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

#include <cmath>
#include <queue>
#include <iomanip>

using namespace std;



template <class T>
void MostrarArbol(const bintree<T> &A, typename bintree<T>::node root)
{
    queue<typename bintree<T>::node> colaNodos;
    int totalNodos = A.size();
    int techo = log2(totalNodos + 1);
    colaNodos.push(root);
    int pot = 0;
    while (colaNodos.size() > 0)
    {
        int niveles = colaNodos.size();
        while (niveles > 0)
        {
            typename bintree<T>::node nodoAux = colaNodos.front();
            colaNodos.pop();
            cout << setw((niveles == pow(2, pot)) ? pow(2, (techo - pot)) : pow(2,
(techo - pot + 1)));
            cout << *nodoAux;
            if (!nodoAux.left().null())
                colaNodos.push(nodoAux.left());
            if (!nodoAux.right().null())
                colaNodos.push(nodoAux.right());
            niveles--;
        }
        pot++;
        cout << endl;
    }
}

int profundidad(bintree<int>::node v)
{
    int prof=0;
    typename bintree<int>::node aux=v;
    while(!aux.parent().null()){
        prof++;
        aux=aux.parent();
    }
    return prof;
}


bintree<int>::node ancestro_comun(bintree<int>::node n1, bintree<int>::node n2)
{
    static int prof_node1=profundidad(n1), prof_node2=profundidad(n2);
    if(prof_node1==-1 && prof_node2==-1)
    {
        prof_node1=profundidad(n1);
        prof_node2=profundidad(n2);
    }
```

```cpp
    if(prof_node1==prof_node2)
    {
        if(n1==n2)
        {
            prof_node1=-1; //RESET
            prof_node2=-1;
            return n1; //DA IGUAL CUAL DEVOLVER PORQUE AMBOS SON IGUALES
        }else
        {
            prof_node1--;
            prof_node2--;
            return ancestro_comun(n1.parent(),n2.parent());
        }
    }else
    {
        if(prof_node1 > prof_node2)
        {
            prof_node1--;
            return ancestro_comun(n1.parent(),n2);
        }else
        {
            prof_node2--;
            return ancestro_comun(n1,n2.parent());
        }
    }

}

int main()
{
    bintree<int> arb2(0);
    arb2.insert_left(arb2.root(), 1);
    arb2.insert_right(arb2.root(), 2);

    bintree<int>::node aux2 = arb2.root().left();
    arb2.insert_left(aux2, 3);
    arb2.insert_right(aux2, 4);
    bintree<int>::node aux= aux2.right();

    aux2 = arb2.root().right();
    arb2.insert_left(aux2, 5);
    arb2.insert_right(aux2, 6);

    bintree<int>::node aux3= aux2;


    cout << "----------ARBOL ORIGINAL----------" << endl;
    MostrarArbol(arb2,arb2.root());

    cout << "----------ANCESTRO COMUN----------" << endl;
    cout << *aux << *aux3 << endl;
    cout << *(ancestro_comun(aux,aux3)) << endl;


}
```