```python
import heapq

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

def astar(grid, start, goal):
    rows, cols = len(grid), len(grid[0])
    open_list = []
    heapq.heappush(open_list, (0 + heuristic(start, goal), 0, start, [start]))
    visited = set()

    while open_list:
        f, g, current, path = heapq.heappop(open_list)
        if current == goal:
            return path
        if current in visited:
            continue
        visited.add(current)
        x, y = current
        for dx, dy in [(-1,0),(1,0),(0,-1),(0,1)]:
            nx, ny = x + dx, y + dy
            if 0 <= nx < rows and 0 <= ny < cols and grid[nx][ny] == 0:
                next_node = (nx, ny)
                if next_node not in visited:
                    g_new = g + 1
                    f_new = g_new + heuristic(next_node, goal)
                    heapq.heappush(open_list, (f_new, g_new, next_node, path + [next_node]))
    return None
grid = [
    [0, 0, 0, 0, 0],
    [1, 1, 0, 1, 0],
    [0, 0, 0, 0, 0],
    [0, 1, 1, 1, 0],
    [0, 0, 0, 0, 0]
]

```

```python
start = (0, 0)
goal = (4, 4)
path = astar(grid, start, goal)

if path:
    print("Path found:")
    print(path)
else:
    print("No path found.")

```

```
>>> %Run -c $EDITOR_CONTENT

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

>>>
```