**Algorithm 1** A* Search Algorithm (Graph)

```
1: function A*(start, goal)
2:     closedset ← the empty set
3:     openset ← start
4:     came_from ← the empty map
5:
6:     q_score[start] ← 0
7:     f_score[start] ← q_score[start] + heuristic_cost_estimate(start, goal)
8:
9:     while openset ≠ ∅ do
10:        current ← the node in openset having the lowest f_score[ ]value
11:
12:        if current = goal then return reconstruct_path(came_from, goal)
13:
14:        remove current from openset
15:        add current to closedset
16:
17:        for all neighbour ∈ neighbour_nodes(current) do
18:            if neighbour ∈ closedset then
19:                continue
20:
21:            neighbour_g_score ← g_score[current] + dist_between(current, neighbour)
22:            if neighbour not in openset || neighbour_g_score < g_score[neighbour] then
23:                came_from[neighbour] ← current
24:                g_score[neighbour] ← neighbour_g_score
25:                f_score[neighbour] ← g_score[neighbour] + heuristic_cost_estimate(neighbour, goal)
26:
27:                if neighbour ∉ openset then
28:                    add neighbour to openset
29:    return failure
```