// 朱刘算法求最小树形图

// 复杂度O(nm)

// 注意不定根的最小树形图可以建一个虚根转化成定根的

namespace zhuliu

{

LL in[MAXN+48];int pre[MAXN+48],bel[MAXN+48];int cirind,tot,i,u,v;

int visited[MAXN+48];

inline LL solve(int root)

{

tot=n;LL res=0;

while (true)

{

for (i=1;i<=tot;i++) in[i]=LINF;

for (i=1;i<=e;i++)

if (edge[i].x!=edge[i].y && in[edge[i].y]>edge[i].c)

{

in[edge[i].y]=edge[i].c;

pre[edge[i].y]=edge[i].x;

}

for (i=1;i<=tot;i++) if (i!=root && in[i]==LINF) return -1;

in[root]=0;cirind=0;

memset(visited,0,sizeof(visited));

memset(bel,0,sizeof(bel));

for (i=1;i<=tot;i++)

{

res+=in[i];

v=i;while (visited[v]!=i && v!=root && !bel[v]) visited[v]=i,v=pre[v];

if (v!=root && !bel[v])

{

++cirind;u=pre[v];

while (u!=v) bel[u]=cirind,u=pre[u];

bel[u]=cirind;

}

}

if (!cirind) break;

for (i=1;i<=tot;i++) if (!bel[i]) bel[i]=++cirind;

for (i=1;i<=e;i++)

{

v=edge[i].y;

edge[i].x=bel[edge[i].x];edge[i].y=bel[edge[i].y];

if (edge[i].x!=edge[i].y) edge[i].c-=in[v];

}

tot=cirind;root=bel[root];

}

return res;

}

}