import java.util.\*;

public class DijkstraMST {

private int numVertices;

private int[] dist;

private boolean[] visited;

private int[][] graph;

public DijkstraMST(int[][] graph, int numVertices) {

this.numVertices = numVertices;

this.graph = graph;

this.dist = new int[numVertices];

this.visited = new boolean[numVertices];

}

public void dijkstra(int startVertex) {

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

dist[i] = Integer.MAX\_VALUE;

visited[i] = false;

}

dist[startVertex] = 0;

for (int i = 0; i < numVertices - 1; i++) {

int u = minDistance(dist, visited);

visited[u] = true;

for (int v = 0; v < numVertices; v++) {

if (!visited[v] && graph[u][v] != 0 && dist[u] != Integer.MAX\_VALUE && dist[u] + graph[u][v] <

dist[v]) {

dist[v] = dist[u] + graph[u][v];

}}}

printMST(startVertex);

}

private int minDistance(int[] dist, boolean[] visited) {

int minDist = Integer.MAX\_VALUE;

int minIndex = -1;

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

if (!visited[i] && dist[i] <= minDist) {

minDist = dist[i];

minIndex = i;

}

}

return minIndex;

}

private void printMST(int startVertex) {

System.out.println("Vertex \t Distance from Source");

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

System.out.println(i + "\t" + dist[i]);

}

}

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

System.out.print("Enter the size of the graph: ");

int n = in.nextInt();

int[][] graph = new int[n][n];

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

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

System.out.print("Enter the weight "+i+ "-> "+j+" of the graph: ");

graph[i][j]=in.nextInt();

}}

DijkstraMST dijkstra = new DijkstraMST(graph,n);

System.out.print("Enter the starting vertex of the graph: ");

int vertex=in.nextInt();

dijkstra.dijkstra(vertex);

}}