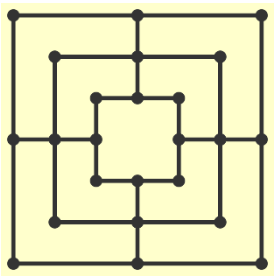
Yunchao Liu **UTD ID: 2021177759**

Project's Short Description

1.Program (Nine Men's Morris) Description:

Game Rules:



Nine Men's Morris game is a board game between two players: White and Black. Each player has 9 pieces, and the game board is as shown. Pieces can be placed on intersections of lines(There are a total of 24 locations for pieces). The goal is to capture opponents pieces by getting three pieces on a single line (a mill). The winner is the first player to reduce the opponent to only 2 pieces, or block the opponent from any further moves. The game has three distinct phrases: opening, midgame, and endgame.

- Opening: Players take turns placing their 9 pieces – one at a time – on any vacant board intersection spot.
- Midgame: Players take turns moving one piece along a board line to any adjacent vacant spot
- Endgame: A player down to only three pieces may move a piece to any open spot, not just an adjacent one(hopping).
- Mills: At any stage if a player gets three of their pieces on the same straight board line(a mill), then one of the opponent's isolated pieces is removed from the board. An isolated piece is a piece that is not part of a mill.

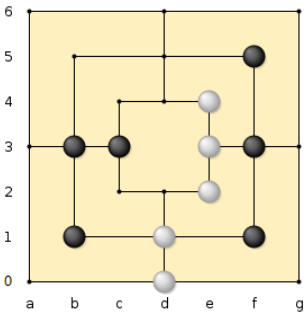
My computer program that plays Nine Men's Morris:

My program will be the user's opponent and playing White. It will take input by reading a .txt file, which contains the current board state information, and generate another .txt file showing the new board state after it makes a move by following the game rules. It uses evaluation function for deciding which move it should make. The user then can make move by editing the .txt file. However, the .txt file is not in a visual form of the board, but instead a string showing the current state(This will be further explained in the solution part below) so another Draw.java program is used for displaying the current board state in a visual form for user's convenience. The program will pop out message when either the user wins or loses.

2. Proposed Solution:

Representing board positions:

the board position is represented by an array of length 24, containing the pieces as the letter W, B, x.(The letter x stands for a "non-piece"). The array specifies the pieces starting from bottom-left and continuing left-right bottom up. Here is an illustration:



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
a0	d0	g0	b1	d1	f1	c2	d2	e2	a3	b3	c3	e3	f3	g3	c4	d4	e4	b5	d5	f5	a6	d6	g6
x	W	x	B	W	B	x	x	W	x	B	B	W	B	x	x	x	W	x	x	B	x	x	x

Implementing game rules:

Game rules will be implemented by using a move generator function to generate legal moves. It will take a board position b as input and generate a list of all positions reachable by a white move(which is the color the program plays with). Then by

evaluating different moves, it will output the optimal solution in the same form as input.

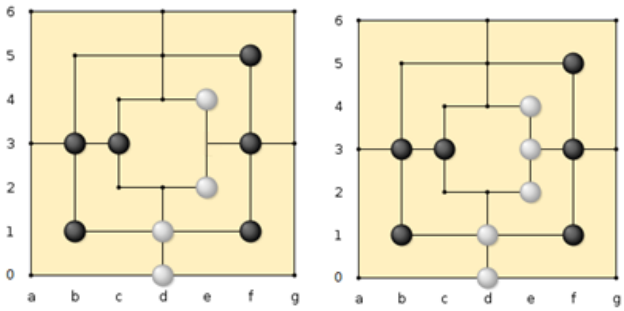
Evaluation function:

In the program, I am going to use static evaluation for each state. For static evaluation, from the computer's point of view(i.e, from the white player's pointer of view), during mid and end period of game, the value of a state is calculated from the number of white pieces, number of black pieces and the numb of moves that black can make. If the number of black pieces is less than 3, the value is 10000, which means white wins. If the number of white pieces is less than 3, the value is -10000, means white lose. If the number of black moves is 0, the value is 10000, which means black has nowhere to move and white wins. Else from the above condition, the value is 1000*(numWhitePieces – numBlackPieces) – numBlackMoves. During Opening, the value is calculated from the numWhitePieces – numBlackPieces.

3. Overview of the implementation details:

Example：

Let's consider the input the following board state in (a), whose corresponding string is xWxBWBxxWxBBxBxxxWxxBxxx. The user should specifies the which period as well(1 for Opening and 2 for Mid and End game). The String will be in a .txt file. The program will get all possible board states after one move by white. Then the program will compare different value of the state and output the one with highest value, which is shown in (b). The corresponding string of (b) is xWxBWBxxWxBBWBxxxWxxBxxx.



(a) (b)

To input, the command line is like following (in Linux environment):

java NineMenMorris board1.txt 2

And the output string will be in a .txt file as well.

To visualize the string in the .txt file, use the following commend:

java Draw board1.txt

Programming tools:

The program is written in Java in Linux environment in the cs1 server. No other third party software will be used.

Architectural Diagram: