Part 1

Copy the Starter code for the Game Class (which represents a Tic Tac Toe game in memory. Complete the following methods:

The following methods are left for you to complete:

```
    /**
     * Attempt to play the current player's token at the given locatio
n
     * @param location board space to be played
     * @return true if the move is successful, false otherwise
     */
    public boolean play(int location) {

    }

    /**
     * Update the status of the board after a valid move is recorded
     */
    private int updateStatus() {

    }

    /**
     * Test three spaces in a line on the board for a game winner
     * @param a location of the first space
     * @param b location of the second space
     * @param c location of the third space
     * @return true if all three spaces contain the same player token
     */
    private boolean testLine(int a, int b, int c) {

    }

    /**
     * @return a list of all available moves on the board
     */
    public List availableMoves() {
```

```
}
```

Part 2 - Interface

Create an interface for the Tic Tac Toe game that will display the board, and provide the ability to create a new game and train the AI.

Part 3 - AI Player

Create the AI Player class. Add the code found in the AI Player Notes to create the Reinforcement Learning based AI player.

Add code to your TicTacToe interface to

- Train the AI agent, by having it play about 10,000 games. After this many games the AI will not be a great player, but it shouldn't be random anymore. Clicking the Train AI button a few more times to have it play about 40,000 games against itself should be enough to have the AI calculate an optimal strategy.
- Have the AI player make moves after the human player chooses their move.

Extensions

Possible ways to extend your program include

- Allow the human player to choose if they will go first or second
- Implement difficulty settings. The AI should always be fully trained. Difficulty can be adjusted by using different values for the "explore" parameter of the AI's `chooseMove` method.
- Save the memory table to disk after training it, and load the memory table when the program starts. This way you will not need to retrain the AI every time.