/\* PRIMS ALGO GREEDY APROACH \*/

#include <iostream>

#include <climits>

using namespace std;

#define V 5 // Number of vertices in the graph

int minKey(int key[], bool mstSet[]) {

int min = INT\_MAX, min\_index;

for (int v = 0; v < V; v++)

if (mstSet[v] == false && key[v] < min)

min = key[v], min\_index = v;

return min\_index;

}

void printMST(int parent[], int graph[V][V]) {

cout << "Edge \tWeight\n";

for (int i = 1; i < V; i++)

cout << parent[i] << " - " << i << "\t" << graph[i][parent[i]] << "\n";

}

void primMST(int graph[V][V]) {

int parent[V]; // Array to store constructed MST

int key[V]; // Key values used to pick minimum weight edge in cut

bool mstSet[V]; // To represent set of vertices not yet included in MST

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

key[i] = INT\_MAX, mstSet[i] = false;

key[0] = 0;

parent[0] = -1; // First node is always the root of MST

for (int count = 0; count < V - 1; count++) {

int u = minKey(key, mstSet);

mstSet[u] = true;

for (int v = 0; v < V; v++)

if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v])

parent[v] = u, key[v] = graph[u][v];

}

printMST(parent, graph);

}

int main() {

int graph[V][V];

cout << "Enter the adjacency matrix of the graph (" << V << "x" << V << "):\n";

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

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

cin >> graph[i][j];

}

}

// Print the solution

primMST(graph);

return 0;

}