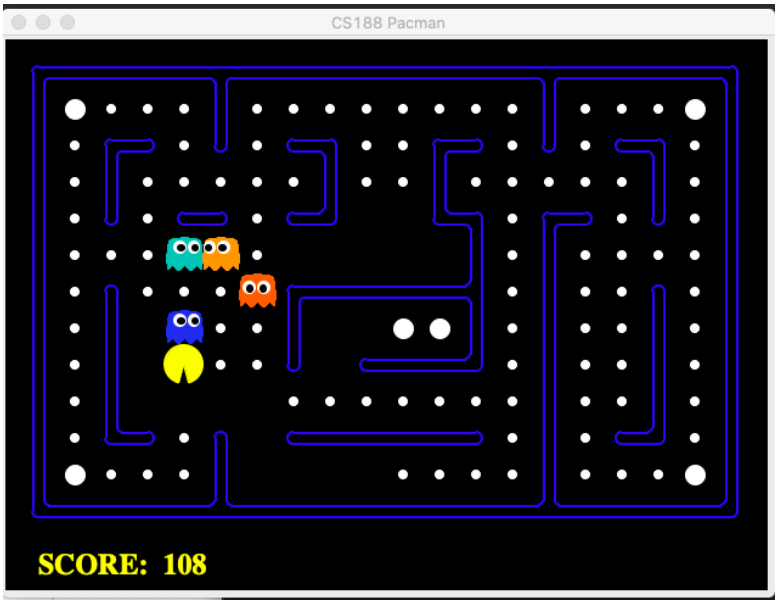
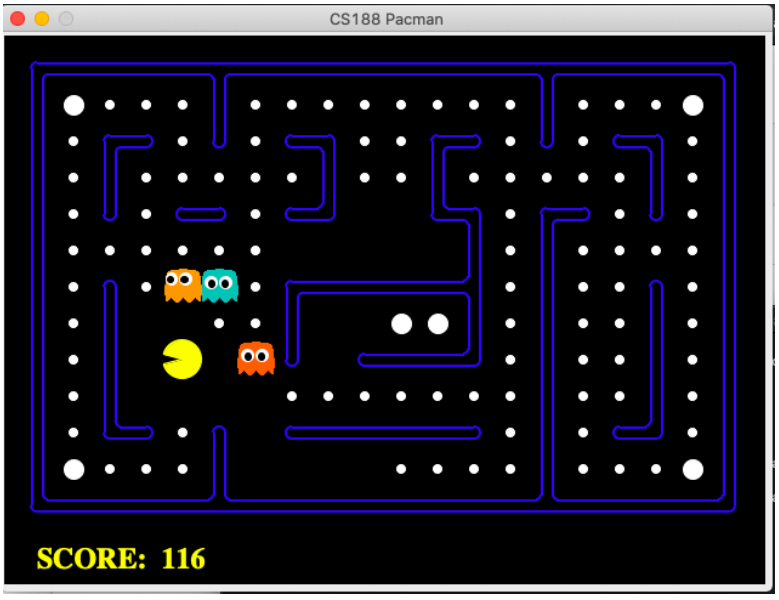
1)

While I was writing this evaluation function, I used the number of remaining capsules in the game, the distance of the closest capsule, the number of remaining foods in the game, the distance to the closest food and the closest ghost distance. I used these features because these fetuses are effective on decision making. I think using the reciprocals or negatives of some values is a good idea. If we give an example to this, ghosts are the best example. Because if the ghost is in the scared time, we have to chase them. In other words, as the distance decreases, the evaluation score increases, but if the ghost is not scared, we try to be as far away from it as possible. Another example is capsules. If the ghosts are frightened, it would be unreasonable to use the next capsule and head towards it. Instead, pacman must chase frightened ghosts.

2)



MinimaxAgent



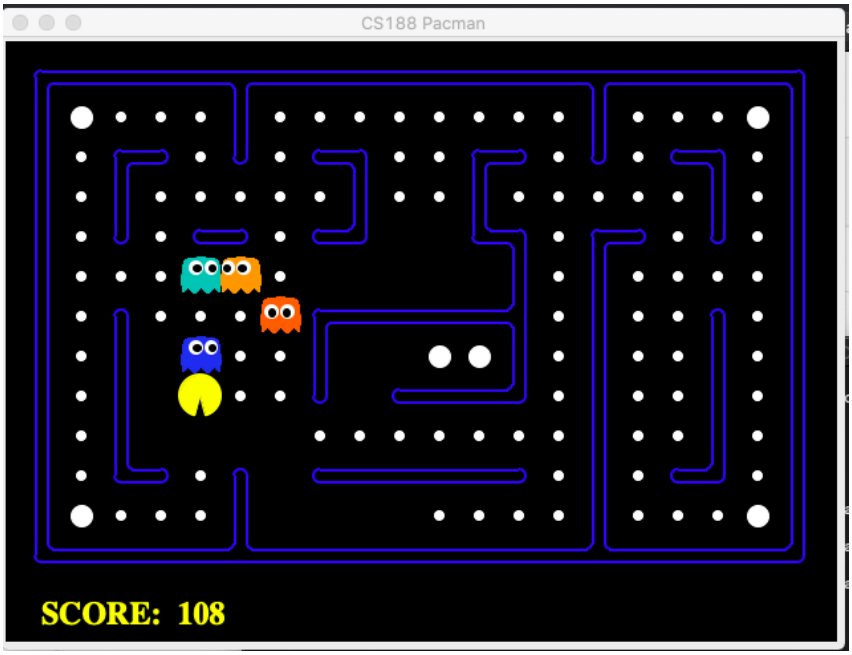
AlphaBetaAgent

When we compare these two Agents, we can see that in AlphaBetaAgent, pacman makes faster decisions and acts faster. AlphaBetaAgent prunes some of the nodes in the minimax tree while calculating the next action. In this case, AlphaBetaAgent time complexity is less than MinimaxAgent.

3)

In both cases, pacman made the same decisions. In fact, both agents work the same as an algorithm. Therefore, both achieve the same optimal result. However, AlphaBetaAgent makes a faster decision because it prunes some unnecessary nodes. As in the Lecture notes, this pruning has no effect on minimax value computed for the root. However, values of intermediate nodes might be wrong.

4)



ExpectimaxAgent

ExpectimaxAgent has come to the same level as MinimaxAgent. It has the same time complexity as MinimaxAgent in operation. You cannot prune chance nodes. Since there is no pruning, it has to expand all nodes.

5)

I thought the evaluation function I did in question 1 worked well enough. That's why I decided to take advantage of it. I used an evaluation function very similar to the one I used for the first evaluation function. This time I got CurrentGameState because we couldn't get SuccessorGameState. I couldn't handle the Stop condition because there are no Action preferences. I got the feature weights in the same 1 question in the same way. I just changed the weight of the scared ghost chase feature (distance to the closest ghost) in the first function.

6)

While doing the Evaluation function, I tried to make Pacman turn to preferences that would increase his score and not get caught by ghosts while doing these. That's why I paid attention to these values.

First, my goal is to finish the remaining dishes and capsules on the screen. That's why I've lowered the score for any food and capsules left on the screen. Our score increases as we go to the nearest food on the screen. But the point I should pay attention to is that eating a food should be more valuable than stopping at the closest point to that food. Also, if ghosts are too close, we should skip that meal. On the other hand, if the ghosts aren't scared, pacman is looking for a capsule. Because it's pointless to go towards the capsule when the frightened ghosts are around. After eating the capsule, Pacman's first goal is to chase ghosts. Because we are trying to reach the best score. In addition, I gave minus the infinite score value to the stop probability in order not to be stuck while doing all these.