vehicle routing and map navigation

programme:

import heapq

def dijkstra(graph, start, goal):

queue = [(0, start)]

distances = {node: float('inf') for node in graph}

distances[start] = 0

shortest\_path = {start: None}

while queue:

current\_distance, current\_node = heapq.heappop(queue)

if current\_node == goal:

break

for neighbor, weight in graph[current\_node].items():

distance = current\_distance + weight

if distance < distances[neighbor]:

distances[neighbor] = distance

shortest\_path[neighbor] = current\_node

heapq.heappush(queue, (distance, neighbor))

path, node = [], goal

while node is not None:

path.insert(0, node)

node = shortest\_path[node]

return path, distances[goal]

# Example usage:

graph = {

'A': {'B': 2, 'C': 4},

'B': {'C': 1, 'D': 7},

'C': {'E': 3},

'D': {'E': 2},

'E': {}

}

route, cost = dijkstra(graph, 'A', 'E')

print("Best Route:", route)

print("Total Distance:", cost)

output:

Best Route: ['A', 'B', 'C', 'E']

Total Distance: 6