University of Asia Pacific

Artificial Intelligence and Expert System Lab (CSE 404)

Department of CSE

Assignment No: 02

Topic/Question: Solve the following Maze game using A* algorithm.

Find the most cost-effective path to reach the final state from initial state A* Algorithm.

The agent will avoid the obstacles during traversing.

Date of Submission: 4 Feb, 2021

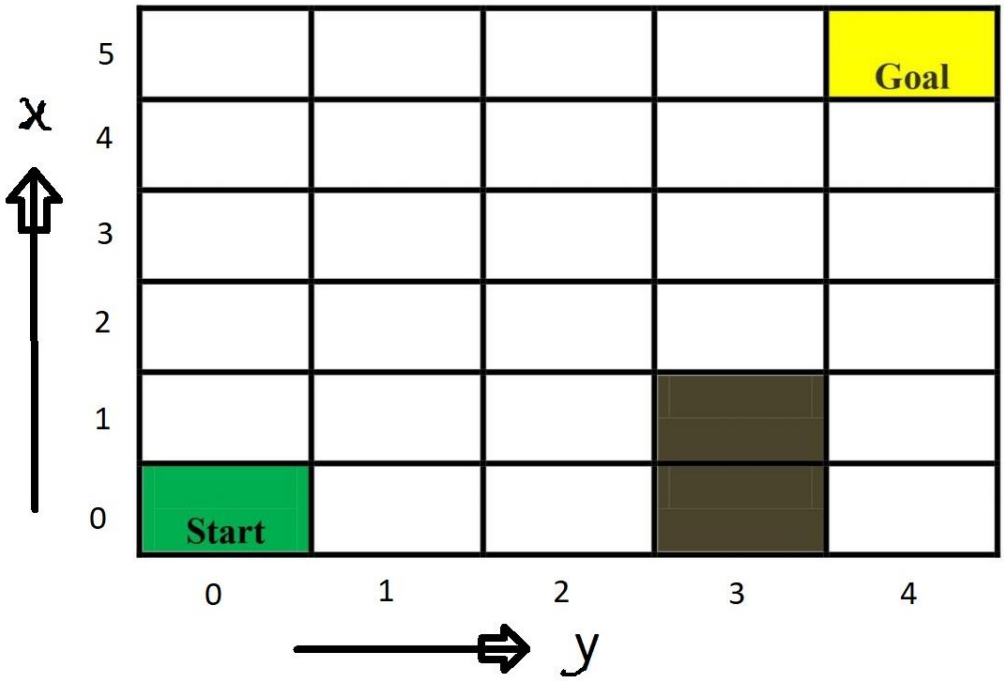
Submitted by	Submitted to
Name: Niamul Hasan Id: **17201026** Semester: 4.1 Section: A1	Dr. Nasima Begum Assistant Professor Department of CSE UAP

Tools:

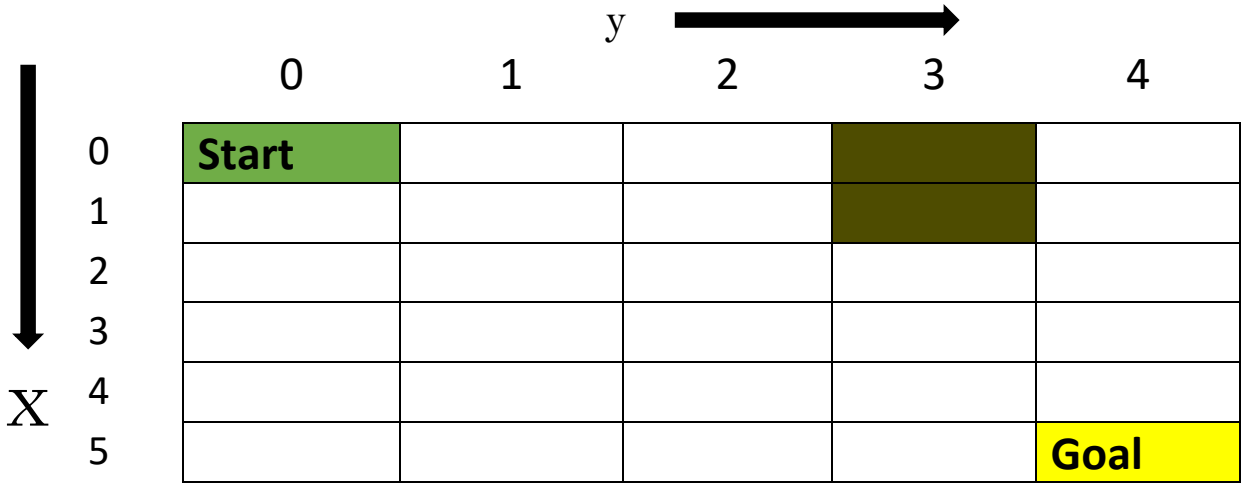
1. Language: Python 3.8 (for coding)
2. IDE: visual studio code (text editor)
3. Drawio (for diagram drawing)

Problem-02(For Even ID Students): Solve the following Maze Game using A* Algorithm.

Find the most cost-effective path to reach the final state from initial state using A* Algorithm. The agent will avoid the obstacles during traversing.



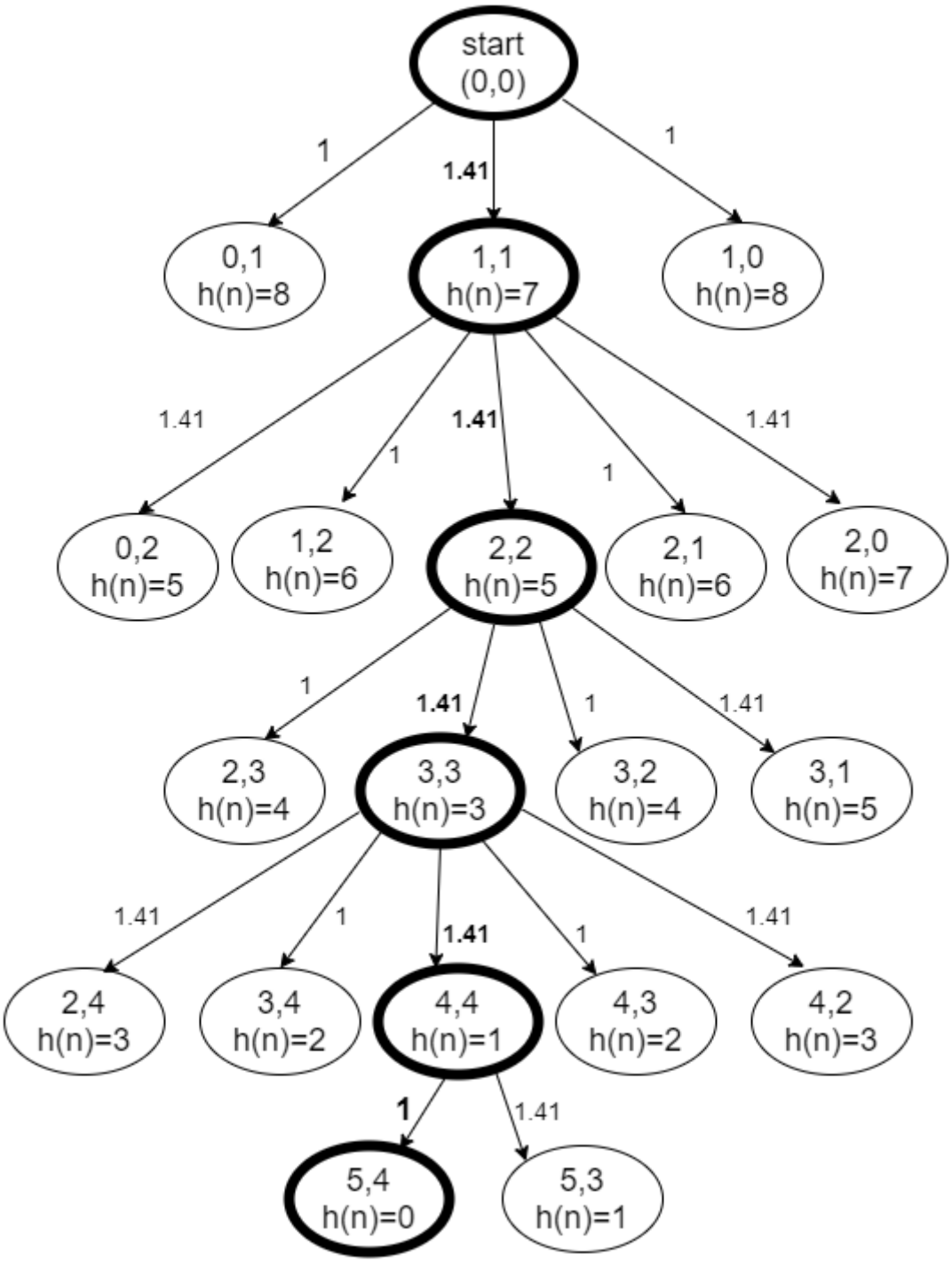
For coding purpose, we are considering the maze in vertical mirror reverse way.



Objective/ Target:

Here the objective is to move from start state to goal state and find the most cost-effective path to reach the final state from initial state by A* Algorithm.

Tree:



Code:

Input:

Here we are solving problem on a specific grid or maze. So, we can initialize our gride like this,

```
maze=[[  (0,0),(0,1),(0,2),       (0,4) ],# gap are the obstacles
      [(1,0),(1,1),(1,2),       (1,4) ],
      [(2,0),(2,1),(2,2),(2,3),(2,4) ],
      [(3,0),(3,1),(3,2),(3,3),(3,4) ],
      [(4,0),(4,1),(4,2),(4,3),(4,4) ],
      [(5,0),(5,1),(5,2),(5,3),(5,4) ]]

visit=[[0,0,0,None,0],
       [0,0,0,None,0],
       [0,0,0,0,0],
       [0,0,0,0,0],
       [0,0,0,0,0],
       [0,0,0,0,0],
       ]
       # 0 means not visited # 1 means vissited # none means can't visit
```

Output:

At first initialize the close fringe with start state: (0, 0)
f(n) of (0, 0) is : 9.00 = 0.00 + 9.00
neighbors of (0, 0) are [(0, 1), (1, 1), (1, 0)]
cost needed form (0, 0) to (0, 1) : 1.00
f(n) of (0, 1) is 9.00 = 1.00 + 8.00
cost needed form (0, 0) to (1, 1) : 1.41
f(n) of (1, 1) is 8.41 = 1.41 + 7.00
cost needed form (0, 0) to (1, 0) : 1.00
f(n) of (1, 0) is 9.00 = 1.00 + 8.00

now, the current position is (1, 1)
neighbors of (1, 1) are [(0, 2), (1, 2), (2, 2), (2, 1), (2, 0)]
cost needed form (1, 1) to (0, 2) : 1.41
f(n) of (0, 2) is 9.83 = 2.83 + 7.00
cost needed form (1, 1) to (1, 2) : 1.00
f(n) of (1, 2) is 8.41 = 2.41 + 6.00
cost needed form (1, 1) to (2, 2) : 1.41
f(n) of (2, 2) is 7.83 = 2.83 + 5.00
cost needed form (1, 1) to (2, 1) : 1.00
f(n) of (2, 1) is 8.41 = 2.41 + 6.00
cost needed form (1, 1) to (2, 0) : 1.41
f(n) of (2, 0) is 9.83 = 2.83 + 7.00

now, the current position is (2, 2)
neighbors of (2, 2) are [(2, 3), (3, 3), (3, 2), (3, 1)]
cost needed form (2, 2) to (2, 3) : 1.00

f(n) of (2, 3) is 7.83 = 3.83 + 4.00
cost needed form (2, 2) to (3, 3) : 1.41
f(n) of (3, 3) is 7.24 = 4.24 + 3.00
cost needed form (2, 2) to (3, 2) : 1.00
f(n) of (3, 2) is 7.83 = 3.83 + 4.00
cost needed form (2, 2) to (3, 1) : 1.41
f(n) of (3, 1) is 9.24 = 4.24 + 5.00

now, the current position is (3, 3)
neighbors of (3, 3) are [(2, 4), (3, 4), (4, 4), (4, 3), (4, 2)]
cost needed form (3, 3) to (2, 4) : 1.41
f(n) of (2, 4) is 8.66 = 5.66 + 3.00
cost needed form (3, 3) to (3, 4) : 1.00
f(n) of (3, 4) is 7.24 = 5.24 + 2.00
cost needed form (3, 3) to (4, 4) : 1.41
f(n) of (4, 4) is 6.66 = 5.66 + 1.00
cost needed form (3, 3) to (4, 3) : 1.00
f(n) of (4, 3) is 7.24 = 5.24 + 2.00
cost needed form (3, 3) to (4, 2) : 1.41
f(n) of (4, 2) is 8.66 = 5.66 + 3.00

now, the current position is (4, 4)
cost needed form (4, 4) to (5, 4) : 1.00
f(n) of (5, 4) is 6.66 = 6.66 + 0.00
cost needed form (4, 4) to (5, 3) : 1.41
f(n) of (5, 3) is 8.07 = 7.07 + 1.00

now, the current position is (5, 4)
Now, we reached to goal (5, 4)
f(n) of (5, 4) is : 6.66 = 6.66 + 0.00

Now, the cost effective path to reach the final state from initial state is:
(0, 0) --> 1.41 --> (1, 1) --> 2.83 --> (2, 2) --> 4.24 --> (3, 3) --> 5.66 --> (4, 4) --> 6.66 --> (5, 4)
Total cost: 6.66