1123 - Trail Maintenance

SUBMIT	PDF (English)	Statistics	Forum
Time Limit: 2 second(s)		Memory Limit: 32 MB	

Tigers in the Sunderbans wish to travel freely among the **N** fields (numbered from **1** to **N**), even though they are separated by trees. The tigers wish to maintain trails between pairs of fields so that they can travel from any field to any other field using the maintained trails. Tigers may travel along a maintained trail in either direction.

The tigers do not build trails. Instead, they maintain deer trails that they have discovered. On any week, they can choose to maintain any or all of the deer animal trails they know about. Always curious, the tigers discover one new deer trail at the beginning of each week. They must then decide the set of trails to maintain for that week so that they can travel from any field to any other field. Tigers can only use trails which they are currently maintaining.

The tigers always want to minimize the total length of trail they must maintain. The tigers can choose to maintain any subset of the deer trails they know about, regardless of which trails were maintained the previous week. Deer trails (even when maintained) are never straight. Two trails that connect the same two fields might have different lengths. While two trails might cross, tigers are so focused; they refuse to switch trails except when they are in a field. At the beginning of each week, the tigers will describe the deer trail they discovered. Your program must then output the minimum total length of trail the tigers must maintain that week so that they can travel from any field to any other field, if there is such a set of trails.

Input

Input starts with an integer **T (≤ 25)**, denoting the number of test cases.

Each case starts with two integers **N (1 ≤ N ≤ 200)** and **W**. **W** is the number of weeks the program will cover **(1 ≤ W ≤ 6000)**.

Each of the next **W** lines will contain three integers describing the trail the tigers found that week. The first two numbers denote the end points (filed numbers) and the third number denotes the length of the trail **(1 to 10000)**. No trail has the same field as both of its end points.

Output

For each case, print the case number in a line. Then for every week, output a single line with the minimum total length of trail the tigers must maintain so that they can travel from any field to any other field. If no set of trails allows the tigers to travel from any field to any other field, output **"-1"**.

Sample Input	Output for Sample Input
1 4 6 1 2 10 1 3 8 3 2 3 1 4 3 1 3 6 2 1 2	Case 1: -1 -1 -1 14 12 8

```cpp
#include<bits/stdc++.h>
using namespace std;
int par[205],n,m,a,cnt;
struct edge{
    int u,v,w;
};
vector<edge>vv,uu;
bool cmp(edge x, edge y){
    return x.w<y.w;
}
int findparent(int x){
    if(par[x]==x) return x;
    return par[x]=findparent(par[x]);
}
void Unionparent(int x,int y){
    par[findparent(y)] = findparent(x);
}

int kruskalMST(){
    for(int i=1; i<=n; i++) par[i]=i;
    int sum=0,cnt=0;

    sort(vv.begin(),vv.end(),cmp);

    uu.clear();
    for(int i=0; i<vv.size(); i++){
        edge top = vv[i];
        int parU = findparent(top.u);
        int parV = findparent(top.v);

        if(parU!=parV){
            Unionparent(parU,parV);
            sum += top.w;
            cnt++;
            edge pp; pp.u=top.u, pp.v=top.v, pp.w=top.w;
            uu.push_back(pp);
        }
    }

    vv.clear();
    vv = uu;

    if(cnt==n-1)return sum;
    return -1;
}
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        printf("Case %d:\n",ks);
        scanf("%d%d",&n,&m);
        vv.clear();

        for(int i=1; i<=m; i++){
            int u,v,w;
            scanf("%d%d%d",&u,&v,&w);
            edge pp; pp.u=u, pp.v=v, pp.w=w;
            vv.push_back(pp);

            int ans = kruskalMST();
            printf("%d\n",ans);
        }
    }
    return 0;
}
```