**Aim:** To find All Pair Shortest Path using Floyd Warshall Algorithm

**Theory:**

The all pair shortest path algorithm is also known as Floyd-Warshall algorithm is used to find all pair shortest path problem from a given weighted graph. As a result of this algorithm, it will generate a matrix, which will represent the minimum distance from any node to all other nodes in the graph.

**Algorithm:**

Floyd Warshall(w)

N = w.rows

D0 = w

For k = 1 to n

Let Dk = (dij) be a new n x n matrix

For I = 1 to n

For j = 1 to n

dijk = min{ dijk-1 , dikk-1 , dkjk-1 }

return Dn

**Example:**

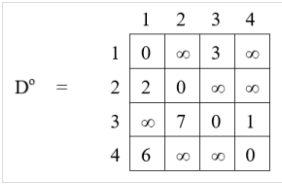
**Problem: Apply Floyd’s method to find the shortest path for the below-mentioned all pairs.**



**Solution:**

Optimal substructure formula for Floyd’s algorithm,

Dk [i, j] = min { Dk – 1 [i, j], Dk – 1 [i, k] + Dk – 1 [k, j] }



**Iteration 1 : k = 1 :**

D1[1, 2] = min { Do [1, 2], Do [1, 1] + Do [1, 2] }

= min {∞, 0 + ∞}

=  ∞

D1[1, 3] = min { Do [1, 3], Do [1, 1] + Do [1, 3] }

= min {3, 0 + 3}

= 3

D1[1, 4] = min { Do [1, 4], Do [1, 1] + Do [1, 4] }

= min {∞, 0 + ∞}

=  ∞

D1[2, 1] = min { Do [2, 1], Do [2, 1] + Do [1, 1] }

= min {2, 2 + 0}

= 2

D1[2, 3] = min { Do [2, 3], Do [2, 1] + Do [1, 3] }

= min {∞, 2 + 3}

= 5

D1[2, 4] = min { Do [2, 4], Do [2, 1] + Do [1, 4] }

= min {∞, 2 + ∞}

= ∞

D1[3, 1] = min { Do [3, 1], Do [3, 1] + Do [1, 1] }

= min {∞, 0 + ∞}

= ∞

D1[3, 2] = min { Do [3, 2], Do [3, 1] + Do [1, 2] }

= min {7, ∞ + ∞}

= 7

D1[3, 4] = min { Do [3, 4], Do [3, 1] + Do [1, 4] }

= min {1, ∞ + ∞}

= 1

D1[4, 1] = min { Do [4, 1], Do [4, 1] + Do [1, 1] }

= min {6, 6 + 0}

= 6

D1[4, 2] = min { Do [4, 2], Do [4, 1] + Do [1, 2] }

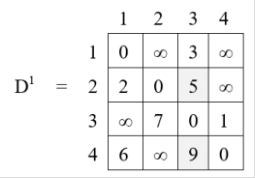
= min {∞, 6 + ∞}

= ∞

D1[4, 3] = min { Do [4, 3], Do [4, 1] + Do [1, 3] }

= min {∞, 6 + 3}

= 9



**Note :** Path distance for highlighted cell is improvement over original matrix.

**Iteration 2 (k = 2) :**

D2[1, 2] = D1 [1, 2]  =  ∞

D2[1, 3] = min { D1 [1, 3], D1 [1, 2] + D1 [2, 3] }

= min {3, ∞ + 5}

= 3

D2[1, 4] = min { D1 [1, 4], D1 [1, 2] + D1 [2, 4] }

= min {∞, ∞ + ∞}

= ∞

D2[2, 1] = D1 [2, 1]  =  2

D2[2, 3] = D1 [2, 3]  =  5

D2[2, 4] = D1 [2, 4]  =  ∞

D2[3, 1] = min { D1 [3, 1], D1 [3, 2] + D1 [2, 1] }

= min {∞, 7 + 2}

= 9

D2[3, 2] = D1 [3, 2]  =  7

D2[3, 4] = min { D1 [3, 4], D1 [3, 2] + D1 [2, 4] }

= min {1, 7 + ∞}

= 1

D2[4, 1] = min { D1 [4, 1], D1 [4, 2] + D1 [2, 1] }

= min {6, ∞ + 2}

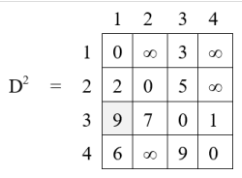
= 6

D2[4, 2] = D1 [4, 2]  =  ∞

D2[4, 3]  =  min { D1 [4, 3], D1 [4, 2] + D1 [2, 3] }

= min {9, ∞ + 5}

= 9

**Iteration 3 (k = 3) :**

D3[1, 2] = min { D2 [1, 2], D2 [1, 3] + D2 [3, 2] }

= min {∞, 3 + 7}

= 10

D3[1, 3] = D2 [1, 3]  =  3

D3[1, 4] = min { D2 [1, 4], D2 [1, 3] + D2 [3, 4] }

= min {∞, 3 + 1}

= 4

D3[2, 1] = min { D2 [2, 1], D2 [2, 3] + D2 [3, 1] }

= min {2, 5 + 9}

= 2

D3[2, 3] = D2 [2, 3]  =  5

D3[2, 4] = min { D2 [2, 4], D2 [2, 3] + D2 [3, 4] }

= min {∞, 5 + 1}

= 6

D3[3, 1] = D2 [3, 1]  =  9

D3[3, 2] = D2 [3, 2]  =  7

D3[3, 4] = D2 [3, 4]  =  1

D3[4, 1] = min { D2 [4, 1], D2 [4, 3] + D2 [3, 1] }

= min {6, 9 + 9}

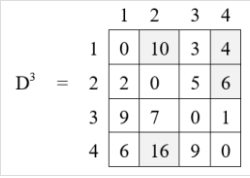
= 6

D3[4, 2] = min { D2 [4, 1], D2 [4, 3] + D2 [3, 2] }

= min {∞, 9 + 7}

= 16

D3[4, 3] = D2 [4, 3]  =  9



**Iteration 4 (k = 4) :**

D4[1, 2] = min { D3 [1, 2], D3 [1, 4] + D3 [4, 2] }

= min {10, 4 + 16}

= 10

D4[1, 3] = min { D3 [1, 3], D3 [1, 4] + D3 [4, 1] }

= min {3, 4 + 9}

= 3

D4[1, 4] = D3 [1, 4]  =  4

D4[2, 1] = min { D3 [2, 1], D3 [2, 4] + D3 [4, 1] }

= min {2, 6 + 6}

= 2

D4[2, 3] = min { D3 [2, 3], D3 [2, 4] + D3 [4, 3] }

= min {5, 6 + 9}

= 5

D4[2, 4] = D3 [2, 4]  =  6

D4[3, 1] = min { D3 [3, 1], D3 [3, 4] + D3 [4, 1] }

= min {9, 1 + 6}

= 7

D4[3, 2] = min { D3 [3, 2], D3 [3, 4] + D3 [4, 2] }

= min {7, 1 + 16}

= 7

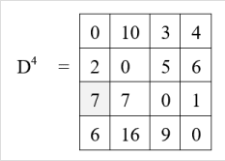
D4[3, 4] = D3 [3, 4]  =  1

D4[4, 1] = D3 [4, 1]  =  6

D4[4, 2] = D3 [4, 2]  = 16

D4[4, 3] = D3 [4, 3]  =  9

Final distance matrix is,



**Code:**

#include <stdio.h>

#define INF 99999

#define V 4

void floydWarshall(int d[V][V]){

int i,j,k;

for(k=0;k<V;k++){

for(i=0;i<V;i++){

for(j=0;j<V;j++){

if(d[i][j]>d[i][k]+d[k][j])

{d[i][j]=d[i][k]+d[k][j];}

}

}

}

printf("Shortest distances for each pair:\n");

for(i=0;i<V;i++){

for(j=0;j<V;j++){

if (d[i][j] == INF)

printf("%s ", "INF");

else

printf("%d ", d[i][j]);

}

printf("\n");

}

}

void main()

{

int graph[V][V] = { { 0, 3, INF, 7 },

{ 8, 0, 2, INF },

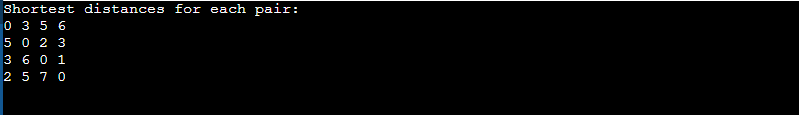
{ 5, INF, 0, 1 },

{ 2, INF, INF, 0 } };

floydWarshall(graph);

}

**Output:**

****

**Conclusion:** Thus we have successfully found the shortest distance from ever source node to every destination node using Floyd Warshall Algorithm.