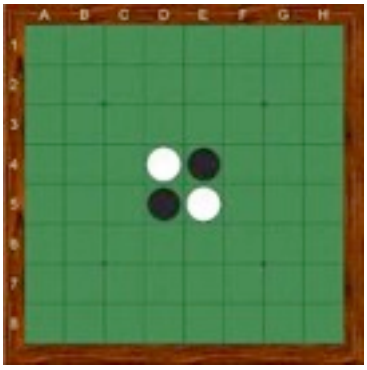
CS4300 Spring 2016 Hauschild

Project 3: Othello
Due Date: Mar 16th, 2016

[100pt] **Othello:**

Your task in this assignment is to write a program that plays as close as optimal Othello (also called Reversi) vs a human player on an 6x6 board as you can do under some constraints:

Othello is a game normally played on 8x8 (doing 6x6 to make the games quicker) board with white and black pieces, with black starting first. The board starts with the center 4 positions occupied by 2 black and 2 white pieces as below:



Black has the first move and can put a piece anywhere that would result in a white piece flipping. After black places its piece, any white pieces between it and a black piece are flipped.

Now it is white's move. It can place its piece anywhere that would result in some black pieces being sandwiched by white. Any ones sandwiched by this move are flipped. Note that any pieces that just happen to end up surrounded by another color (after this process) are not flipped.
This alternating of moves repeats until the board is filled.

Specifications:

1) You must ensure the optimality of your player by using minimax (or alpha-beta pruning). You are not allowed to simply use lookup rules or something similar.

2) Prompt the user if they want to go first or second against the computer.

3) Each turn, display the board and let them enter a x,y coordinate position on your board to indicate their move. This should be x,y from the top left. So the top left on the board is 0,0, bottom right is 5,5

4) Display the board state between moves, as well as the minimax result from that move.

5) You should do this using iterative deepening minimax. That is, do it to depth 1 (and keep track of the move), then to depth 2 (take this move), and so on, keeping track of total time. Every expansion down minimax, check the time. If more than 5 seconds have elapsed, take the previous best move found and terminate your current search.

6) As you are terminating your search and not going all the way, you need a heuristic. A simple one might be the number of your own pieces on the board.

I strongly suggest you get the game running with you making all the moves, then once the game is playable, implement minimax for one player.