#include <iostream>

using namespace std;

class graph{

int a[10][10],b[10][10];

int n;//Number of vertices.

public:

graph(int x){

n=x;

for(int i=0;i<10;i++){

for(int j=0;j<10;j++){

a[i][j]=b[i][j]=0;

}

}

}

void prims(){

int visited[n],from[n],dist[n];

int sum = 0;

//visited[]: marks if a vertex is added to MST.

//from[]: tracks parent node in MST.

//dist[]: stores current minimum weight edge to each vertex from the MST.

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

visited[i] = 0;

from[i] = 0;

dist[i] = a[0][i];

}

visited[0] = 1;

for(int edges = 0; edges < n - 1; edges++){

int min = 999999, u = -1, v = -1;

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

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

min = dist[i];

v = i;

u = from[i];

}

}

if(u != -1 && v != -1){

b[u][v] = b[v][u] = min;

sum += min;

visited[v] = 1;

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

if(!visited[i] && a[v][i] < dist[i]){

dist[i] = a[v][i];

from[i] = v;

}

}

}

}

cout << "The sum of the Minimum Spanning Tree is : " << sum << endl;

}

void input(){

cout<<"Enter the adjacency matrix weights(Enter -1 for no edge)"<<endl;

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

for(int j = i+1;j<n;j++){

cout<<"Enter the weight of the edge("<<i<<","<<j<<") : ";

cin>>a[i][j];

if(a[i][j] == -1){

a[i][j]=a[j][i]=100;// Representing no edge with large value

}

else{

a[j][i]=a[i][j];//// Symmetric for undirected graph

}

}

a[i][i] = 0;

}

}

void disp(){

cout<<"Minimum Spanning Tree adjacency matrix : "<<endl;

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

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

cout<<b[i][j]<<" ";

}

cout<<endl;

}

}

};

int main(){

int n;

cout<<"Enter the number of vertices : ";

cin>>n;

graph g(n);

g.input();

g.prims();

g.disp();

return 0;

}