/\*

优先队列维护已染色的中DFS序最大与最小的点，每次查询分别找到最大的点与最小的点，这两点的LCA即为答案

\*/

#include<bits/stdc++.h>

#define P pair<int,int>

using namespace std;

int n,m,s,num=0,head[1000001],deep[1000001],f[1000001][23];

int pos[1000001];

int logT[500001];

bool vis[1000001];

int a1,a2;

inline int read(){

int x=0,f=1;

char ch=getchar();

while(ch<'0'||ch>'9'){

if(ch=='-')

f=-1;

ch=getchar();

}

while(ch>='0'&&ch<='9'){

x=(x<<1)+(x<<3)+(ch^48);

ch=getchar();

}

return x\*f;

}

struct edg{

int next,to;

}edge[1000001];

void edge\_add(int u,int v)

{

num++;

edge[num].next=head[u];edge[num].to=v;head[u]=num;

edge[++num].next=head[v];edge[num].to=u;head[v]=num;

}

int stack\_[1000001]={0};

void dfs(int u,int father)

{

int top=-1;

stack\_[++top]=u;

while(top!=-1)

{

int u=stack\_[top];

top--;

pos[u]=++pos[0];

deep[u]=deep[f[u][0]]+1;

for(int i=1;i<=logT[deep[u]];i++)

{

f[u][i]=f[f[u][i-1]][i-1];

}

for(int i=head[u];i;i=edge[i].next)

{

int v=edge[i].to;

if(v==f[u][0])continue;

f[v][0]=u;

stack\_[++top]=v;

}

}

}

int lca(int x,int y)

{

if(deep[x]<deep[y])swap(x,y);

for(int i=logT[deep[x]-deep[y]];i>=0;i--)

{

if(deep[f[x][i]]>=deep[y])x=f[x][i];

if(x==y)return x;

}

for(int i=logT[deep[x]];i>=0;i--)

{

if(f[x][i]!=f[y][i])

{

x=f[x][i];y=f[y][i];

}

}

return f[x][0];

}

int main(){

int T;

cin>>T;

for(int i=1;i<=200000;i++)logT[i]=log2(i)+1;

while(T--)

{

num=0;

scanf("%d",&n);

memset(head,0,sizeof(head));

pos[0]=0;

memset(vis,0,sizeof(vis));

memset(f,0,sizeof(f));

memset(pos,0,sizeof(pos));

for(int i=1;i<n;i++)

{

a1=read()+1;

a2=read()+1;

edge\_add(a1,a2);

}

dfs(1,0);

char ch;

int a;

scanf("%d",&m);

priority\_queue<P,vector<P>,less<P> > minn;

priority\_queue<P,vector<P>,greater<P> >maxx;

for(int i=1;i<=m;i++)

{

ch=0;

while(!(ch=='-'||ch=='+'))ch=getchar();

a=read()+1;

if(ch=='+')

{

minn.push(P(pos[a],a));

maxx.push(P(pos[a],a));

vis[a]=true;

}

else vis[a]=false;

if(minn.empty()&&maxx.empty())

{

cout<<-1<<endl;

continue;

}

P l=minn.top();

P r=maxx.top();

while(!vis[l.second])

{

minn.pop();

if(minn.empty())break;

l=minn.top();

}

while(!vis[r.second])

{

maxx.pop();

if(maxx.empty())break;

r=maxx.top();

}

if(minn.empty()&&maxx.empty())

{

cout<<-1<<endl;

continue;

}

printf("%d\n",lca(l.second,r.second)-1);

}

}

}