```cpp
#include "bintree.h"

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

#include <iostream>

using namespace std;

template <class T>
bool esHoja(const bintree<T> &A, const typename bintree<T>::node &v)
{
    return ( v.left().null() && v.right().null() );
}

template <class T>
int profundidad(const bintree<T> &A, const typename bintree<T>::node &v)
{
    int prof=0;
    typename bintree<T>::node aux=v;
    while(A.root()!=aux){
        prof++;
        aux=aux.parent();
    }
    return prof;
}

template <class T>
typename bintree<T>::node getHojaProfunda(const bintree<T> &A, typename
bintree<T>::node v){
    static int max_profundidad=0;
    static typename bintree<T>::node hoja_prof=A.root();

    if(v==A.root()) //ES DECIR HEMOS LLAMADO DE NUEVO A LA FUNCION CON OTRO ARBOL
    {
        max_profundidad=0;
        hoja_prof=A.root();
    } //HAY QUE RESETEAR LOS DATOS

    if(!v.null())
    {
        if(esHoja(A,v))
        {
            int prof=profundidad(A,v);
            if(prof>max_profundidad)
            {
                hoja_prof=v;
                max_profundidad=prof;
            }
        }
        getHojaProfunda(A,v.left());
        getHojaProfunda(A,v.right());
    }
    return hoja_prof;
}

template <class T>
void MostrarArbol(const bintree<T> &A,typename bintree<T>::node root){
    queue<typename bintree<T>::node> colaNodos;
```

```
60      int totalNodos=A.size();
61      int techo=log2(totalNodos+1);
62      colaNodos.push(root);
63      int pot=0;
64      while(colaNodos.size() > 0){
65              int niveles = colaNodos.size();
66              while(niveles > 0){
67                  typename bintree<T>::node nodoAux = colaNodos.front();
68                  colaNodos.pop();
69                  cout<<setw((niveles==pow(2,pot))?pow(2, (techo-pot)):pow(2, (techo-
pot+1)));
70                  cout<<*nodoAux;
71                  if(!nodoAux.left().null()) colaNodos.push(nodoAux.left());
72                  if(!nodoAux.right().null()) colaNodos.push(nodoAux.right());
73                  niveles--;
74              }
75              pot++;
76              cout << endl;
77      }
78  }
79
80  int main()
81  {
82      bintree<int> arb(5);
83      arb.insert_left(arb.root(), 22);
84
85      bintree<int>::node n = arb.root().left();
86
87      arb.insert_left(n, 481);
88      arb.insert_right(n, 125);
89
90      bintree<int>::node mas_prfu1=getHojaProfunda(arb,arb.root());
91      MostrarArbol(arb,arb.root());
92      cout << *mas_prfu1 << endl;
93
94      bintree<int> aux(1);
95      aux.insert_right(aux.root(),128);
96      aux.insert_right(aux.root().right(),209);
97
98      bintree<int>::node mas_prfu2=getHojaProfunda(aux,aux.root());
99      MostrarArbol(aux,aux.root());
100     cout << *mas_prfu2 << endl;
101 }
```