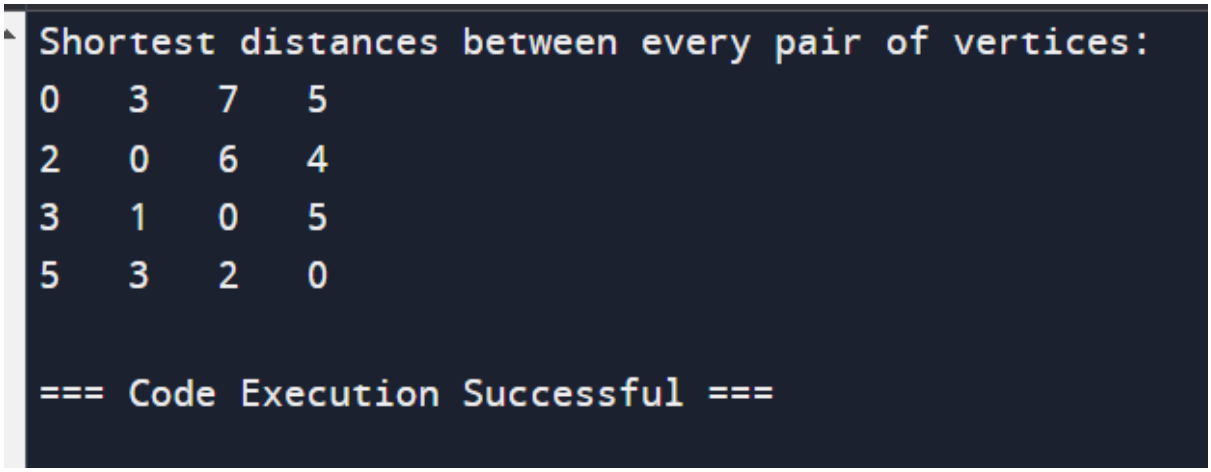
110. Warshalls algorithm

PROGRAM:-

```
# Define a large value for infinity
INF = float('inf')

def floyd_warshall(graph):
    # Number of vertices in the graph
    V = len(graph)

    # Initialize the solution matrix the same as the input graph matrix
    dist = [[INF] * V for _ in range(V)]
    for i in range(V):
        for j in range(V):
            dist[i][j] = graph[i][j]

    # Put 0 on the diagonal
    for i in range(V):
        dist[i][i] = 0

    # Update the solution matrix by considering all vertices as intermediate vertices
    for k in range(V):
        for i in range(V):
            for j in range(V):
                if dist[i][k] + dist[k][j] < dist[i][j]:
                    dist[i][j] = dist[i][k] + dist[k][j]

    print_solution(dist)

def print_solution(dist):
    print("Shortest distances between every pair of vertices:")
    for i in range(len(dist)):
        for j in range(len(dist)):
            if dist[i][j] == INF:
                print("INF", end="\t")
            else:
                print(dist[i][j], end="\t")
        print()

# Example usage
if __name__ == "__main__":
    graph = [
        [0, 3, INF, 5],
        [2, 0, INF, 4],
        [INF, 1, 0, INF],
        [INF, INF, 2, 0]
    ]
    floyd_warshall(graph)
```

OUTPUT:-



TIME COMPLEXITY:-O(V^3)