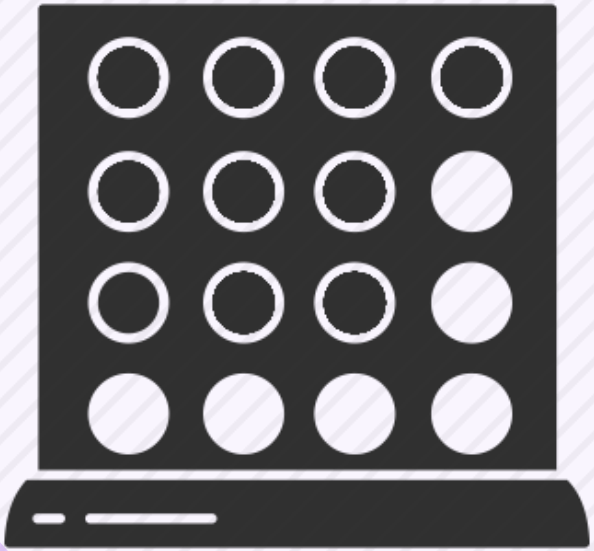


Greedy Connect Four

Alejandro Sherman

Core Idea

Initial Design Idea:

- Within a matrix or board a human player may play connect four against an AI
- The AI makes decisions based off of a set of greedy choices
- The idea is for the AI to attempt to perform an optimal choice

Specifications

Initial Design Specifications:

- Four markers of the same type in a row functions as a win
- Initially implemented with only matches in the vertical counting as a win

Greedy Choices

Opponent Highest match

The AI keeps track of the opponents highest number of matches:

- Once Opponent has three matches in a row

AI Highest match

The AI keeps track of it's highest number of matches:

- When AI has three in a row
- When opponent doesn't have three in a row

Implementation

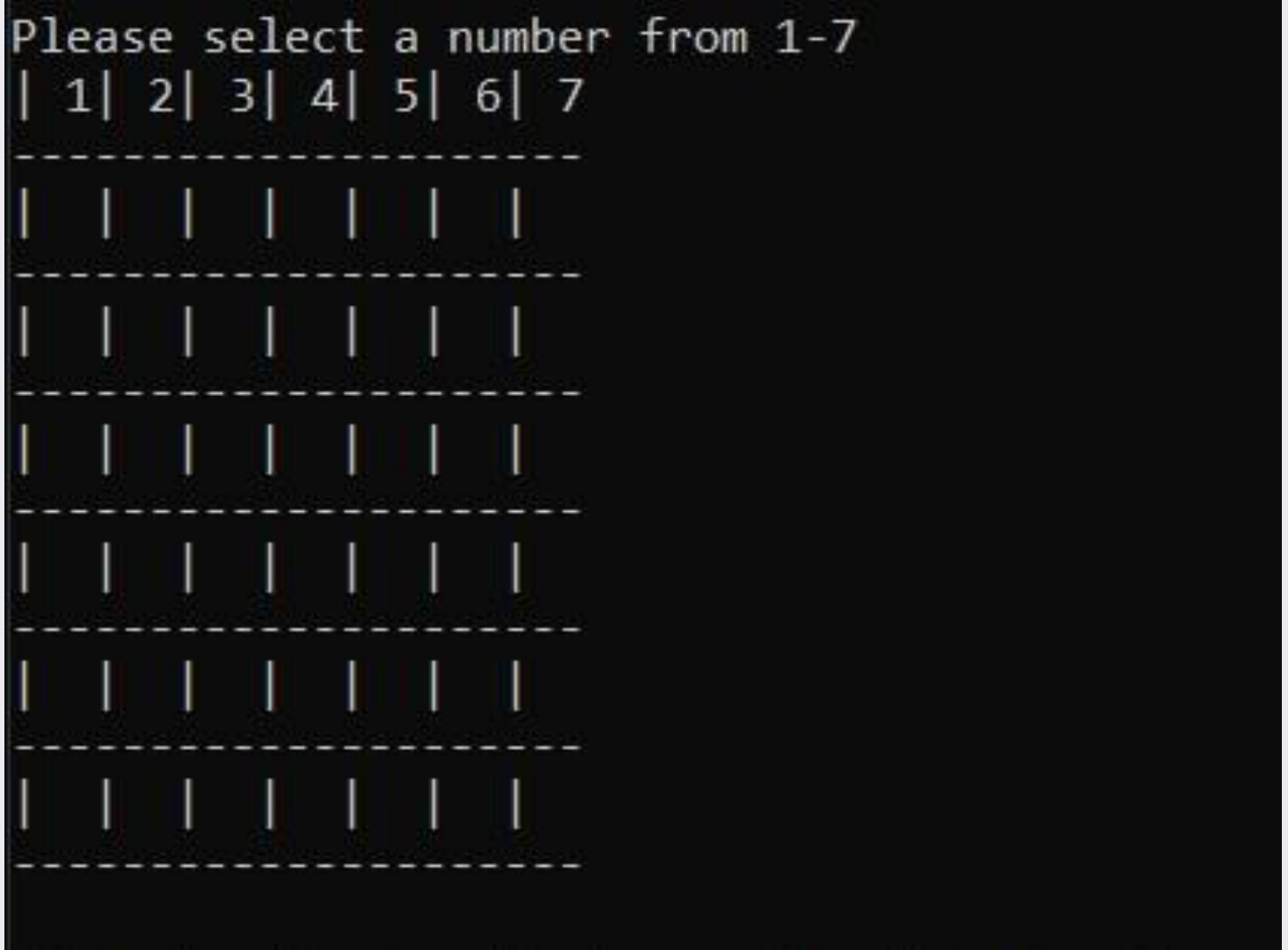
```
#define HEIGHT 6
#define WIDTH 7

void draw_board();
void player_movement(int player);
void ai_movement(int player);
bool check_for_winner(int x, int y, int player);
bool check_vertical(int x, int y, int player);

int board_info[HEIGHT][WIDTH] = {{0,0,0,0,0,0,0},
                                 {0,0,0,0,0,0,0},
                                 {0,0,0,0,0,0,0},
                                 {0,0,0,0,0,0,0},
                                 {0,0,0,0,0,0,0},
                                 {0,0,0,0,0,0,0}};
```

```
void draw_board(){
    cout << endl;
    for (int y = 0; y < HEIGHT; y++){
        for (int x = 0; x < WIDTH; x++){
            cout << "| ";
            if (board_info[y][x] == 0) cout << " ";
            else if (board_info[y][x] == 1) cout << "X";
            else if (board_info[y][x] == 2) cout << "O";
        }
        cout << "\n---------------------" << endl;
    }
}
```

Example Game


Please select a number from 1-7
| 1| 2| 3| 4| 5| 6| 7

| | | | | | |

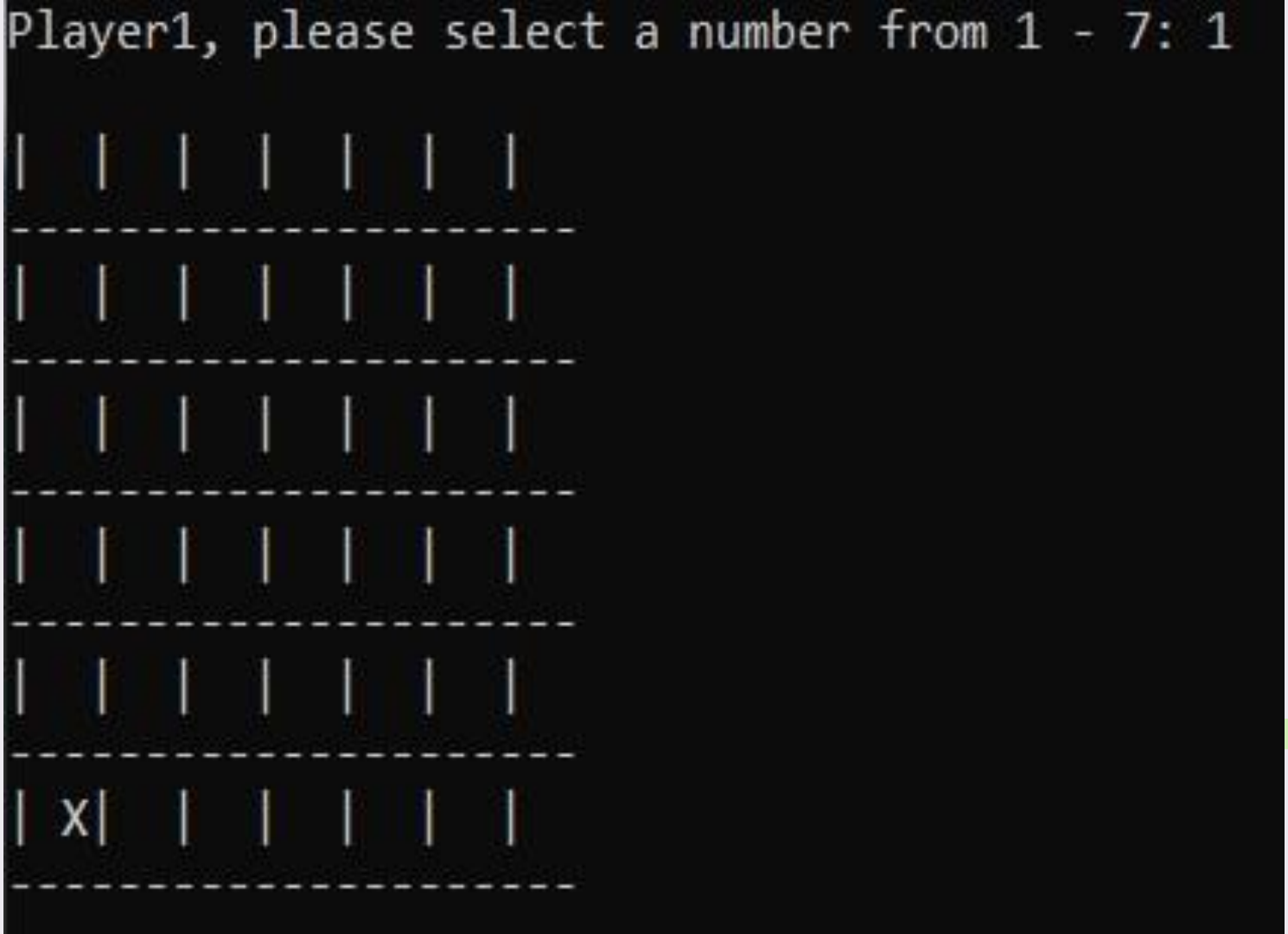
| | | | | | |

| | | | | | |

| | | | | | |

| | | | | | |

| | | | | | |


Player1, please select a number from 1 - 7: 1
| | | | | | |

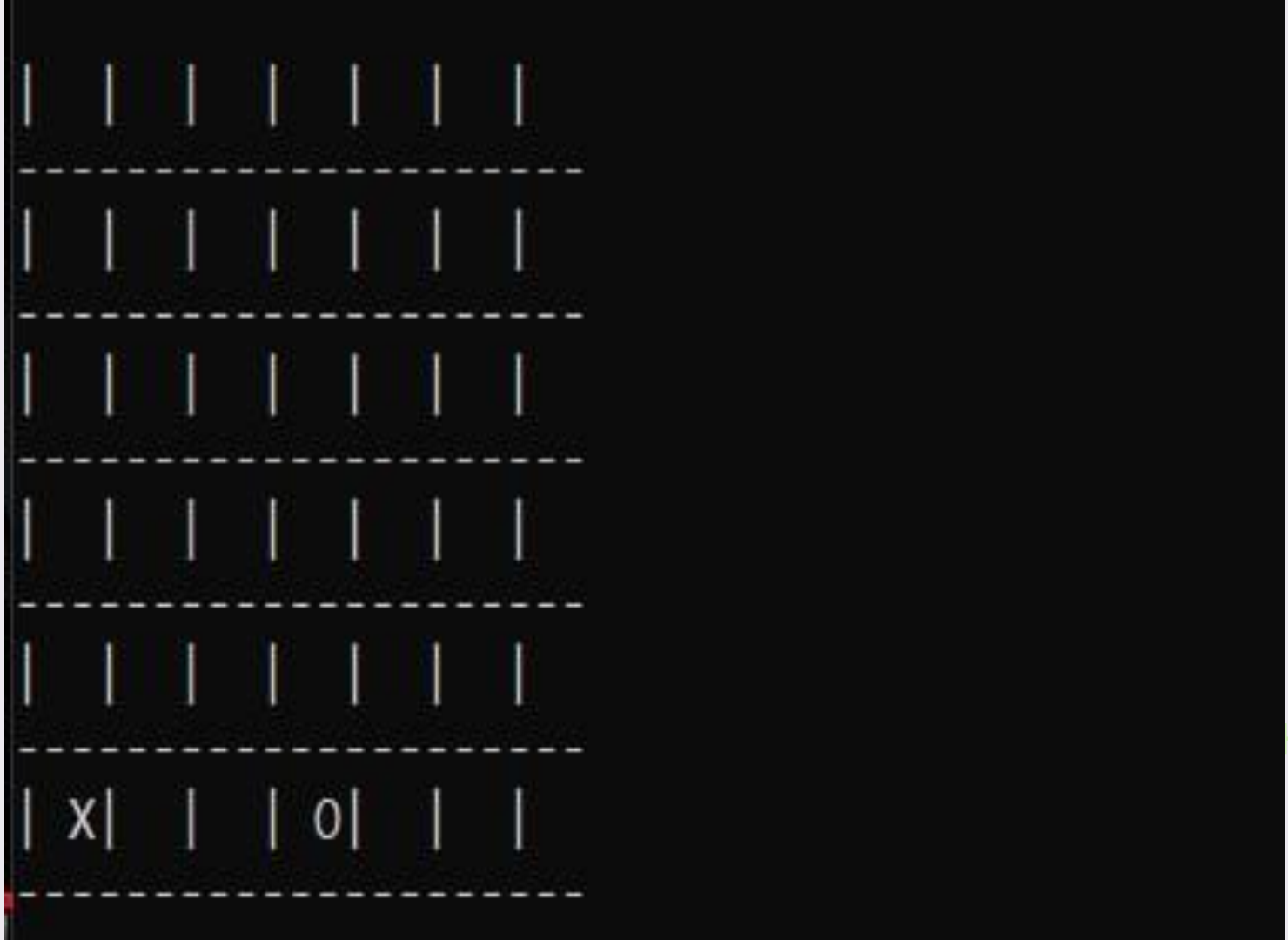
| | | | | | |

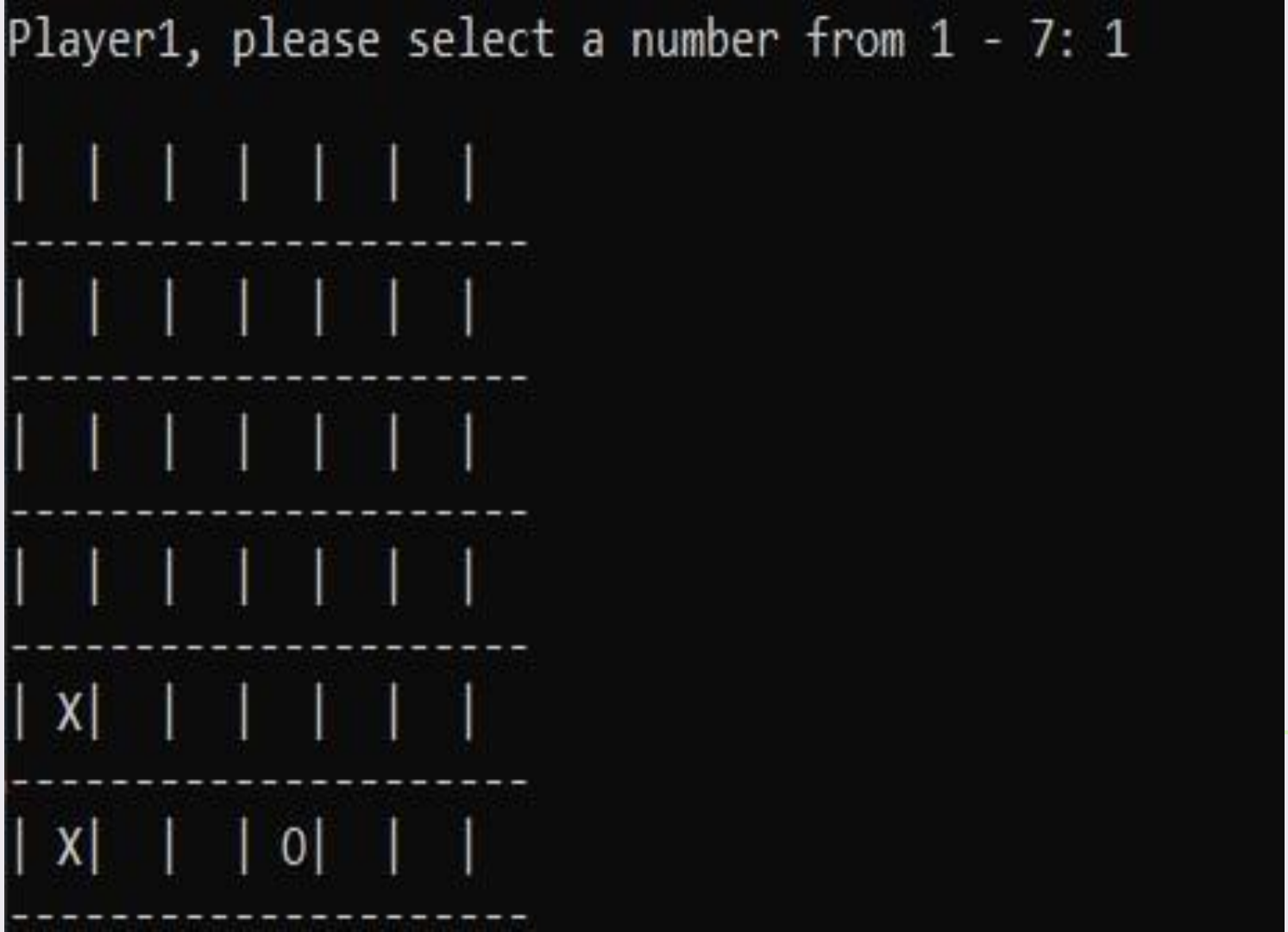
| | | | | | |

| | | | | | |

| | | | | | |

| X| | | | | |


X			O		


Player1, please select a number from 1 - 7: 1
| | | | | | |

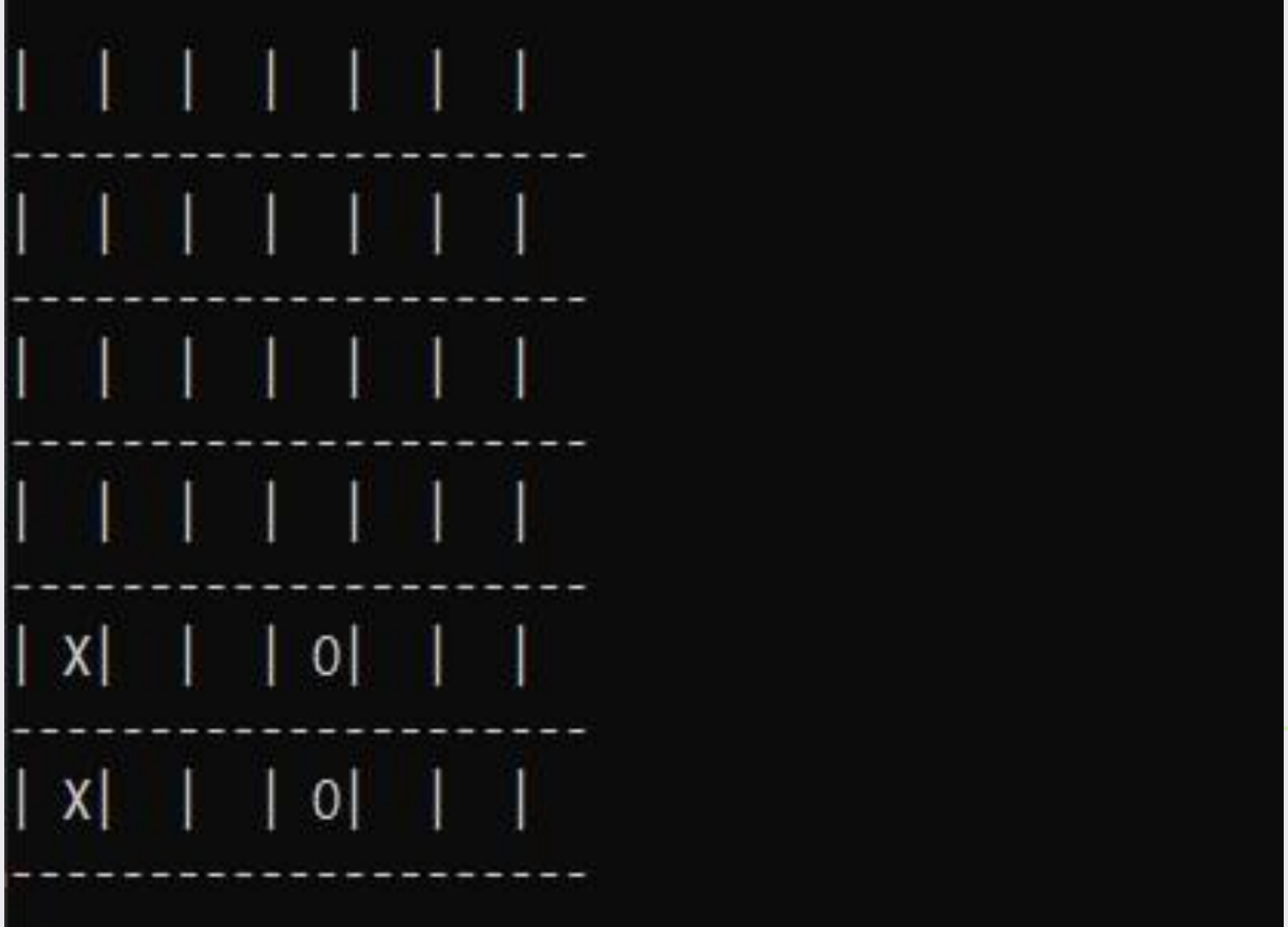
| | | | | | |

| | | | | | |

| | | | | | |

| X| | | | | |

| X| | | O| | |


| | | | | | |

| | | | | | |

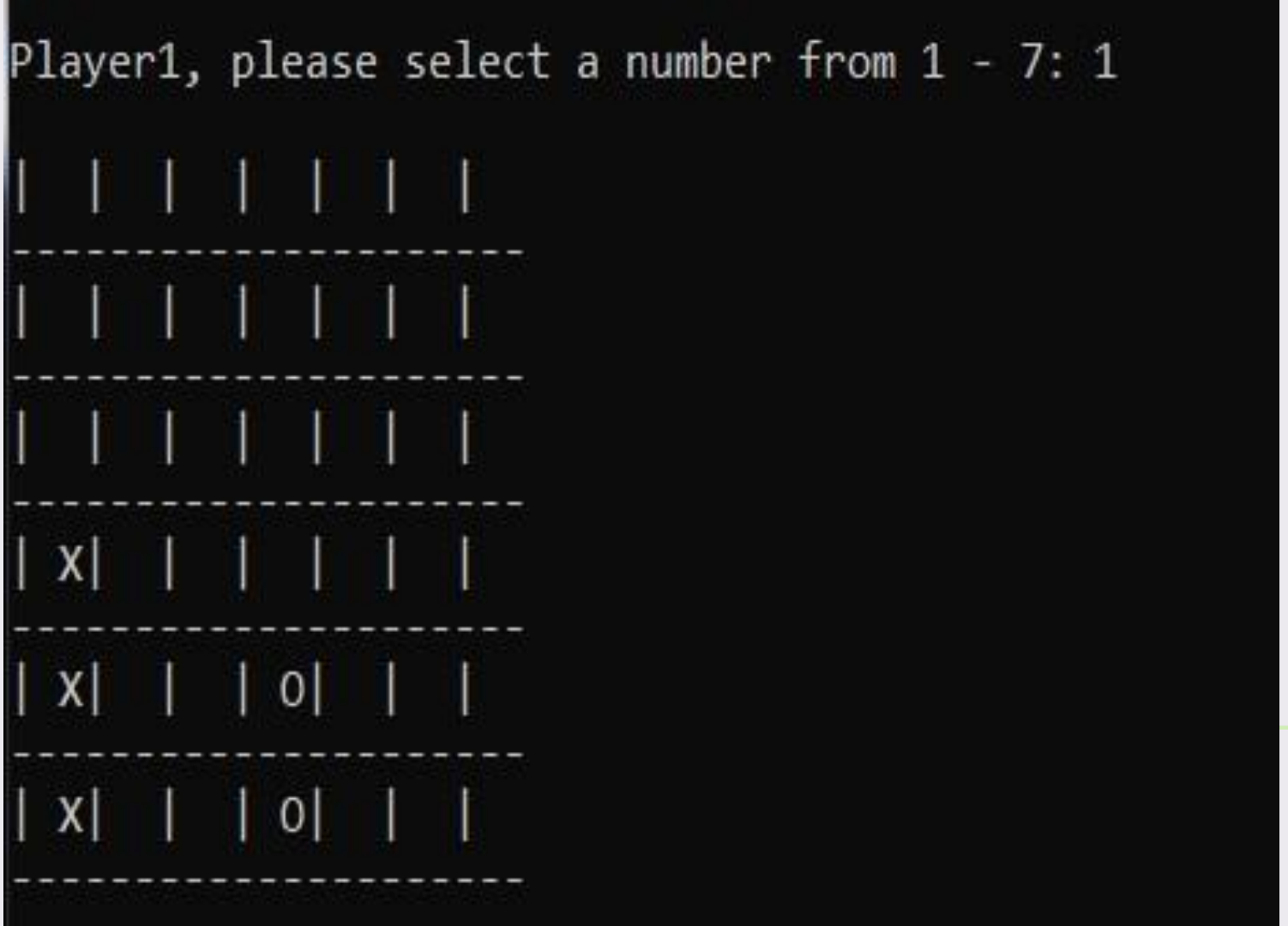
| | | | | | |

| | | | | | |

| | | | | | |

| X| | | O| | |

| X| | | O| | |


Player1, please select a number from 1 - 7: 1
| | | | | | |

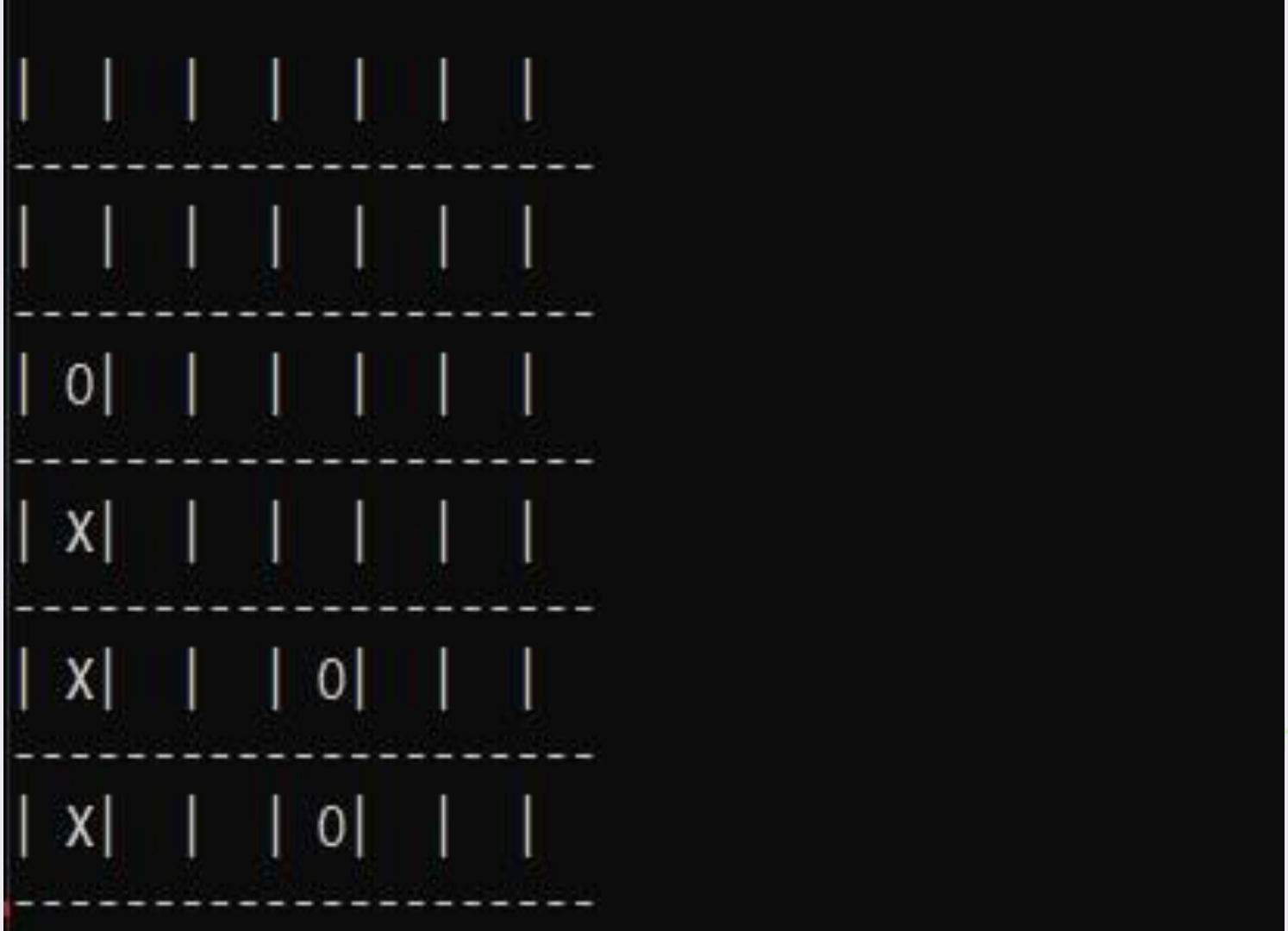
| | | | | | |

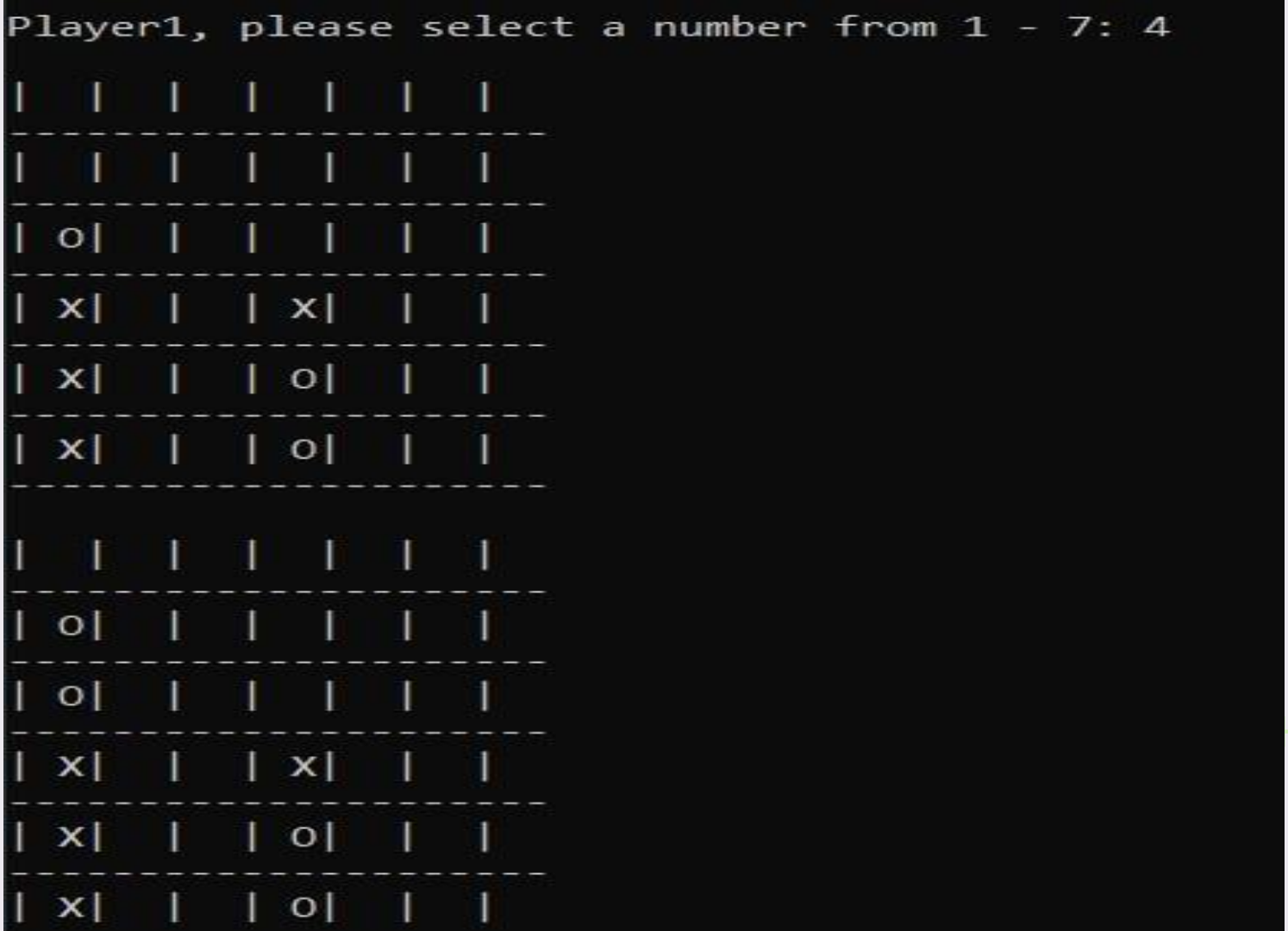
| | | | | | |

| X| | | | | |

| X| | | O| | |

| X| | | O| | |


O					
X					
X			O		
X			O		


Player1, please select a number from 1 - 7: 4
o					
x			x		
x			o		
x			o		
o					
o					
x			x		
x			o		
x			o		

```
| O| O| O| O| O| O| O
---------------------
| O| O| X| X| X| X| X
---------------------
| O| O| O| O| O| O| O
---------------------
| X| X| X| X| X| X| X
---------------------
| X| X| X| O| O| X| O
---------------------
| X| X| X| O| O| X| X
---------------------

Draw
```

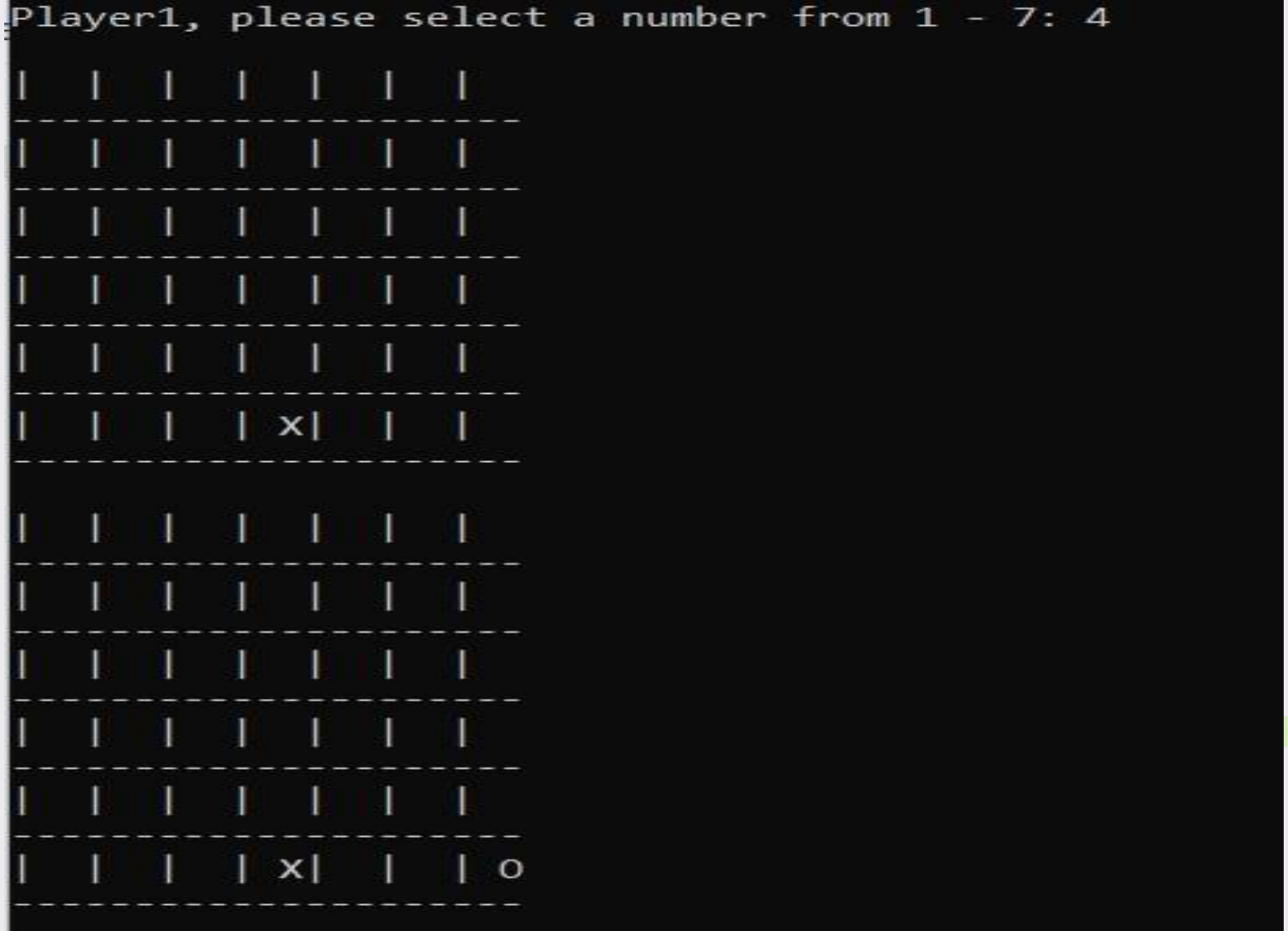
Limitation

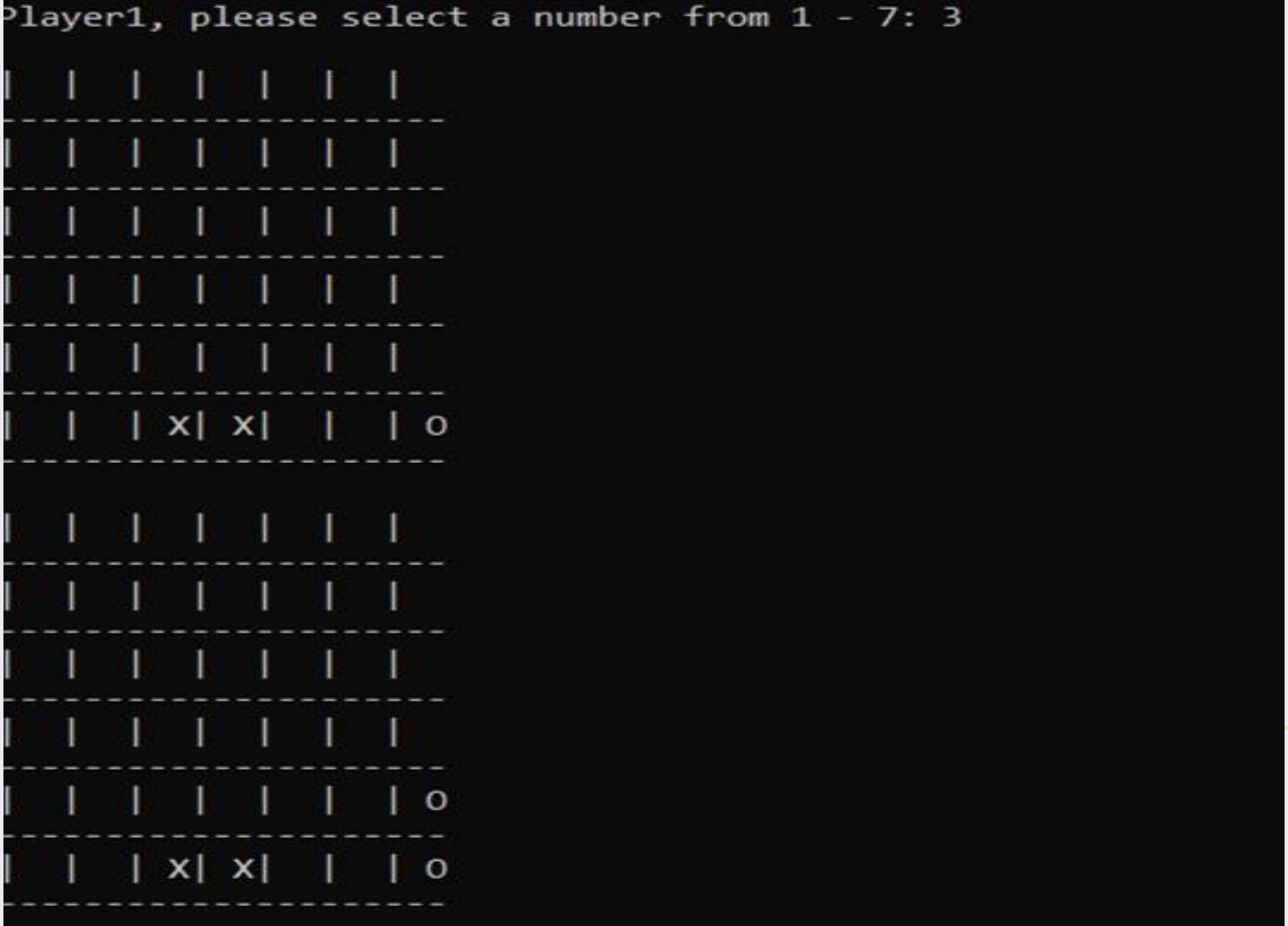
- In the simple implementation where only the vertical direction is considered, the greedy choice will always lead to a scenario where one player can block the other with ease

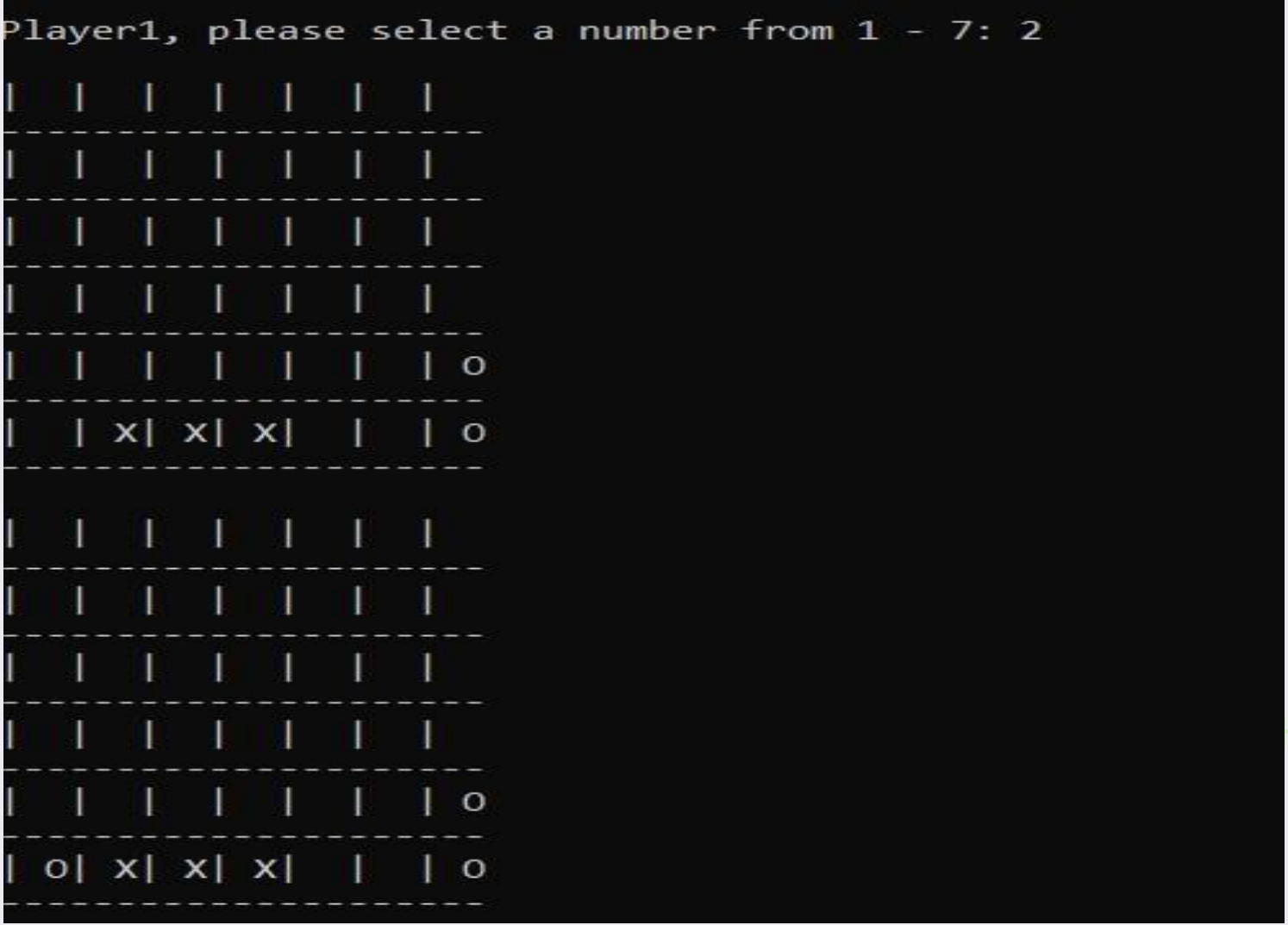
Consider:

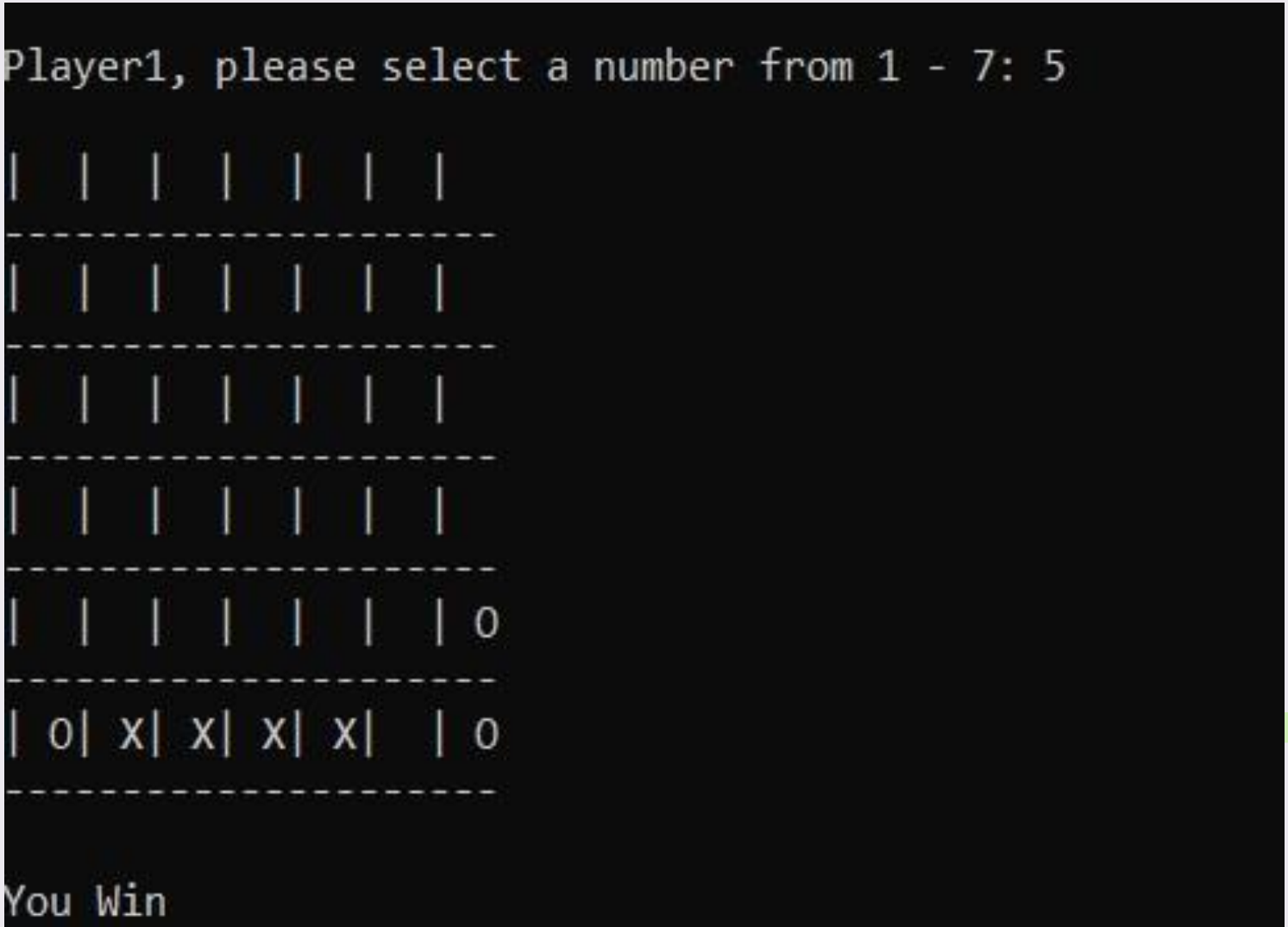
How the AI behavior may change with the implementation of the horizontal axis

See the following example


Player1, please select a number from 1 - 7: 4
x
x
o


Player1, please select a number from 1 - 7: 3
		x	x			o
						o
		x	x			o


Player1, please select a number from 1 - 7: 2
						o
	x	x	x			o
						o
o	x	x	x			o


Player1, please select a number from 1 - 7: 5
						O
O	X	X	X	X		O
You Win

What gives?

- This is the second time we've encountered the AI following the greedy protocol but still resulting in a suboptimal outcome

Conclusion

The greedy solution is this case is insufficient

With the AI making decisions based solely on greedy choices, it can react to the current state of the game and act accordingly, but cannot account for the state of the game following it's greedy choice

Implementations

Greedy Solution

- The AI keeps track of the opponents highest number of matches
- AI makes snap decisions based on the current state of the game
- Can be lured into a trap

Dynamic Programming

- The AI keeps track of the opponents highest number of matches
- AI considers all possible combinations following each choice and makes the optimal one
- Cannot be lured into a trap

Complexity

Space Complexity: O(w * h) for both

Greedy Solution

- Two for loops up to the size of the width (w) and height (h)
- Other operations, but are much smaller in comparison
- $O(w * h) \approx O(n^2)$

Dynamic Programming

- Same two for loops, but with one more that compares the values of one column with the rest following each possible choice
- Other operations, but are much smaller in comparison
- $O(w * h * w) \approx O(n^3)$

Additions

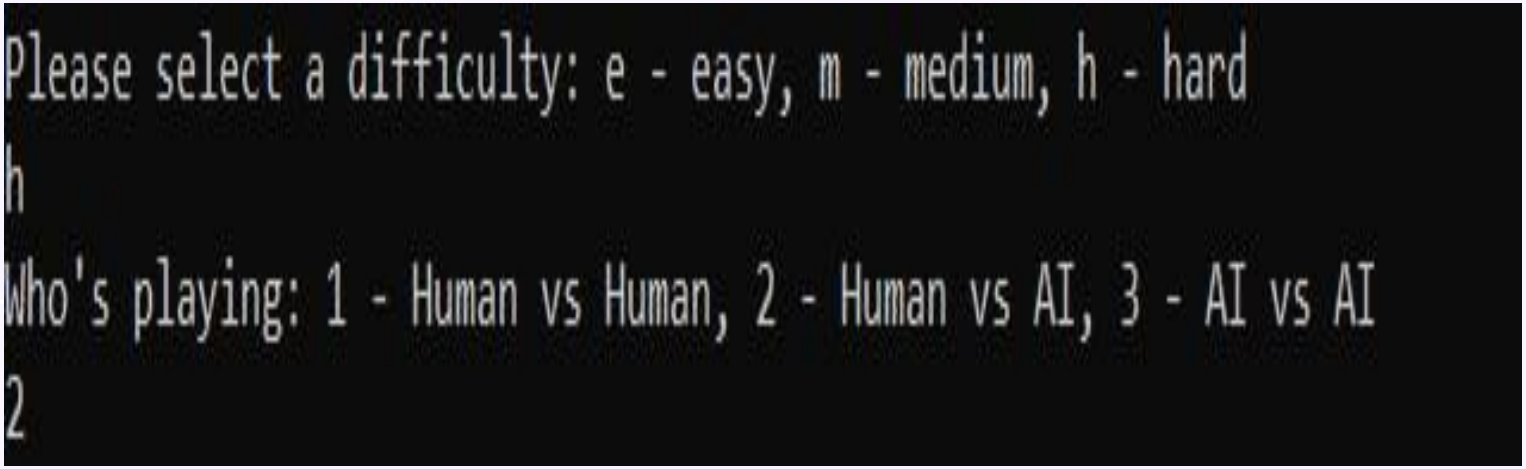
Additions

Choice of Difficulty

Option 1:
Easy
Option 2:
Medium
Option 3:
Hard

Choice of Players

Option 1:
Human vs Human
Option 2:
Human vs AI
Option 3:
AI vs AI



e

easy

AI chooses randomly from the available spots

m

medium

AI will only focus on the greedy choice that allows it to increase it's score

h

hard

AI will choose to the greedy choice to increase it's score, unless the other player is about to win, in which case make that greedy choice

```
Please select a difficulty: e - easy, m - medium, h - hard
h
Who's playing: 1 - Human vs Human, 2 - Human vs AI, 3 - AI vs AI
2
```

01

Human vs Human

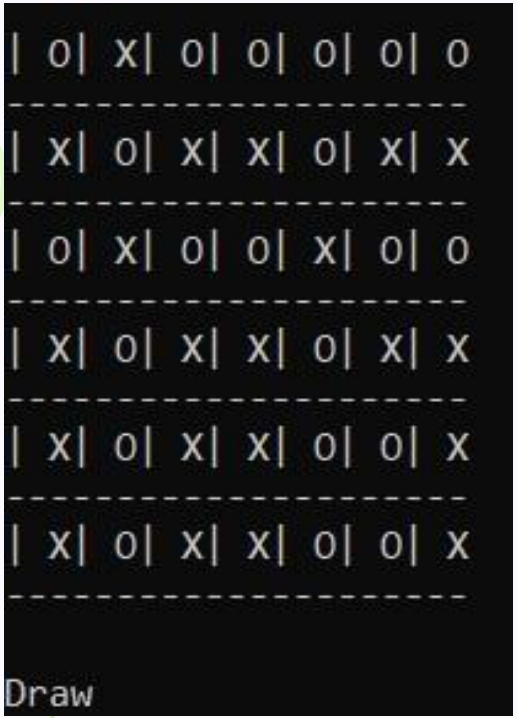
02

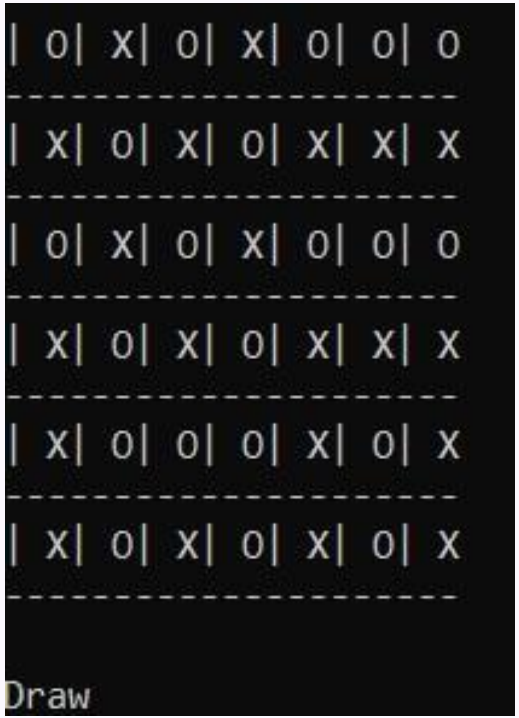
Human vs AI

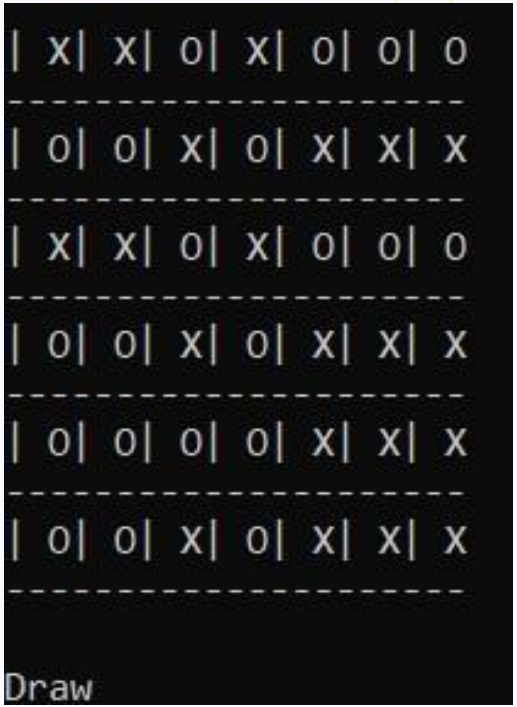
03

AI vs AI

Sample AI vs AI







As for connecting four on the diagonal:

Once again, a dynamic programming implementation would be the way to an optimal AI

Thanks