**1. Discuss how well the standard approach to game playing (Minimax, Expectiminimax) would apply to games such as tennis, pool, and soccer, which take place in a continuous, physical state space. Here are some hints for the discussion:**

**a) Would the approach work?**

Approach like Minimax, is for minimizing the possible [loss](https://en.wikipedia.org/wiki/Loss_function) for a [worst case (maximum loss) scenario](https://en.wikipedia.org/wiki/Worst-case_scenario). So it would work for improve the calculate speed and efficiency.

**b) What would be the state space size?**

The state is any given configuration of the given board. So the state space consists of the set of all possible game states. For instance, the state space size for TicTacToe will be very large.

**c) What would it take to make the approach work in theory?**

Calculating the maximin value of a player is done in a worst-case approach: for each possible action of the player, we check all possible actions of the other players and determine the worst possible combination of actions—the one that gives player i the smallest value. Then, we determine which action player i can take in order to make sure that this smallest value is the highest possible.

**d) Would it work in practice? [6 points]**

Yes minimax and expectiminimax can be used for the game playing such as TicTacToe and Five-in-a-row. Because these algorithms provide better search results and are efficient in game playing.

**2. Describe how the minimax and alpha-beta algorithms change for two-player, nonzero-sum games, in which each player has his or her own utility function (for the leaf nodes). You may assume that each player knows the other’s utility function. If there are no constraints on the two terminal utilities, is it possible for any node to be pruned by alpha-beta? Discuss this subject. [6 points]**

Minimax: if each player has his or her own utility function, they are going to choose what is best for them. Even if it’s also good for another one. We can assign increasingly positively value for states which increase better for Max. Similarly, assign increasingly negatively values for states increasingly better for Min. In this situation, the algorithm works unchanged. If in case both players have increasingly positive values, then each player can optimize to choose the maximum value every time to win.

Alpha-beta: In alpha-beta, we go by notion that what is good for max and what’s bad for min. Min won’t allow Max to go down a path since Min can force something worse, hence Max knowing this does not have to explore the path. Since, we have been given that this is a zero-sum assumption, the same state cannot be good for both Max and Min. You cannot just assume that what Min likes, Max won't.