**Floyd's Program**

**Design and implement C/C++ Program to solve All-Pairs Shortest Paths problem using Floyd's algorithm.**

#include <stdio.h>

// defining the number of vertices

#define nV 4

#define INF 999

void printMatrix(int matrix[][nV]);

// Implementing floyd warshall algorithm

void floydWarshall(int graph[][nV])

{

int matrix[nV][nV], i, j, k;

for (i = 0; i < nV; i++)

for (j = 0; j < nV; j++)

matrix[i][j] = graph[i][j];

// Adding vertices individually

for (k = 0; k < nV; k++)

{

for (i = 0; i < nV; i++)

{

for (j = 0; j < nV; j++)

{

if (matrix[i][k] + matrix[k][j] < matrix[i][j])

matrix[i][j] = matrix[i][k] + matrix[k][j];

}

}

}

printMatrix(matrix);

}

void printMatrix(int matrix[][nV])

{

printf("All pairs shortest path is\n");

for (int i = 0; i < nV; i++)

{

for (int j = 0; j < nV; j++)

{

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

printf("%4s", "INF");

else

printf("%4d", matrix[i][j]);

}

printf("\n");

}

}

int main() {

int graph[nV][nV] = {{INF, 1,4,6},

{INF, INF,7, 4},

{2,INF, INF,3},

{INF, INF, INF, INF}};

floydWarshall(graph);

}

Output

All pairs shortest path is

6 1 4 5

9 10 7 4

2 3 6 3

INF INF INF INF



int main()

{

int graph[nV][nV] = {{0, 3, INF, 5},

{2, 0, INF, 4},

{INF, 1, 0, INF},

{INF, INF, 2, 0}};

floydWarshall(graph);

}

All pairs shortest path is

0 3 7 5

2 0 6 4

3 1 0 5

5 3 2 0

3)

int main()

{

int graph[nV][nV] = {{0,INF,3,INF},

{2, 0, INF, INF},

{INF, 7, 0, 1},

{6, INF, INF, 0}};

floydWarshall(graph);

}

All pairs shortest path is

0 10 3 4

2 0 5 6

7 7 0 1

6 16 9 0