198) Given a graph represented by an adjacency matrix, implement Dijkstra's Algorithm to find the shortest path from a given source vertex to all other vertices in the graph. The graph is represented as an adjacency matrix where graph[i][j] denote the weight of the edge from vertex i to vertex j. If there is no edge between vertices i and j, the value is Infinity (or a very large number).

Test Case 1:
Input:
n = 5
graph = [[0, 10, 3, Infinity, Infinity], [Infinity, 0, 1, 2, Infinity], [Infinity, 4, 0, 8, 2],
[Infinity, Infinity, Infinity, 0, 7], [Infinity, Infinity, Infinity, 9, 0]]
source = 0
Output: [0, 7, 3, 9, 5]
Test Case 2:
Input:
n = 4
graph = [[0, 5, Infinity, 10], [Infinity, 0, 3, Infinity], [Infinity, Infinity, 0, 1],
[Infinity, Infinity, Infinity, 0]]
source = 0
Output: [0, 5, 8, 9]

AIM: To write a python program for the graph is represented as an adjacency matrix where graph[i][j] denote the weight of the edge from vertex i to vertex j. If there is no edge between vertices i and j, the value is Infinity (or a very large number).

PROGRAM:

```
import sys

def dijkstra(graph, src):
    n = len(graph)
    dist = [sys.maxsize] * n
    dist[src] = 0
    visited = [False] * n

    for _ in range(n):
        # Find the vertex with the minimum distance that hasn't been visited yet
        u = min_distance(dist, visited)
        visited[u] = True

        # Update the distance value of the adjacent vertices of the picked vertex
```

```
        for v in range(n):
            if not visited[v] and graph[u][v] and dist[u] != sys.maxsize and dist[u] +
graph[u][v] < dist[v]:
                dist[v] = dist[u] + graph[u][v]

    return dist

def min_distance(dist, visited):
    min_val = sys.maxsize
    min_index = -1

    for v in range(len(dist)):
        if not visited[v] and dist[v] < min_val:
            min_val = dist[v]
            min_index = v

    return min_index

n = 5
graph = [
    [0, 10, 3, sys.maxsize, sys.maxsize],
    [sys.maxsize, 0, 1, 2, sys.maxsize],
    [sys.maxsize, 4, 0, 8, 2],
    [sys.maxsize, sys.maxsize, sys.maxsize, 0, 7],
    [sys.maxsize, sys.maxsize, sys.maxsize, 9, 0]
]
source = 0
print("Test Case 1 Output:", dijkstra(graph, source))
```

OUTPUT:

```
A module you have imported isn't available at the moment. It will be available
    soon.
```

TIME COMPLEXITY : O(n^2)