Name :Ganesh

USN : 2VX23UE015

Title : 08 : Dijkstra's Algorithm

In [13]:

```
graph = [
    [0, 0, 0, 7, 0],
    [3, 0, 4, 0, 0],
    [0, 0, 0, 5, 6],
    [0, 2, 0, 0, 0],
    [0, 0, 4, 0, 0]
]

source = 0

def dijkstra(graph, source):
    V = len(graph)
    dist = [float('inf')] * V
    dist[source] = 0
    visited = [False] * V

    for _ in range(V):

        min_dist = float('inf')
        min_index = -1
        for v in range(V):
            if not visited[v] and dist[v] < min_dist:
                min_dist = dist[v]
                min_index = v



        visited[min_index] = True


        for v in range(V):
            if not visited[v] and graph[min_index][v] :
                dist[v] = min(dist[v], dist[min_index] + graph[min_index][v])

    return dist


shortest_paths = dijkstra(graph, source)
print("Shortest distances from node", source, "to each node:")
for i, d in enumerate(shortest_paths):
    print("Vertex", i, "Distance =", d)
```

```
Shortest distances from node 0 to each node:
Vertex 0 Distance = 0
Vertex 1 Distance = 9
Vertex 2 Distance = 13
Vertex 3 Distance = 7
Vertex 4 Distance = 19
```

In [15]:

```python
n = int(input("Enter number of vertices in the graph: "))
print("Enter the adjacency matrix where each row separated by spaces: ")
graph =[]
for i in range(n):
    row = input().split()
    graph.append([int(x) for x in row])
source = int(input("Enter the source vertex: "))
def dijkstra(graph, source):
    V = len(graph)
    dist = [float('inf')] * V
    dist[source] = 0
    visited = [False] * V

    for _ in range(V):

        min_dist = float('inf')
        min_index = -1
        for v in range(V):
            if not visited[v] and dist[v] < min_dist:
                min_dist = dist[v]
                min_index = v



        visited[min_index] = True


        for v in range(V):
            if not visited[v] and graph[min_index][v] :
                dist[v] = min(dist[v], dist[min_index] + graph[min_index][v])

    return dist


shortest_paths = dijkstra(graph, source)
print("Shortest distances from node", source, "to each node:")
for i, d in enumerate(shortest_paths):
    print("Vertex", i, "Distance =", d)
```

```
Enter the adjacency matrix where each row separated by spaces:
Shortest distances from node 0 to each node:
Vertex 0 Distance = 0
Vertex 1 Distance = 9
Vertex 2 Distance = 13
Vertex 3 Distance = 7
Vertex 4 Distance = 19
```

```
In [ ]:
```