**2.FLOYDS-WARSHALL**

#include <stdio.h>

#include <stdlib.h>

void floydWarshall(int \*\*graph, int n)

{

int i, j, k;

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

{

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

{

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

{

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

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

}

}

}

}

int main(void)

{

int n, i, j;

printf("Enter the number of vertices: ");

scanf("%d", &n);

int \*\*graph = (int \*\*)malloc((long unsigned) n \* sizeof(int \*));

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

{

graph[i] = (int \*)malloc((long unsigned) n \* sizeof(int));

}

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

{

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

{

if (i == j)

graph[i][j] = 0;

else

graph[i][j] = 100;

}

}

printf("Enter the edges: \n");

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

{

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

{

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

scanf("%d", &graph[i][j]);

}

}

printf("The original graph is:\n");

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

{

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

{

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

}

printf("\n");

}

floydWarshall(graph, n);

printf("The shortest path matrix is:\n");

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

{

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

{

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

}

printf("\n");

}

return 0;

}

