105.Floyd Algorithm

AIM: To find the shortest path between all pairs

PROGRAM:

```
INF = float('inf')


def floyd_warshall(graph):
    n = len(graph)
    dist = [[0] * n for _ in range(n)]
    for i in range(n):
        for j in range(n):
            dist[i][j] = graph[i][j]
    for k in range(n):
        for i in range(n):
            for j in range(n):
                if dist[i][k] != INF and dist[k][j] != INF and dist[i][k] + dist[k][j] < dist[i][j]:
                    dist[i][j] = dist[i][k] + dist[k][j]


    return dist
graph = [
    [0, 3, INF, 7],
    [8, 0, 2, INF],
    [5, INF, 0, 1],
    [2, INF, INF, 0]
]


result = floyd_warshall(graph)
for row in result:
    print(row)
```

OUTPUT:

```
[0, 3, 5, 6]
[5, 0, 2, 3]
[3, 6, 0, 1]
[2, 5, 7, 0]
```

TIME COMPLEXITY: $O(n^3)$