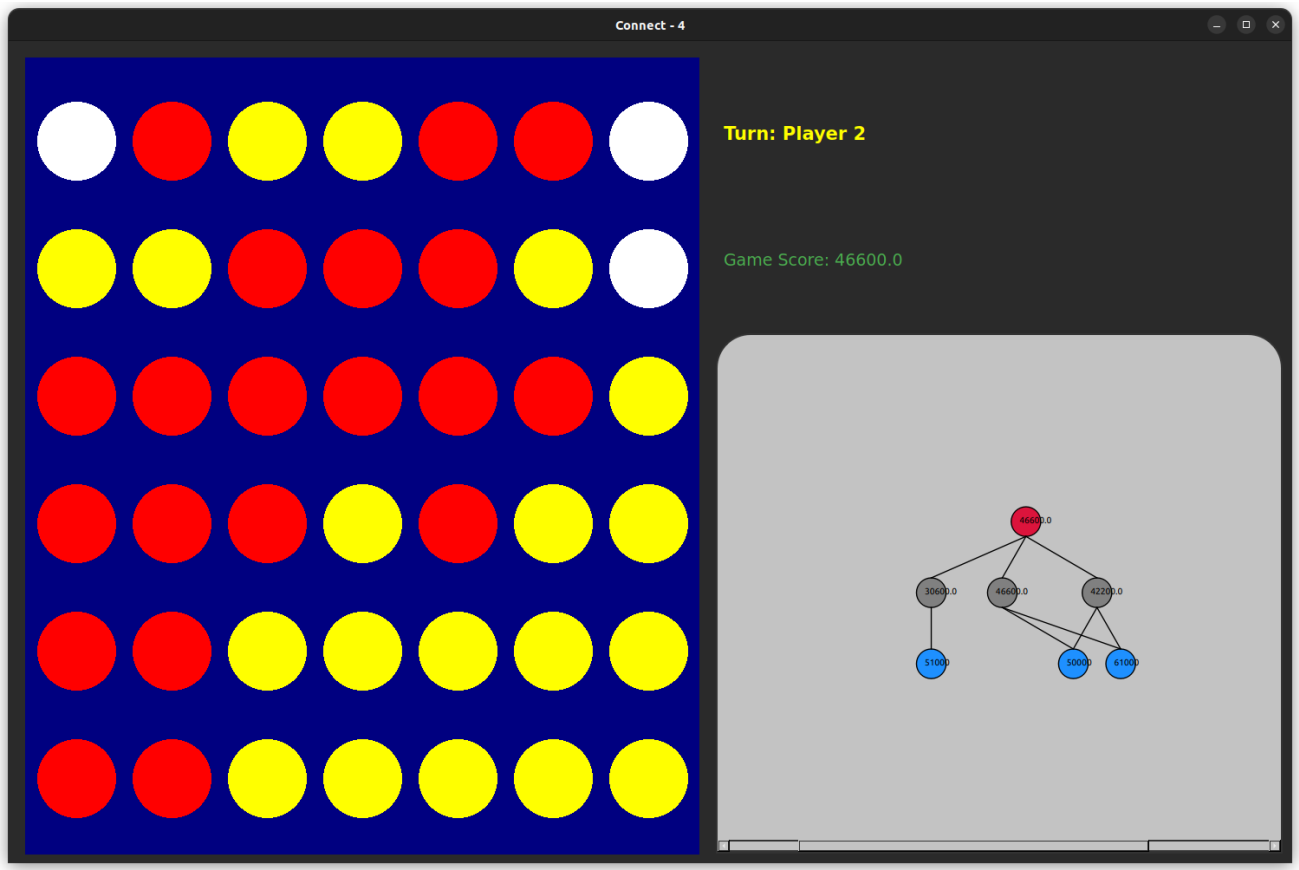
```
State Count: 1650
2 2 0 2 0 0 1
1 2 2 1 2 1 1
1 1 2 2 1 2 2
1 1 1 1 1 2 1
1 1 2 2 1 2 2
1 1 2 2 2 1 2

Agent Chooses: 0
Game Score:  59000
Time Elased:  0.007721
```

- ExpectiMinmax



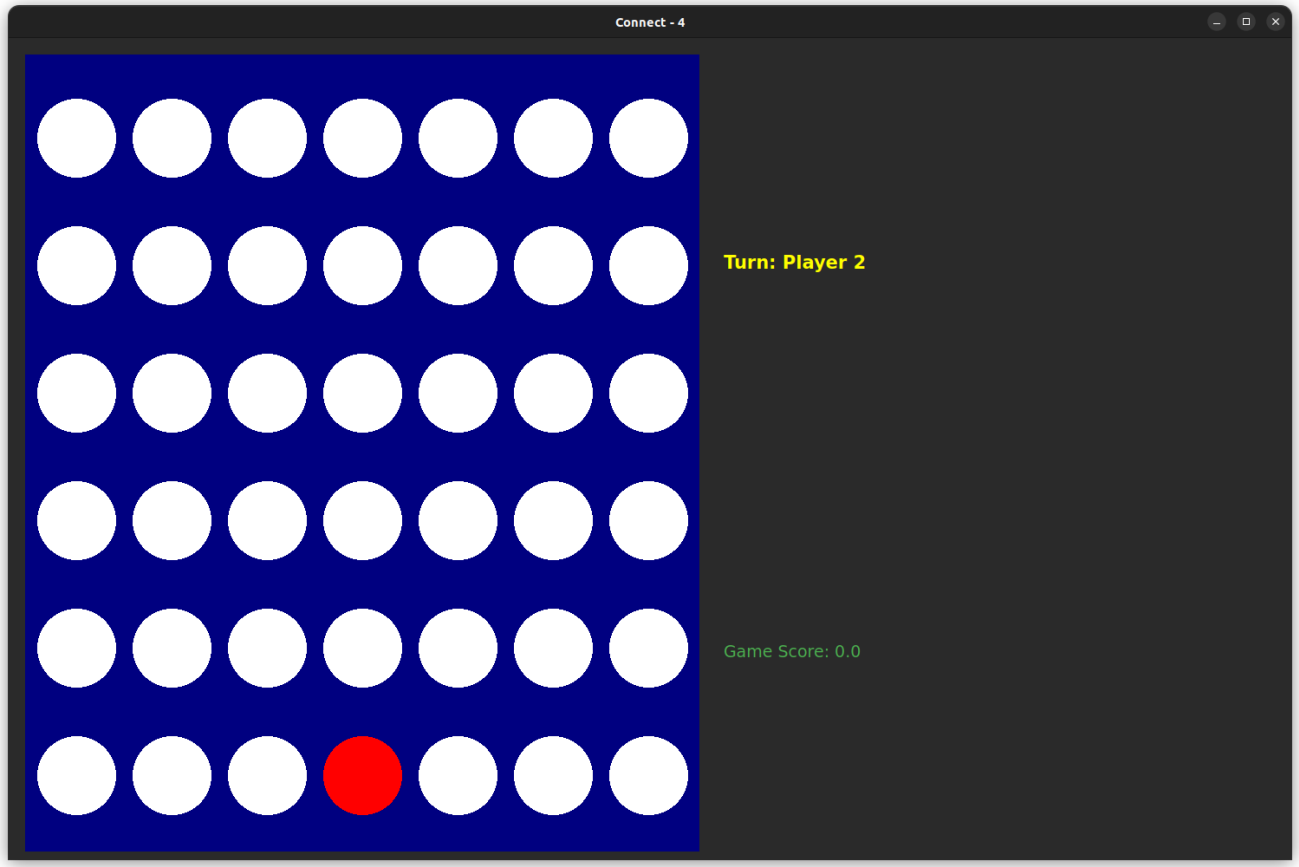

Connect - 4
Turn: Player 2
Game Score: 11000.0
11000.0
11000.0
10000.0
10600.0
10800.0
10000
12000
9000
11000


Connect - 4
Turn: Player 2
Game Score: 23400.0
23400.0
19000.0
19800.0
20800.0
23400.0
15000
20000
20000
19000


Connect - 4
Turn: Player 2
Game Score: 46600.0
46600.0
30600.0
46600.0
42200.0
51000
50000
61000

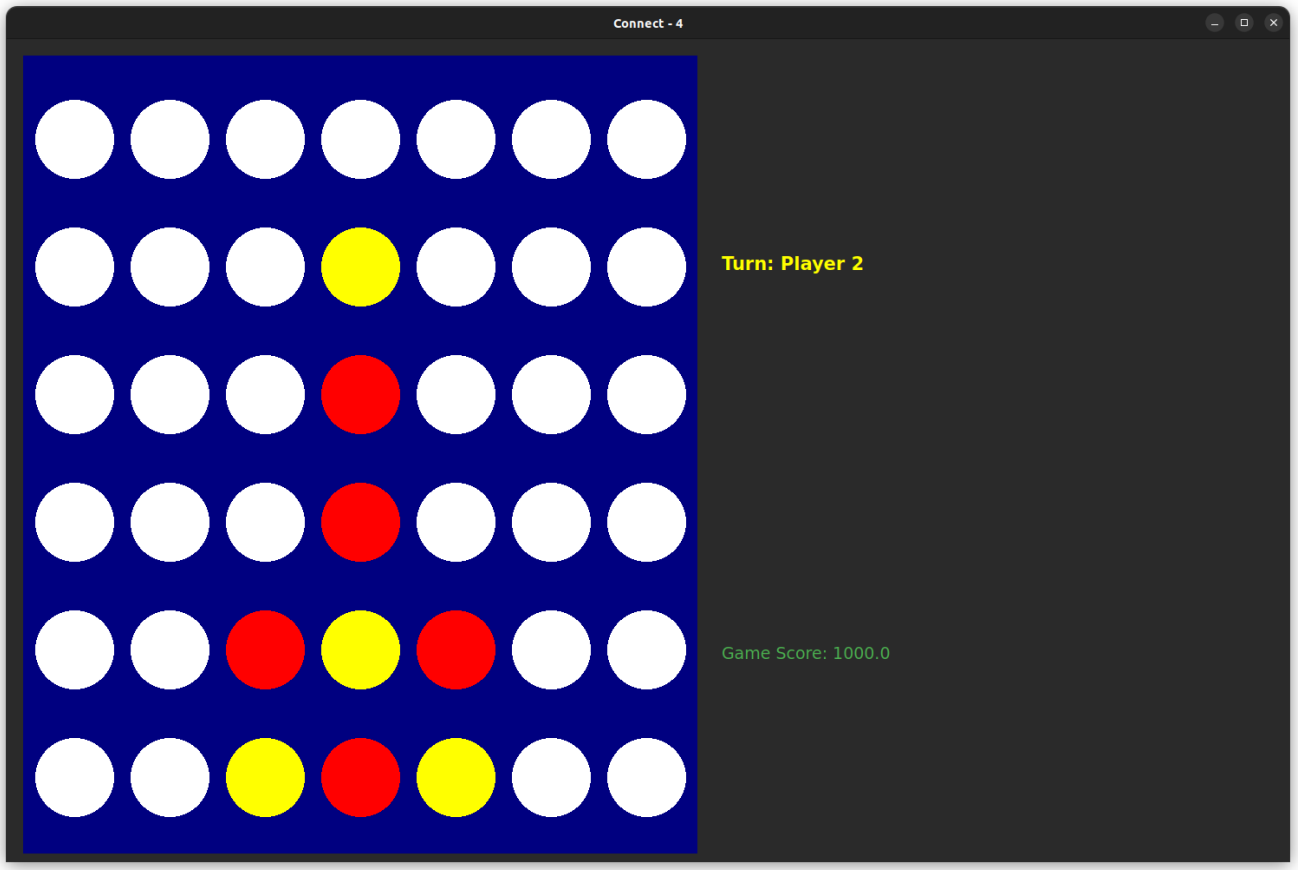
3.2 K = 6

- Minmax

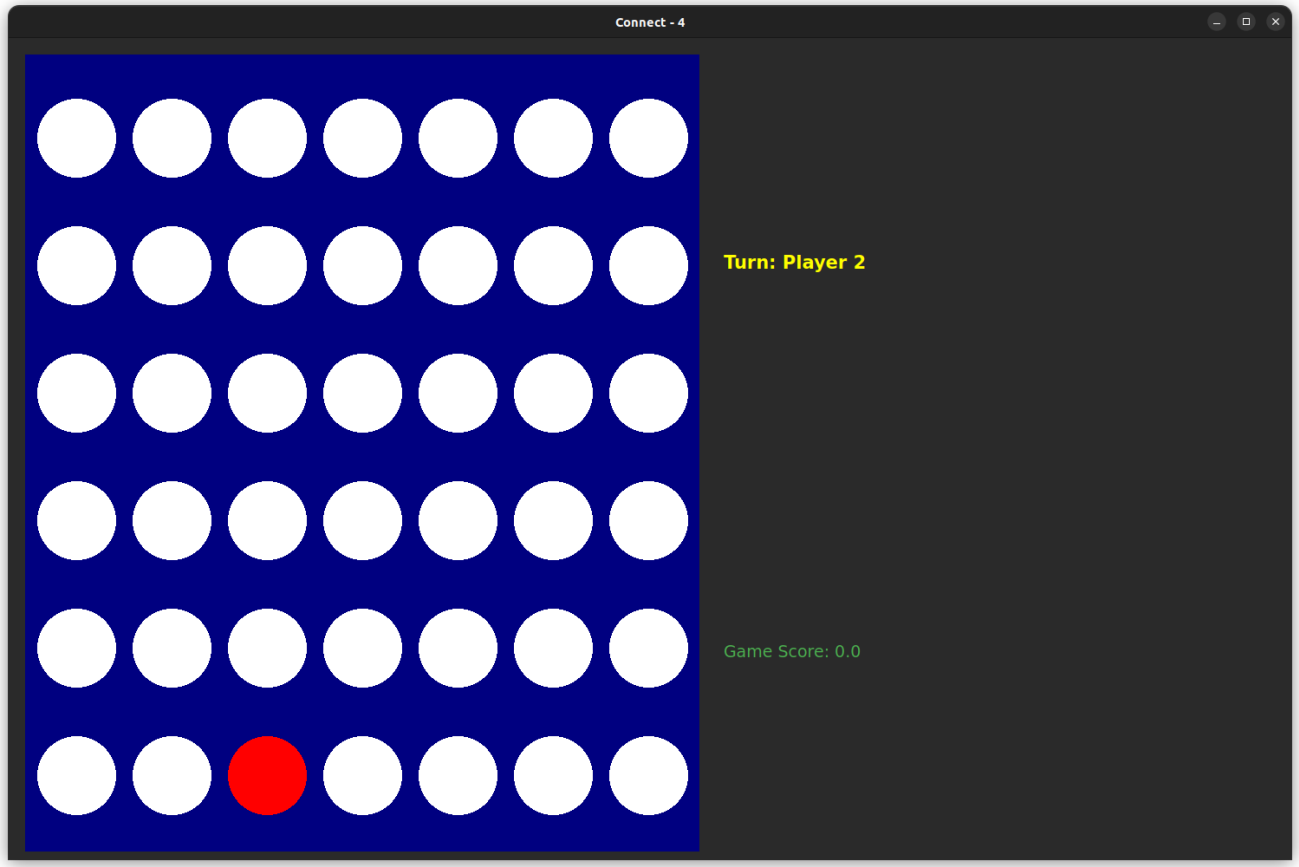



```
State Count: 22100
0 0 0 0 0 0 2
0 0 0 0 0 0 1
0 0 0 0 0 0 2
0 0 0 0 0 0 1
0 0 0 0 0 0 2
0 0 0 0 0 0 1

Agent Chooses: 3
Game Score:  0
Time Elased:  1.090937
```

```
State Count: 104909
0 0 0 0 0 0 2
0 0 0 2 0 0 1
0 0 0 1 0 0 2
0 0 0 1 0 0 1
0 0 0 2 1 0 2
0 0 2 1 2 0 1

Agent Chooses: 2
Game Score:  1000
Time Elased:  1.420425
```

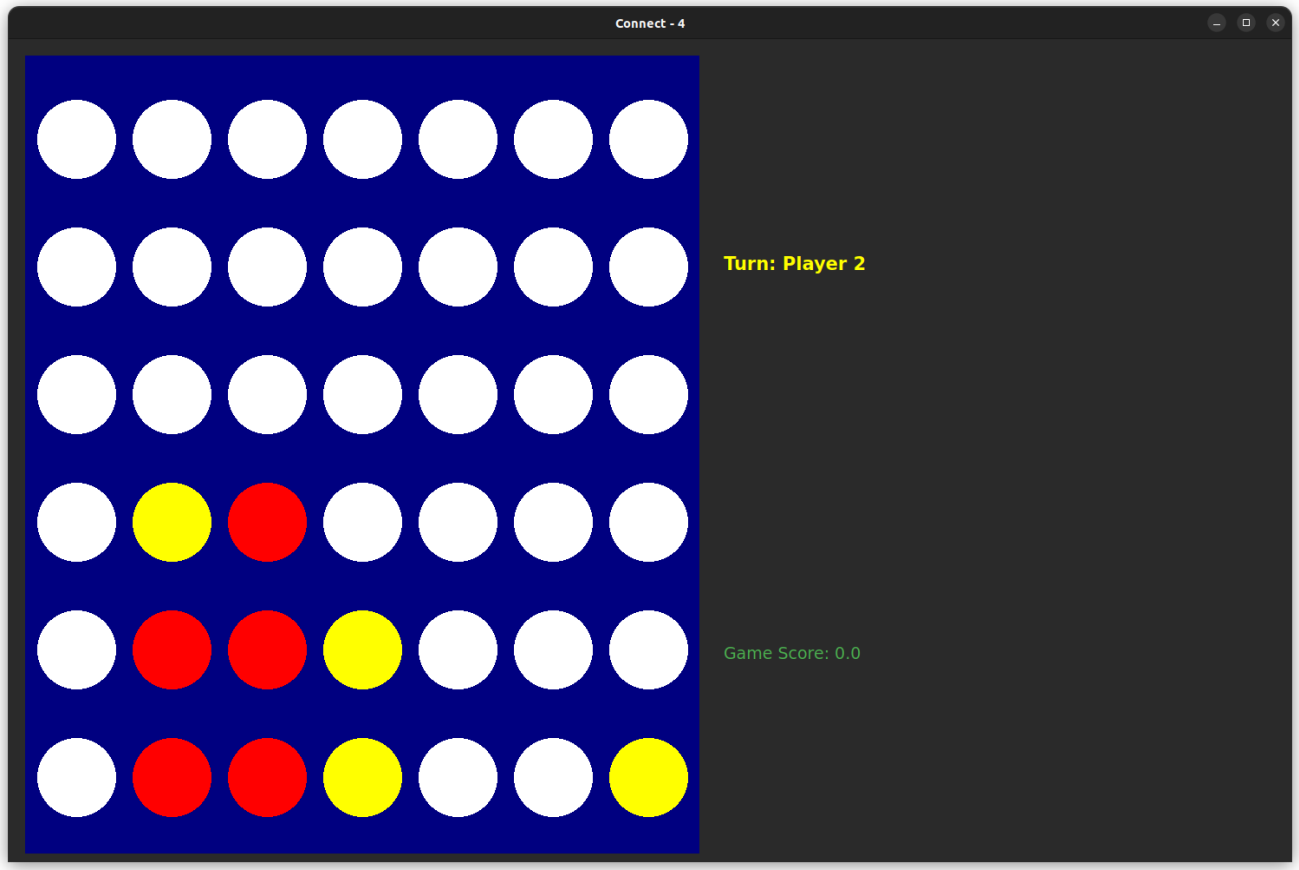


- Alphabeta Pruning




```
State Count: 6612
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 1
2 0 0 0 0 0 1
2 0 0 0 0 0 1
2 0 1 0 0 0 2

Agent Chooses: 1
Game Score:  1000
Time Elased:  0.165722
```

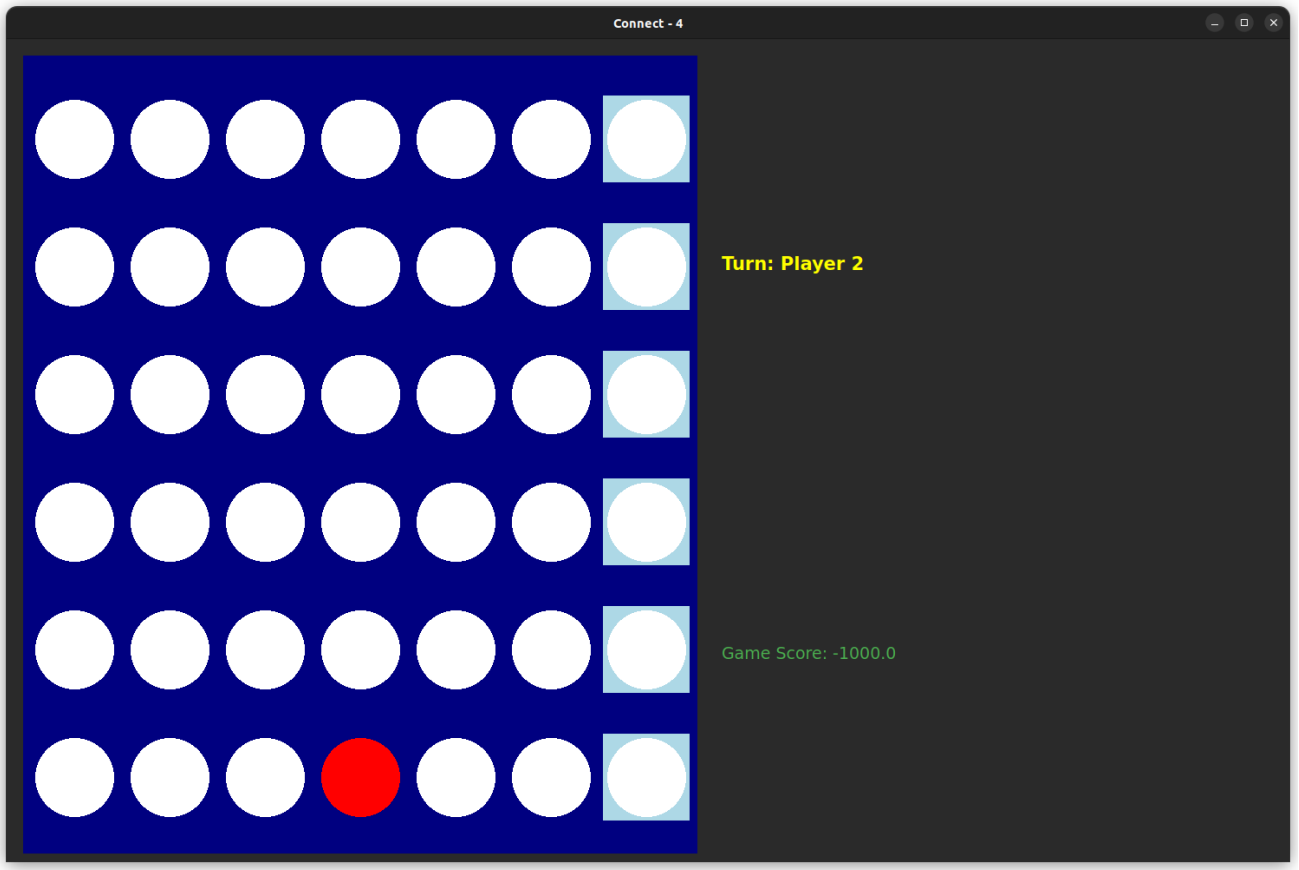
```
State Count: 21622
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 2 0 0 0 0 1
0 2 2 0 0 0 1
0 1 1 2 0 0 1
2 1 1 2 0 0 2

Agent Chooses: 2
Game Score:  0
Time Elased:  0.346117
```



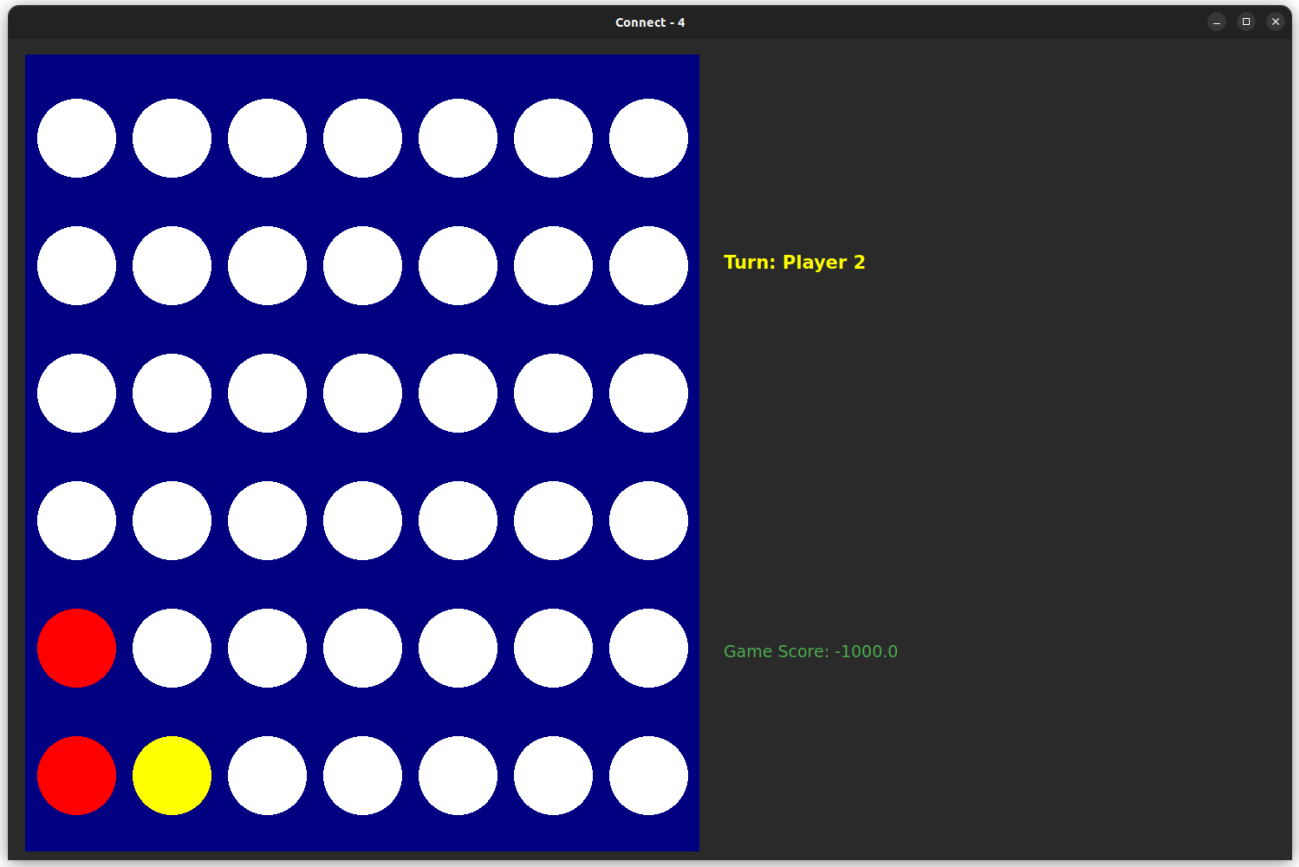
3.3 K = 8

- Minmax

```
State Count: 263348
0 0 0 0 0 0 2
0 0 0 0 0 0 1
0 0 0 0 0 0 2
0 0 0 0 0 0 1
0 0 0 0 0 0 2
1 0 0 0 0 0 1

Agent Chooses: 3
Game Score:  -1000
Time Elased:  15.235981
```



- Alphabeta Pruning




```
State Count: 35359
0 0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 0 0 0 0 1
0 0 0 0 0 0 1
0 0 0 0 0 0 1
0 2 2 2 0 0 1

Agent Chooses: 0
Game Score:  -1000
Time Elased:  1.887777
```