CIS*3700 Winter 2023
Assignment 2
Due March 3rd, 2023 (+ 2 day grace period @ -25% per day)
Submission requirements

- Submission **must** include a README file with instructions for the marker to run code
- Code must be properly commented such that the marker can find and mark each requirement easily
- All required files must be uploaded to CourseLink in a zipped folder
- No additional libraries may be used to implement required algorithms, outside of basic helpers. i.e. scipy.eucldist(a,b) is okay, scipy.a_star(my_graph) is not. Trust your judgement
- Work is to be completed individually, and of your own creation

Basic Chess Playing

In this assignment, you will be creating a basic chess AI. Basic being the key word.

For your AI, you will need to implement a Min-Max tree with A-B pruning. Recall that Min-Max trees require an associated cost with its decision making. The evaluation function of a board state will be up to you to create. The most basic cost is assigning each piece on the board a value, and comparing the total value of the pieces on the board. E.g.

Pawn = 1
Rook = 4
Knight = 3
Bishop = 3
Queen = 5

Where a terminal condition is the King being taken. You can get as complex as you wish for judging board states. Just remember that adding more complexity to your costs increases the time required to make a decision, and your AI should be able to make a move within a few seconds. No one wants to play chess by mail with their computer. Use a reasonable depth for your search.

Include in your README a basic outline of the evaluation function you used.

You will be required to create the environment for your chess game entirely yourself. This includes board representation, rules, and output. The following rules will need to be implemented:

Piece movement
Piece taking
Promotion

Check
Castling
Stalemate (No legal moves)
Checkmate

We will be forgoing En Passant and 3-peat board state for simplicity sake. Sorry chess purists.

Output of the boardstate will be in text format. You will need to designate characters to each piece, then output the current board to the console.

Example board:

1	wR	wN	wB	wQ	wK	wB	wN	wR
2	wP	wP	wP	wP	wP	wP	wP	wP
3	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
5	.	.	.	.	.	.	.	.
6	.	.	.	.	.	.	.	.
7	bP	bP	bP	bP	bP	bP	bP	bP
8	bR	bN	bB	bQ	bK	bB	bN	bR
	A	B	C	D	E	F	G	H

On this board, the lowercase prefix indicates the colour, and the capital letter the piece.

P = Pawn
R = Rook
N = Knight
B = Bishop
Q = Queen
K = King

The numbers on the side and letters on the bottom are used to indicate location on the board, and are used to input moves. The first pair will indicate the piece to be moved, with the second pair being the location the piece is moving to. You will need to make sure each move is legal before updating the board. For example:

BLACK TO PLAY

Input:
C7 C5

Output:

1	wR	wN	wB	wQ	wK	wB	wN	wR
2	wP	wP	wP	wP	wP	wP	wP	wP
3	.	.	.	.	.	.	.	.
4	.	.	.	.	.	.	.	.
5	.	.	bP	.	.	.	.	.
6	.	.	.	.	.	.	.	.
7	bP	bP	.	bP	bP	bP	bP	bP
8	bR	bN	bB	bQ	bK	bB	bN	bR
	A	B	C	D	E	F	G	H

WHITE TO PLAY

Your AI player will update the board state internally, without the need for location pairs.
Helpful suggestion:

Start this assignment by first creating the chess game with 2 players. Having a working game before you start the min-max tree will make things much easier than trying to test both at the same time. This will also be used to test your implementation of the rules.

The game should start by asking if the game will be 1 player or 2. If 1 player is selected, it will then prompt for if the player wishes to be black or white. Remember that white always plays first (despite my example).

BONUS: Champions of 3700

Details to follow.