UNIVERSITÉ
BISHOP'S
UNIVERSITY

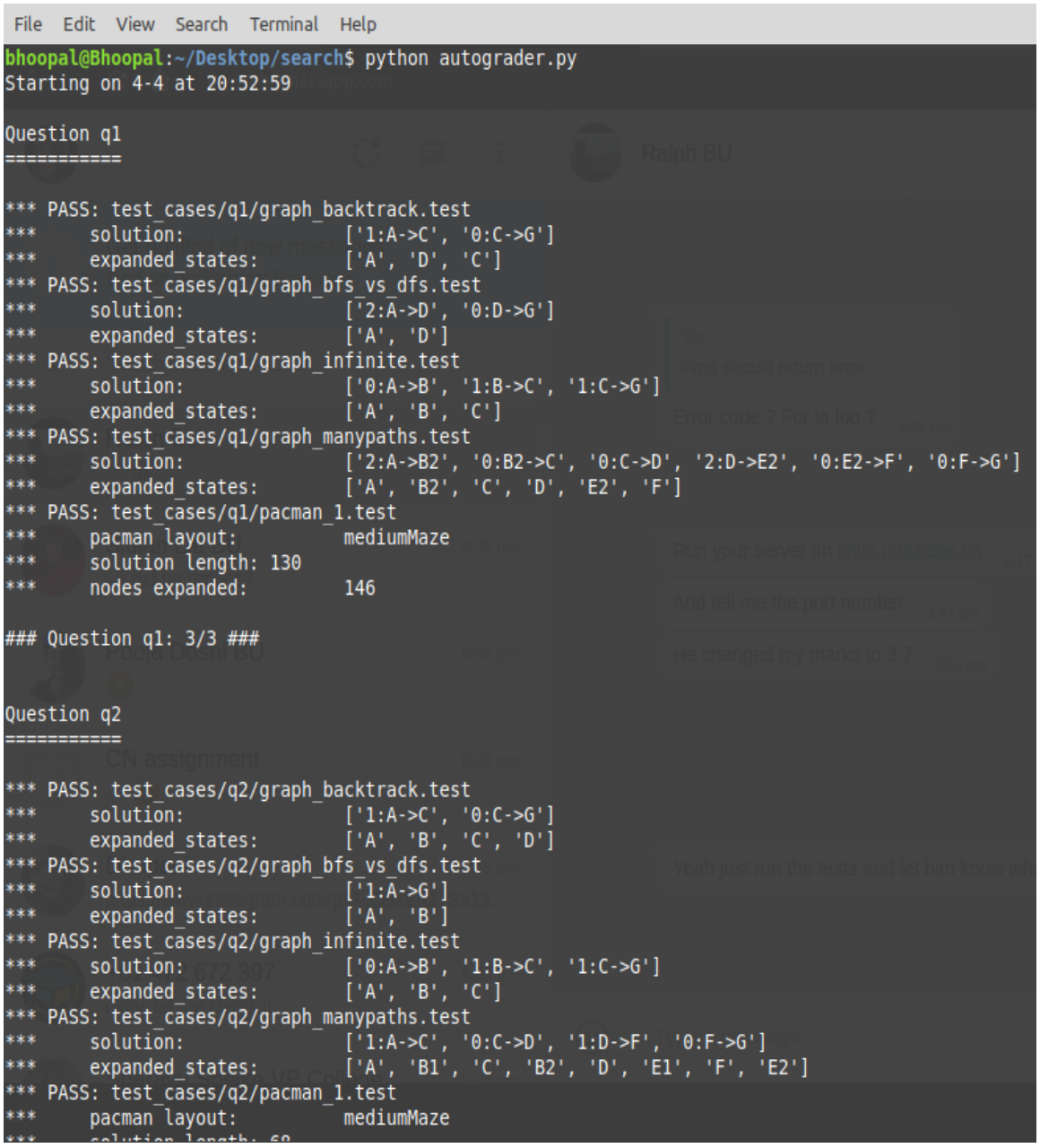
CS 520 – Advanced Topics in Software: Artificial Intelligence

Search strategies

I. Contribution of each member of your group

Full name	Contribution
Bhoopalsinh Musale (002269332)	Coding
Ahmad Hamdan (002232832)	Documentation
Ali Nourzad (002270328)	Writing pseudocode
Niloufar Karimifar (002270292)	Writing pseudocode

I. Autograder score



```
File  Edit  View  Search  Terminal  Help
bhoopal@Bhoopal:~/Desktop/search$ python autograder.py
Starting on 4-4 at 20:52:59

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
```

File Edit View Search Terminal Help

```
***     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
```

File Edit View Search Terminal Help

```
*** 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
```

File Edit View Search Terminal Help

```
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', 'No
 'West', 'West', 'West', 'West', 'South', 'South', 'East', 'East', 'East', 'East', 'South', 'South', 'South', 'South', 'South', 'South', 'West', 'West', 'South', 'S
, 'South', 'West', 'West', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'East', 'East', 'East', 'East', 'East', 'East', 'South', 'South', 'East', 'Eas
East', 'East', 'East', 'North', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'North', 'North', 'East', 'East', 'East', 'East', 'South', 'South', 'Sout
South', 'East', 'East', 'North', 'North', 'East', 'East', 'South', 'South', 'South', 'South', 'South', 'North', 'North', 'North', 'North', 'North', 'North', 'North'
st', 'West', 'North', 'North', 'East', 'East', 'North', 'North']
path length: 106
*** PASS: Heuristic resulted in expansion of 801 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
```

```
File   Edit   View   Search   Terminal   Help
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
*** PASS: test_cases/q7/food_heuristic_grade_tricky.test
***     expanded nodes: 4137
***     thresholds: [15000, 12000, 9000, 7000]

### Question q7: 5/4 ###


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
```

File Edit View Search Terminal Help

```
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
```

File Edit View Search Terminal Help

```
*** 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 20:53:39

Provisional grades
==================
Question q1: 3/3
Question q2: 3/3
```

File Edit View Search Terminal Help

```
[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 20:53:39

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: 5/4
Question q8: 3/3
------------------
Total: 26/25

Your grades are NOT yet registered.  To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.

bhoopal@Bhoopal:~/Desktop/search$
```

II. Answers to questions

1. Question 1

- **Depth-first search (DFS)**

It is a traversing algorithm for searching for a graph or tree. DFS algorithm starts at the root node and move as far as it can down a given branch, then backtracks till it finds an unexplored path, and then searches it. The algorithm does this until the whole graph has been traversed.

- **Pseudo code for DFS:**
  - DFS(G,v) (v is the vertex where the search starts)
  - Stack S := {}; (start with an empty stack)
  - for each vertex u, set visited[u] := false;
  - push S, v;
  - while (S is not empty) do
  - u := pop S;
  - if (not visited[u]) then
  - visited[u] := true;
  - for each unvisited neighbour w of u
  - push S, w;
  - end if
  - end while
  - END DFS()

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q1

Starting on 4-4 at 21:37:15

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

Finished at 21:37:15

Provisional grades

==========

Question q1: 3/3

Total: 3/3

Your grades are NOT yet registered. To register your grades, make sure

to follow your instructor's guidelines to receive credit on your project.

2. Question 2

- Breadth First Search (BFS)

BFS is a traversing algorithm that starts traversing from a selected node (source node) and traverses the graph level-wise thus exploring the neighbour nodes. Then it moves towards the next-level neighbour nodes.

- **Pseudo code for BFS:**
  - Input: s as the source node
  - BFS (G, s)
  - let Q be queue.
  - Q.enqueue(s)
  - mark s as visited
  - while (Q is not empty)
  - v = Q.dequeue( )
  - for all neighbors w of v in Graph G
  - if w is not visited
  - Q.enqueue(w)
  - mark w as visited.

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q2
Starting on 4-4 at 21:37:19

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

Finished at 21:37:19

Provisional grades

==========

Question q2: 3/3

Total: 3/3

Your grades are NOT yet registered. To register your grades, make sure to follow your instructor's guidelines to receive credit on your project.

3. Question 3

- **Uniform-cost search**

Uniform-cost search is a searching algorithm used for traversing a weighted tree or graph. This algorithm is needed when a different cost is assigned to every edge. The aim of this search algorithm is to find a path to the goal node which has the lowest cumulative cost. The uniform-cost algorithm expands nodes according to their path costs form the root node.

Pseudo code for Uniform-cost search :

- ufs(p) returns a answer
- if p's starting state is a goal then return empty path to initial state
- frontier = a priority queue ordered by pathCost, with a node for the initial state
- reached = a table of {state: the best path that reached state}; initially empty
- solution = failure
- while frontier is not empty and top(frontier) is cheaper than solution do
- parent = pop(frontier)
- for child in successors(parent) do
- s = child.state
- if s is not in reached or child is a cheaper path than reached[s] then
- reached[s] = child
- add child to the frontier
- if child is a goal and is cheaper than solution then
- solution = child
- return solution

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q3

Starting on 4-4 at 21:37:23

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

Finished at 21:37:23

Provisional grades

==========

Question q3: 3/3

Total: 3/3

Your grades are NOT yet registered. To register your grades, make sure to follow your instructor's guidelines to receive credit on your project.

4. Question 4

- **aStarSearch**

Similar Dijkstra, A* process by making the lowest-cost path from the start node to the destination node. A* different and better from the Dijkstra algorithm is by for each node, this algorithm have function f(n). Function f(n) returns an estimate of the total cost of a path using that node. Therefore, A* is a heuristic function, which differs from an algorithm in that a heuristic is more of an estimate and is not necessarily provably correct.

A* expands paths that are already less expensive by using this function:

$$\mathbf{f(n) = g(n) + h(n)}$$

where

- f(n)= total estimated cost of path through node n
- g(n) = cost so far to reach node n
- h(n) = estimated cost from n to goal. This is the heuristic part of the cost function, so it is like a guess.

Pseudo code for aStarSearch :

- Do
- for each s ∈ S
- Do
- g(s): =0
- End
- g(start): = 0;
- Open = Close: = ∅;
- Insert start into Open;
- expansions: = 0;
- While expansion < Lookahead
- Do
- expansions: = expansion+1;
- Delete a state s with the largest f-value (g(s)+h(s)) from Open;
- Close: =Close U {s};

- For each E A(s)
- Do
- g(succ(s,a)):= g(s) + constant; tree (succ (s, a));
- Ifsucc(s, a) is not in Open then insert into Open;
- End
- End
- End

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q4

Starting on 4-4 at 21:37:28

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

Finished at 21:37:28

Provisional grades

==========

Question q4: 3/3

Total: 3/3

Your grades are NOT yet registered. To register your grades, make sure to follow your instructor's guidelines to receive credit on your project.

5. Question 5

Pseudo code for CornersProblem:

- In __init__() I add goal state ((1,1), (1,top), (right, 1), (right, top)) and change startingGameState
- Define getStartState(self) which returns current startingPosition and goal
- In isGoalState() check whether this search state is a goal state of the problem.
- In getSuccessors(self, state) just added the given code
- getCostOfActions(self, actions) is same as given

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q5

Note: due to dependencies, the following tests will be run: q2 q5

Starting on 4-4 at 21:37:44

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 q5

==========

*** PASS: test_cases/q5/corner_tiny_corner.test

*** pacman layout: tinyCorner

*** solution length: 28

Question q5: 3/3

Finished at 21:37:44

Provisional grades

==========

Question q2: 3/3

Question q5: 3/3

Total: 6/6

Your grades are NOT yet registered. To register your grades, make sure to follow your instructor's guidelines to receive credit on your project.

6. Question 6

Pseudo code for cornersHeuristic:

- Check if current state is goal state if yes then return 0 otherwise move next
- Define a distance List which appends calculated maze distance with current state[0],problem, starttingGameState
- Finally return max value from distance list.

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q6
Note: due to dependencies, the following tests will be run: q4 q6
Starting on 4-4 at 21:37:48

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 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 801 nodes

Question q6: 3/3

Finished at 21:37:56

Provisional grades
==========
Question q4: 3/3
Question q6: 3/3

Total: 6/6

Your grades are NOT yet registered. To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.

7. Question 7

Pseudo code for food Heuristic:

- Check if problem.is Goal State if yes return 0 otherwise continue
- Define maze list and food list from foodGridas
- Run a for loop over food_list and append mazeDistance calculated based on position, pos, problem, problem.startingGameState
- Return max from maze_list

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q7
Note: due to dependencies, the following tests will be run: q4 q7
Starting on 4-4 at 21:38:01

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 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
*** PASS: test_cases/q7/food_heuristic_grade_tricky.test
*** expanded nodes: 4137
*** thresholds: [15000, 12000, 9000, 7000]

Question q7: 5/4

Finished at 21:38:35

Provisional grades
==========
Question q4: 3/3
Question q7: 5/4

Total: 8/7

Your grades are NOT yet registered. To register your grades, make sure to follow your instructor's guidelines to receive credit on your project.

8. Question 8

Pseudo code for findPathToClosestDot() of AnyFoodSearchProblem:

- Returns search.ucs(problem)

Pseudo code for isGoalState() of AnyFoodSearchProblem:

- Return self.food[x][y]

- **Output**

bhoopal@Bhoopal:~/Desktop/search$ python autograder.py -q q8

Starting on 4-4 at 21:38:44

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 21:38:44

Provisional grades

==========

Question q8: 3/3

Total: 3/3

Your grades are NOT yet registered. To register your grades, make sure

to follow your instructor's guidelines to receive credit on your project.