



## CHATGPT CODE

```
def hasPath(maze, start, destination):
    rows, cols = len(maze), len(maze[0])

    # Helper function to perform DFS
    def dfs(x, y):
        # Check if the current position is the destination
        if [x, y] == destination:
            return True

        # Mark the current position as visited
        maze[x][y] = -1

        # Explore four possible directions: up, down, left, and right
        directions = [(0, 1), (0, -1), (1, 0), (-1, 0)]
        for dx, dy in directions:
            new_x, new_y = x, y

            # Keep rolling in a direction until hitting a wall or the boundary
            while 0 <= new_x + dx < rows and 0 <= new_y + dy < cols and
maze[new_x + dx][new_y + dy] != 1:
                new_x += dx
                new_y += dy

            # If the new position is not visited, perform DFS on it
            if maze[new_x][new_y] == 0 and dfs(new_x, new_y):
                return True

        # If no path found, return False
        return False

    # Call DFS from the starting position
    return dfs(start[0], start[1])

# Test data
maze = [[0, 0, 1, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0], [1, 1, 0, 1, 1], [0,
0, 0, 0, 0]]
start = [0, 4]
destination = [4, 4]

# Check if there is a path from start to destination
```

```
print(hasPath(maze, start, destination)) # Output: True
```

## TEST CASE

```
1 def hasPath(maze, start, destination):
2     rows, cols = len(maze), len(maze[0])
3     def dfs(x, y):
4         if [x, y] == destination:
5             return True
6         maze[x][y] = -1
7         directions = [(0, 1), (0, -1), (1, 0), (-1, 0)]
8         for dx, dy in directions:
9             new_x, new_y = x, y
10            while 0 <= new_x + dx < rows and 0 <= new_y + dy < cols and maze[new_x + dx][new_y + dy] != 1:
11                new_x += dx
12                new_y += dy
13                if maze[new_x][new_y] == 0 and dfs(new_x, new_y):
14                    return True
15            return False
16        return dfs(start[0], start[1])
17 maze = [[0, 0, 1, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0], [1, 1, 0, 1, 1], [0, 0, 0, 0, 0]]
18 start = [0, 4]
19 destination = [4, 4]
20 print(hasPath(maze, start, destination)) # Output: True
21 maze = [[0, 0, 1, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0], [1, 1, 0, 1, 1], [0, 0, 0, 0, 0]]
22 start = [0, 4]
23 destination = [3, 2]
24 print(hasPath(maze, start, destination)) # Output: True
25 maze = [[0, 0, 1, 0, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0], [1, 1, 0, 1, 1], [0, 0, 0, 0, 0]]
26 start = [4, 3]
27 destination = [0, 1]
28 print(hasPath(maze, start, destination)) # Output: True
```

True  
False  
False

[Done] exited with code=0 in 0.186 seconds

Activate Windows  
Go to Settings to activate Windows.

Ln 14, Col 32 Spaces: 4 UTF-8 CRLF Python 3.9.13 64-bit (microsoft store) 4:04 PM 8/3/2023

