amazon Coding Challenge

Bright Network Internship, June 2022

I have implemented Dijkstra's algorithm to find the path with a Minimum Heap for the priority queue.

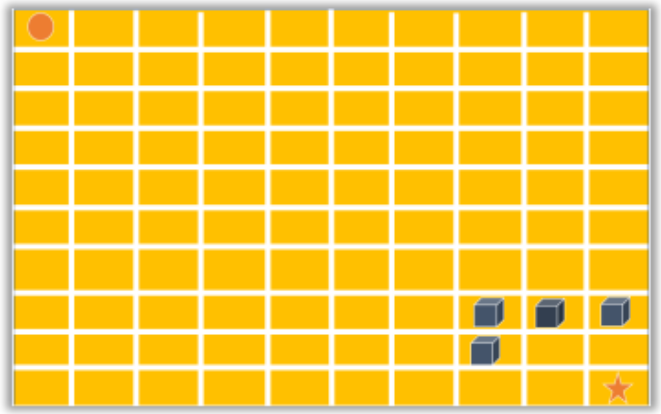
Overview:

Thank you for choosing to take part in Amazon's Coding Challenge for Bright Network. We are very happy to have you here. We hope you have as much fun implementing this challenge as we had creating it!

In this challenge, you are going to implement Amazon's pathfinding algorithm for Amazon's self-driving delivery vehicles. The self-driving vehicle will need to create a path on a 2D-grid that contains a starting point *(x,y)*, a delivery point *(x,y)* and a number of obstacles. Your vehicle can navigate to any of the adjacent squares (even diagonally), as long as the squares are inbound and do not contain an obstacle.

General notes:

You can use any language and ideally the output is to a command line.



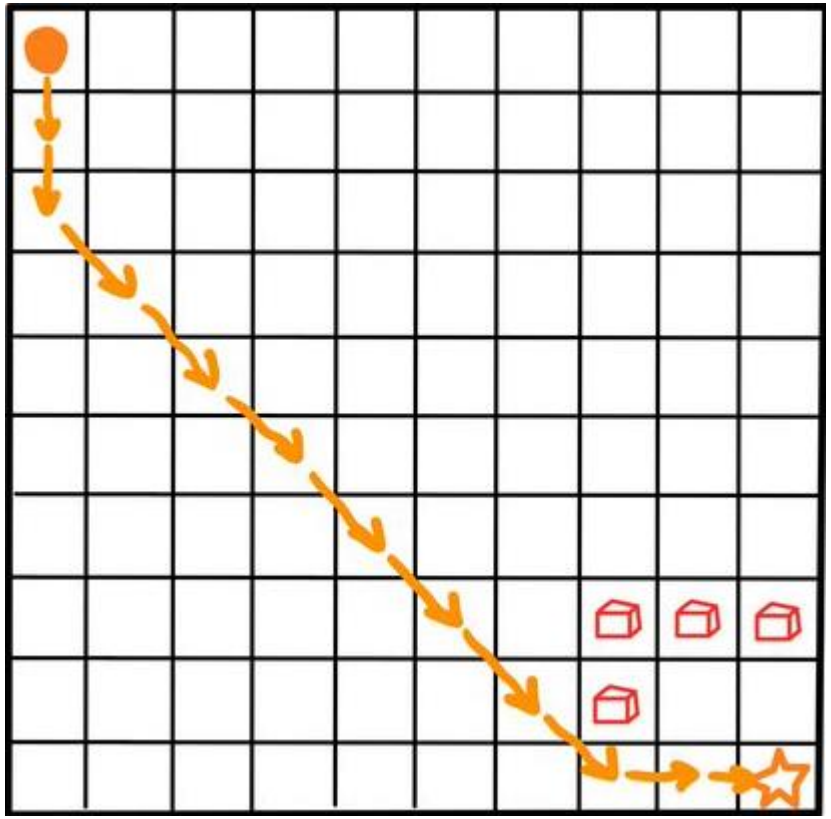
Phase 1:

Implement a 10x10 grid that contains a starting point on (0,0), the delivery point on (9,9) and the following obstacles on locations (9,7) (8,7) (6,7) (6,8).

Your algorithm should calculate a valid path avoiding the obstacles and reaching the delivery point.

Your solution should print the path in the format of
[(x1,y2), (x2,y2)…..] and also the number of steps.

Visualization



Input

The input were the four obstacles as visible on the 10x10 grid in the overview on the previous page: `[8,7],[7,7],[7,8],[7,9]`

Output

```
#############
## Phase 1 ##
#############
-
Optimal path is [(0, 0), (1, 0), (2, 0), (3, 1), (4, 2), (5, 3), (6, 4), (7, 5), (8, 6), (9, 7), (9, 8), (9, 9)]
-
Number of steps is 12
-
The optimal path is length 13.899494936611667
-
```

Phase 2:

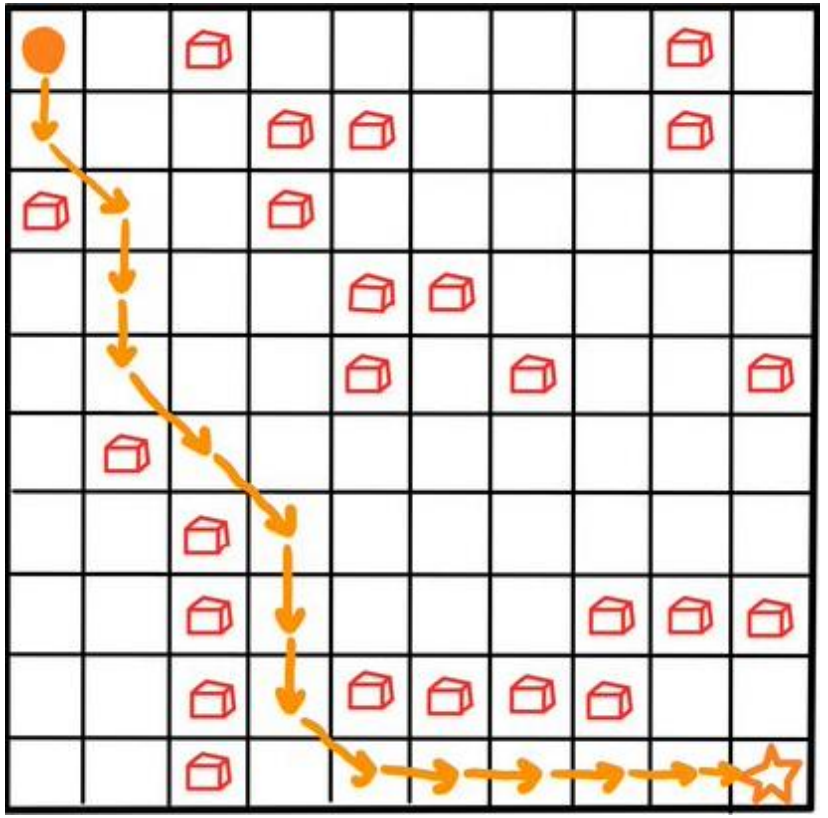
Add an additional 20 randomly placed obstacles and print their location using the format [(x1,y1),(x2,y2)….].

The obstacles should not overlap existing ones and should not be places at the start and delivery points.

Your algorithm should calculate a valid path avoiding the obstacles and reaching the delivery point.

Your solution should print the path in the format of
[(x1,y2), (x2,y2)…..]

Visualization



Input

The input were the randomly generated obstacles as outlined in the below screenshot in the first section of Phase 2, as well as 4 obstacles from Phase 1 on the previous page

Output

```
##############
## Phase 2 ##
##############
-
Randomly generated blocks are:
[[8, 5], [6, 2], [3, 4], [3, 5], [7, 2], [0, 2], [8, 6], [9, 2], [2, 3], [4, 6], [4, 4], [0, 8], [2, 0], [5, 1], [1, 4], [1, 8], [8, 4], [4, 9], [8, 2], [1, 3]]
-
Optimal path is [(0, 0), (1, 0), (2, 1), (3, 1), (4, 1), (5, 2), (6, 3), (7, 3), (8, 3), (9, 4), (9, 5), (9, 6), (9, 7), (9, 8), (9, 9)]
-
Number of steps is 15
-
The optimal path is length 15.65685424949238
```

Github

https://github.com/MaksKret/amazonCCbrightN