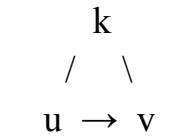
1. **All Pair Shortest Path :**
   Every nodes single source shortest path. For this we need Floyd - Warshall Algorithm. **It is an Incremental Algorithm.**
   For this we will use Adjacency-Matrix.
   Intermediate node = not using the direct edge, by using other bypassable nodes from u to v node are the intermediate nodes.

2. **Floyd-Warshall Simulation part 1 :**
   d[u][v] = d[u][k] + d[k] + d[v] // k is the intermediate node.



   d[u][v] = min(d[u][v], d[u][k] + d[k][v]

3. **Floyd-Warshall Simulation part 2 :**
   Increasing the intermediate nodes number will produce the more shortest distance of each node.

4. **Pseudocode & Complexity :**
   - Input → A weighted graph as an adjacency matrix.
   - Output → All pair shortest path/distance.
   - Create a distance matrix, d. Where d[i][j] = x, where there is a direct edge from i to j which cost is x.
   - for all node "i" d[i][i] = 0.
   - for all nodes i & j where there isn't any direct edge from i→j : d[i][j] = INF
   - for all node "k" :
     - for all node u:
       - for all node v:
         - d[u][v] = min(d[u][v] , d[u][k] + d[k][v])
   - output all pair shortest distance, 'd'

   Time Complexity : O(V^3)
   Space Complexity : O (V^2)

5. **Floyd Warshall Code : Implementation**
6. **Problem Solving :** From CSES
   Shortest Routes II : https://cses.fi/problemset/task/1672