CS4750/7750 HW #4 (20 points)

Implement minimax algorithm to play a two-player, four-in-a-row game, which is a variation of tic-tac-toe: two players, *X* and *O*, take turns marking the spaces in a 5×6 grid. The player who succeeds in placing 4 of their marks consecutively in a horizontal, vertical, or diagonal row wins the game. See an example below where X player plays first and wins the game.

	o	x		x	
	o	o	x		
o	x	x	o		
	x				

This is a group assignment. You may use any programming language in your implementation.

You are asked to do the following tasks:

1) (6 points) Implement the minimax algorithm to play the game. Break ties based on left to right and top to bottom order. Describe the implementation in your report.
2) (4 points) Implement Player 1 (X player) as follows.
   - Player 1 makes the first move and puts ‘x’ at [3,4] position of the board.
   - Run the minimax algorithm on a 2-ply game tree, i.e., looking ahead 2 moves (one move by the player and one move by the opponent).
   - Use the following heuristic evaluation function on the resulting board, if it is not a terminal state. Breaking ties randomly. For terminal nodes, return their utility value: -1000, 0, or 1000.

h(n) = 200[number of two-side-open-3-in-a-row for me]*
– 80[number of two-side-open-3-in-a-row for opponent]*
+ 150[number of one-side-open-3-in-a-row for me]*
– 40[number of one-side-open-3-in-a-row for opponent]*
+ 20[number of two-side-open-2-in-a-row for me]*
– 15[number of two-side-open-2-in-a-row for opponent]*
+ 5[number of one-side-open-2-in-a-row for me]*
– 2[number of one-side-open-2-in-a-row for opponent]*

where

- “one-side-open-3-in-a-row”: there is an empty space next to one end of a 3-in-a-row to potentially make it 4-in-a row in the next move.
- “two-side-open-3-in-a-row”: there are empty spaces next to both ends of a 3-in-a-row to potentially make it 4-in-a row in the next move.
- “one-side-open-2-in-a-row”: there is an empty space next to one end of a 2-in-a-row to potentially make it 3-in-a row in the next move.
- “two-side-open-2-in-a-row”: there are empty spaces next to both ends of a 2-

in-a-row to potentially make it 3-in-a row in the next move.

For example, for player 'X', the value of the following state is
*h = 200*0 - 80*1 + 150*1 - 40*0 + 20*1 - 15*0 + 5*0 - 2*3 = 84*

		o	x		
		o	o	x	
	o	x	x	o	
		x			

3) (4 points) Implement Player 2 (O player) as follows.
   - Use the same heuristic function as Player 1.
   - Run minimax algorithm on a 4-ply game tree, i.e., looking ahead 4 moves (two moves by the player and two moves by the opponent).

4) (4 points) Play a game between Player 1 and Player 2. Print out the following:
   a) every move made by the two players for the whole game; and
   b) for each move, the number of nodes generated by minimax and the CPU execution time.

Submission:
   a) (18 points) A pdf file of your report containing descriptions of your implementation and result.
   b) (2 points) A zip file containing your code with appropriate comments.