CONNECT-K FINAL REPORT ***[TEMPLATE --- do not exceed two pages total]***

Partner Names and ID Numbers:

Team Name:

1. Describe your heuristic evaluation function, Eval(S). This is where the most “smarts” comes into your AI, so describe this function in more detail than other sections. Did you use the dot product of a vector of weights with a vector of features? What features? How did you set the weights? Did you simply write a block of code to make a good guess? What heuristic did you use? Please use a half a page of text or more for your answer to this question.

2. Describe how you implemented Alpha-Beta pruning. Since you put it on a switch so that you can turn it on and off, please evaluate how much it helped you, if any.

3. Describe how you implemented Iterative Deepening Search (IDS). Were there any surprises or difficulties?

4. Did you remember the values associated with each node in the game tree at the previous IDS depth limit, then sort the children at each node of the current iteration so that the best values for each player are (usually) found first? Describe the data structure you used. Did it help?

5. Describe your quiescence test, Quiescence(S). Did it help?

6. Any suggestions for improving this project? (One suggestion is to remove the first-player advantage: the first player initially places one single mark, and then the players alternate each placing two marks per turn. But, this would square your branching factor for each ply. I hope to compensate for this in the tournament by having each pair play both sides in alternation.)

1. Asdf
2. Asdf
3. For Iterative Deepening Search, we started with a depth of zero to see what possible moves were available to us. After evaluating each move, we finish the Search so that we at least have a move to give once the turn’s time limit has been reached. Then we proceed to the depth of one to see if the moves that we analyzed at the previous level could lead to a better state than the possible moves from the previous depth limit. If we find a better outcome, then we change the move we intended to return to the move that would lead us to our new best state. We continue to move deeper down the search with increasing iterations, revising our intended move as we analyze more and more BoardStates into the future until we run out of time. At this point, we make the move that we found leads to the best BoardState that we had analyzed before the deadline was reached. (Did it help?)
4. We used a TreeMap to store and sort the values associated with each BoardState so that we can expand the best node each time when we move to the next level of the IDS limit. Because a TreeMap by default sorts from least to greatest, we had to adjust the evaluation so that the best move for the AI would appear first when iterating from the front of the TreeMap to the back. (Did it help?)
5. Asdf
6. One possible solution for improving this project is to compensate for first-player advantage by decreasing the worth of a win by the first player, and increasing the worth of a win by the second player. A solution to help students implementing the AI would be to offer code examples of how to perform Iterative Deepening Search, or Quiescence Tests and other similar searches. (Did it help?)