CS4750/7750 HW #4 (20 points)

Fall 2016

(Due 10/6, Thursday)

In this programming assignment, you are asked to implement programs to play a two-player game similar to tic-tac-toe: two players, *X* and *O*, take turns marking the spaces in a 4×4 grid. The player who succeeds in placing 3 of their marks consecutively in a horizontal, vertical, or diagonal row wins the game.

You may form a team of up to three people. The team can be different from that for the HW#2. One solution is submitted by each team electronically in Blackboard. You may use any programming language in your implementation.

Part I. Beginner (6 points)

Implement a simple player, called Beginner, who places marks sequentially in a blank square in increasing order of row number and then column number, i.e., (1,1), (1,2), (1,3), (1,4), (2,1), (2,2), …, unless it or the opponent wins the game immediately. Its algorithm is as follows:

```
function Beginner-Decision (state) returns an action
    if the player has an open 2-in-a-row
        return marking the position to get a 3-in-a-row // "Win"
    if the opponent has an open 2-in-a-row
        return marking a position next to the 3-in-a-row to block the opponent
    return marking sequentially a blank square in increasing order of row
number and then column number.
```

"open 2-in-a-row" means that there is a blank space at one end of the 2-in-a-row, making it possible to become a 3-in-a-row.

Submission:

a) A brief description of your implementation.
b) Step-by-step of one game played between you (human) and the player Beginner. Beginner plays first.
c) Your code with appropriate comments.

Part II. Advanced (7 points)

Implement a minimax player, called Advanced, who runs 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). The heuristic evaluation function for cutoff nodes (non-terminals) is

h(n) = [# of open 2-in-a-row for me] - [# of open 2-in-a-row for opponent].

For example, for player 'x', the value of the following state is

h = (3-1) = 2

	o	x	
x	o	o	x
o	x	x	

When *h* values are the same, the search breaks tie randomly.

Submission:

a) A brief description of your algorithm and implementation.
b) Step-by-step of one game played between Beginner and Advanced. Beginner plays first. For every step played by Advanced, print the # of expanded nodes and the CPU execution time in milliseconds.
c) Same as (b) except Advanced plays first.

Part III. Master (7 points)

Implement a player, called Master, who runs the minimax algorithm on a 4-ply game tree, i.e., looking ahead 4 moves (2 moves by the player and 2 moves by the opponent). The heuristic evaluation function for cutoff nodes is the same as in part II.

Submission:

a) Step-by-step of one game played between Advanced and Master. Advanced plays first. For every step played by Advanced and Master, show the # of expanded nodes and the CPU execution time in milliseconds.
b) Same as (b) except Master plays first.
c) Your code with appropriate comments.

Your submission should be a single pdf file with file name containing your name and assignment number. For example, *firstnameInitial_lastname_hw4.pdf* for HW4.