4.13 Implement Floyd's Algorithm to find the shortest path between all pairs of cities. Display the distance matrix before and after applying the algorithm. Identify and print the shortest path

**AIM**

To implement Floyd’s Algorithm to compute the shortest paths between all pairs of cities in a weighted graph, display the distance matrix before and after applying the algorithm, and identify the shortest path..

**ALGORITHM**

**1**. Start

2.Input the number of routers n and cost of each link (∞ if no direct link).

3. Build an initial distance matrix:

* Distance to itself = 0
* Distance to neighbors = given link cost
* If no direct connection = ∞.

4. Display the initial distance matrix.

5.Apply Floyd’s Algorithm

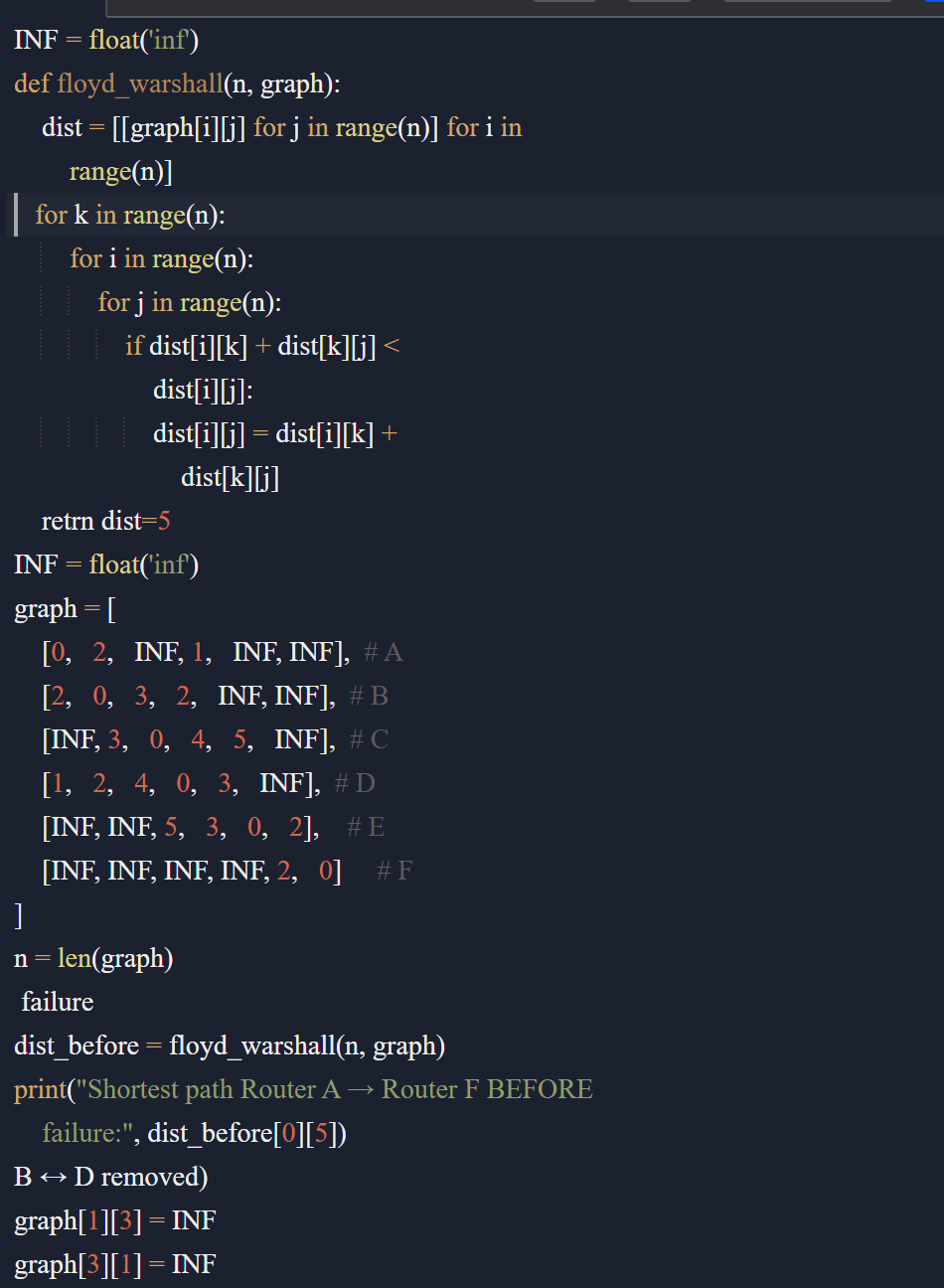
6. Display shortest path distance from Router A to Router F.

7.Simulate **link failure**: set dist[B][D] = dist[D][B] = ∞.

8.Run Floyd’s Algorithm again to recompute all-pairs shortest paths.

9.Display shortest path distance from Router A to Router F after failure.

10.End

****

Input:

Routers: A, B, C, D, E, F

Links:

A–B = 2

A–D = 1

B–C = 3

B–D = 2

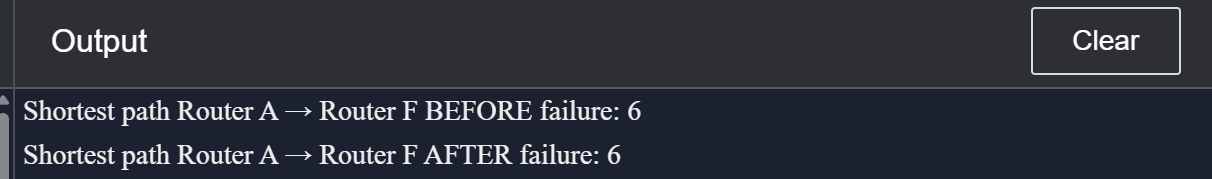
C–D = 4

C–E = 5

D–E = 3

E–F = 2

Output:



**RESULT:**

The program correctly implements Floyd’s Algorithm to calculate shortest paths in the router network.

* Before the **B–D link failure**, the shortest path from **Router A → Router F = 5**.
* After the **B–D link failure**, the shortest path increases to **6**.

**PERFORMANCE ANALYSIS:**

* **Time Complexity: O(n³)**
* **Space Complexity: O(n²)**