D. Maximum Distributed Tree

time limit per test

2 seconds

memory limit per test

256 megabytes

input

standard input

output

standard output

You are given a tree that consists of nn nodes. You should label each of its n−1n−1 edges with an integer in such way that satisfies the following conditions:

* each integer must be greater than 00;
* the product of all n−1n−1 numbers should be equal to kk;
* the number of 11-s among all n−1n−1 integers must be minimum possible.

Let's define f(u,v)f(u,v) as the sum of the numbers on the simple path from node uu to node vv. Also, let ∑i=1n−1∑j=i+1nf(i,j)∑i=1n−1∑j=i+1nf(i,j) be a *distribution index* of the tree.

Find the maximum possible distribution index you can get. Since answer can be too large, print it modulo 109+7109+7.

In this problem, since the number kk can be large, the result of the prime factorization of kk is given instead.

**Input**

The first line contains one integer tt (1≤t≤1001≤t≤100) — the number of test cases.

The first line of each test case contains a single integer nn (2≤n≤1052≤n≤105) — the number of nodes in the tree.

Each of the next n−1n−1 lines describes an edge: the ii-th line contains two integers uiui and vivi (1≤ui,vi≤n1≤ui,vi≤n; ui≠viui≠vi) — indices of vertices connected by the ii-th edge.

Next line contains a single integer mm (1≤m≤6⋅1041≤m≤6⋅104) — the number of prime factors of kk.

Next line contains mm prime numbers p1,p2,…,pmp1,p2,…,pm (2≤pi<6⋅1042≤pi<6⋅104) such that k=p1⋅p2⋅…⋅pmk=p1⋅p2⋅…⋅pm.

It is guaranteed that the sum of nn over all test cases doesn't exceed 105105, the sum of mm over all test cases doesn't exceed 6⋅1046⋅104, and the given edges for each test cases form a tree.

**Output**

Print the maximum distribution index you can get. Since answer can be too large, print it modulo 109+7109+7.

**Example**

**input**

**Copy**

3

4

1 2

2 3

3 4

2

2 2

4

3 4

1 3

3 2

2

3 2

7

6 1

2 3

4 6

7 3

5 1

3 6

4

7 5 13 3

**output**

**Copy**

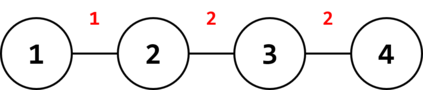
17

18

286

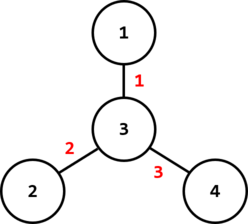
**Note**

In the first test case, one of the optimal ways is on the following image:



In this case, f(1,2)=1f(1,2)=1, f(1,3)=3f(1,3)=3, f(1,4)=5f(1,4)=5, f(2,3)=2f(2,3)=2, f(2,4)=4f(2,4)=4, f(3,4)=2f(3,4)=2, so the sum of these 66 numbers is 1717.

In the second test case, one of the optimal ways is on the following image:



In this case, f(1,2)=3f(1,2)=3, f(1,3)=1f(1,3)=1, f(1,4)=4f(1,4)=4, f(2,3)=2f(2,3)=2, f(2,4)=5f(2,4)=5, f(3,4)=3f(3,4)=3, so the sum of these 66 numbers is 1818.

1. **#include** <bits/stdc++.h>
2. **using** **namespace** std;
3. **typedef** **long** **long** ll;
4. **const** **int** N=1000005;
5. ll sz[N],a[N],ans,mod=ll(1e9+7),b[N];
6. **vector<int>** G[N];
7. **void** dfs(**int** p,**int** fa)
8. {
9. sz[p]=1;
10. **for**(**int** i=0;i<G[p].size();i++) **if**(G[p][i]!=fa) dfs(G[p][i],p),sz[p]+=sz[G[p][i]];
11. }
12. **int** main()
13. {
14. **int** t;
15. cin>>t;
16. **while**(t--)
17. {
18. **int** n,m,x,y;
19. cin>>n;
20. ans=0;
21. **for**(**int** i=1;i<=n;i++) G[i].clear(),a[i]=1;
22. **for**(**int** i=1;i<n;i++) cin>>x>>y,G[x].push\_back(y),G[y].push\_back(x);
23. dfs(1,0);
24. cin>>m;
25. **for**(**int** i=1;i<=m;i++) cin>>a[i];
26. m=max(n-1,m);
27. sort(a+1,a+1+m);
28. **if**(m>n-1) **for**(**int** i=n;i<=m;i++) a[n-1]=a[n-1]%mod\*a[i]%mod,a[n-1]%=mod;
29. **for**(**int** i=1;i<n;i++) b[i]=(n-sz[i+1])\*sz[i+1];
30. sort(b+1,b+n);
31. **for**(**int** i=1;i<n;i++) ans=ans%mod+(b[i]%mod\*a[i]%mod)%mod,ans%=mod;
32. cout<<ans%mod<<**"\n"**;
33. }
34. }