**Constructing Roads**

**Time Limit: 2000/1000 MS (Java/Others)    Memory Limit: 65536/32768 K (Java/Others)  
Total Submission(s): 19724    Accepted Submission(s): 7536**

Problem Description

There are N villages, which are numbered from 1 to N, and you should build some roads such that every two villages can connect to each other. We say two village A and B are connected, if and only if there is a road between A and B, or there exists a village C such that there is a road between A and C, and C and B are connected.   
  
We know that there are already some roads between some villages and your job is the build some roads such that all the villages are connect and the length of all the roads built is minimum.

Input

The first line is an integer N (3 <= N <= 100), which is the number of villages. Then come N lines, the i-th of which contains N integers, and the j-th of these N integers is the distance (the distance should be an integer within [1, 1000]) between village i and village j.  
  
Then there is an integer Q (0 <= Q <= N \* (N + 1) / 2). Then come Q lines, each line contains two integers a and b (1 <= a < b <= N), which means the road between village a and village b has been built.

Output

You should output a line contains an integer, which is the length of all the roads to be built such that all the villages are connected, and this value is minimum.

Sample Input

3

0 990 692

990 0 179

692 179 0

1

1 2

Sample Output

179

Source

[kicc](http://acm.hdu.edu.cn/search.php?field=problem&key=kicc&source=1&searchmode=source)

分析：

给出n个城市之间的距离 然后在给出q 接下来的q行 每行给出两个城市的序号 表示这两个城市已经建过路了 求还没有建路的城市之间的最小的即可

根据题意是求最小生成树的 直接套用prim算法即可

但是要注意某些城市已经有路了

AC代码：

#include <iostream>

#include <cstring>

#include <cstdio>

#define INF 0x3f3f3f3f

**using namespace** std**;**

**int** map**[**1001**][**1001**],** t**,** vis**[**1001**],**dis**[**1001**];**

**void** prim**()**

**{**

memset**(**dis**,**1**,sizeof(**dis**));**

**int** sum **=** 0**;**

**int** k**;**

**for(int** i **=** 0**;** i **<** t**;** i**++)**

**{**

vis**[**i**] =** map**[**0**][**i**];**

**}**

**int** min**,**min1 **=** INF**;**

dis**[**0**] =** 0**;**

**for(int** i **=** 1**;** i **<** t**;** i**++)**

**{**

min **=** INF**;**

**for(int** j **=** 0**;** j **<** t**;** j**++)**

**{**

**if(**dis**[**j**]!=**0**&&**vis**[**j**]<**min**)**

**{**

min **=** vis**[**j**];**

k **=** j**;**

**}**

**}**

dis**[**k**] =** 0**;**

sum**+=**min**;**

**for(int** j **=** 0**;** j **<** t**;** j**++)**

**if(**dis**[**j**]!=**0**&&**map**[**k**][**j**] <** vis**[**j**])**

**{**

vis**[**j**] =** map**[**k**][**j**];**

**}**

**}**

printf**(**"%d\n"**,**sum**);**

**}**

**int main()**

**{**

**while(~**scanf**(**"%d"**,&**t**))**

**{**

**for(int** i **=** 0**;** i **<** t**;** i**++)**

**for(int** j **=** 0**;** j **<** t**;** j**++)**

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

**int** n**,** x**,** y**;**

scanf**(**"%d"**,&**n**);**

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

**{**

scanf**(**"%d%d"**,&**x**,&**y**);**

map**[**x**-**1**][**y**-**1**] =** map**[**y**-**1**][**x**-**1**] =** 0**;**

**}**

prim**();**

**}**

**return** 0**;**

**}**