2048 Pseudo-AI

**Project Description**

The purpose of this project is to create an open-sourced educational game of 2048 that introduces the concept of algorithm-AI relationship with customizable algorithmic parameters written in the Python language. By exploring the optimal strategies for solving the popular game 2048, this game will offer a complete playable game, on top of which users are able to learn how the expectimax AI algorithm works and design their own parameters to better the algorithm.

Players will be given guidelines on how the top-down design of the AI algorithm is implemented, alter the totally customizable parameters of each helper function of the algorithm, and finally, test the efficiency of their strategies by running the algorithm many times and obtain the probability of highest scores. At the later stages of the game, users may even be able to change the fundamental design of the game—for example, to have 3s and 5s, or to have multiple boards simultaneously—in order to learn the general purpose of this AI algorithm in solving complex problems.

**Competitive Analysis**

Several similar projects exist online and on GitHub, most of which are open-sourced and written in either JavaScript or C++. This project, however, uses a different language (Python) to design and solve the AI algorithm because Python is beginner-friendly for other learners to improve my source code. The approach will also differ, since most online projects do not simulate a board but rather use an online web game and register only key strokes to solve the game. This project will solve the algorithm by iteratively constructing the board and precisely monitor the complexity of the game state.

Moreover, most open-sourced 2048 projects on GitHub have limited readability, meaning their algorithmic design is unclear and accessibility for beginners is slow. To avoid this “black box” of AI algorithm, this project will aim to create a user-friendly UI with customizable parameters so even novice programmers will be able to be inspired and re-create the codes by themselves. Furthermore, this game will add levels to change the fundamental design of the game—for example, to have 3s and 5s, or to have multiple boards simultaneously—in order to encourage algorithmic thinking.

**Structural Plan**

The final project will consist of multiple files:

1. A main \_\_init\_\_.py file for the basic, functional game constructed in tkinter. This file will also include guidelines on how to customize both the game and the algorithm.
2. An ai.py file to solve the basic game of 2048.
3. An advancedMain.py file to make the game entirely customizable, including features like different numbers of base tiles and multiple boards.
4. An advancedAI.py file to solve the advanced game mechanisms.

**Algorithmic Plan**

The hardest parts of making the game include merging, making the board entirely customizable, and introducing a general-purpose AI algorithm.

Merging the board will require a top-down approach to solve how the game works fundamentally. Similarly, making the board customizable also requires to means to use OOPy animation, clear design instructions, and maybe multi-threading to accelerate the testing process.

The general-purpose AI algorithm is determined to be expectimax (a variant of expectiminimax, or minimax.) By inventing a scoring scheme, each game state of the board will be assigned a score. The algorithm will recursively find the single branch that scores highest in all possible moves within a certain number of depths.

**Timeline Plan**

**Version Control Plan**

The entire project will be open-sourced and constantly backed up on GitHub.

URL: <https://github.com/philxhuang/AI2048>

**Module List**