Name: Alambek Gulamidinov ID: 20192012

GAME-TREE SEARCH: I implemented game-Tree search using alpha-betta minimax algo + memorization with hash function to optimize my code.
I created hash function that maps every cell number to some power of prime number so that when we sum up mapping values of defined cells we will have unique sum that corresponds to exactly for those cells.
This will cut running time since I need to store just one int to keep sum of the hash of the cells where players moved, instead of tracking every move on the board.
And Also this helps to track only unique values (we can make different multiply random moves in the board that could come up to the same final state, however when keeping sum of hash will be always same, not having permutation and others. So it will be really fast.

PLAYING AS THE FIRST and SECOND PLAYER:
I Couldn't check how my code runes when playing this functions.
However I made them using same strategy for counting minimax.

MAXIMUM PROBLEM SIZE: I don't know exact limit of my program. But I suppose my code can count 6x6 tables too. It run sample5x5a.txt in 13 seconds.

GAME-TREE PRINTOUT: Printout was made by different method.
I didn't store hash for this, since I need to print the board. That's why printing is very slow. However it uses minimax algo with alpha beta.

BONUS:
I want to claim for bonus point.
I used hash function to optimize my game-tree search. This hash function will enable to cut the running time a lot, since instead of traversing the board and keeping values for Player1 and Player2, I just have the values (two integers) that can track every move made by players in O(1): adding hash function of cell to the Player's move data.