```cpp
#include "bintree.h"
#include <iostream>

using namespace std;

template <class T>
void postorden(const bintree<T> &A, const typename bintree<T>::node &v)
{
    if (!v.null())
    {
        postorden(A,v.left());
        postorden(A,v.right());
        cout << *v;
    }
}

string postorden_pre_ino(string pre,string ino)
{
    string post="";
    string aux1,aux2;

    int raiz=ino.find_first_of(pre[0]);

    if(raiz!=0) //SI HAY ARBOL IZQUIERDO
    {
        aux1=pre.substr(1,raiz);
        aux2=ino.substr(0,raiz);
        //cout << "ARBOL IZQUIERDO ->>>> " << aux1 << "<->" << aux2 << endl;
        post+=postorden_pre_ino(aux1,aux2);
    }
    if(raiz!=ino.length()-1) //SI HAY ARBOL DERECHO
    {
        aux1=pre.substr(raiz+1,pre.length()-1);
        aux2=ino.substr(raiz+1,ino.length()-1);
        //cout << "ARBOL DERECHO ->>>> " << aux1 << "<->" << aux2 << endl;
        post+=postorden_pre_ino(aux1,aux2);
    }

    post+=pre[0];

    return post;

}


int main()
{
    bintree<char> arb('G');

    //RAMA IZQUIERDA

    arb.insert_left(arb.root(), 'E');
    bintree<char>::node node_aux = arb.root().left();

    arb.insert_left(node_aux, 'A');

    node_aux = node_aux.left();

    arb.insert_left(node_aux, 'I');
    arb.insert_right(node_aux, 'B');

    //RAMA DERECHA

    arb.insert_right(arb.root(), 'M');
    node_aux = arb.root().right();

    arb.insert_left(node_aux, 'C');
    arb.insert_right(node_aux, 'K');

    node_aux = node_aux.left();
    arb.insert_left(node_aux, 'L');
    arb.insert_right(node_aux, 'F');

    node_aux = node_aux.left();
    arb.insert_right(node_aux, 'D');

    node_aux = arb.root().right().right();
    arb.insert_right(node_aux, 'J');

    node_aux = node_aux.right();
    arb.insert_left(node_aux, 'H');

    postorden(arb,arb.root());
    cout << endl;
    string preorder=postorden_pre_ino("GEAIBMCLDFKJH","IABEGLDCFMKHJ");
    cout << preorder << endl;
}
```