Tvrtko Zadro, NIA

State:

Everything working, but I had problem finding good heuristics for Q6 and Q7.
Heuristic for Q6 goes greedy and finds closest corner from current position and then looks from closest remaining corner from that corner, etc. and the heuristic is sum of those manhattan distances.
Using heuristic from Q6 in Q7 wasn't passing admissibility test so I tried using one of two:

1) Heuristic is equal to number of food remaining to collect
2) Recursive function that would return minimum of approximations. For every food point sum manhattan distance to it, minimum number of walls to be encountered on shortest path and recursive function repeated for that food node and rest of the food.

These two would return the same result. I didn't know how to improve them.

Q1:

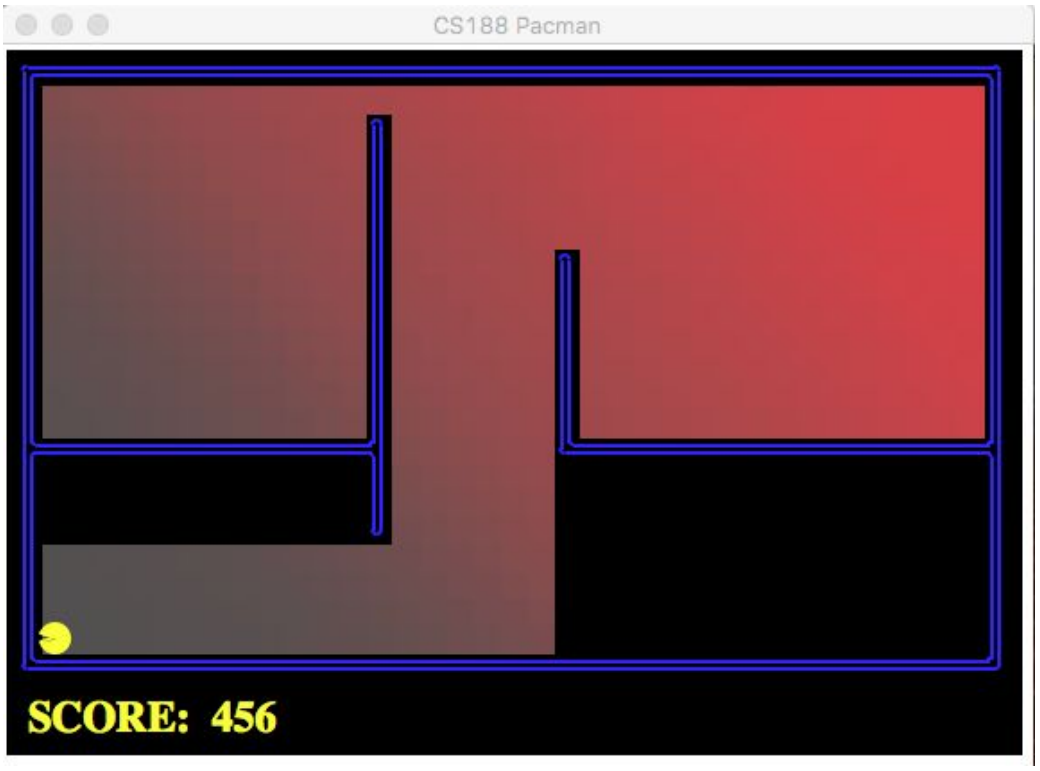
Exploration order was to be expected, algorithm kind of goes "all in" on one direction until it can. Pacman does not go through all explored states because at the end they were not the part of the optimal path. They were only considered when constructing optimal path.
Cost of 130 for *mediumMaze* is not least cost solution. Reason is because DFS goes all in on one direction he might find the solution before giving a chance to explore paths that would cost less.

Q2:

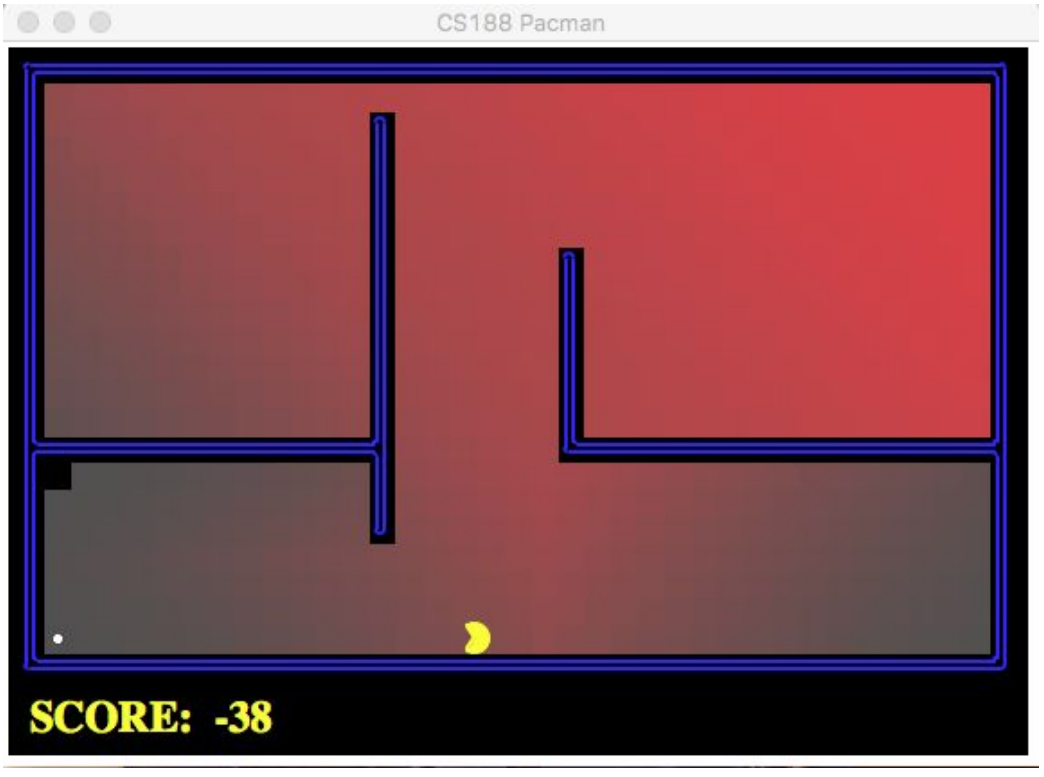
BFS finds least cost solution because he is slowly branching from the start. Because queue is FIFO priority based we well always explore nodes that came earlier, meaning we will prioritize nodes that are closer. That's why when we find goal state we could construct the least cost solution.

Q4:

For *openMaze* UCS and A* both find best solution, but UCS expands more nodes. My guess is that is because UCS in this case gives priority to nodes that are closer to the start, so it explores big part of the map that is closer to the start but further from the goal (*Picture 2.*). While UCS is guided by the heuristic and at the end reaches the goal by expanding less nodes (*Picture 1.*). In top left corner both were stuck because A*'s heuristic is manhattan distance, which does not take walls into account, so it couldn't know he was guiding search into a dead end.



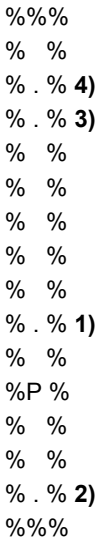
Picture 1. A* on *openMaze*



Picture 2. UCS on *openMaze*

Q8:

Example for where repeatedly going to the closest dot does not result in finding the shortest path:



The algorithm would first go to collect food slightly above him (marked with 1), then he would collect food on the bottom (2) and then he would have to go all the way up again for *3* and *4*. Optimally, he would first collect *2* and then go for *1*, *3* and *4*.