

main.py

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

Output

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

=== Code Execution Successful ===

```

main.py
25     return path[::-1]
26
27     closed_set.add(current_node.position)
28     for dr, dc in [(-1, 0), (1, 0), (0, -1), (0, 1)]:
29         new_pos = (current_node.position[0] + dr, current_node
30                     .position[1] + dc)
31         if (0 <= new_pos[0] < rows and 0 <= new_pos[1] < cols
32             and
33             grid[new_pos[0]][new_pos[1]] == 0 and new_pos not in
34             closed_set):
35             new_node = Node(new_pos, current_node, current_node
36                             .g + 1, heuristic(new_pos,
37                                     goal))
38             heapq.heappush(open_list, new_node)
39
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

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

=== Code Execution Successful ===

Clear

31°C
Mostly cloudy

Search

ENG
IN