**Marks from the Auto Grader:**

**Q1: 3/3**

**Q2: 3/3**

**Q3: 3/3**

**Q4: 3/3**

**Q5: 3/3**

**Q6: 3/3**

**Q7: 3/4**

**Q8: 3/3**

**Total: 24/25**

**Explanations:**

**Q1:** For this question,since we are usingDFS, I had to use stack (a data structure that follow LIFO methodology). Each node, when visited is stored in ‘*visited*’ array.   
Through the inbuilt function *getStartState()* the first movement is recorded in the movements array where all the following, possible, movements will be stored. Now, until the goal state is retrieved a while loop is run where we go deeper and deeper. Each nodes child is retrieved using the inbuilt function *getSuccessors()* and are expanded to its deepest form.   
If, the node which is getting expanded is the goal state, evaluated using *isGoalState().*

**Q2:** For BFS, the same code as DFS was used except that a Queue was used in place of a Stack.  
Each node’s all children were expanded one by one. Each nodes child is retrieved using the inbuilt function *getSuccessors()* and are expanded level by level.   
If, the node which is getting expanded is the goal state, evaluated using *isGoalState().*

**Q3:** Uniform Cost Search: A priority queue was used. This data structure was used as this gives priority to each element, no matter the order of elements. This priority is set by calculating the cost which is exactly what we wanted. The inbuilt function *getCostofAction()* is used to get the cost of going form position ‘A’ to position ‘B’.

**Q4:** A-Star search problem: For this problem, I chose to use a Priority Queue as the nodes are getting visited based on their heuristic values. The priority queue named ‘Movements’ stores the starting node and the cost of it and then the nodes which are to be expanded in the future. All the visited nodes are stored in an array data structure called ‘visited’. A while loop is run till the goal node is not found (predefined function *isGoalState()* is used for this purpose). Further, if the *current\_state* is not in the visited array then that particular node is expanded. Now, for selecting the minimum heuristic values, a for loop is used and I have used, *getCostOfActions()* and *heuristic(),* and have added both so as to calculate the total heuristic value to travel that node.

**Q5:** The CornersProblem uses the BFS function (Q2) for solving its problem. The only code which was edited by me were the two functions in the *class CornersProblem,* which were: *getStartState()* and  *isGoalState().* For the isGoalState function, all we need to do is check if it has visited all the 4 corners, if yes, return true or else return False. To check if it has visited all 4 corners, we can check the length of visited array, if it is ==4, then TRUE.

**Q6:** CornersProblem heuristic, uses the A-Star method defined for Q4. It is similar to the previous question as the objective of this problem is to reach each and every corner. Except, in this method we consider the cost of each step. Hence, since even A-Star follows the same methodology, we use the same function. The only function which was to be edited was the cornerHeuristics function in the searchAgents file. In this function, using for loops, we make a list of corners that haven’t been visited yet. Further, we calculate the heuristic value for each and every corner in the unvisited array and return the corner with max heuristic value.

**Q7:** For this problem,no change in the code was required. It only relies on A-Star problem defined in question 3.

**Q8:** For this problem, only around 2 lines of code was required. First, as mentioned on the UCB website, we needed to write 1 line of code in the *AnyFoodSearchProblem* – *IsGoalState().* Here, we only had to return the food item present on the x,y location. Further, to find the path to the closest dot we use the bfs function and only 1 line of code was to be added which is ‘*return search.bfs(problem)*’.

**Auto Grader output:**

D:\Nishit\Master's\UCF\Notes\Semester 1\AAI\Pac-Man 1\search>python autograder.py

Starting on 9-25 at 17:27:19

Question q1

===========

\*\*\* PASS: test\_cases\q1\graph\_backtrack.test

\*\*\* solution: ['1:A->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'D', 'C']

\*\*\* PASS: test\_cases\q1\graph\_bfs\_vs\_dfs.test

\*\*\* solution: ['2:A->D', '0:D->G']

\*\*\* expanded\_states: ['A', 'D']

\*\*\* PASS: test\_cases\q1\graph\_infinite.test

\*\*\* solution: ['0:A->B', '1:B->C', '1:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C']

\*\*\* PASS: test\_cases\q1\graph\_manypaths.test

\*\*\* solution: ['2:A->B2', '0:B2->C', '0:C->D', '2:D->E2', '0:E2->F', '0:F->G']

\*\*\* expanded\_states: ['A', 'B2', 'C', 'D', 'E2', 'F']

\*\*\* PASS: test\_cases\q1\pacman\_1.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 130

\*\*\* nodes expanded: 146

### Question q1: 3/3 ###

Question q2

===========

\*\*\* PASS: test\_cases\q2\graph\_backtrack.test

\*\*\* solution: ['1:A->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C', 'D']

\*\*\* PASS: test\_cases\q2\graph\_bfs\_vs\_dfs.test

\*\*\* solution: ['1:A->G']

\*\*\* expanded\_states: ['A', 'B']

\*\*\* PASS: test\_cases\q2\graph\_infinite.test

\*\*\* solution: ['0:A->B', '1:B->C', '1:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C']

\*\*\* PASS: test\_cases\q2\graph\_manypaths.test

\*\*\* solution: ['1:A->C', '0:C->D', '1:D->F', '0:F->G']

\*\*\* expanded\_states: ['A', 'B1', 'C', 'B2', 'D', 'E1', 'F', 'E2']

\*\*\* PASS: test\_cases\q2\pacman\_1.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 68

\*\*\* nodes expanded: 269

### Question q2: 3/3 ###

Question q3

===========

\*\*\* PASS: test\_cases\q3\graph\_backtrack.test

\*\*\* solution: ['1:A->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C', 'D']

\*\*\* PASS: test\_cases\q3\graph\_bfs\_vs\_dfs.test

\*\*\* solution: ['1:A->G']

\*\*\* expanded\_states: ['A', 'B']

\*\*\* PASS: test\_cases\q3\graph\_infinite.test

\*\*\* solution: ['0:A->B', '1:B->C', '1:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C']

\*\*\* PASS: test\_cases\q3\graph\_manypaths.test

\*\*\* solution: ['1:A->C', '0:C->D', '1:D->F', '0:F->G']

\*\*\* expanded\_states: ['A', 'B1', 'C', 'B2', 'D', 'E1', 'F', 'E2']

\*\*\* PASS: test\_cases\q3\ucs\_0\_graph.test

\*\*\* solution: ['Right', 'Down', 'Down']

\*\*\* expanded\_states: ['A', 'B', 'D', 'C', 'G']

\*\*\* PASS: test\_cases\q3\ucs\_1\_problemC.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 68

\*\*\* nodes expanded: 269

\*\*\* PASS: test\_cases\q3\ucs\_2\_problemE.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 74

\*\*\* nodes expanded: 260

\*\*\* PASS: test\_cases\q3\ucs\_3\_problemW.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 152

\*\*\* nodes expanded: 173

\*\*\* PASS: test\_cases\q3\ucs\_4\_testSearch.test

\*\*\* pacman layout: testSearch

\*\*\* solution length: 7

\*\*\* nodes expanded: 14

\*\*\* PASS: test\_cases\q3\ucs\_5\_goalAtDequeue.test

\*\*\* solution: ['1:A->B', '0:B->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C']

### Question q3: 3/3 ###

Question q4

===========

\*\*\* PASS: test\_cases\q4\astar\_0.test

\*\*\* solution: ['Right', 'Down', 'Down']

\*\*\* expanded\_states: ['A', 'B', 'D', 'C', 'G']

\*\*\* PASS: test\_cases\q4\astar\_1\_graph\_heuristic.test

\*\*\* solution: ['0', '0', '2']

\*\*\* expanded\_states: ['S', 'A', 'D', 'C']

\*\*\* PASS: test\_cases\q4\astar\_2\_manhattan.test

\*\*\* pacman layout: mediumMaze

\*\*\* solution length: 68

\*\*\* nodes expanded: 221

\*\*\* PASS: test\_cases\q4\astar\_3\_goalAtDequeue.test

\*\*\* solution: ['1:A->B', '0:B->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C']

\*\*\* PASS: test\_cases\q4\graph\_backtrack.test

\*\*\* solution: ['1:A->C', '0:C->G']

\*\*\* expanded\_states: ['A', 'B', 'C', 'D']

\*\*\* PASS: test\_cases\q4\graph\_manypaths.test

\*\*\* solution: ['1:A->C', '0:C->D', '1:D->F', '0:F->G']

\*\*\* expanded\_states: ['A', 'B1', 'C', 'B2', 'D', 'E1', 'F', 'E2']

### Question q4: 3/3 ###

Question q5

===========

\*\*\* PASS: test\_cases\q5\corner\_tiny\_corner.test

\*\*\* pacman layout: tinyCorner

\*\*\* solution length: 28

### Question q5: 3/3 ###

Question q6

===========

\*\*\* PASS: heuristic value less than true cost at start state

\*\*\* PASS: heuristic value less than true cost at start state

\*\*\* PASS: heuristic value less than true cost at start state

path: ['North', 'East', 'East', 'East', 'East', 'North', 'North', 'West', 'West', 'West', 'West', 'North', 'North', 'North', 'North', 'North', 'North', 'North', 'North', 'West', 'West', 'West', 'West', 'South', 'South', 'East', 'East', 'East', 'East', 'South', 'South', 'South', 'South', 'South', 'South', 'West', 'West', 'South', 'South', 'South', 'West', 'West', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'East', 'East', 'East', 'East', 'East', 'East', 'South', 'South', 'East', 'East', 'East', 'East', 'East', 'North', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'East', 'East', 'South', 'South', 'South', 'South', 'East', 'East', 'North', 'North', 'East', 'East', 'South', 'South', 'South', 'South', 'South', 'North', 'North', 'North', 'North', 'North', 'North', 'North', 'West', 'West', 'North', 'North', 'East', 'East', 'North', 'North']

path length: 106

\*\*\* PASS: Heuristic resulted in expansion of 978 nodes

### Question q6: 3/3 ###

Question q7

===========

\*\*\* PASS: test\_cases\q7\food\_heuristic\_1.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_10.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_11.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_12.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_13.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_14.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_15.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_16.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_17.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_2.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_3.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_4.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_5.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_6.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_7.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_8.test

\*\*\* PASS: test\_cases\q7\food\_heuristic\_9.test

\*\*\* FAIL: test\_cases\q7\food\_heuristic\_grade\_tricky.test

\*\*\* expanded nodes: 9551

\*\*\* thresholds: [15000, 12000, 9000, 7000]

### Question q7: 3/4 ###

Question q8

===========

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_1.test

\*\*\* pacman layout: Test 1

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_10.test

\*\*\* pacman layout: Test 10

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_11.test

\*\*\* pacman layout: Test 11

\*\*\* solution length: 2

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_12.test

\*\*\* pacman layout: Test 12

\*\*\* solution length: 3

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_13.test

\*\*\* pacman layout: Test 13

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_2.test

\*\*\* pacman layout: Test 2

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_3.test

\*\*\* pacman layout: Test 3

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_4.test

\*\*\* pacman layout: Test 4

\*\*\* solution length: 3

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_5.test

\*\*\* pacman layout: Test 5

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_6.test

\*\*\* pacman layout: Test 6

\*\*\* solution length: 2

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_7.test

\*\*\* pacman layout: Test 7

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_8.test

\*\*\* pacman layout: Test 8

\*\*\* solution length: 1

[SearchAgent] using function depthFirstSearch

[SearchAgent] using problem type PositionSearchProblem

\*\*\* PASS: test\_cases\q8\closest\_dot\_9.test

\*\*\* pacman layout: Test 9

\*\*\* solution length: 1

### Question q8: 3/3 ###

Finished at 17:28:10

Provisional grades

==================

Question q1: 3/3

Question q2: 3/3

Question q3: 3/3

Question q4: 3/3

Question q5: 3/3

Question q6: 3/3

Question q7: 3/4

Question q8: 3/3

------------------

Total: 24/25

**GITLOG:**

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Mon Sep 24 16:18:17 2018 -0500

BFS Update

commit 1b837cd46b32d780d88387e4b7733f6a693d0f26

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Mon Sep 24 13:29:51 2018 -0500

BFS Update

commit e638f1ca7bc63e723b35f6d71e836a4b66cb9f92

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Sun Sep 16 19:33:19 2018 -0500

UCS and AsTar

commit 503be8410de101bec9776c55e9567505a749bf49

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Sun Sep 16 18:55:02 2018 -0500

DFS and BFS

commit a1d8907ac423dc7db88fc68812606eee97734fec

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Sun Sep 16 15:19:35 2018 -0500

Bigns

commit 1e6eeccac34cd3368e67d99fd2aff3e20f6fe210

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Sat Sep 15 14:17:51 2018 -0500

Update to python 2.7

commit e3fa79486c48ad252ca8e0e05545814f564b5b43

Author: nishitmehta1 <nishit.mehta1@gmail.com>

Date: Tue Sep 11 19:03:53 2018 -0500

Initial Commit