This project was made with the goal of solving the problem of being able to play the 4-in-a-line game with the computer being able to determine the best possible moves through a certain mini-max alpha beta pruning search. My view of being able to solve this is through with one other additional class, which is Board, containing all pertinent operations on the current 8x8 2D array state of the game.

Ideally, I would have preferred to implement a very simple yet deeper ply search. Thus, I attempted to implement an evaluation function such that it would only search through the 2D array board and look for consecutives to give the utility of the best possible move. Quite simply, all the function does is loop through the 2D array row by row, and column by column looking for consecutives, incrementing the respective characters, resetting the respective character consecutive variables back to 0 when necessary, and checking for every consecutive 2 or 3 after every loop iteration and incrementing the utility value of the board by a certain amount based upon the amount of consecutives. Unfortunately given more time, I think it would certainly be more possible to implement bigger and more complex evaluation functions; however this type of game can only be so complex even with such an ideal evaluation function.

My attempt at the alpha beta pruning algorithm was simply to just follow the pseudocode as close as possible given in the lectures. It worked out for the most part, but there were some glaring issues that I had along the path to implementing this. One of such obstacles was adding in the check for a certain depth. I think this was successfully done through the terminal_test() method in my Board class. One other obstacle was being able to match the utility that gets returned from terminal_test() to the action that the computer is supposed to have moved to. A possible solution for that obstacle was to have a global variable keep track of the action that was to be chosen; however, I'm persuaded that I did not do this part well. The rest of the algorithm was just mostly copying from said pseudocode.

Despite all this, I was not able to finish the project completely. This is just the result of my own pitiful effort and a shameful attempt.