<u>Project 1</u>

Greg Rosich(u0917936), Atul Sharma (u1001513)

1.

1.2. We expected the exploration order to take the first path it encountered, whether it be the longest or the shortest, however, with the implementation we had, the path taken ended up exploring the deepest node first, which in some cases happened to be the shortest path. Not every node was explored each time.

1.3. DFS expands the nodes to the maximum extent that they can be expanded until it finds the goal node. The cost does not play a factor. The path found may be the cheapest, but that would be a coincidence.

2.

2.2. No because BFS just finds the shortest path. It tries to minimize the number of transitions instead of the cost (which is not considered).

3.

3.2. PriorityQueue

4.

4.2. We considered the aStar algorithm with the Manhattan Distance heuristic. 535 nodes were expanded and goal was found in 0.1 seconds with aStar. The UCS algorithm expanded 682 nodes and took longer with 0.2 seconds. This is because in this case, including the manhattan distance in prioritization of expansion helped us find the goal faster with aStar. UCS only had access to the transition costs and was forced to expand cheap transitions that may have been leading away from the goal.

5.

5.2. The spaces were represented as tuples with a combination of an x,y coordinate and boolean markers that would indicate whether the cell contained food (dots) or not. For corner detection, All we did was use the direction obtained from the action list specified, and we ignored any actions that may have caused pacman to walk into a wall. The commented out section contains some code hints for this problem.

6.

6.2. We used the Manhattan distance between our current position and the farthest particle of food to be found. In some cases, we tried iterating through just the corners and that also worked to an extent.

7.

7.2. We used the actual maze distance between the current position (or state) and the farthest particle of food in the food grid.

8.

8.2. It won't always find the shortest path because the closest dot will not always be the most optimal solution. It may be more beneficial to target a certain food particle that falls in the middle of a path rather than backtrack and return to food particles that may have been skipped.

Self Analysis

1. Maneuvering the API was the hardest part of the assignment for us. Finding the order of expansion was also slightly challenging as opposed to finding just the path from start to goal.
2. Implementing A-Star was extremely straightforward once we figured out BFS, then UCS.
3. The corner heuristic modeling problem helped us understand the idea behind heuristics better. Also contrasting and telling the difference between A-Star and UCS helped me understand how heuristics help make searching more efficient.
4. Problem 3.7 seemed a little repetitive, other than that everything was extremely helpful to understand course content.
5. The question about openMaze could have contained more detail on how to run and use the information that we get.