Prathamsinh
Batch – CSE54
Enrollment No. - 22162171004

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

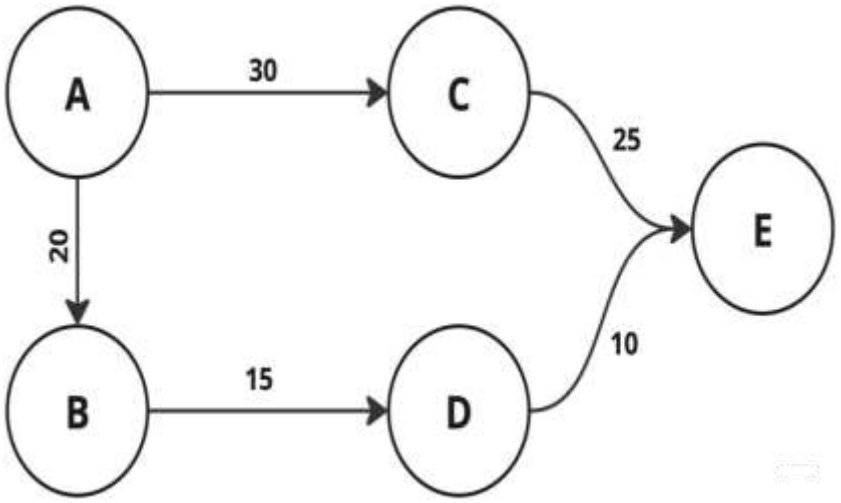
Practical 11

AIM:

A government official needs to visit several cities within a state. To minimize travel costs, they want to find the shortest path between their starting city and each destination city.

Task:

Given a graph representing the cities and their connecting roads, determine the minimum cost path from a given starting city to all other cities.



Input:

Enter total number of nodes: 5

Enter the node from where you want to calculate the distance: A

Enter Data (Weight):

	A	B	C	D	E
A	0	20	30	∞	∞
B	∞	0	∞	15	∞
C	∞	∞	0	∞	25
D	∞	∞	∞	0	10
E	∞	∞	∞	∞	0

Output:

	A	B	C	D	E
A	0	20	30	35	45
B	∞	0	∞	15	25

C	∞	∞	0	∞	25
D	∞	∞	∞	0	10
E	∞	∞	∞	∞	0

OR

Source	Destination	Cost
A	A	0
	B	20
	C	30
	D	35
	E	45

Code :-

```
import sys


def dijkstra(graph,
    start_node):n = len(graph)
    visited = [False] * n
    distance = [sys.maxsize] *
    ndistance[start_node] = 0


    for _ in range(n):
        min_distance =
        sys.maxsize min_index =-1
```

```
        for i in range(n):
            if not visited[i] and distance[i] <
                min_distance:min_distance = distance[i]
                min_index = i
        visited[min_index] =
        True

        for j in range(n):
            if graph[min_index][j] != float('inf') and not visited[j]:
                new_dist = distance[min_index] + graph[min_index][j]
                if new_dist < distance[j]:
                    distance[j] = new_dist
    return distance

def print_distances(distance, cities):
    print("Source Destination Cost")

    for i in range(len(cities)):
        if distance[i] ==
            sys.maxsize:
            print(f"{cities[i]}: ∞")
        else:
```

```
        print(f"{cities[i]}: {distance[i]}")


cities = ['A', 'B', 'C', 'D', 'E']
graph = [
    [0, 20, 30, float('inf'), float('inf')],
    [float('inf'), 0, float('inf'), 15, float('inf')],
    [float('inf'), float('inf'), 0, float('inf'), 25],
    [float('inf'), float('inf'), float('inf'), 0, 10],
    [float('inf'), float('inf'), float('inf'), float('inf'), 0]
]

start_city = 'A'
start_node = cities.index(start_city)
distances = dijkstra(graph, start_node)
print_distances(distances, cities)
```

Output :-