```
def dfs(maze, start, end):
    # Directions: up, right, down, left
    directions = [(-1, 0), (0, 1), (1, 0), (0, -1)]

    visited = set()

    def dfs_helper(current, path):
        if current == end:
            return path

        visited.add(current)

        for dx, dy in directions:
            next_cell = (current[0] + dx, current[1] + dy)

            if (0 <= next_cell[0] < len(maze) and
                0 <= next_cell[1] < len(maze[0]) and
                maze[next_cell[0]][next_cell[1]] != '#' and
                next_cell not in visited):

                result = dfs_helper(next_cell, path + [next_cell])
                if result:  # Path found
                    return result

        return None  # No path found from this branch

    return dfs_helper(start, [start])


# Example maze
maze = [
    ['S', '.', '.', '#', '.', '.', '.'],
    ['.', '#', '.', '#', '.', '#', '.'],
    ['.', '#', '.', '.', '.', '.', '.'],
    ['.', '.', '#', '#', '#', '.', '.'],
    ['.', '#', '.', '.', '.', '#', '.'],
    ['.', '#', '#', '#', '.', '#', '.'],
    ['.', '.', '.', '.', '.', '.', 'E'],
]

start = (0, 0)
end = (6, 6)

# Run DFS to find the path
```

```
path = dfs(maze, start, end)

if path:
    print("Path found!")
    print(" -> ".join(map(str, path)))
else:
    print("No path exists.")
```