Project 1

Assignment Overview

In this assignment you will implement two games using the easyAI framework:
http://zulko.github.io/easyAI/

Project Specification

This is a group assignment.

Students are encouraged (but not required) to work in groups of **max 3 students**. No exceptions.

Ideally, the group should be organized around three main tasks / duties:

- Design of the solution ("architect" role)
- Coding of the solution ("developer" role)
- Documentation of the solution ("reporter" role)

You are required to indicate in your report "who did what" and document the entire process, from sketching the original plans and dividing up the tasks all the way to polishing the interface, testing the solution, and preparing the report.

The basic functionality for each part is specified below.

Part 1 – Setup (10%)

1. **Python installation and configuration**
   You can use either method, but **method (2) is preferred.**

   (1) Using pip (recommended method)
   i. Download python for your operating system (OS), see here[1]
   ii. Installed downloaded file to your OS.
   iii. Validate that you have python install make sure that you have python by running the following command below.
   `python --version`
   It will print current version of your python interpreter.
   iv. Install pip with python, see here[2]
   `python -m pip install --upgrade pip`
   v. Validate that you have pip install make sure that you have pip by running the following command below
   `python -m pip --version`

[1] https://www.python.org/downloads/
[2] https:// pip.pypa.io/en/stable/installation/

(2) Using Conda

a. Download conda for you operating system, see here[3]
b. Open terminal and check that conda is installed.
   Note: if you use Windows, you must search for "Anaconda Prompt"
c. Create conda environment create conda environment and install python version 3.8 (you can use any python version that is compatible with easyAI).
   `conda create -n py38 python=3.8`
d. Activate conda environment
   `conda activate py38`
e. Check that your conda environment is activated.
   There are many ways to check. One way is to run the following command.
   `conda info --envs`

2. **EasyAI installation**
   Follow instructions in the easyAI GitHub repo: https://github.com/Zulko/easyAI

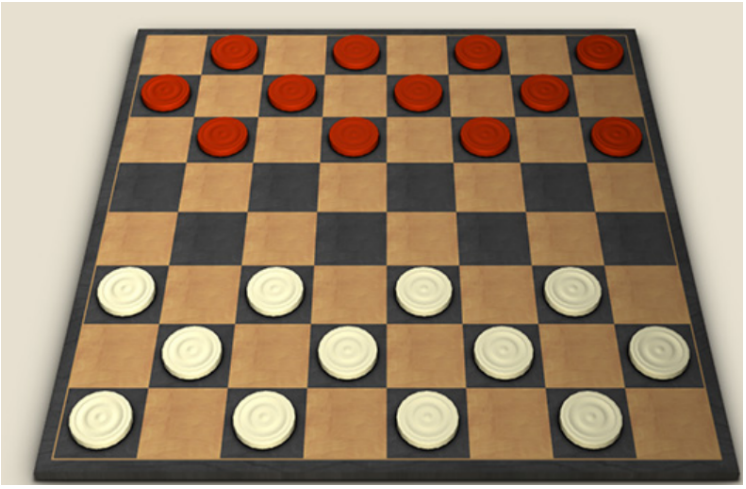
   `sudo python -m pip install easyAI`

Part 2: Run tic-tac-toe and connect-four example presented in easyAI documentation (40%)

Note that code in the easyAI documentation has typos. Github links for both games are pasted below
Tic-Tac-Toe and Connect Four: https://github.com/Awannaphasch2016/easy-ai-examples-fixed

Part3: Implement a modified version of the game of checkers (50%)

In this part you will implement a two-player game based on the game of Checkers. See references for implementation from the examples given in Easy AI, here.



[3] https://docs.conda.io/projects/conda/en/latest/user-guide/getting-started.html

Objective:

The objective of the game is to get as many pieces as you can from the opponent.

Material

An 8 x 8 board of checkers is used with two colors one for each opponent.

How to Win

The game can be won when the opponent is unable to make a move. This can be done in two ways: The entirety of a player's pieces was captured by the opponent (when the piece is a King)

Additional Rules

1. Only one jump is possible per move.
2. Forward move is only allowed for each opponent.
3. When a player jumps over another opponent's piece, the piece is not removed by the opponent.
4. There are two types of moves Step or Jump.
5. The pieces always move diagonally only on dark colored squares.
6. When a player's piece reaches the last row on the opponent's side of the board, they can use one of their captured pieces to crown the piece as a king.
7. The first player who has a piece promoted to king wins immediately.

Implement the following functions:

See starter code at
https://github.com/Awannaphasch2016/checker_easyAI/blob/main/checker_questions.py
and complete the code for the functions (12.5 pts each):

- `make_move()`
- `lose()`
- `is_over()`
- `scoring()`

Part 4: (BONUS): Post solution to Part 3 on GitHub (10% bonus)[4].

Note that there is a difference between Git and GitHub. Git is used for version control and code sharing and GitHub is used for hosting the source code in a centralized manner.

Git important Terminologies

i) Commit: commit command is used to save changes to your local repository
ii) Branch: A branch is a version repository that diverges from the main working project. The default branch name in Git is known as Master.
iii) Origin: Origin is referred to the address of the remote repository.

Below are the step-by-step instructions (commands) to set up a Git repository for the BONUS portion of the assignment.

1. Initialize the project folder as Git Repository
   git init

2. Add Git remote branch
   git remote add origin <your-git-repo-url>

3. Create branch called 'Main'
   git branch -M main

4. Push from the current local branch (Main) to remote branch (origin)
   git push -u origin main

To learn more about Git and GitHub, please go through the references given below.

1. https://learngitbranching.js.org/
2. https://ohmygit.org/
3. https://www.git-tower.com/learn/git/faq/git-create-repository

[4] A common mistake students make is to attempt bonus items ***before*** producing a "perfect 100" baseline solution. **Make no mistake:** if your baseline solution doesn't meet **all** grading criteria, you are ***not*** eligible for bonus points. Period.

Deliverables

You must submit (via Canvas):

- **One** file **p1_FAUusername.py** (where "**FAUusername**" is the username of one of the team members); in my case the file would be called **p1_omarques.py**) that allows me to call the code for either solution.
    - This is your source code solution; be sure to include your names, date, assignment number and comments describing your code.
    - Only one submission per team is enough, if the names of the team members appear in the comments and report.
- (Optional) A **README.md file** with installation instructions, dependencies, etc.
- A **detailed report** (PDF, markdown, and/or HTML) with detailed "project notes" (describing what my TA and I cannot see by looking at your source code and/or running your program), screenshots, references, etc.
    - Examples: design decisions, documented limitations, future improvements, etc.

Notes and Hints:

- Start by breaking the problems down into parts and solve smaller problems before producing the final solution.
- Try to handle special cases and prevent runtime errors to the best of your knowledge.
- **Don't overdo it!** Try not to risk breaking a good solution by adding bells and whistles to it.