虚树相关

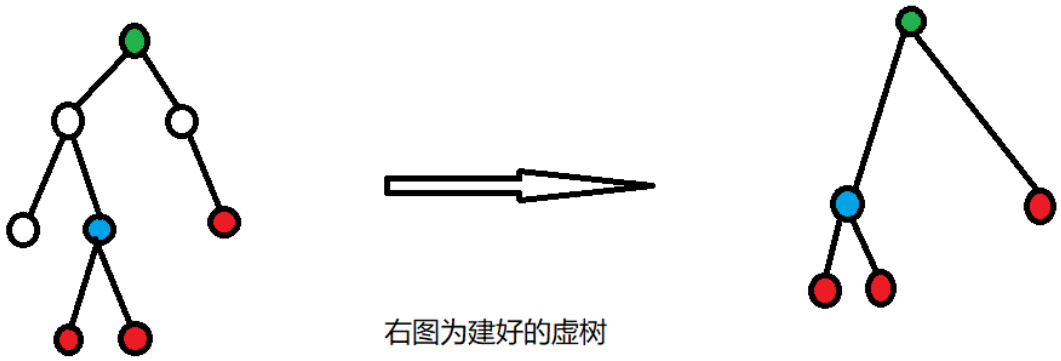
前置技能

欧拉序：就是从根结点出发，按dfs的顺序在绕回原点所经过所有点的顺序。 **LCA**：最近公共祖先。 **栈**：不会就去面壁。

相关概念

> 虚树就是将原问题中的树形结构选取关键点，建成的这棵新树。 通过选取关键点可以降低复杂度（优化暴力）。

虚树有效地保留了原树的主要特征，精简了原树结构，所以称其优化暴力。



[SDOI2011]消耗战

建立虚树

先找出关键点，加入虚树，再将关键点按欧拉序排序，求出两两之间的LCA加入虚树，最后判断根节点是否加入虚树即可。

```
for(int i=1;i<=K;i++)
    read(x),vis[St[++Top]=x]=1,ft[x]=m[x];//初始化
sort(St+1,St+Top+1,Cmp);//按欧拉序排序
for(int i=2;i<=Top;i++){
    int fa=Lca(St[i],St[i-1]);
    if(!vis[fa])vis[St[++Top]=fa]=1;
}
int Kp=Top;
for(int i=1;i<=Kp;i++)St[++Top]=-St[i];//正为dfs中的入度，负为出度
if(!vis[1])St[++Top]=1,St[++Top]=-1;//判断根节点是否加入虚树
```

使用虚树

按照欧拉序DP即可（利用栈模拟DFS的过程）。

```
sort(St+1,St+Top+1,Cmp);
for(int i=1;i<=Top;i++){
    if(St[i]>0){Pop[++Cts]=St[i];continue;}//入栈
    int Now=Pop[Cts],fa=Pop[--Cts];
    if(Now^1)ft[fa]+=min(1ll*m[Now],ft[Now]);//DP
    else {
        printf("%lld\n",ft[1]),vis[1]=ft[1]=0;
        break;
    }
    vis[Now]=ft[Now]=0;
}
```

完整代码

```
#include <cstdio>
#include <cctype>
#include <algorithm>
using namespace std;

char tc(){
    static char tr[10000],*A=tr,*B=tr;
    return A==B&&(B=(A=tr)+fread(tr,1,10000,stdin),A==B)?EOF:*A++;
}
void read(int &x){
    char c;for(;!isdigit(c=tc()););
    for(x=c-48;isdigit(c=tc());x=(x<<1)+(x<<3)+c-48);
}

const int S=2e5+5e4+5;
int head[S],nxt[S<<1],to[S<<1],w[S<<1],Cnt;
void Add(int x,int y,int c){
    to[++Cnt]=y,w[Cnt]=c,nxt[Cnt]=head[x],head[x]=Cnt;
    to[++Cnt]=x,w[Cnt]=c,nxt[Cnt]=head[y],head[y]=Cnt;
}
```

```
long long ft[S];
int N,M,x,y,c,tot,P,Dfn,Top,Cts,Pop[S];
int vis[S],St[S<<2],f[S][27],m[S],Gi[S],Gt[S],d[S];
void dfs(int x,int fa,int dep){
    Gi[x]=++Dfn,m[x]=min(dep,m[fa]);
    d[x]=d[fa]+1,f[x][0]=fa;
    for(int i=head[x];i;i=nxt[i])
        if(to[i]^fa)dfs(to[i],x,w[i]);
    Gt[x]=++Dfn;
}
void init(void){
    m[1]=m[0]=2e9,dfs(1,0,2e9);
    for(int i=1;(1<<i)<=N;i++)
        for(int j=1;j<=N;j++){
            f[j][i]=f[f[j][i-1]][i-1];
        }
    return ;
}
int Lca(int x,int y){
    if(d[x]<d[y])swap(x,y);
    for(int i=26;~i;i--)if(d[f[x][i]]>=d[y])x=f[x][i];
    if(x==y)return x;
    for(int i=26;~i;i--)if(f[x][i]^f[y][i])x=f[x][i],y=f[y][i];
    return f[x][0];
}
int Cmp(int x,int y){
    return (x>0?Gi[x]:Gt[-x])<(y>0?Gi[y]:Gt[-y]);
}
int main()
{
    read(N);
    for(int i=1;i<N;i++)read(x),read(y),read(c),Add(x,y,c);
    for(init(),read(M);M;M--){
        int K;read(K),Top=0;
        for(int i=1;i<=K;i++)
            read(x),vis[St[++Top]=x]=1,ft[x]=m[x];
        sort(St+1,St+Top+1,Cmp);
        for(int i=2;i<=Top;i++){
            int fa=Lca(St[i],St[i-1]);
            if(!vis[fa])vis[St[++Top]=fa]=1;
        }
        int Kp=Top;
        for(int i=1;i<=Kp;i++)St[++Top]=-St[i];
        if(!vis[1])St[++Top]=1,St[++Top]=-1;
        sort(St+1,St+Top+1,Cmp);
        for(int i=1;i<=Top;i++){
            if(St[i]>0){Pop[++Cts]=St[i];continue;}
            int Now=Pop[Cts],fa=Pop[--Cts];
            if(Now^1)ft[fa]+=min(1ll*m[Now],ft[Now]);
            else {
                printf("%lld\n",ft[1]),vis[1]=ft[1]=0;
                break;
            }
```

```
            vis[Now]=ft[Now]=0;
        }
    }
    return 0;
}
```