```cpp
#include          <bits/stdc++.h>
#pragma GCC       optimize ("Ofast")
#pragma GCC       optimize ("unroll-loops")
#pragma GCC       target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,tune=native")

#define IOS       ios_base::sync_with_stdio(false); cin.tie (nullptr)
#define PREC      cout.precision (10); cout << fixed
#define x         first
#define y         second

using namespace std;

const int N = (int) 2e5 + 10;
const int Mod = (int) 1e9 + 7;

int n, V;
vector <vector <int> > Adj;
long long f[N], g[N], dp[N];
vector < vector <long long> > pref, suff;

long long mul (long long a, long long b) {
    a *= b;
    a %= Mod;
    return a;
}

void read() {
    cin >> n;
    Adj.assign(n, vector <int>());
    pref.assign(n, vector <long long> ());
    suff.assign(n, vector <long long> ());

    for (int i = 0; i < n; ++i)
        dp[i] = f[i] = g[i] = 0;

    for (int i = 1; i < n; ++i) {
        int u = i, v;
        cin >> v; --v;
        Adj[u].emplace_back(v);
        Adj[v].emplace_back(u);
    }
    for (int i = 0; i < n; ++i)
        dp[i] = f[i] = g[i] = 0;
}

void rootAt (int u, int pr) {
    for (int i = 0; i < (int) Adj[u].size(); ++i) {
        int v = Adj[u][i];
        if (v == pr)
            Adj[u].erase (Adj[u].begin() + i);
    }
    for (int i = 0; i < (int) Adj[u].size(); ++i) {
        int v = Adj[u][i];
        rootAt (v, u);
    }
}

void dfsF(int u) {
    for (auto v : Adj[u])
        dfsF (v);

    long long fval = 1;
    for (int i = 0; i < (int) Adj[u].size(); ++i) {
        int v = Adj[u][i];
        fval = mul(fval, 1 + f[v]);
    }
    f[u] = fval;

    pref[u].assign((int)Adj[u].size(), 0);
    suff[u].assign((int)Adj[u].size(), 0);

    for (int i = 0; i < (int)Adj[u].size(); ++i) {
        int v = Adj[u][i];
```

```
        if (i == 0)
            pref[u][i] = 1 + f[v];
        else pref[u][i] = mul (pref[u][i - 1], (1 + f[v]));
    }

    for (int i = (int)Adj[u].size() - 1; i >= 0; --i) {
        int v = Adj[u][i];
        if (i == (int)Adj[u].size() - 1)
            suff[u][i] = 1 + f[v];
        else suff[u][i] = mul (suff[u][i + 1], (1 + f[v]));
    }
}

void dfsG(int u, int pr) {
    if (pr == -1) g[u] = 1;

    for (int i = 0; i < (int) Adj[u].size(); ++i) {
        int v = Adj[u][i];
        long long gval = 1;
        if (i != 0)
            gval = mul (gval, pref[u][i - 1]);
        if (i != (int) Adj[u].size() - 1)
            gval = mul (gval, suff[u][i + 1]);
        gval = mul (gval, g[u]);
        g[v] = gval + 1;
    }

    dp[u] = (f[u] * g[u]) % Mod;

    for (auto v : Adj[u])
        dfsG (v, u);
}

void solve() {
    for (int v = 0; v < n; ++v)
        cout << dp[v] << ' ';
    cout << '\n';
}

signed main() {
    IOS; PREC;

    read();
    rootAt(0, -1);
    dfsF(0);
    dfsG(0, -1);
    solve();
    return EXIT_SUCCESS;
}
```