#include <stdio.h>

#define INF 999

#define MAX 100

int p[MAX], c[MAX][MAX], t[MAX-1][2];

int find(int v) {

if (p[v] == v)

return v;

return p[v] = find(p[v]); // Path compression

}

void union1(int i, int j) {

p[j] = i;

}

void kruskal(int n) {

int i, j, k, u, v, min, res1, res2, sum = 0;

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

min = INF;

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

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

if (i == j) continue;

if (c[i][j] < min) {

u = find(i);

v = find(j);

if (u != v) {

res1 = i;

res2 = j;

min = c[i][j];

}

}

}

}

union1(find(res1), find(res2));

t[k][0] = res1;

t[k][1] = res2;

sum += min;

}

printf("\nCost of spanning tree is = %d\n", sum);

printf("Edges of spanning tree are:\n");

for (i = 0; i < n - 1; i++)

printf("%d -> %d\n", t[i][0], t[i][1]);

}

int main() {

int i, j, n;

printf("\nEnter the n value: ");

scanf("%d", &n);

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

p[i] = i;

printf("\nEnter the graph data:\n");

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

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

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

kruskal(n);

return 0;

}