CS2180 : Artificial Intelligence
Name : Vaibhav B. Nagrale
Roll no. : 112001046

Question q1 DFS

"""
Explanation:
first check whether our start state is our goal state
then apply DFS to reach our goal state (food)
after getting goal state trace back from past vertex and store nodes in answer
then return this answer
"""

tinyMaze : Search nodes expanded: 15
mediumMaze : Search nodes expanded: 146
bigMaze : Search nodes expanded: 390

1. graph_backtrack.test
   *** solution: ['1:A->C', '0:C->G']
   *** expanded_states: ['A', 'D', 'C']
2. graph_bfs_vs_dfs.test
   *** solution: ['2:A->D', '0:D->G']
   *** expanded_states: ['A', 'D']
3. graph_infinite.test
   *** solution: ['0:A->B', '1:B->C', '1:C->G']
   *** expanded_states: ['A', 'B', 'C']
4. 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']
5. pacman_1.test
   *** pacman layout: mediumMaze
   *** solution length: 130
   *** nodes expanded: 146

Question q2 BFS

"""
Explanation:
first check whether our start state is our goal state
then apply BFS to reach our goal state (food)
after getting goal state trace back from past vertex and store nodes in answer
then return this answer
"""
mediumMaze : Search nodes expanded: 269
bigMaze : Search nodes expanded: 620
1. graph_backtrack.test
   *** solution: ['1:A->C', '0:C->G']
   *** expanded_states: ['A', 'B', 'C', 'D']
2. graph_bfs_vs_dfs.test
   *** solution: ['1:A->G']
   *** expanded_states: ['A', 'B']

3. graph_infinite.test
    *** solution: ['0:A->B', '1:B->C', '1:C->G']
    *** expanded_states: ['A', 'B', 'C']
4. 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']
5. pacman_1.test
    *** pacman layout: mediumMaze
    *** solution length: 68
    *** nodes expanded: 269

Question q3 UCS

"""
Explanation:
first check whether our start state is our goal state
then apply UCS to reach our goal state (food)
    calculate the cost
    if more cost then do nothing else push to queue and change the cost
after getting goal state trace back from past vertex and store nodes in answer
then return this answer
"""

mediumMaze : Search nodes expanded: 269
mediumDottedMaze : Search nodes expanded: 186
mediumScaryMaze : Search nodes expanded: 108

1. graph_backtrack.test
    *** solution: ['1:A->C', '0:C->G']
    *** expanded_states: ['A', 'B', 'C', 'D']

2. graph_bfs_vs_dfs.test
    *** solution: ['1:A->G']
    *** expanded_states: ['A', 'B']
3. graph_infinite.test
    *** solution: ['0:A->B', '1:B->C', '1:C->G']
    *** expanded_states: ['A', 'B', 'C']
4. 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']
5. ucs_0_graph.test
    *** solution: ['Right', 'Down', 'Down']
    *** expanded_states: ['A', 'B', 'D', 'C', 'G']
6. ucs_1_problemC.test
    *** pacman layout: mediumMaze
    *** solution length: 68
    *** nodes expanded: 269
7. ucs_2_problemE.test
    *** pacman layout: mediumMaze

*** solution length: 74
*** nodes expanded: 260

8. ucs_3_problemW.test
   *** pacman layout: mediumMaze
   *** solution length: 152
   *** nodes expanded: 173
9. ucs_4_testSearch.test
   *** pacman layout: testSearch
   *** solution length: 7
   *** nodes expanded: 14
10. ucs_5_goalAtDequeue.test
    *** solution: ['1:A->B', '0:B->C', '0:C->G']
    *** expanded_states: ['A', 'B', 'C']

Question q4 A* search

"""
Explanation:
same as UCS
first check whether our start state is our goal state
then apply A* search to reach our goal state (food)
calculate the f_n = cost + heuristic
if more f_n then do nothing else push to queue and change the cost
after getting goal state trace back from past vertex and store nodes in answer
then return this answer
"""

bigMaze : Search nodes expanded: 549

1. astar_0.test
   *** solution: ['Right', 'Down', 'Down']
   *** expanded_states: ['A', 'B', 'D', 'C', 'G']
2. astar_1_graph_heuristic.test
   *** solution: ['0', '0', '2']
   *** expanded_states: ['S', 'A', 'D', 'C']
3. astar_2_manhattan.test
   *** pacman layout: mediumMaze
   *** solution length: 68
   *** nodes expanded: 222
4. astar_3_goalAtDequeue.test
   *** solution: ['1:A->B', '0:B->C', '0:C->G']
   *** expanded_states: ['A', 'B', 'C']
5. graph_backtrack.test
   *** solution: ['1:A->C', '0:C->G']
   *** expanded_states: ['A', 'B', 'C', 'D']
6. 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 q5 searchAgent for BFS

"""
        Explanation:
        here we make start_posn
        return start_posn
        return 1 or 0 for goal state
        for successor
            we calculate next coordinate (nextx,nexty)
            if it do not hits wall then remove this coordinate from our left elements using a for loop
            append this next coordinate and left elements in successor list
            then return successor
"""
tinyCorners : Search nodes expanded: 252
mediumCorners : Search nodes expanded: 1966

1. corner_tiny_corner.test
    *** pacman layout: tinyCorner
    *** solution length: 28

Question q6 cornerHeuristic for A* search

"""
    Explanation:
    this heuristic is mainly defined by adding shortest distance
    [from current to our first food] return if no elements left by
    adding distance to our heuristic
    [and if there are elements left
    then calculating distance between nearest food and farthest food]
    and then adding to our heuristic and then returning heuristic
1. Heuristic --> here it is minimum distance from one vertex to another (food)
    (current to nearest) and (nearest to farthest) coordinate
2. this heuristic is Admissable and Consistent
3. Behaviour : it goes through all nodes and calculates Heuristic and follows that path to reach goal
4. Comparison (original(heuristic=0) and our heuristic) :
    Original => Search nodes expanded: 1966
    Our's => Search nodes expanded: 783
"""
mediumCorners : Search nodes expanded: 783

*** 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', 'East', 'East', 'North', '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 783 nodes

Question q7 foodHeuristic

"""
    Explanation:
    same as que 6 heuristic is difined
    this heuristic is mainly defined by adding shortest distance
    [from current to our first food] return if no elements left by
    adding distance to our heuristic
    [and if there are elements left
    then calculating distance between nearest food and farthest food]
    and then adding to our heuristic and then returning heuristic
1. Heuristic --> here it is minimum distance from one vertex to another (food)
    (current to nearest) and (nearest to farthest) coordinate
2. this heuristic is Admissable and Consistent
3. Behaviour : it goes through all nodes and calculates Heuristic and follows that path to reach goal
4. Comparison (original(heuristic=0) and our heuristic) :

testSearch	Original	=> Search nodes expanded: 14
	Our's	=> Search nodes expanded: 12
trickySearch	Original	=> Search nodes expanded: 16688
	Our's	=> Search nodes expanded: 8366

"""
testSearch : Search nodes expanded: 12
trickySearch : Search nodes expanded: 8366

1. food_heuristic_1.test
2. food_heuristic_10.test
3. food_heuristic_11.test
4. food_heuristic_12.test
5. food_heuristic_13.test
6. food_heuristic_14.test
7. food_heuristic_15.test
8. food_heuristic_16.test
9. food_heuristic_17.test
10. food_heuristic_2.test
11. food_heuristic_3.test
12. food_heuristic_4.test
13. food_heuristic_5.test
14. food_heuristic_6.test
15. food_heuristic_7.test
16. food_heuristic_8.test
17. food_heuristic_9.test
18. food_heuristic_grade_tricky.test
    *** expanded nodes: 8366
    *** thresholds: [15000, 12000, 9000, 7000]