B. Appleman and Tree

time limit per test

2 seconds

memory limit per test

256 megabytes

input

standard input

output

standard output

Appleman has a tree with *n* vertices. Some of the vertices (at least one) are colored black and other vertices are colored white.

Consider a set consisting of *k* (0 ≤ *k* < *n*) edges of Appleman's tree. If Appleman deletes these edges from the tree, then it will split into (*k* + 1) parts. Note, that each part will be a tree with colored vertices.

Now Appleman wonders, what is the number of sets splitting the tree in such a way that each resulting part will have exactly one black vertex? Find this number modulo 1000000007 (109 + 7).

**Input**

The first line contains an integer *n* (2  ≤ *n* ≤ 105) — the number of tree vertices.

The second line contains the description of the tree: *n* - 1 integers *p*0, *p*1, ..., *pn*- 2 (0 ≤ *pi* ≤ *i*). Where *pi* means that there is an edge connecting vertex (*i* + 1) of the tree and vertex *pi*. Consider tree vertices are numbered from 0 to *n* - 1.

The third line contains the description of the colors of the vertices: *n* integers *x*0, *x*1, ..., *xn*- 1 (*xi* is either 0 or 1). If *xi* is equal to 1, vertex *i* is colored black. Otherwise, vertex *i* is colored white.

**Output**

Output a single integer — the number of ways to split the tree modulo 1000000007 (109 + 7).

**Examples**

**input**

**Copy**

3  
0 0  
0 1 1

**output**

**Copy**

2

**input**

**Copy**

6  
0 1 1 0 4  
1 1 0 0 1 0

**output**

**Copy**

1

**input**

**Copy**

10  
0 1 2 1 4 4 4 0 8  
0 0 0 1 0 1 1 0 0 1

**output**

**Copy**

27

States

1. No. of black vectices € [0,1].
2. Vertex v € [1,N].

Approach:

So here we need to think for the subtree at node v dp[v][0] -> number of ways to obtain 0 black vertices in the subtree at node v, dp[v][1] -> number of ways to obtain 1 black vertices in the subtree at node v. So our final answer is dp[1][0].

Now let v[1]….v[x] be children of v.

#include<bits/stdc++.h>

#define int long long int

#define pb push\_back

#define ppb pop\_back

#define pf push\_front

#define ppf pop\_front

#define F first

#define S second

#define inf 1e18

#define vec vector<int>

#define pii pair<int,int>

using namespace std;

vector<int> adj[100001];

int col[100001], dp[100001][2];

int mod = 1e9 + 7;

void dfs(int v, int par)

{

dp[v][0] = (!col[v]);

dp[v][1] = col[v];

for (int to : adj[v])

{

if (to == par)

continue;

dfs(to, v);

int white = dp[to][0],

black = dp[to][1];

dp[v][1] = (dp[v][0] \* black) % mod + (dp[v][1] \* white) % mod + (dp[v][1] \* black) % mod;

dp[v][1] %= mod;

dp[v][0] = (dp[v][0] \* white) % mod + (dp[v][0] \* black) % mod;

dp[v][0] %= mod;

}

}

int32\_t main()

{

#ifndef ONLINE\_JUDGE

// for getting input from input.txt

freopen("input.txt", "r", stdin);

// for writting output to output.txt

freopen("output.txt", "w", stdout);

#endif

ios\_base::sync\_with\_stdio(NULL);

cin.tie(NULL);

cout.tie(NULL);

int tt = 1;

//cin >> tt;

while (tt--)

{

int n;

cin >> n;

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

{

int x;

cin >> x;

adj[i].pb(x);

adj[x].pb(i);

}

for (int i = 0; i < n; i++)

cin >> col[i];

dfs(0, -1);

cout << dp[0][1];

}

}