1.What is the difference in the definition of a heuristic value for a game state, and for a state in A* search? What properties make a heuristic in either situation 'good'? (3 points total)

A heuristic value for a game state varies based on opponents. In some cases, a maximum value is desired, in the other the minimum value is desired. In the Pacman example, we want to find the closest path to the food and the furthest path from the ghosts. In an A* search, our goal is always to get to the goal state in as few searched states as possible. In A*, a good heuristic finds the solution with a minimal number of expanded nodes. In a game state, a good heuristic prevents the opponent from winning.

2. Note that Pacman will have suicidal tendencies when playing in situations where death is imminent. Why do you think this is the case? Briefly explain in one or two sentences. (2 points total)

After evaluating all it's option, Pacman determines that the final state is a terminal isLose() state. In this case, it finds the shortest solution to this path which is why it may have suicidal tendencies.

3. Consider an evaluation function that returns the square of a state's true minimax value. Consider three cases:

(i) In planning a move (or strategy) for a certain state S, the agent does not hit any terminal state.

(ii) In planning a move (or strategy) for a certain state S, the agent only hits a terminal state.

(iii) In planning a move (or strategy) for a certain state S, the agent sometimes hits the terminal state.

Answer the following questions:

(a) Will the search result in the same strategy in cases (i) and (ii)? [Same/Not Same] (1 point)
Not same. Squaring a negative value can give varied results when it hits a terminal state.

(b) Will the search result in the same strategy in cases (i) and (iii)? [Same/Not Same] (1 point)
Same

(c) Will the search result in the same strategy in cases (ii) and (iii)? [Same/Not Same] (1 point)
Not same

4. You should notice a speed-up compared to your MinimaxAgent. Consider a game tree constructed for our Pacman game, where b is the branching factor and where depth is greater than d. Say a minimax agent (without alpha-beta pruning) has time to explore all game states up to and including those at level d. At level d, this agent will return estimated minimax values from the evaluation function.

(a) In the best case scenario, to what depth would alpha-beta be able to search in the same amount of time? (1 point total)

It would be able to search to two times the depth d. The time complexity for alpha – beta is O($b^{d/2}$) and the runtime for minimax is O(b^d).

(b) In the worst case scenario, to what depth would alpha-beta be able to search in the same amount of time? How might this compare with the minimax agent without alpha-beta pruning? (2 points total)

In the worst case scenario, none of the nodes get pruned and so the algorithm still has to search through all the nodes, like Minimax. In this situation, the depth of the alpha-pruning would be the same as the depth of the Minimax search in the same amount of time.

5.True or False: Consider a game tree where the root node is a max agent, and we perform a minimax search to terminals. Applying alpha-beta pruning to the same game tree may alter the minimax value of the root node. (1 point total)

False. Applying alpha-beta pruning speeds up the computation but essentially gives you the same result as a minimax search.