Code speciality:

1. Pointers used for better space complexity and memory management
2. Very efficient memory management, memory created only when reaquired and all memory created on heap.

Code readability:

1. No global variables used

Algorithm:

In this problem the given input is a map (binary occupancy grid with 1 representing accessible locations and 0 representing inaccessible locations). Having a start and end location which are points in the grid and the task is to perform shortest path navigation from the start to end. The motion model for the node is only in 4 directions. If the path is not possible return -1 and if the path is bigger than allocated limited space available then return the length of the path, but not fill the path as the allocated space is short.

For this problem I have coded the A-star algorithm. For every point in the grid I have made a node class with attributes such as location of the node (idx), distance travelled from the start node (dist), total accumulated heuristics (f) and pointer to its parent.

Firstly, I make the edges (vector <vector<int>>) which is nothing but the map and make the start and end index as accessible ie.1 and if the basic case as start == end then return 0

I make a list distances which tells me how far the node is from the start node and closed list which is a map of node index to its parent, the closed list will help me to trace back the path

As per the algorithm, I maintain a priority queue which gives me the node with least value of heuristic function (f=g+h), then I pop the node out see if it is the target or not and some basic checks as per requirements, I then propagate the node in four directions as per the question, and fill the respective node attributes until the path is found and if not found I return -1. If path is found I fill the path in the poutBuffer array.