백준 1507번 - 궁금한 민호

시간제한	메모리 제한	제출	정답	맞은 사람	정답 비율
2초	128MB	4674	2331	1832	48.775%

문제

강호는 N개의 도시로 이루어진 나라에 살고 있다. 각 도시는 M개의 도로로 연결되어 있으며, 각 도로를 지날 때 필요한 시간이 존재한다. 도로는 잘 연결되어 있기 때문에, 도시 A에서 B로 이동할 수 없는 경우는 존재하지 않는다.

도시 A에서 도시 B로 바로 갈 수 있는 도로가 있거나, 다른 도시를 거쳐서 갈 수 있을 때, 도시 A에서 B를 갈 수 있다고 한다.

강호는 모든 쌍의 도시에 대해서 최소 이동 시간을 구해놓았다. 민호는 이 표를 보고 원래 도로가 몇 개 있는지를 구해보려고 한다.

예를 들어, 예제의 경우에 모든 도시 사이에 강호가 구한 값을 가지는 도로가 존재한다고 해도 된다. 하지만, 이 도로의 개수는 최솟값이 아니다. 예를 들어, 도시 1-2, 2-3, 1-4, 3-4, 4-5, 3-5를 연결하는 도로만 있다고 가정해도, 강호가 구한 모든 쌍의 최솟값을 구할 수 있다. 이 경우 도로의 개수는 6개이고, 모든 도로의 시간의 합은 55이다.

모든 쌍의 도시 사이의 최소 이동 시간이 주어졌을 때, 이 나라에 존재할 수 있는 도로의 개수의 최솟값일 때, 모든 도로의 시간의 합을 구하는 프로그램을 작성하시오.

입력

첫째 줄에 도시의 개수 N($1 \le N \le 20$)이 주어진다. 둘째 줄부터 N개의 줄에 각각의 도시 사이에 이동하는데 필요한 시간이 주어진다. A에서 B로 가는 시간과 B에서 A로 가는 시간은 같다. 또, A와 B가 같은 경우에는 0이 주어지고, 그 외의 경우에 필요한 시간은 2500보다 작거나 같은 자연수이다.

출력

첫째 줄에 도로 개수가 최소일 때, 모든 도로의 시간의 합을 출력한다. 불가능한 경우에는 -1을 출력한다.

예제 입력1

```
5
0 6 15 2 6
6 0 9 8 12
15 9 0 16 18
2 8 16 0 4
6 12 18 4 0
```

예제 출력1

```
55
```

(이하 생략)

출처

출처

알고리즘 분류

- 그래프 이론
- 플로이드-와샬

접근 방법

각 정점들 쌍의 최단경로가 주어졌을 때, 간선의 개수를 최소화하는 방법들에 대해서 생각해보자.

먼저, N개의 정점들이 **최대** $\frac{N(N-1)}{2}$개의 undirected edge를 갖는다는 것을 알 수 있다.

여기서 불필요한 간선을 제거하는 식으로 문제를 해결해보자.

가정에 따라 초기 상태에서 간선은 총 $\frac{N(N-1)}{2}$ 개가 존재한다. 불필요한 간선은 다음과 같을 때 발생한다.

> 서로 다른 3개의 정점 a, b, c가 있을 때, (a $\rightarrow$ c 의 가중치 = a $\rightarrow$ b $\rightarrow$ c 의 가중치)

이 때는, a $\rightarrow$ c로 가는 간선을 제거해주어도 a에서 c로 가는 최단 시간을 보장할 수 있게 된다.

만약, a $\rightarrow$ b $\rightarrow$ c 로 가는 간선을 제거한다면 b $\rightarrow$ c 로 가는 간선에 영향을 주게 되고 이것은 추후에 b로부터 출발하는 간선이 유실되어 b와 a 또는 b와 c의 연결성을 잃어버릴 수 있으므로, a $\rightarrow$ c로 가는 간선을 삭제한다.

즉, Complete Graph에서 삭제될 수 있는 간선들은 다음과 같은 조건을 갖는다.

- i $\neq$ k and j $\neq$ k 이고 `minDist[i][j] = minDist[i][k] + minDist[k][j]` 일 때, i $\rightarrow$ j 사이의 간선을 제거

`불가능한 경우에는 -1을 출력한다`라는 문장은 문제에서 주어지는 입력에서 모순이 존재할 수 있다는 것을 함의한다.

입력에서 모순이 발생하는 경우는 아래와 같을 것이다.

`minDist[i][j] > minDist[i][k] + minDist[k][j]`

이 경우는, i → j로 가는 최단 경로(`minDist[i][j]`)가 i로 부터 출발하여 경유점 k 를 경유하여 도착하는 것 보다 더 느리다는 사실을 나타내고, 이것은 `minDist[i][j]`의 정의에 모순된다는 것을 의미하므로 이 경우가 존재할 경우 그 즉시 -1을 출력하고 프로그램을 종료한다.

소스코드

```cpp
#define FASTIO cin.tie(0)->sync_with_stdio(false), cout.tie(0)
////////////////////////////////////////////////////////////////
#include <bits/stdc++.h>
using namespace std;
int N, minDist[25][25];
bool isValid[25][25] = {false, };
int main(void){
    FASTIO;
    cin >> N;
    for(int i=0; i<N; i++){
        for(int j=0; j<N; j++){
            cin >> minDist[i][j];
        }
    }
    memset(isValid, true, sizeof(isValid));
    for(int k=0; k<N; k++){
        for(int i=0; i<N; i++){
            for(int j=0; j<N; j++){
                if(minDist[i][j] == minDist[i][k] + minDist[k][j] && k != i && k != j)
                    isValid[i][j] = false;
                else if(minDist[i][j] > minDist[i][k] + minDist[k][j]){
                    cout << -1 << '\n';
                    exit(0);
                }
            }
        }
    }

    int ans = 0;
    for(int i=0; i<N; i++){
        for(int j=0; j<N; j++)
            if(isValid[i][j])
                ans += minDist[i][j];
    }
    cout << ans / 2;
    return 0;
}
```