Computer Science 1 – Group Project
Reversi
Assigned: 2/15/23 Thursday
Due: 4/4/24 (Thursday) at 11:59pm (WebCourses time)

Problem

Reversi is a game with two teams: BLACK and WHITE played on an 8 x 8 board. The initial board has two BLACK pieces and two WHITE PIECES. (Black pieces are at positions (3, 3) and (4, 4) and white pieces are at (3, 4) and (4, 3), where rows and columns are labeled 0 to 7, inclusive.

On a move, a player can go in any empty square such that doing so forms some consecutive streak of pieces where the opponents' pieces end up in between two of your pieces, one of which, you just placed, with no empty pieces in between. When you move in such a square, for all new streaks formed, all of the opponents' pieces turn into your piece.

The goal of the game is to have as many pieces as possible on the end board.

Black goes first and players alternate turns. If no valid move is available for a player, the other player gets another turn. If neither player has a valid move, the game ends. The winner is the player who has more of her pieces on the board at the end of the game. Most average games fill the board completely and it's relatively rare to not have any possible moves for a team.

Ultimately, your team's project will be to write a function that is given a board state and the team that is currently playing and return one of the valid moves for the team in question. A framework has been posted online. You should add two new files (one .c file and one .h file) to that framework for your own testing and implement the required function. You will only submit to me your one .c file and one .h file.

Many competitive games are timed, thus, for your project, in addition to considering the board position, you'll also have to consider how much time you have left to spend on all of the moves left in the game. In particular, your computer player will be given 3 minutes to complete all of its moves in a particular game. Whenever it's asked to move, it will be told exactly how much time is left in the whole game.

Implementation Requirements

The code framework is online to download in a zip file called "ReversiFramework.zip" There are two players included in the file team20.c/team20.h and team21.c/team21.h. You should add your two files teamAB.c/teamAB.h, where AB is the two digit representation of your team number. Your file must have the following function, where AB is replaced by the appropriate digits.

```
position* teamABMove(const enum piece board[][SIZE], enum piece
mine, int secondsleft);
```

You may call any of the functions in reverse_functions.c, but make sure you free any memory you allocate during the time your function gets called.

To make sure that no function names conflict, every function you write must have the prefix teamAB.

Here are some further restrictions:

Your function is NOT allowed to call the function for other teams. In essence, you are NOT allowed to find out what another team "would do" if you were to make a particular move, when deciding your own move.

Your function can't read from or write to a text file.

Your function can't connect to the internet.

Note: I may add other restrictions that I haven't thought of yet. (All three of these were prompted by student questions.

How to Compile

With two players playing, the following line on the command line will compile:

```
gcc -o reversi.exe reversi.c reversi_functions.c team20.c
team21.c
```

When you are testing of course, use one of team20.c or team21.c and then substitute your team file.

All files should be in the same directory.

How the Winner and Loser in a game will be determined

You have no responsibility to write code that determines when a game is over, who has won a game, or how much time any team has used in the course of a game. Your ONLY responsibility is to write the function specified previously.

I will write the code that calls each team's move function. (Note: Each team will have a function with a different name. These will be given out in class.) My code will "manage" the game by keeping track of how much time each team has spent on their moves, determining whether a team has returned an illegal move, and determining which team has won (if any) after each move. You can download this code from the course website.

The way my code will work is as follows: If a team has exceeded its time limit, returned an invalid move, OR made a move that has resulted in a losing board position, then the game will immediately be halted and the other team will be declared the winner. Similarly, if a team returns a move that results in a winning board position, the game will be halted and the team will be declared the winner. In all other cases, the other team will simply be called to move on the adjusted game board.

The Tournament

We will have fourteen teams in the tournament, and there will be some "pool play" on Tuesday, April 9th. This play will be used to "rank" the teams for the single-elimination tournament on Thursday, April 11th. In pool play each team will play the same number of games and each team will be BLACK the same number of times to ensure fairness in comparing outcomes.

In the single elimination contest, in the first couple rounds, the team that goes second (presumably better) will be based on seeding. Both the semifinals and finals will be a best of 3 series. In particular, the first two games of the series will have each team go first once. If these games are split, then the starter for the last game will be determined by a coin flip. (Note that if both players are deterministic, then in each encounter where the starting team is the same, the outcome will be identical. So, maybe introduce some randomness into your player!!!)

Deliverables: Due over Webcourses ON April 4, 2024 11:59 pm.

Turn in two files over WebCourses that contains your solution to the given problem. One file should be teamab.c file with all of your functions (no main should be included), and the other file should be teamab.h file which includes all of your function prototypes and constants. (Again, ab is the two digit representation of your team number.) You must have the one function specified in the description and may include any other functions you deem necessary. Make sure to include ample comments and use good programming style, on top of the requirements that are given in the program description above.

Deliverables: Due over Webcourses ON April 10th, 2024 3:00 pm.

Based on your pool play, you may make changes (but are not required to do so) and resubmit your edited player by Tuesday at 3 pm. If you don't submit anything, your player from pool play will be used. If you submit something but I can't get it working with my framework Tuesday night, then I'll also use the player from pool play.

Grading Specifications

Documentation and Style: 15%
Following Implementation Restrictions: 10%
Performance in Contest: 30%
Effort and Sophistication of Computer Player: 45%
(approximate breakdown only)