```cpp
// ej  Arboles bien codificados


#include <iostream>
#include <fstream>

#include "bintree_eda.h"

using namespace std;

struct tSol{
    bool es;
    int ceros;
};

tSol bienCodificado(bintree<int> const& arbol) {

    if (arbol.empty()) return {true, 0};
    else {

        tSol izq = bienCodificado(arbol.left());
        tSol der = bienCodificado(arbol.right());

        if (izq.es && der.es) {

            tSol sol;

            bool esta;

            int dif0;

            dif0 = izq.ceros - der.ceros;

            if (dif0 < -1 || dif0 > 1) {
                esta = false;
            }
            else {
                esta = true;
            }

            if (!esta) return { false, 0};
            else {

                sol.es = true;

                if (arbol.root() == 0) {

                    sol.ceros = izq.ceros + der.ceros + 1;
                }
                else {

                    sol.ceros = izq.ceros + der.ceros;
                }
```

```
                return sol;
            }
        }
        else {

            return {false, 0};
        }
    }
}


void resuelveCaso() {

    bintree<int> arbol = leerArbol(-1);


    tSol sol = bienCodificado(arbol);

    if (sol.es) cout << "SI\n";
    else cout << "NO\n";
}


int main() {
    // ajustes para que cin extraiga directamente de un fichero
#ifndef DOMJUDGE
    std::ifstream in("datos.txt");
    auto cinbuf = std::cin.rdbuf(in.rdbuf());
#endif

    int casos = 0;

    cin >> casos;

    for (int i = 0; i < casos; i++) {
        resuelveCaso();
    }

    // para dejar todo como estaba al principio
#ifndef DOMJUDGE
    std::cin.rdbuf(cinbuf);
    system("PAUSE");
#endif
    return 0;
}
```