**DATA STRUCTURES AND ITS APPLICATIONS**

**UE22CS252A**

**3rd Semester, Academic Year 2023**

**MINI PROJECT**

**SUPER XO**

**Team Members** :

Kalyanishetti Praneet Vijay – PES2UG22CS251

Kartik Sadanand Naik – PES2UG22CS258

Kousthubh N Gowda – PES2UG22CS270

Aayush Kumar – PES2UG22CS908

**SYNOPSIS :**

**What is this project about?**

The Super XO Game is a dynamic and engaging implementation of the classic Tic-Tac-Toe, which is developed using 2D array concept of Data Structures and Algorithm. Even though we are doing this for marks and our future, but still it was so fun and enjoyable doing this.

1. Talking about the feature of the game :

* The game features a standard 3x3 Tic-Tac-Toe grid where two players take turns.
* Players can choose to play as either 'X' or 'O' and compete to form a winning sequence.
* The game ends when a player wins or when the board is filled, resulting in a draw.

1. Data structure used in creating this :

* A 2D array, or two-dimensional array, is a fundamental data structure that organizes data in a grid-like fashion. Unlike a 1D array, which is a linear collection of elements, a 2D array allows data to be stored in rows and columns, creating a structured matrix.
* In the context of the "Super XO Game" project, a 2D array is employed to efficiently store and manage the game board, providing a systematic approach to tracking the game state and enabling easy access and manipulation of individual cells.

Super XO Game is a well-structured, educational, and fun DSA mini project that not only demonstrates strong coding and algorithmic skills but also provides an enjoyable experience for players. It showcases the practical application of data structures and algorithms in real-world scenarios, making it an excellent addition to the university's computer science program.

**ADTs definition of all DS used :**

1. **Printing the board :**

* It prints the 3 x 3 board in which each segment contains the an internal 3 x 3 board with the help of numberings using the 2D array initialised.

1. **Big board move :**

* Activates the corresponding block of the big board.

1. **Small board move :**

* Activates the corresponding block of the small board.

1. **Restrict Big board move :**

* Checks if the given big board value is valid or not and if the big board block is filled then asks the player to give another big board move till the big board move is validated.

1. **Check win of small board :**

* If the ‘X’ or ‘O’ entered fulfils the win condition in the small board then calls the block fill function for the big block board.

1. **Block fill function :**

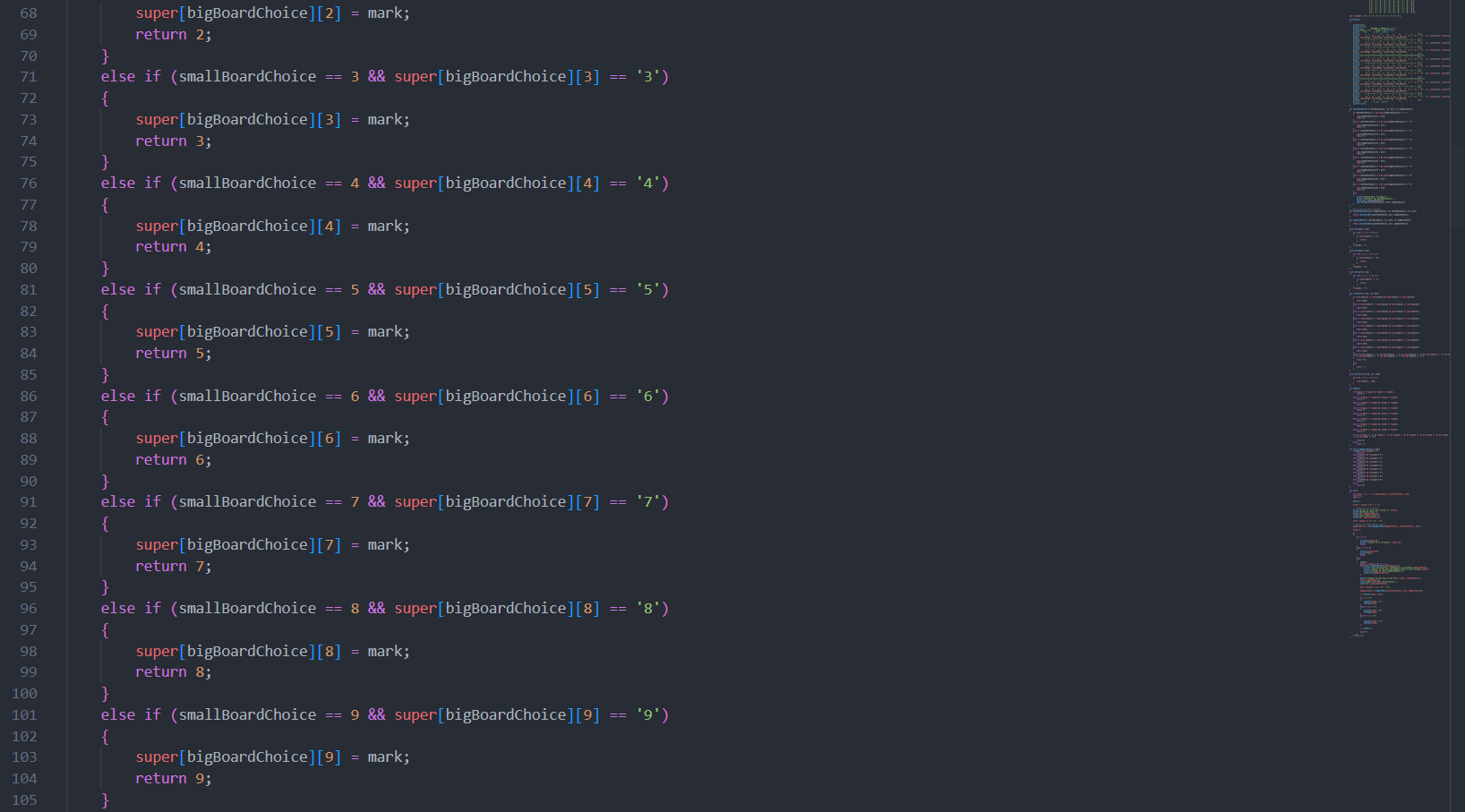
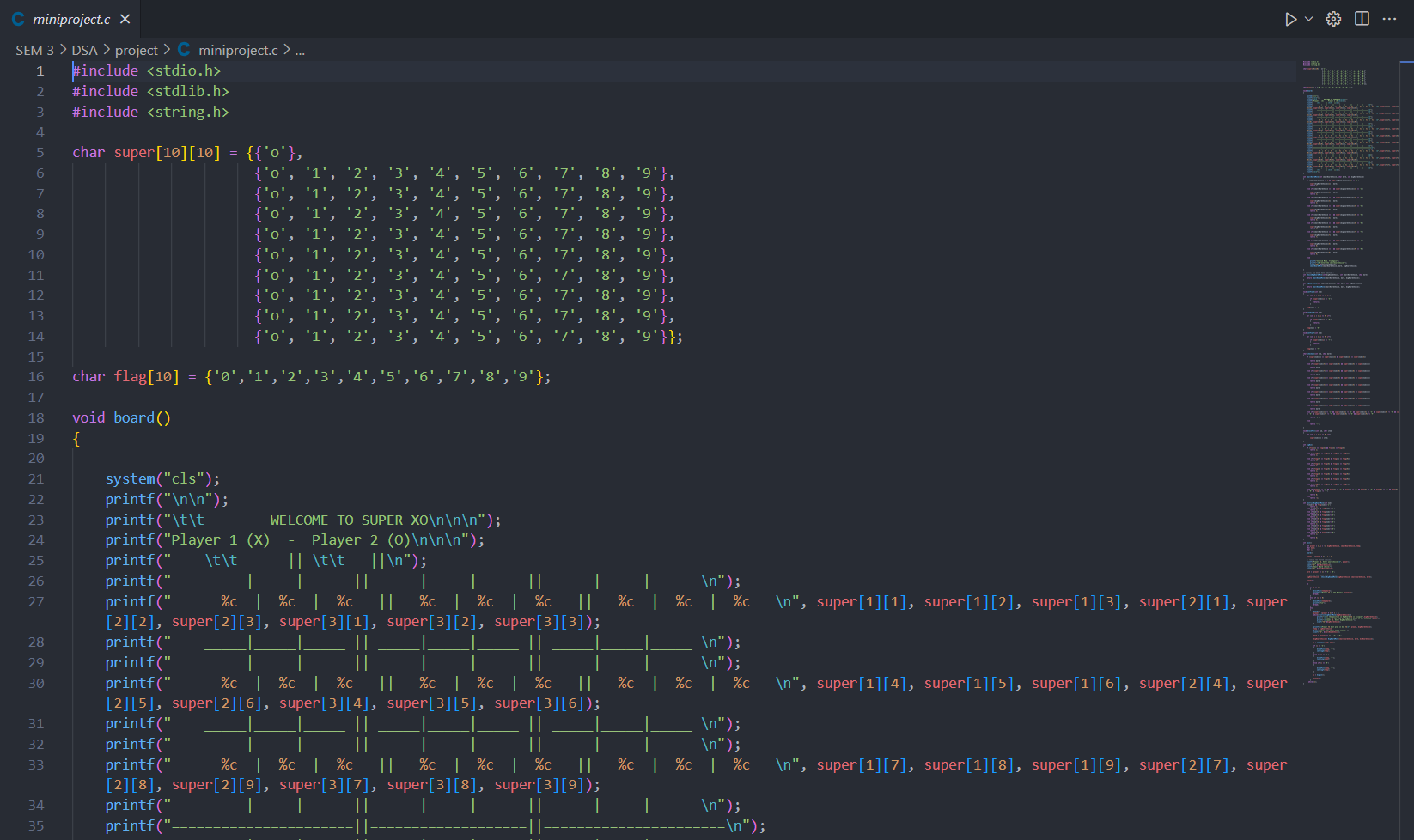
* If a player wins in a particular block of the big board , then all the positions of the small board in that particular big board is made ‘X’ or ‘O’ according to the particular player, or ‘T’ if it’s a tie.

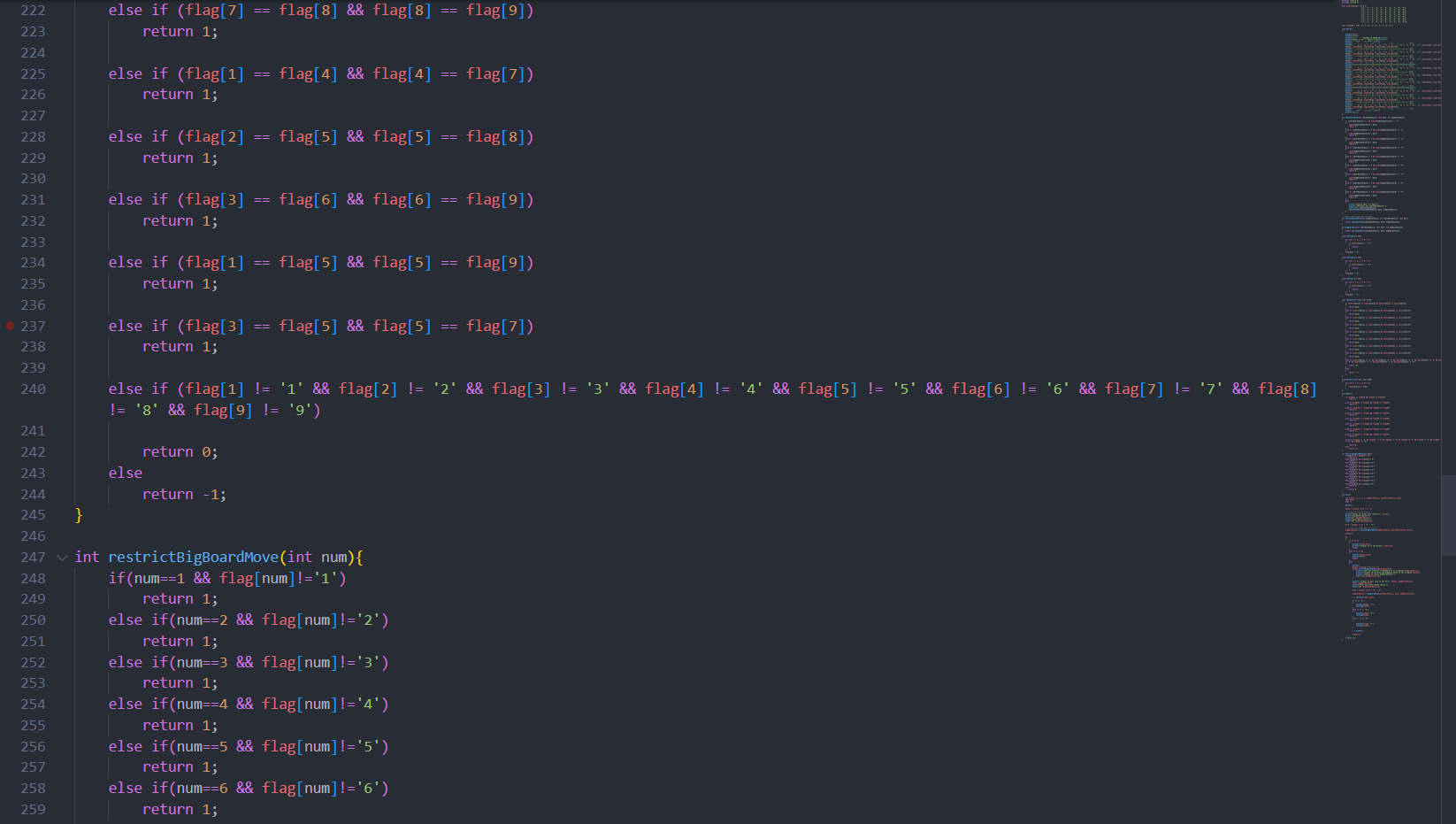
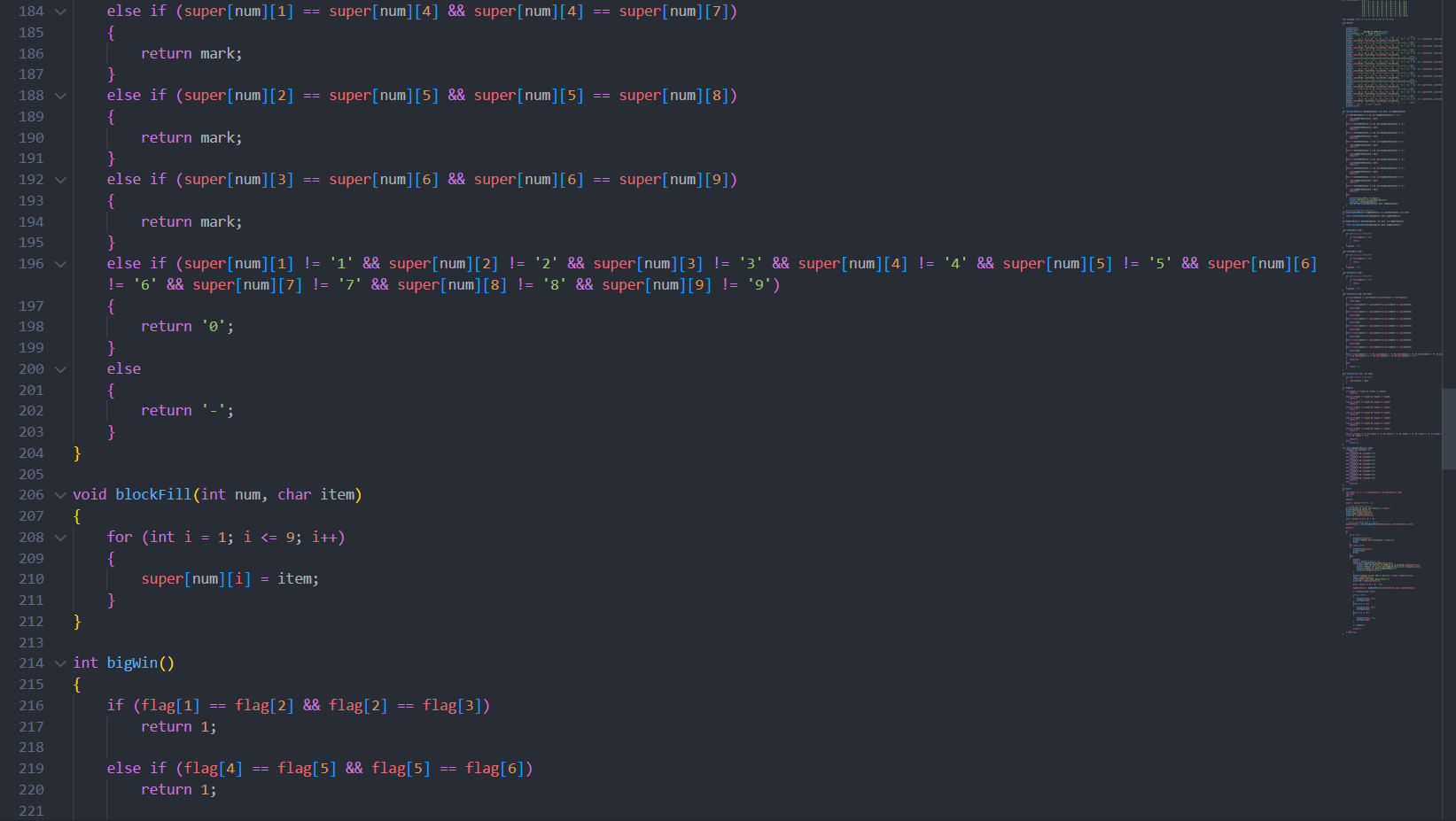
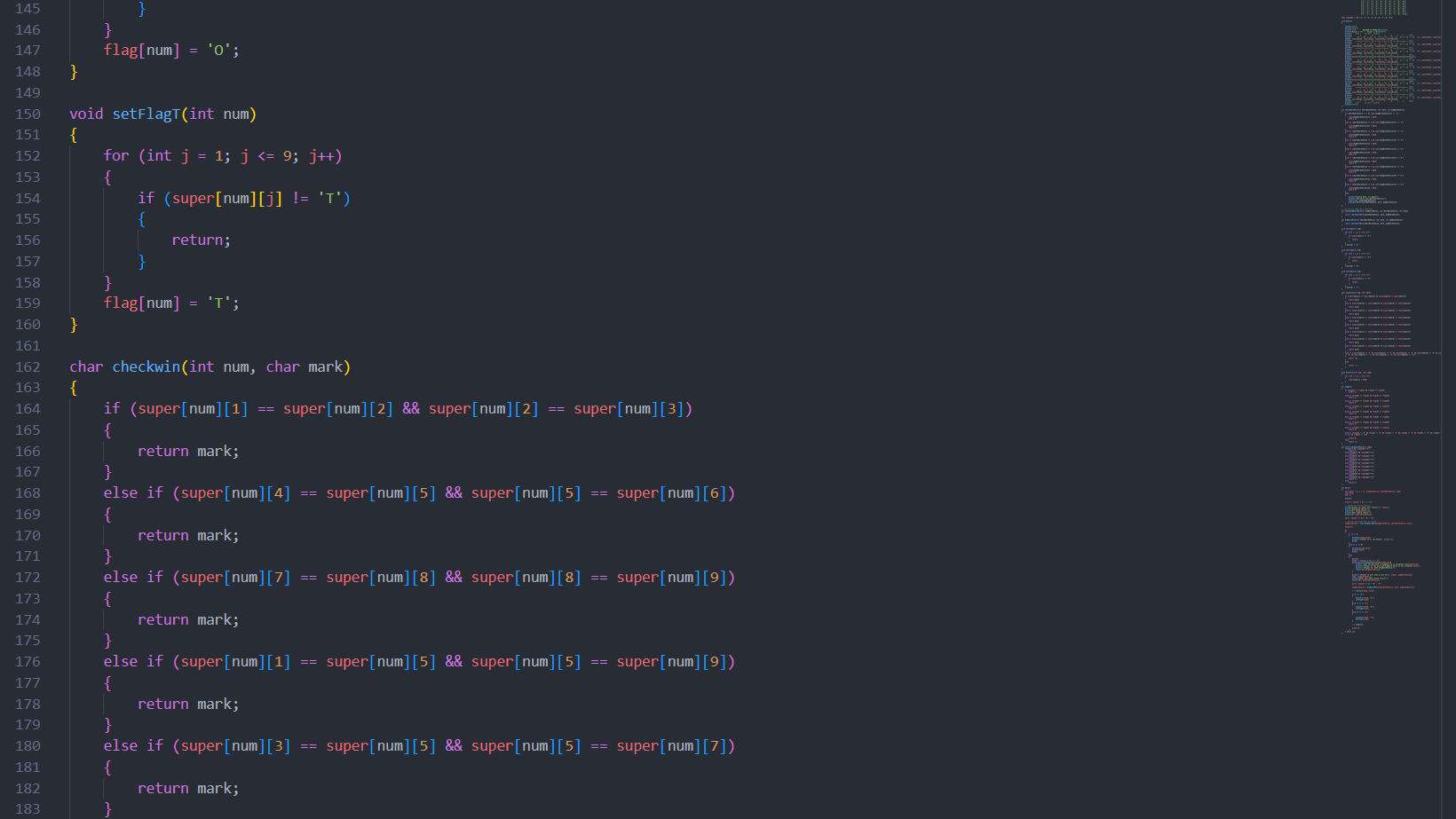
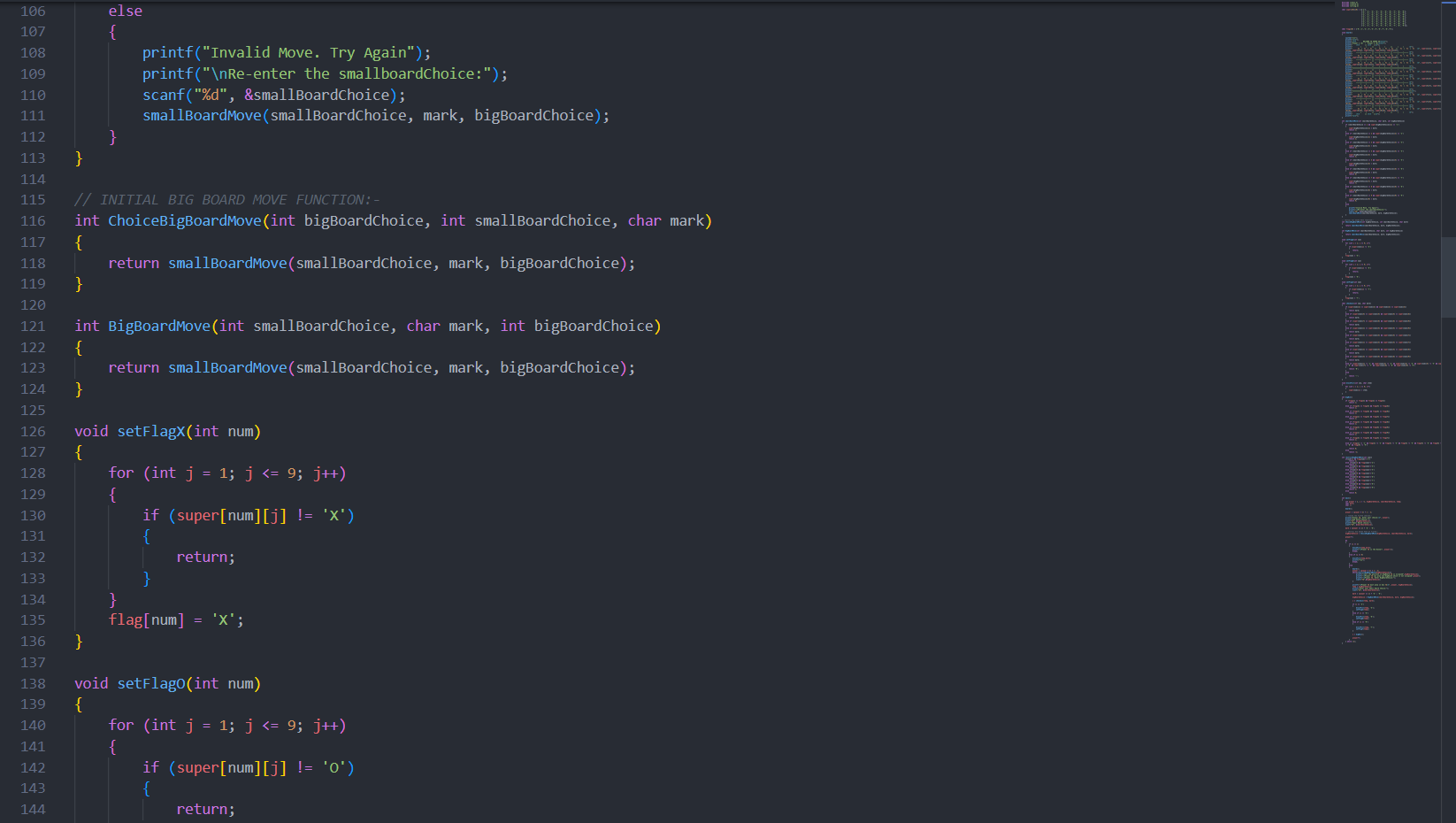
1. **Setting the flag :**

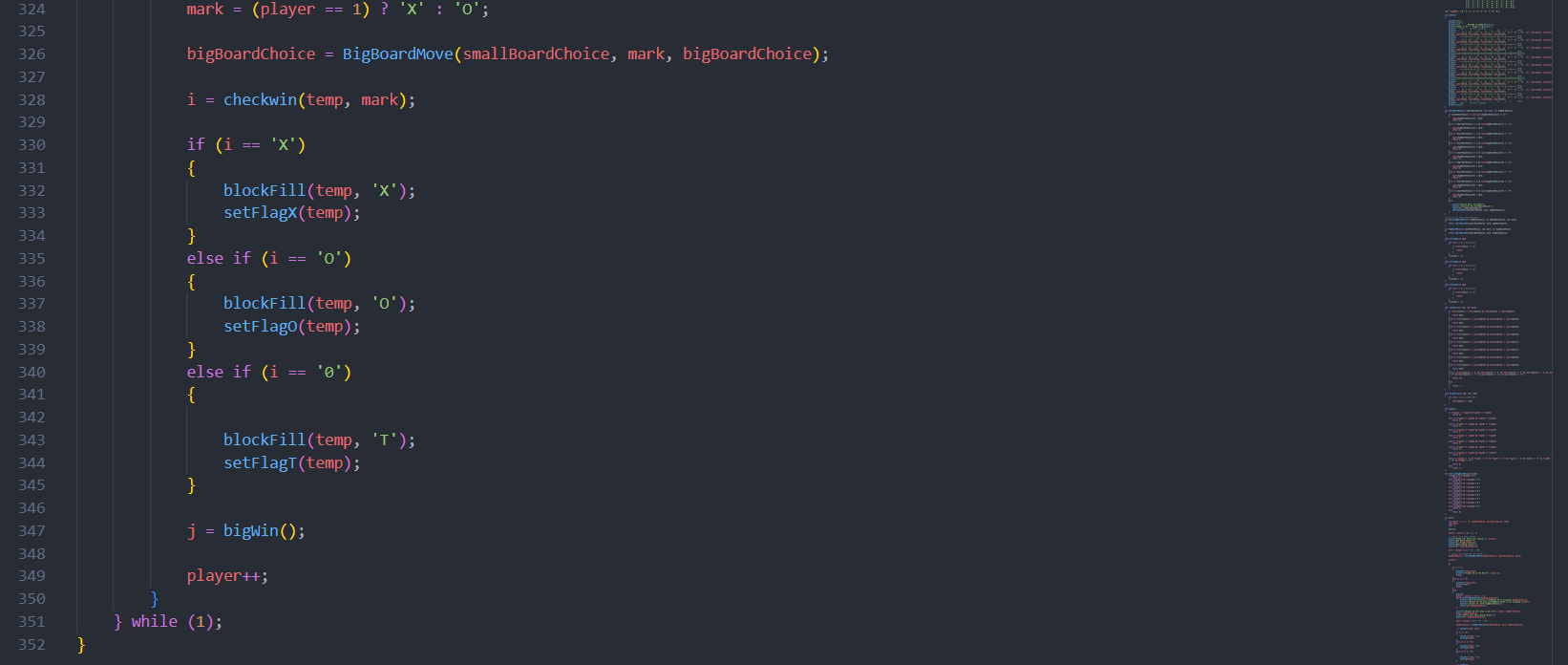
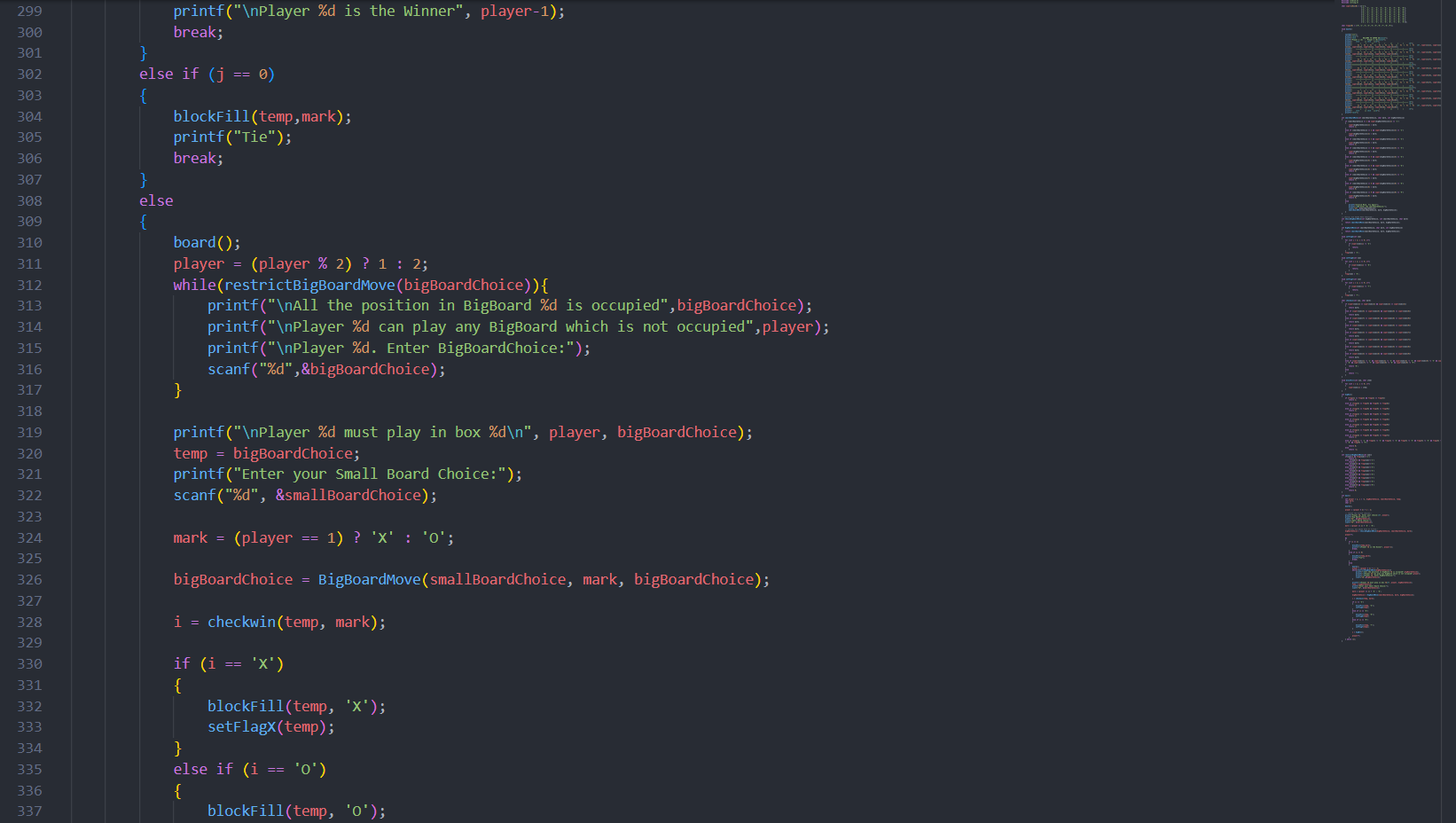
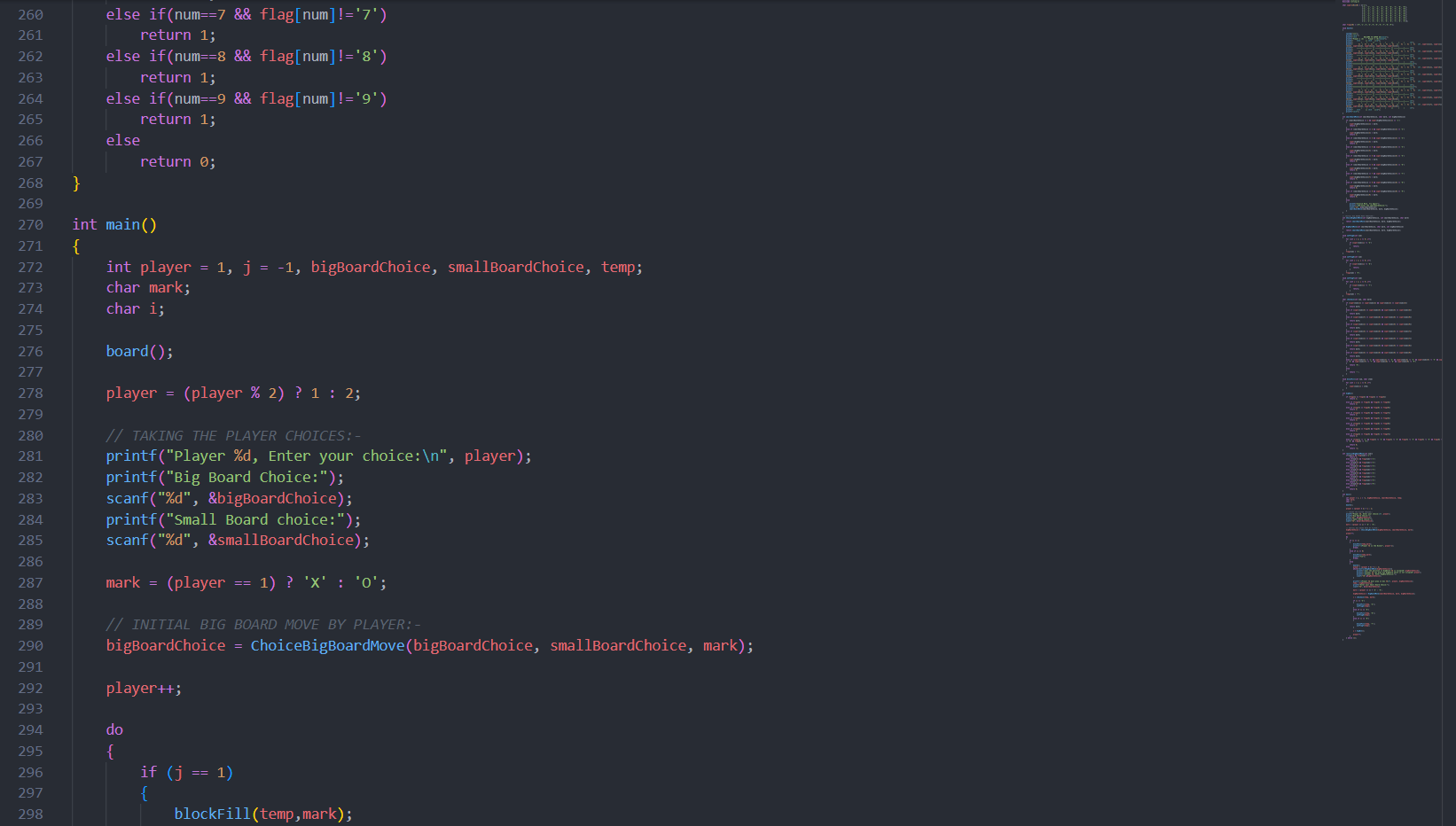
* Having an array named “flag”, if after the small board move, if a player wins in a particular block of the big board then, after the lock fill the position corresponding to the block of big board is flag array is set to ‘X’ or ‘O’ depending on the player or ‘T’ if it’s a tie.

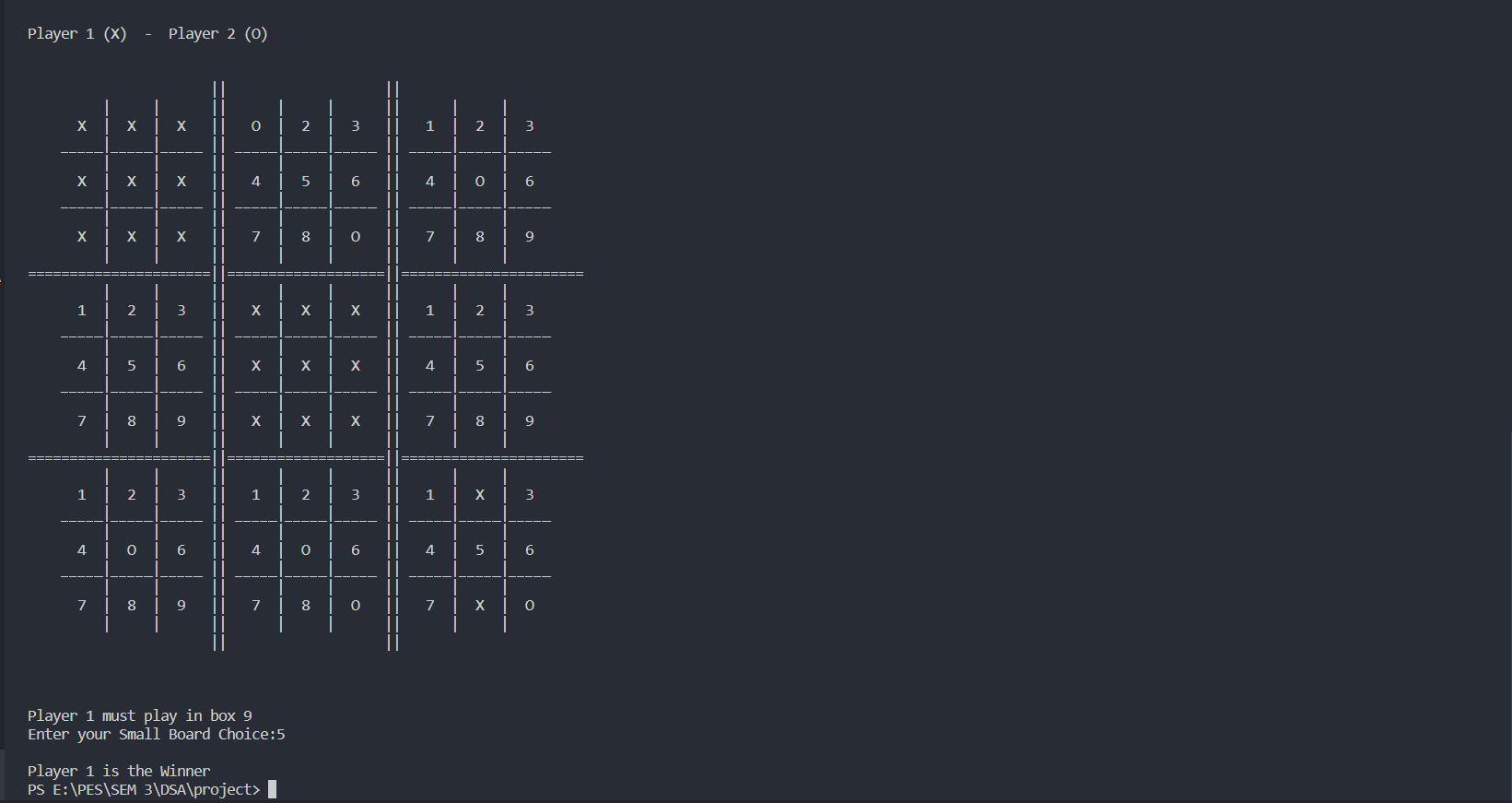
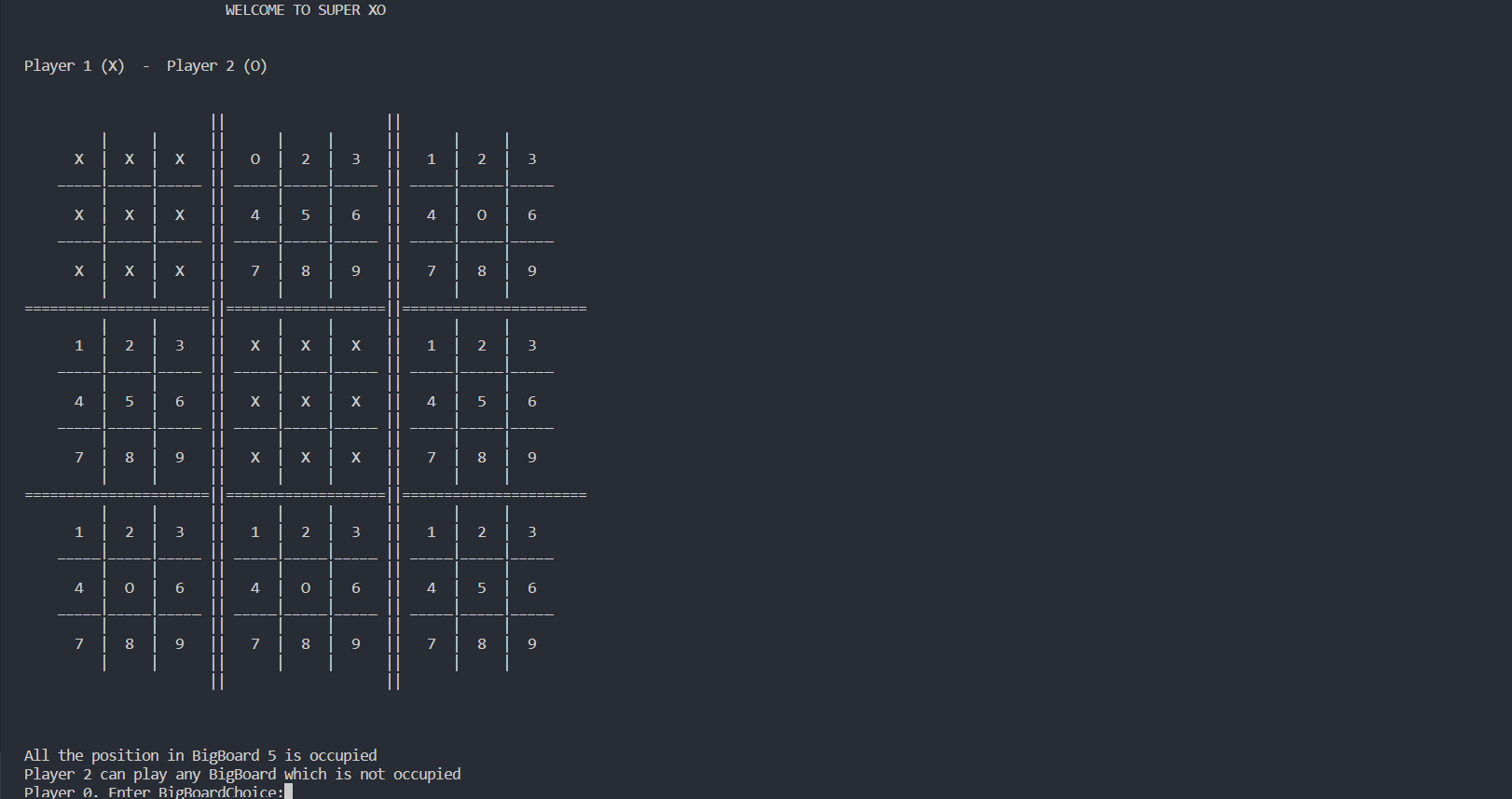
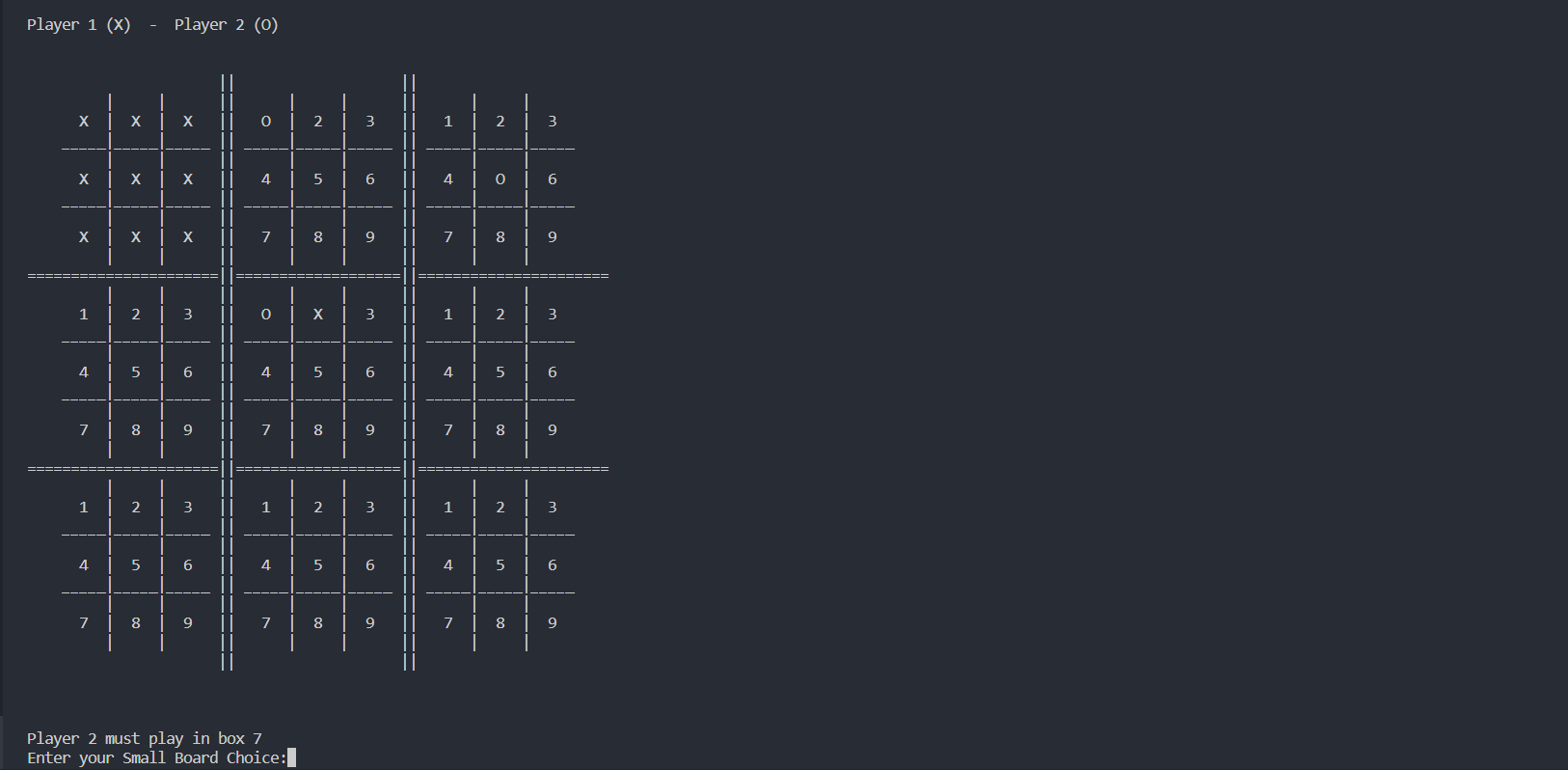
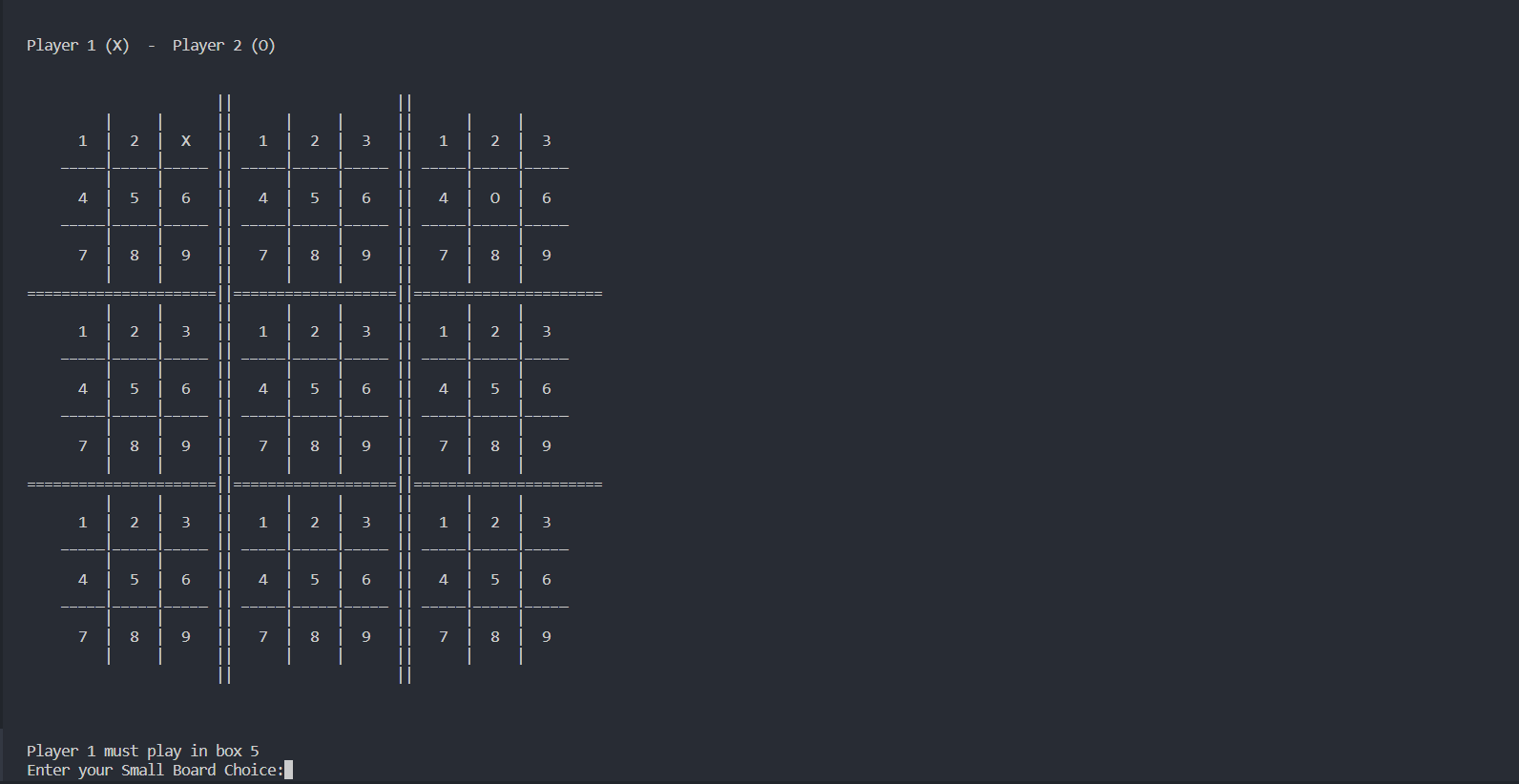
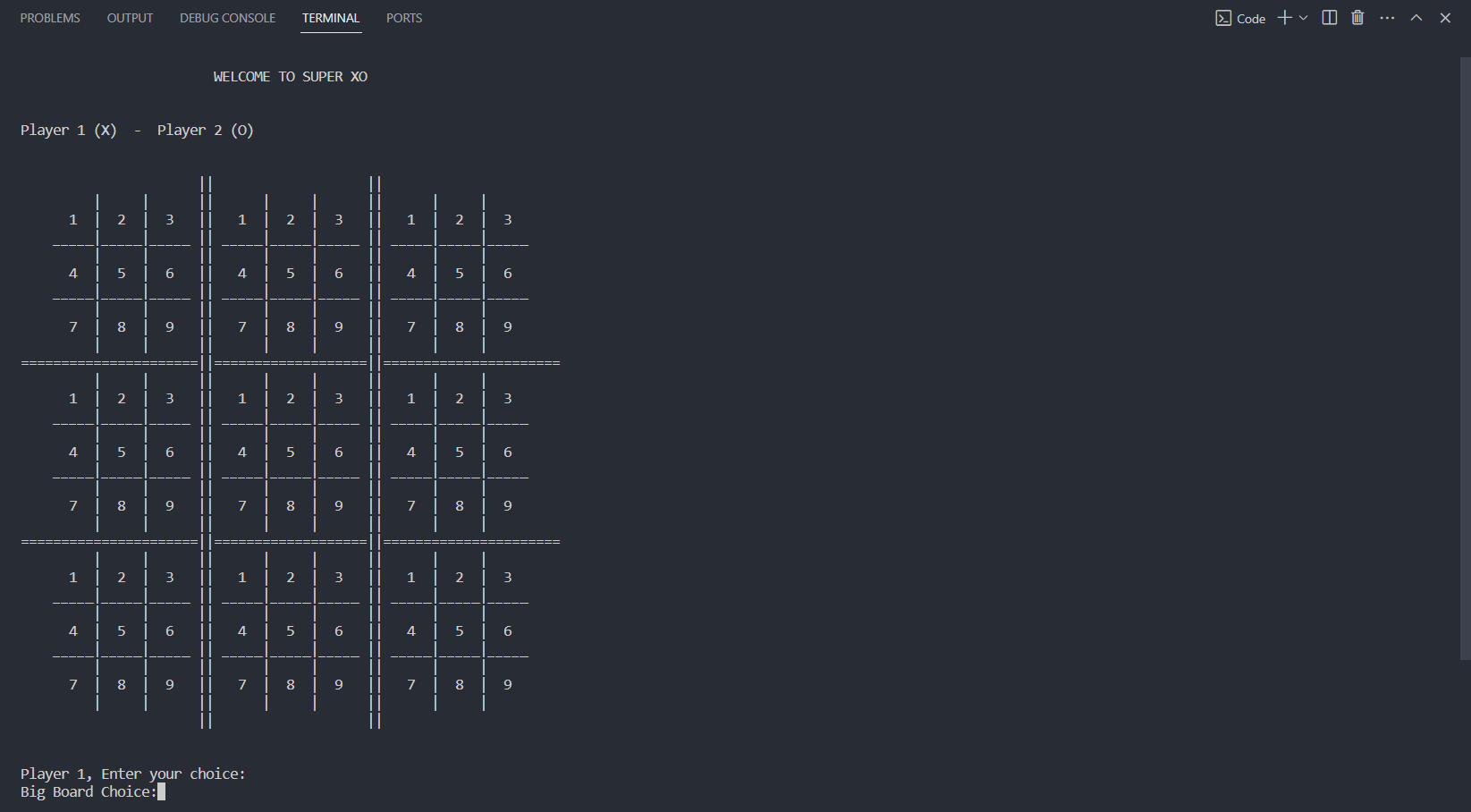
1. **Check win of big board :**

* If the ‘X’ or ‘O’ entered fulfils the win condition in the big board then declares the corresponding player as the winner

**Screenshots of the CODES :**





**Few screenshots of the output of the code : **