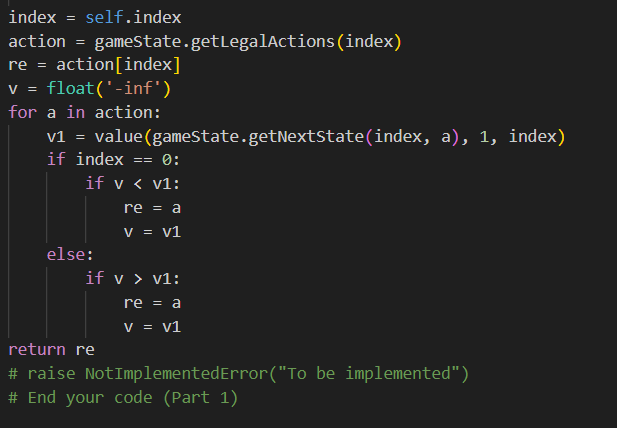
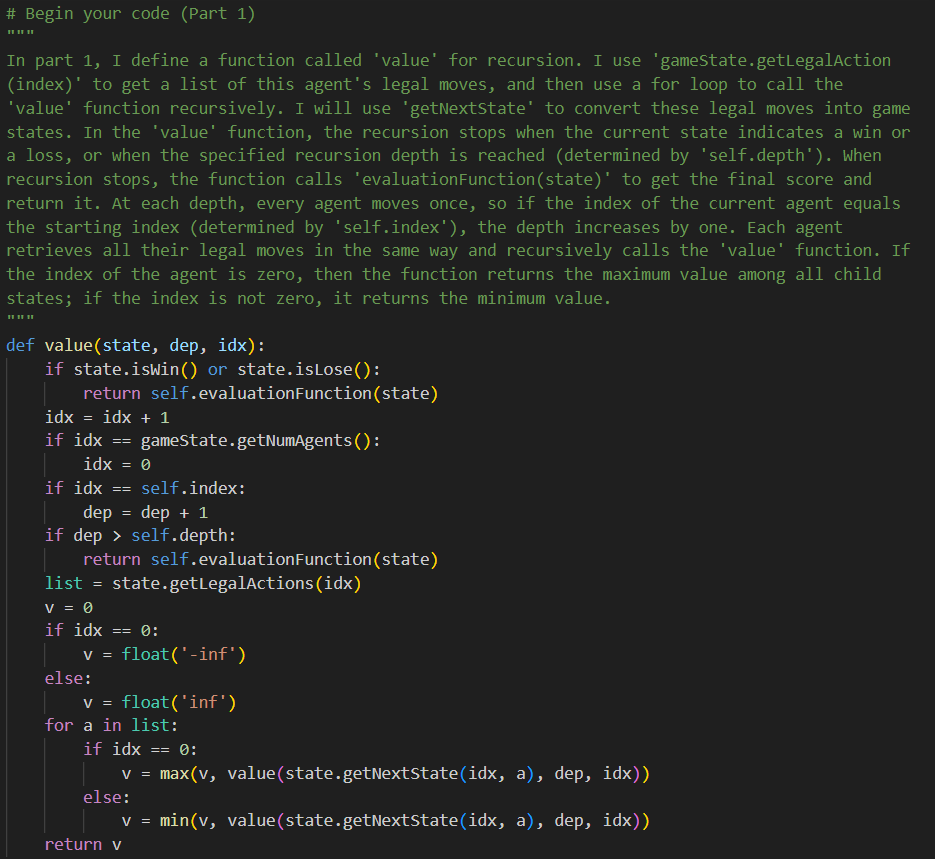
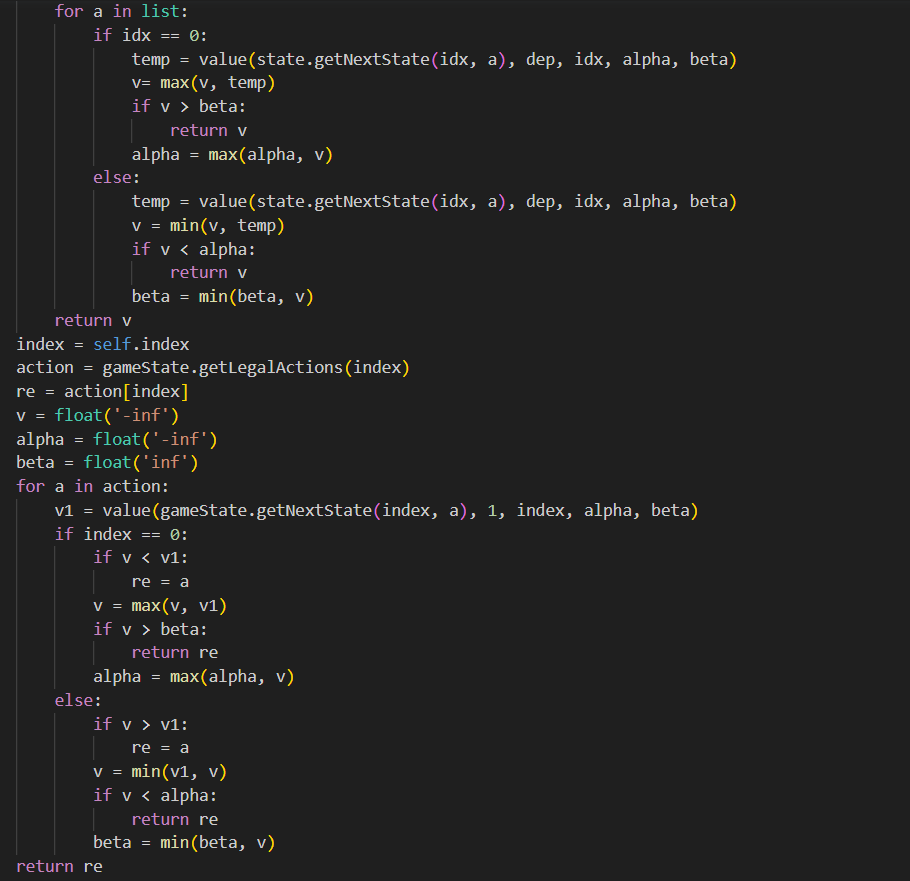
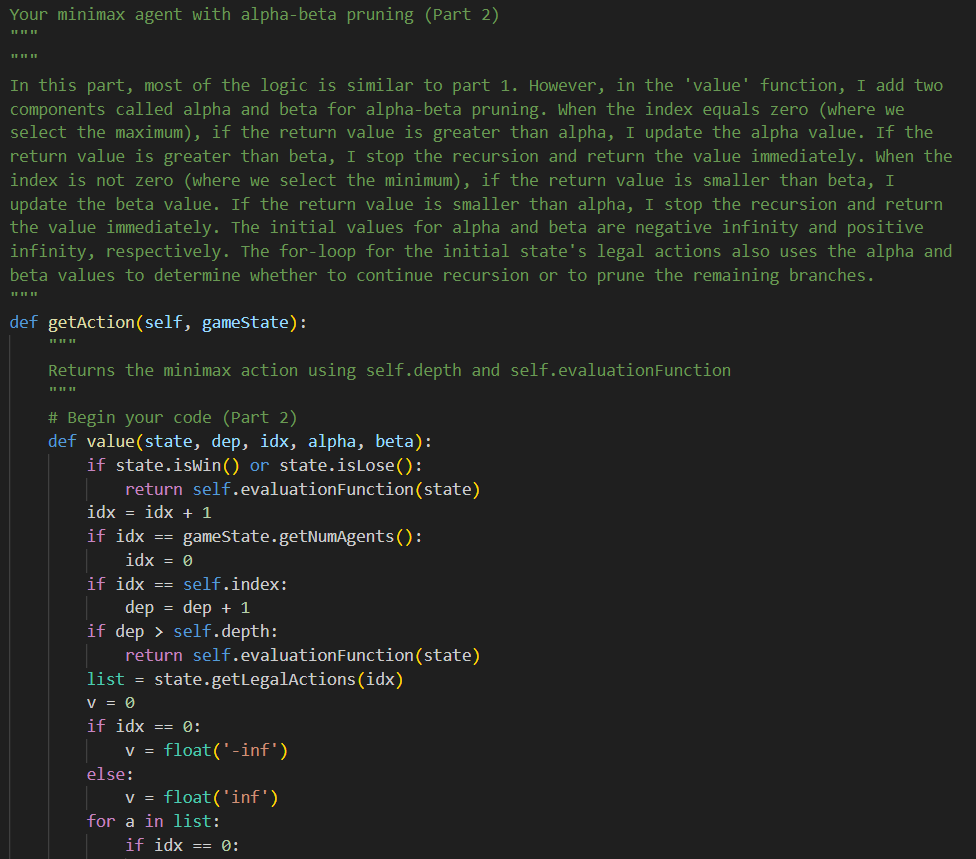
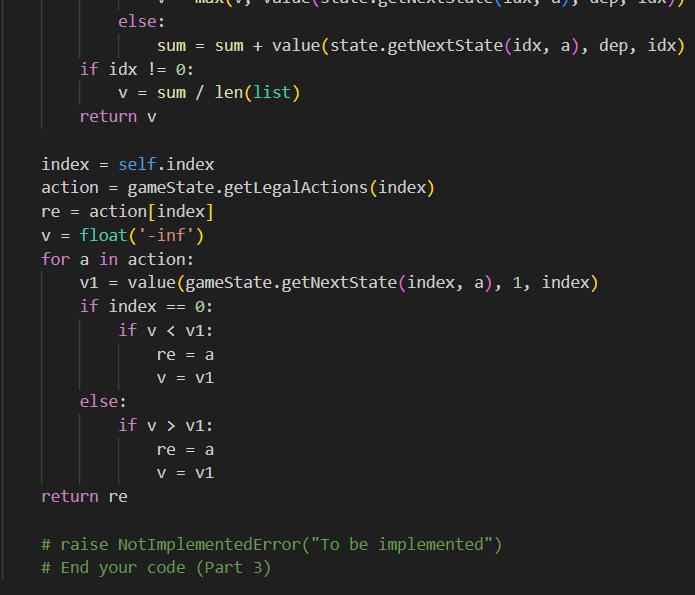
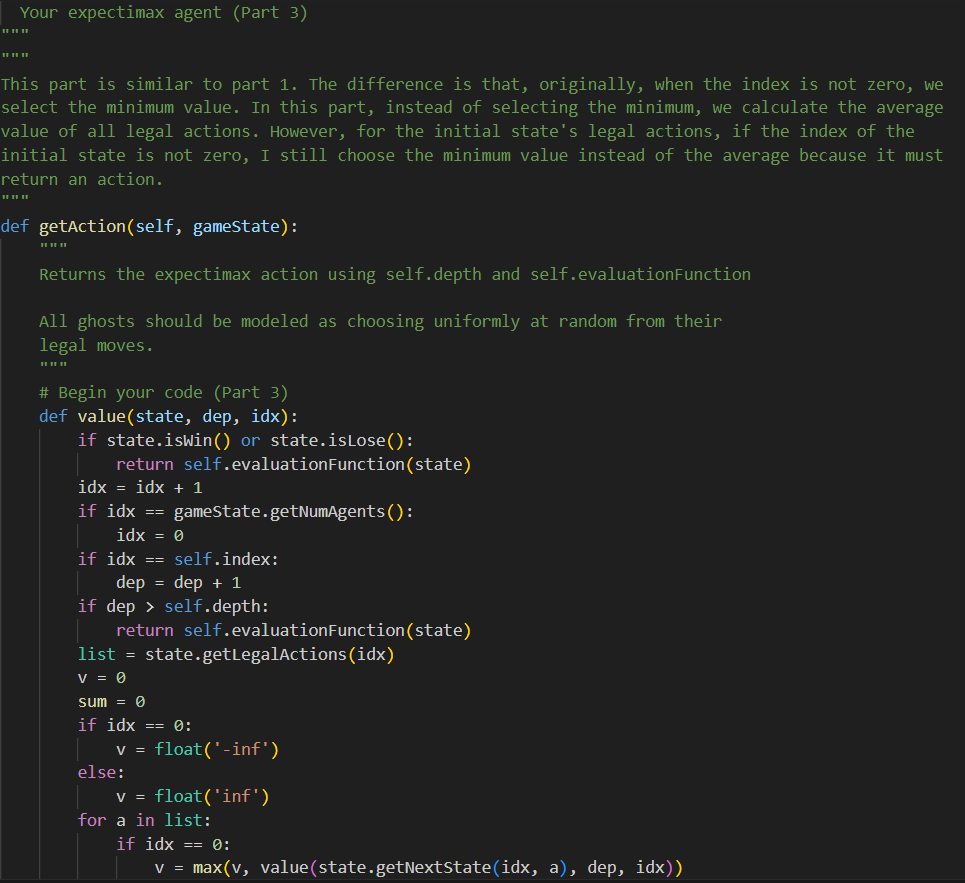
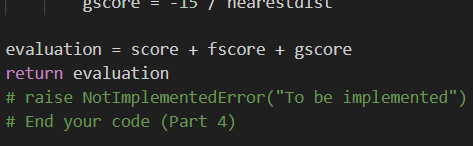
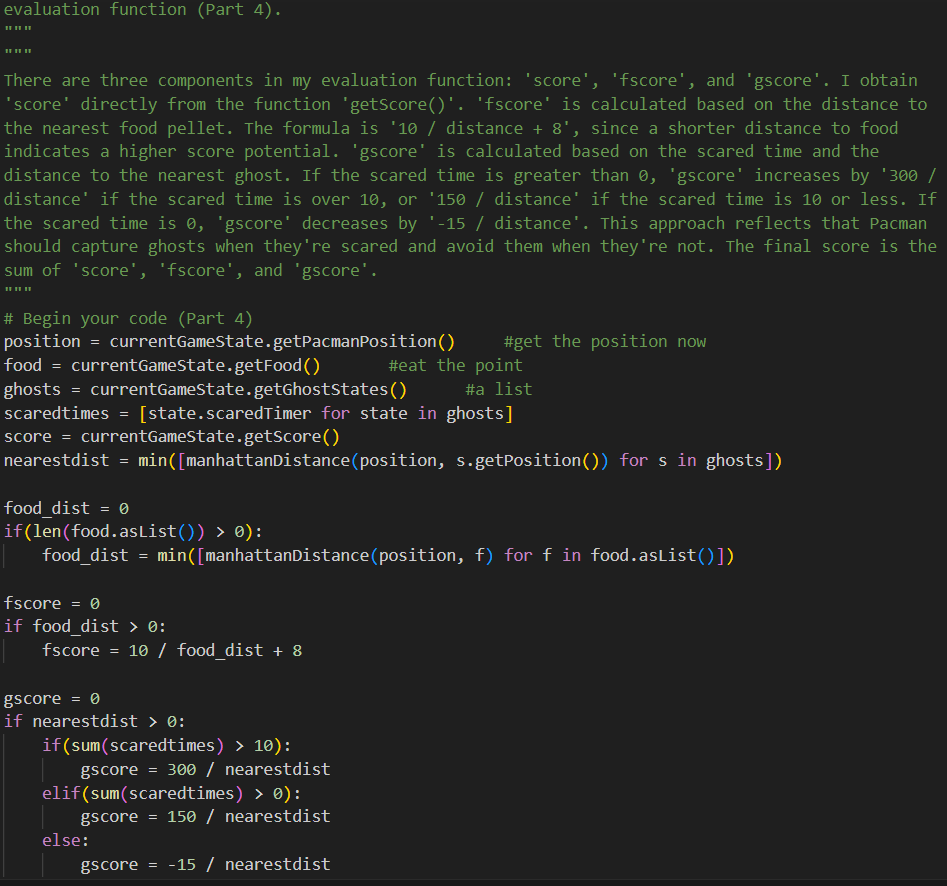
Homework 3: Multi-Agent Search

111550088 張育維

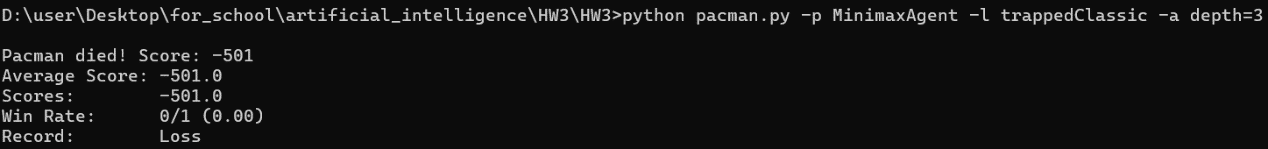
Part I. Implementation (20%):



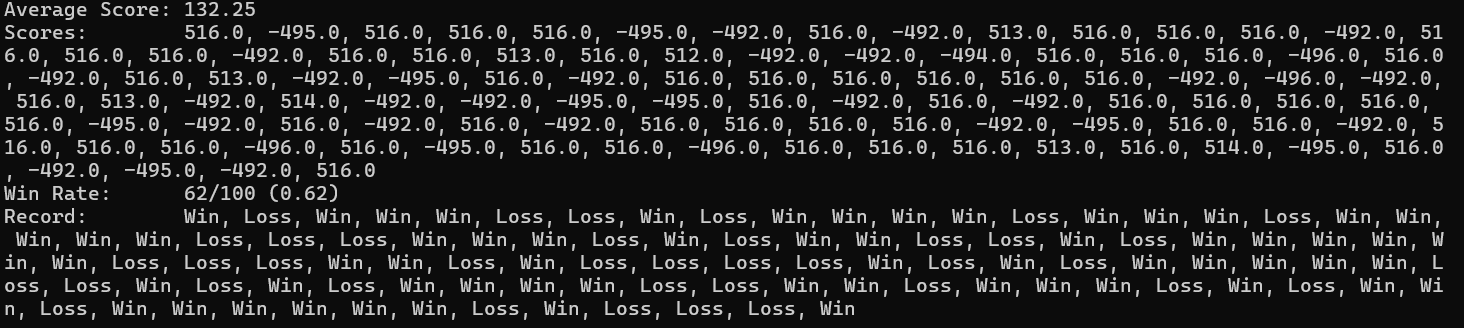




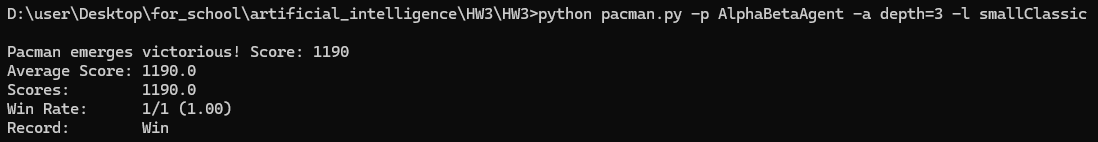
Part II. Results & Analysis (10%):



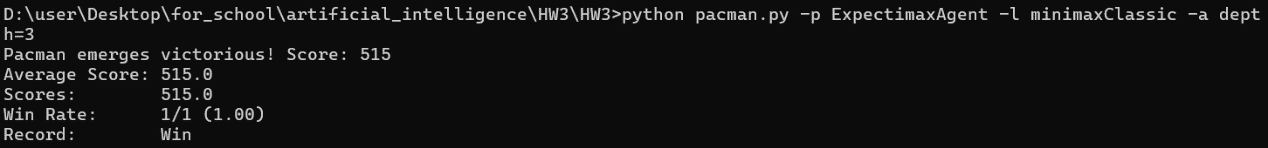
Why the pacman will rush to the ghost is that it can’t escape from these two ghosts, so it die earlier to make the rest score higher.



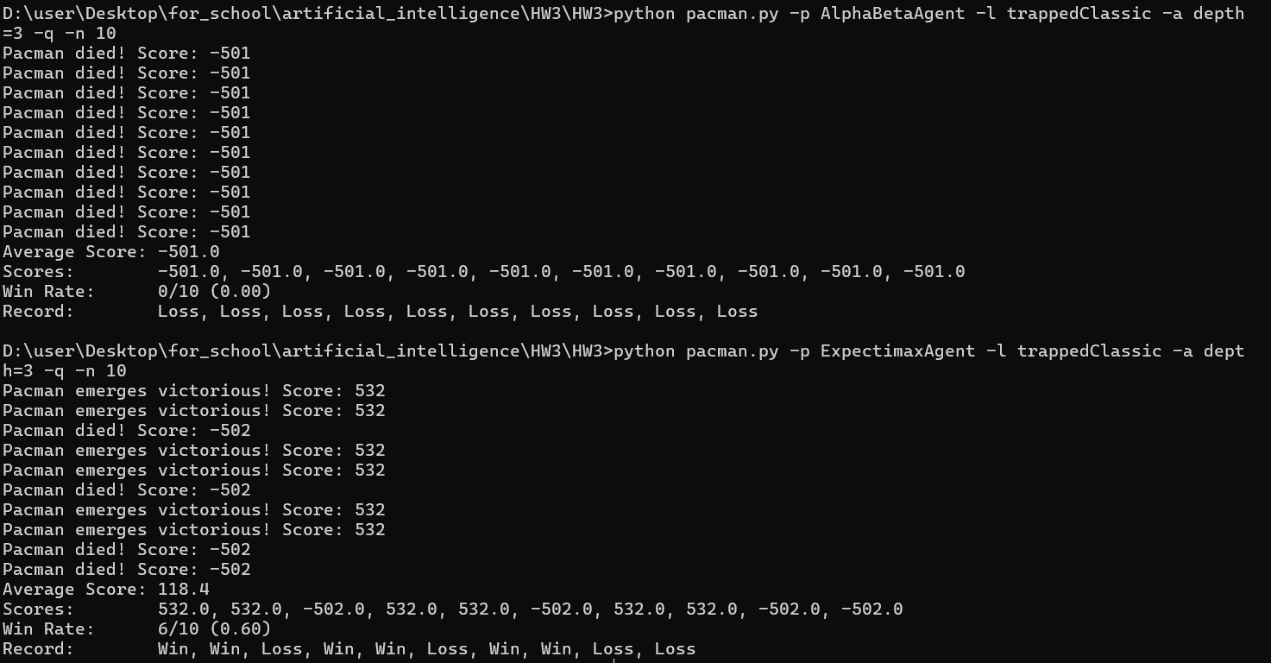
I test the command “python pacman.py -p MinimaxAgent -l minimaxClassic -a depth=4”. The win rate is 62%, fit the question request.



Run in ‘python pacman.py -p AlphaBetaAgent -a depth=3 -l smallClassic’



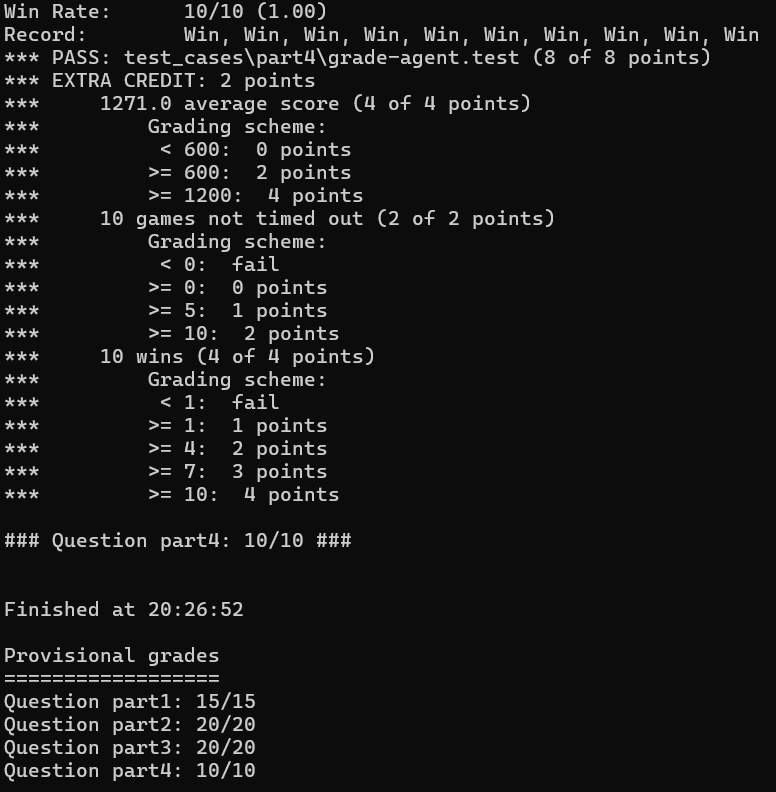
Run in ‘python pacman.py -p ExpectimaxAgent -l minimaxClassic -a depth=3’



In the alpha-beta agent, the win rate is 0%, while in the expectimax agent, the win rate is 60%. The reason the alpha-beta agent loses every time is that its behavior is similar to a minimax agent—it rushes toward the nearest ghost to minimize the potential loss in score. This leads to frequent collisions with the ghosts, resulting in losing the game.

On the other hand, the expectimax agent has a 60% win rate due to its use of average value calculations, which means that it doesn't always choose the most conservative path. Instead, it considers the probabilistic outcomes of each move. This approach can lead to scenarios where the farthest ghost moves even farther away from Pacman. In those instances, Pacman can collect all the pellets before the ghost changes direction, resulting in a win.

The critical difference lies in how these agents make decisions: the alpha-beta agent leans toward the worst-case scenario, often leading to aggressive ghost-avoiding strategies, while the expectimax agent's approach allows for more nuanced outcomes, sometimes favoring riskier paths that can lead to successful pellet collection.



I test all by autograder.py at the same time, and all of them pass the test.