5B) Design and implement C/C++ Program to find the transitive closure using Warshal's algorithm.

#include <stdio.h>

// Function to print the transitive closure matrix

void printTransitiveClosure(int reach[][10], int V) {

printf("Transitive closure of the graph is:\n");

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

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

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

}

printf("\n");

}

}

// Function to find transitive closure using Warshall's algorithm

void transitiveClosure(int graph[][10], int V) {

int reach[10][10];

int i, j, k;

// Initialize the reach[][] matrix same as the input graph matrix

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

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

reach[i][j] = graph[i][j];

// Add all vertices one by one to the set of intermediate vertices

// Before start of a iteration, we have reachability values for all

// pairs of vertices such that the reachability values consider

// only the vertices in set {0, 1, 2, .. k-1} as intermediate vertices

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

// Pick all vertices as source one by one

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

// Pick all vertices as destination for the above picked source

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

// If vertex k is on a path from i to j,

// then make sure that the value of reach[i][j] is 1

reach[i][j] = reach[i][j] || (reach[i][k] && reach[k][j]);

}

}

}

// Print the transitive closure matrix

printTransitiveClosure(reach, V);

}

int main() {

int V;

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

scanf("%d", &V);

int graph[10][10];

printf("Enter the adjacency matrix:\n");

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

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

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

}

}

// Find transitive closure using Warshall's algorithm

transitiveClosure(graph, V);

return 0;

}

OUTPUT:

Enter the number of vertices: 5

Enter the adjacency matrix:

1 0 0 1 0

0 0 0 0 0

0 1 1 1 0

0 1 0 1 0

0 1 1 1 1

1 0 0 1 0

0 0 0 0 0

0 1 1 1 0

0 1 0 1 0

0 1 1 1 1

Transitive closure of the graph is:

1 1 0 1 0

0 0 0 0 0

0 1 1 1 0

0 1 0 1 0

0 1 1 1 1