Recon Blind Multi-Chess

Important Dates

1) April 21st, 2019 @ 11:59 PM – Testing submission deadline of **my_agent.py** and all associated files necessary for running your code. Note, this should be *plug and play* (i.e. no modifications should be necessary on our end to anything).
2) April 23rd, 2019 @ 9:30 AM – FINAL submission deadline and Round Robin tournament in class. Latest submission is used towards final grade.
3) April 29th, 2019 @ 11:20 AM – Final tournament during final period.

Project Description

Recon Blind Multi-Chess (RBMC) is a version of chess where each opponent cannot see the other's moves in real time. At the beginning of each turn, a player may "sense" a 3x3 section of the board to know the true state of that board for that moment in time, giving a glimpse of any opponent activity in the area. Each player will have five minutes on their clock to play an entire game of RBMC – totaling 10 minutes per game. For a complete comprehensive rule guide, see this link: https://reconchess.readthedocs.io/en/latest/rules.html

For this assignment, you will be developing an AI that plays RBMC using the information you have learned from this course. You may work in groups of two people and may utilize any resources online so long as they are credited in your submission.

Take note of the important dates above. The testing submission deadline is the final deadline to submit your code where we will "soft test" it on our final system and give you feedback within 24 hours. The FINAL submission deadline is the final deadline all code should be turned in by for competition. **Be warned that if your code does not run on this final deadline, it will receive zero credit.** It is in your best interest to submit by the testing deadline to ensure this does not happen.

What We Provide

You will receive a package containing several files (short descriptions listed below). These files simulate the environment we will be running during the main competition. Download the assignment zip file from Canvas and unzip it in your workspace.

You will also need to install the python-chess library. You can install python-chess by using the pip command line:

```
pip install python-chess
```

Note: you might need to use 'pip3' instead of 'pip' to make sure it installs in your Python3 environment.

Once installed, to import the python-chess library to any additional file, write:

```
import chess
```

Files:

game.py
The game object class; methods included help with interacting with the game board. Do not modify.

human_agent.py
Agent that allows for user input in the console to play RBMC. Do not modify.

my_agent.py
Your assignment file containing methods that must be implemented. This is the file that you are submitting for this project.

- **NOTE: Before submitting, you must rename your my_agent.py file to your team members' Georgia Tech usernames, separated by an underscore, followed by '.py' as follows:**
    - my_agent.py → becomes → mghuy3_mjohnson435.py
- Fun Fact: Rename the class "MyAgent" to the name you wish your bot to appear as in the tournament.

play_game.py
Plays a game of RBMC between two agents passed in as arguments. This also saves a .TXT file to the "GameHistory" folder with a complete history of the match. Do not modify.

player.py
Helper class for implementing player agents. Do not modify.

random_agent.py
Agent that randomly selects moves; used as a dummy agent to play against. Do not modify.

Your Responsibilities

*****VERY IMPORTANT NOTE*****

This code was written using **Python 3.5**. For this reason, you are required to write your code in Python 3.5.x (any version 'x' should work). We will be running your code with a Python 3.5 interpreter. Any code that does not compile with this interpreter will be rejected. You can find how to add a Python 3.5 interpreter to your machine through simple online searches. Please contact the TAs if you have any issues doing this.

****

You must implement the methods in my_agent.py to play RBMC with an artificial intelligence agent of your own design. Of the provided files, **this is the only file that may be updated**. Any alteration to any other file will not be beneficial to you as the originals will be used during the competition to evaluate your agent.

Additionally, the method headers and return statement of the methods that my_agent.py come with also should not be altered as this will cause them to not be called properly nor return properly.

In order to run the game:

```
python play_game.py [path to WHITE player] [path to BLACK player]
```

*If you want to the option to play as any color as human player, make sure to pass in the path to the human player first.

If you would like to provide any other methods or python files to aid in your agent, you may do the following:

- Add additional methods or classes to my_agent.py that are called within **my_agent.py**.
- Add additional python files containing functions or classes that are imported and used in **my_agent.py**. See additional notes on this below.
- **NOTE: This last stipulation must be approved:** Import additional helpful libraries not already on the approved library list (a pinned post on Piazza) to my_agent.py for use. To do this, you must reply to the post detailing the provide the libraries you wish to use, how you plan to implement them, and why it is necessary you accomplish this with this library. **You must get approval to do this before the competition deadline and should be prepared for rejection of approval should that occur. We reserve the right to approve, reject, and limit the use of any library.**

For potential training purposes, you will have access to several functions and attributes within **game.py**, including the truth board. This will not be the case during competition so please take care that your agent not attempt to call these variables or methods directly from **game.py**, else it will fail the match. We are passing the information directly to you through the handle methods, but if you find that there are other useful information for the decision making process, please email the TAs with what additional information you would like provided and why. **We reserve the right to approve, reject, or limit the information requested.**

Naming Conventions for Additional Files:

For each additional python file you wish to use in your game, please do the following:

1) Make sure your **my_agent.py** file has been renamed to your group members Georgia Tech usernames, separated by an underscore.
2) Using the same order of names, name each additional file as

   member1_member2_filename.py

   where "*filename*" is whatever identifying characteristic you wish the file to have.

   Ex:

   mghuy3_mjohnson435_neural_net.py

Your Submission

Submit all your python files to the dropbox on Canvas **by April 21st at 11:59 PM.** Of course, as mentioned above, this is the testing deadline and the dropbox really closes before the Round Robin tournament. However, it is better to think about it as being due on this date. The box will only accept python files with extension "**.py**" (so do not zip your files before uploading), but you can upload multiple python files. If Canvas happens to rename your upload, do not worry as when we download it, it will be named the way you submitted it.

Good luck, and please contact the TAs or Prof. Gombolay with any questions!