Assignment 4: Tic Tac Toe search space representation using **game tree**

Due: Thursday March 24th at 11:59 PM

In this assignment, we will borrow a problem from the field of AI to sharpen our tree manipulation skills.

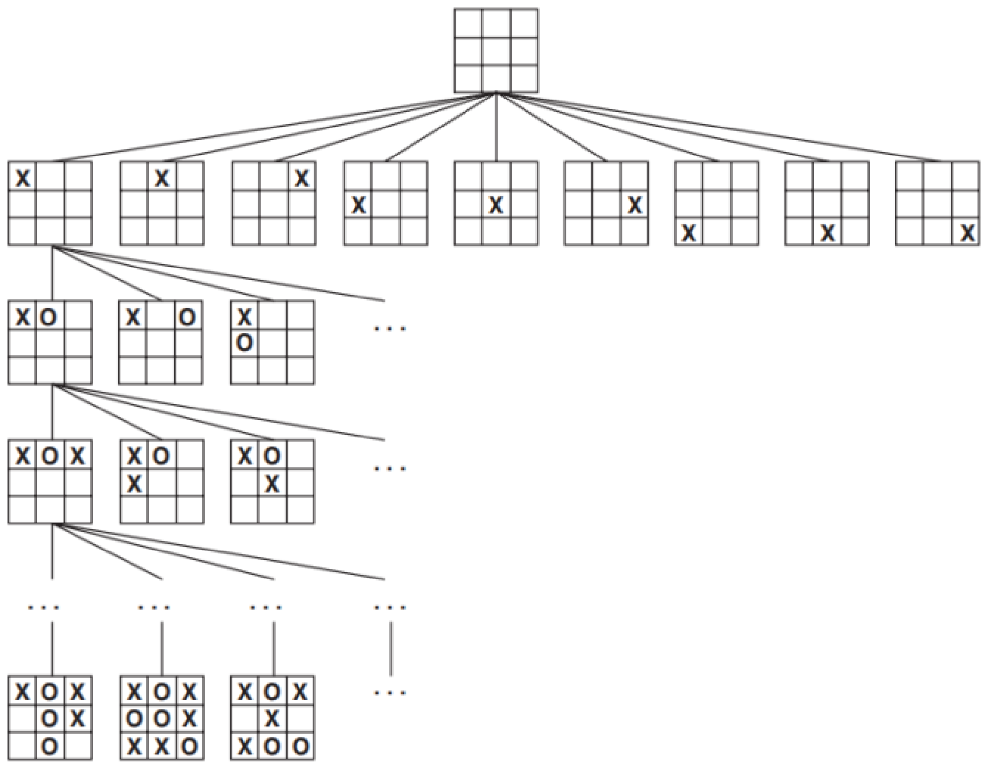
Tic Tac Toe is a two-player, turn-based game that takes place on a 3x3 grid (see below). At each turn, a player 'X' or 'O' places a marker on an unoccupied position on the 3x3 grid. The game ends when

a. One of the players makes a row, column or diagonal with their markers ('X' or 'O' is the winner)

or

b. The board is full, and no player has made a row, column, or diagonal (this is a tie).

The game can be represented using a **game tree**. The root of the game tree is the initial state (empty board) and every subsequent level represents a move by one of the players.



Part 1: 50% write a program in python to build this game tree. The tree should explore all possible states. Be sure to stop adding levels to the current branch when one of the players achieves victory (the leaves of the tree are all ties, victories by X, or victories by O). Player X moves fist. Modify the `LinkedBinaryTree` class from lecture 13 notes (page 320) to store your game states. Your game tree is not binary, on page 327 of lecture 13 you see the outline for 'Linked Structure for General Tree', use these concepts to implement the class that will store your game tree.

Part 2: 50% implement the breadth-first and depth-first traversals for your game tree (can use code from lecture 13 notes). In the main of your program, you should write some code to display a prompt to the user. The prompt will ask the user if they want to see a printout of the breadth-first or depth-first traversal of the tree. If they choose breadth-first, print out the first level of the tree and keep prompting the user to press 'enter' to see each subsequent level. If they choose depth-first, print out the depth-first traversal of the leftmost child (in the above example it is the board with 'X' in the top left square) and then keep prompting the user to see the other 8 children by pressing 'enter'.

Bonus +5% implement the minimax algorithm for tic tac toe.