```cpp
#include "bintree.h"

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

#include <iostream>

using namespace std;

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>
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 main()
{
    bintree<int> arb(5);
    arb.insert_left(arb.root(),29);
    arb.insert_left(arb.root().left(),30);
    arb.insert_left(arb.root().left().left(),30);

    MostrarArbol(arb,arb.root());

    cout << profundidad(arb,arb.root().left().left().left()) << endl;
}
```