猴王解报

考虑 a_i 和 b_i 是朋友意味着如果 a_i 在 **原神party** 中那么 b_i 在 c_i 次 **原神party** 后就也要在 **原神party** 中。

我们将猴子抽象成点，朋友关系抽象成长度为 c_i 的边。那么我们发现整个 **原神party** 过程就相当于从点 1 开始依次向周围“扩散”，求“扩散”到点 n 的最早时间。也即点 1 到点 n 的最短路。

1 与 n 不联通即输出 `-1`。

代码使用堆优化的dijkstra算法求单源最短路。

Code

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
inline ll read()
{
    ll x(0),f(1);char c=getchar();
    while(c<'0'||c>'9')f=c=='-'?-1:1,c=getchar();
    while(c<='9'&&c>='0')x=x*10+c-48,c=getchar();
    return x*f;
}
const int N=100010;
ll f[N];
vector<pair<int,ll> >G[N];//vector存图
void add(int x,int y,ll z){G[x].push_back({y,z});G[y].push_back({x,z});}
priority_queue<pair<ll,int>,vector<pair<ll,int>>,greater<pair<ll,int>>>pq;//小根pair<ll,int>堆
int main()//使用dijkstra堆优化算法求最短路
{
    freopen("MK.in","r",stdin);freopen("MK.out","w",stdout);
    int n=read(),m=read();
    for(int i=1;i<=m;i++){int x=read(),y=read(),z=read();add(x,y,z);}
    memset(f,0x3f,sizeof(f));f[1]=0;pq.push({f[1],1});
    while(!pq.empty())
    {
        pair<ll,int>nw=pq.top();pq.pop();
        int x=nw.second;ll fx=nw.first;if(fx!=f[x])continue;
        for(auto [y,z]:G[x])//新式语法。相当于for(int i=0,y=G[x][i].first,ll z=G[x][i].second(显然这
        if(f[y]>f[x]+z)f[y]=f[x]+z,pq.push({f[y],y});
    }
    if(f[n]==f[0])puts("-1");//判断是否无法到达
    else printf("%lld",f[n]);//直接输出 1~n 的最短路长度
    return 0;
}
```