main.py

Share Run

```python
import heapq

grid = [
    [0, 1, 0],
    [0, 1, 0],
    [0, 0, 0]
]

start = (0, 0)
goal = (2, 2)

def heuristic(a, b):
    return abs(a[0]-b[0]) + abs(a[1]-b[1])

def astar(start, goal):
    heap = [(0 + heuristic(start, goal), 0, start)]
    came_from = {start: None}
    cost_so_far = {start: 0}

    while heap:
        _, cost, current = heapq.heappop(heap)
        if current == goal:
            path = []
            while current:
                path.append(current)
                current = came_from[current]
            return path[::-1]
        for dx, dy in [(-1,0),(1,0),(0,-1),(0,1)]:
            x, y = current[0]+dx, current[1]+dy
            neighbor = (x, y)
            if 0<=x<3 and 0<=y<3 and grid[x][y]==0:
                new_cost = cost + 1
                if neighbor not in cost_so_far or new_cost < cost_so_far[neighbor]:
                    cost_so_far[neighbor] = new_cost
                    heapq.heappush(heap, (new_cost + heuristic(neighbor, goal), new_cost, neighbor))
                    came_from[neighbor] = current

print(astar(start, goal))
```

Output

Clear

```
[(0, 0), (1, 0), (2, 0), (2, 1), (2, 2)]

=== Code Execution Successful ===
```