```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <sstream>
#include "bintree_eda.h"

using namespace std;


bintree<int> crearArbol(vector<int > const &pre, vector<int > const &in, int iniP, int
finP, int iniI, int finI) {

    if (iniP > finP) {

        return {};
    }
    else if (iniP == finP || iniI == finI) return { {}, pre[iniP], {} };
    else {

        int raiz = pre[iniP];

        bool e = false;
        unsigned int i = iniI;

        while (!e && i < finI) {

            if (raiz == in[i]) e = true;
            else i++;
        }

        int finP2 = iniP;

        for (int j = iniI; j < i; j++) {

            finP2++;
        }

        iniP++;

        bintree<int> izq = crearArbol(pre, in, iniP, finP2, iniI, i - 1);

        bintree<int> der = crearArbol(pre, in, finP2 + 1, finP, i + 1, finI);

        return { izq, raiz, der };
    }
}



bool resuelveCaso() {

    vector<int> preorden;
    vector<int> inorden;
```

```
    int num;

    string pre, in;

    getline(cin, pre);

    if (!cin) return false;

    getline(cin, in);

    stringstream c(pre);

    while (c >> num) {

        preorden.push_back(num);
    }

    stringstream i(in);

    while (i >> num) {

        inorden.push_back(num);
    }

    bintree<int> arbol = crearArbol(preorden, inorden, 0, preorden.size() - 1, 0,
inorden.size() - 1);

    vector<int > postorden = arbol.postorder();

    for (unsigned int i = 0; i < postorden.size(); i++) {

        cout << postorden[i] << " ";
    }

    cout << endl;

    return true;
}


int main() {
    // Para la entrada por fichero.
    // Comentar para acepta el reto
#ifndef DOMJUDGE
    ifstream in("datos.txt");
    auto cinbuf = std::cin.rdbuf(in.rdbuf()); //save old buf and redirect std::cin to
casos.txt
#endif


    while(resuelveCaso()){}


    // Para restablecer entrada. Comentar para acepta el reto
```

```
#ifndef DOMJUDGE // para dejar todo como estaba al principio
    std::cin.rdbuf(cinbuf);
    system("PAUSE");
#endif

    return 0;
}
```