Assignment 3

One of the intriguing features of AlphaZero is that it learned to play chess (and Go, and Shogi) extremely well given *only* the rules of chess (or Go, or Shogi), and no special knowledge about how to make good moves. Here, we want you to write a program that does the same thing but for a much simpler game: Tic-Tac-Toe.

Your task is to implement (in Python) a Tic-Tac-Toe playing program where a human can play against the computer, and the computer makes all its moves using random playouts as described below.

Importantly, your program should not use *any* information about Tic-Tac-Toe beyond the essential rules of how to play it, how to determine legal moves, and how to recognize if the game is a win, loss, or draw.

Your program should work as follows. When it's the computers turn to make a move on a board, it should make a list of all legal moves. Then for each of the moves it does some number of **random playouts**. A random playout is when the computer simulates playing the game until it is over. During a random playout, the computer makes random moves for each player until a win, loss, or draw is reached. When a playout is done, the result (win, loss, or draw) is recorded, and then some more random playouts are done. After random playouts are done for all legal moves, it choses the move that resulted in the greatest number of wins (or least number of losses, or most number of wins + draws, etc. — whatever formula you find is the best way to make a decision based on these win/loss/draw statistics).

Do enough random playouts so that your program **never loses** against a smart player.

The exact number of random playouts that the computer should do is an interesting question, and you should determine this number by doing some experiments. Aim for the fewest number of playouts that results in your program never losing.

Remember: Do *not* add *any* knowledge about how to choose a good move in your program! When playing the game, your program must make its move decisions entirely by random playouts.

Your game should have a neat and easy-to-use interface in plain text.

Don't use any code from the textbook, or any special libraries for this: stick to regular Python, and code imported from its standard library. Write all the other code yourself.

What to Submit

When you are ready, put all your code into a single file named `a3.py` and submit it on Canvas. Please set up the file so that running the command "python3 a3.py" will launch a new game, e.g.:

```
# a3.py

# ...

if __name__ == '__main__':
  play_a_new_game()
```