PACMAN Project
AI Course, Fall 2020
Due Date: November 12, 2020.

In this project, you will implement adversarial search using alpha-beta pruning on top of a PACMan game. Specifically, you will use an implementation of Pac-Man projects developed at UCBerkely for the introductory AI course (link: http://ai.berkeley.edu/project_overview.html)

To Download
1. Download python (please download python 2.x and NOT python version 3.x since the project does not work with 3.x) (https://www.python.org/download/releases/2.7.2/)
2. Download the Pac-man projects folder from ecourseware

To Run
1. To run a simple reflex agent implemented in multiagents.py,
a. python pacman.py -p ReflexAgent
2. You can choose a different layout using
a. python pacman.py -p ReflexAgent –l testClassic
3. You can choose the number of ghosts in pacman using the parameter k
a. python pacman.py -p ReflexAgent –l testClassic –k 1

Tasks
1. Implement alpha-beta pruning in the class provided in multiagents.py. Assume that there is only 1 ghost. Therefore, it becomes a two player game with pacman as the max node and the ghost as the min node. Use the existing evaluation function. Here depth will refer to the number of plys. For example, depth of 2 means Pacman and ghost each make two moves before returning the score.
a. python pacman.py -p AlphaBetaAgent -a depth=2 -l smallClassic –k 1
2. Score the leaves of your tree with the supplied self.evaluationFunction, which defaults to scoreEvaluationFunction. AlphaBetaAgent extends MultiAgentSearchAgent, which gives access to self.depth and self.evaluationFunction. Make sure your code makes reference to these two variables where appropriate as these variables are populated in response to command line options.
3. Test your code against at least five layouts specified in the layouts/ folder. Experiment with depths 1,2 and 3, and report in each case, the pacman score and whether pacman won or lost.
4. Implement a new evaluation function under betterEvaluationFunction in multiagents.py. Example of a new evaluation function might be reciprocal of distance to the closest food. Re-run your experiments. Did the results improve? (Can you think of anything else which works even better?). Describe the evaluation function you implemented and the experimental results.

Important Files in the Pacman project folder
multiAgents.py Where the alpha-beta code is to be implemented
pacman.py
The main file that runs Pacman games. This file also describes a Pacman GameState type, which you will use extensively in this project
game.py
The logic behind how the Pacman world works. This file describes several supporting types like AgentState, Agent, Direction, and Grid.
util.py Useful data structures for implementing search algorithms.

To Submit
Source code (just the modified files in the project) and a brief report providing the experimental results.

Reference
The above is an abridged version of the description in http://ai.berkeley.edu/multiagent.html.

Please read the whole description since it will be useful in your implementation.

Plagiarism Notice

Please DO NOT take copy code from the internet or from others in the class. **You can work in groups of 2 for this project (if you can collaborate virtually).**