Assignment 2

Game Playing Problem

Max possible score:

- 4308: 100 Points [+20 Points EC]
- 5360: 100 Points [+20 Points EC]

Task 1

Max: [4308: 100 Points, 5360: 100 Points]

Your task is to build an agent to play two versions (standard and misére) of the variant of a game called nim (called red-blue nim against a human player). The game consists of two piles of marbles (red and blue). On each players turn they pick a pile and remove one marble from it. If on their turn, either pile is empty then they lose in the standard version and win in the misére version. The amount they lose (or win) is dependent on the number of marbles left (2 for every red marble and 3 for every blue marble). So if on the computer player turn, it has 0 red marbles and 3 blue marbles, it loses 9 points in the standard version (or wins 9 points in the misére version).

Your program should be called red_blue_nim and the command line invocation should follow the following format:

red_blue_nim.py <num-red> <num-blue> <version> <first-player> <depth>

- <num-red> and <num-blue> are required. (Number of red and blue marbles respectively)
- <version> is either
    - standard - Player loses if either pile empty on their turn [default option if <version> is not given]
    - misere - Player wins if either pile empty on their turn
- <first-player> can be
    - computer - play a full game from given state with computer starting the game followed by human [default option if <first-player> is not given]
    - human - play a full game from given state with human starting the game followed by computer
- <depth> only used if depth limited search (Extra Credit) is implemented.

For a full game,

- On Computer turn, the program should use MinMax with Alpha Beta Pruning to determine the best move to make and perform the move.
    - Move ordering is pick blue marble followed by pick red marble for standard

        - Move ordering is pick red marble followed by pick blue marble for misére
    - On Human turn, the program should use a prompt to get the move from the human user and perform the move.

The program should alternate between these turns till the game ends (when the players run out of either red or blue marbles). Once the game ends, calculate who won and their final score and display it to the user.

Extra Credit (20 Points):

If your program determines computer move by using depth limited MinMax search with alpha beta pruning then you will be given 20 points extra credit. You will need to come up with a eval function to use with the program also. Please submit a text file describing the reasoning behind your eval function for full credit.

How to submit

Implementations in C, C++, Java, and Python will be accepted. Points will be taken off for failure to comply with this requirement unless previously cleared with the Instructor.

Create a ZIPPED directory called <net-id>_assmt2.zip (no other forms of compression accepted, contact the instructor or TA if you do not know how to produce .zip files).
The directory should contain the source code for the task (no need for any compiled binaries). Each folder should also contain a file called readme.txt, which should specify precisely:

- Name and UTA ID of the student.
- What programming language is used for this task. (Make sure to also give version and subversion numbers)
- How the code is structured.
- How to run the code, including very specific compilation instructions, if compilation is needed. Instructions such as "compile using g++" are NOT considered specific if the TA needs to do additional steps to get your code to run.
    - If your code will run on the ACS Omega (not required) make a note of it in the readme file.
- Insufficient or unclear instructions will be penalized.
- Code that the TA cannot run gets AT MOST 75% credit (depending on if the student is able to get it to run during a Demo session).