

main.py

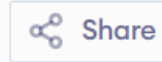


Share

Run

```
1 import heapq
2
3 class Node:
4     def __init__(self, position, parent=None, g=0, h=0):
5         self.position = position
6         self.parent = parent
7         self.g = g
8         self.h = h
9         self.f = g + h
10    def __lt__(self, other):
11        return self.f < other.f
12
13    def heuristic(a, b):
14        return abs(a[0] - b[0]) + abs(a[1] - b[1])
15
16    def a_star(grid, start, goal):
17        rows, cols = len(grid), len(grid[0])
18        open_list = []
19        heapq.heappush(open_list, Node(start, None, 0, heuristic(start,
20            goal)))
21        closed_set = set()
22        while open_list:
23            current_node = heapq.heappop(open_list)
24            if current_node.position == goal:
25                path = []
26                while current_node:
```

main.py



Run

```
25     while current_node:
26         path.append(current_node.position)
27         current_node = current_node.parent
28         return path[::-1]
29     closed_set.add(current_node.position)
30     for dr, dc in [(-1, 0), (1, 0), (0, -1), (0, 1)]:
31         new_pos = (current_node.position[0] + dr, current_node
32                     .position[1] + dc)
33         if (0 <= new_pos[0] < rows and 0 <= new_pos[1] < cols
34             and
35             grid[new_pos[0]][new_pos[1]] == 0 and new_pos not in
36                 closed_set):
37             new_node = Node(new_pos, current_node, current_node
38                             .g + 1, heuristic(new_pos, goal))
39             heapq.heappush(open_list, new_node)
40     return None
41
42 warehouse_grid = [
43     [0, 0, 0, 0, 1],
44     [1, 1, 0, 1, 0],
45     [0, 0, 0, 0, 0],
46     [0, 1, 1, 1, 0],
47     [0, 0, 0, 0, 0]
48 ]
49
50 start_position = (0, 0)
51 goal_position = (4, 4)
52 path = a_star(warehouse_grid, start_position, goal_position)
53 print("Optimal Path:", path)
```

Output

Clear

Optimal Path: [(0, 0), (0, 1), (0, 2), (1, 2), (2, 2), (2, 3), (2, 4), (3, 4), (4, 4)]

=== Code Execution Successful ===