DATE: PAGE:

Sumanth.KV
1BM18CS112
5C

fro

```
from collections import defaultdict

class Graph():
    def __init__(self):
        self.edges = defaultdict(list)
        self.weights = {}

    def addEdge(self, from_node, to_node,
                weight):
        self.edges[from_node].append(to_node)
        self.edges[to_node].append(from_node)
        self.weights[(from_node, to_node)] =
                                    weight
        self.weights[(to_node, from_node)] = weight

def dijsktra(graph, initial, end):
    shortest_path = {initial: (None, 0)}
    current_node = initial
    visited = set()
    while current_node != end:
        visited.add(current_node)
        destinations = graph.edges[current_node]
        weight_to_current_node =
            shortest_paths[current_node][1]

        for next_node in destinations:
            weight = graph.weights
                [(current_node, next_node)] +
                weight_to_current_node
```

Sumanth.ks

```
        if next_node not in shortest_path :
            shortest_paths[next_node] = (current_node, weight)
    else:
        current_shortest_weight = shortest_paths[next_node][1]
        if current_shortest_weight > weight:
            shortest_paths[next_node] = (current_node, weight)

next_destination = {node: shortest_paths[node] for node in shortest_paths if node not in visited}

if not next_destination:
    return "Route not possible"
current_node = min(next_destinations, key=lambda k: next_destinations[k][1])

path = []
while current_node is not None:
    path.append(current_node)
    next_node = shortest_path[current_node][0]
    current_node = next_node

path = path[::-1]
print("shortest weight", current_shortest_weight)
print(path)
```

(2) Sumanth.ks