

$\text{\text{GYM}}$ 链接 : [Matrix Exponentiation](#)

F

给一个无向图，求包含 k 条边的最小路径。

考虑 dp 定义 $f[i][j][k]$ 为从第 i 个点出发经过 k 条边到第 j 个点的最短路。则会有如下的转移方程

$$f[i][j][k] = \min_u \{f[i][u][k-1] + g[u][j]\}$$

但 k 值很大，所以不能直接 dp 因为取最小值的操作和乘法都满足结合律交换律，且上述形式和矩阵乘法类似，所以可以把矩阵乘法改写成上面的转移方程，再用矩阵快速幂加速。

```
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define db double
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<" "; <<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 2e5+5;

inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}

int n = 100;

struct Matrix
{
    ll mat[100][100];
    //Matrix() {memset(mat,0,sizeof(mat));}
    /*Matrix operator * (Matrix b)
    {
        Matrix res;
        rep(i,0,n-1) rep(j,0,n-1) rep(k,0,n-1) res.mat[i][j] =
        (res.mat[i][j] + mat[i][k]*b.mat[k][j]%M)%M;
        return res;
    }*/
    Matrix() {rep(i,0,n-1) rep(j,0,n-1) mat[i][j] = INF;}
    Matrix operator * (Matrix b)
```

```
{
    Matrix res;
    rep(k,0,n-1){
        rep(i,0,n-1) rep(j,0,n-1) {
            res.mat[i][j] = min(res.mat[i][j],mat[i][k]+b.mat[k][j]);
        }
    }
    return res;
}
Matrix operator ^ (ll b)
{
    Matrix res,A=*this;
    rep(i,0,n-1) rep(j,0,n-1) res.mat[i][j] = (i!=j)*INF;
    while(b){
        if(b&1) res = res*A;
        A = A*A;
        b>>=1;
    }return res;
}
};
int main()
{
    fastio();
    int m,k; cin>>n>>m>>k;
    Matrix A;
    while(m--){ int u,v,w;
        cin>>u>>v>>w; u--,v--;
        A.mat[u][v] = w;
    }
    A = A^k;
    ll ans = INF;
    rep(i,0,n-1) rep(j,0,n-1) ans = min(ans,A.mat[i][j]);
    if(ans>=1e18) cout<<"IMPOSSIBLE"<<endl;else cout<<ans<<endl;
    return 0;
}
```

From:
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:wangzai_milk:matrix_exponentiation&rev=1597167879

Last update: 2020/08/12 01:44