LAB:06 A* Algorithm using "pygame" OR "pyamaze

```
sers > shiza > Downloads > astar_maze.py > ...
from pyamaze import maze, agent, COLOR
from queue import PriorityQueue

def h(cell1, cell2):
    """Manhattan distance heuristic."""
    x1, y1 = cell1
    x2, y2 = cell2
    return abs(x1 - x2) + abs(y1 - y2)

def aStar(m):
    start = (m.rows, m.cols)
    goal = (1,1)

    open_set = PriorityQueue()
    open_set.put((0, start))
    g_score = {cell: float('inf') for cell in m.grid}
    g_score[start] = 0
    f_score = {cell: float('inf') for cell in m.grid}
    f_score[start] = h(start, goal)

    came_from = {}

    while not open_set.empty():
        current = open_set.get()[1]

        if current == goal:
            break

        for direction in 'NSEW':
            if m.maze_map[current][direction] == 1:
                if direction == 'N':
                    neighbor = (current[0] - 1, current[1])
                if direction == 'S':
                    neighbor = (current[0] + 1, current[1])
                if direction == 'E':
                    neighbor = (current[0], current[1] + 1)
                if direction == 'W':
                    neighbor = (current[0], current[1] - 1)

                temp_g_score = g_score[current] + 1

                if temp_g_score < g_score[neighbor]:
                    came_from[neighbor] = current
                    g_score[neighbor] = temp_g_score
                    f_score[neighbor] = temp_g_score + h(neighbor, goal)
                    open_set.put((f_score[neighbor], neighbor))

    # Reconstruct path
    path = {}
    cell = goal
    while cell != start:
        path[came_from[cell]] = cell
        cell = came_from[cell]

    return path

# Create the maze and run the algorithm
m = maze(5, 5)  # You can change size here
m.CreateMaze()

path = aStar(m)

# Draw path with an agent
a = agent(m, footprints=True, color=COLOR.red)
m.tracePath({a: path})

m.run()
```