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

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

using namespace std;

void Expr_post(string expr, int pos, bintree<char> &A, bintree<char>::node aux_node)
{
    if (pos >= 0)
    {
        A.insert_right(aux_node, expr[pos]);
        pos--;
        A.insert_left(aux_node, expr[pos]);

        pos--;
        if (*aux_node.right() == '+' || *aux_node.right() == '-' || *aux_node.right() == '*' || *aux_node.right() == '/')
        {
            Expr_post(expr, pos, A, aux_node.right());
            pos--;
        }
        if (*aux_node.left() == '+' || *aux_node.left() == '-' || *aux_node.left() == '*' || *aux_node.left() == '/')
        {
            Expr_post(expr, pos, A, aux_node.left());
        }
    }
}
bintree<char> Expr_post(string expr)
{
    bintree<char> resultado(expr[expr.size() - 1]);
    int i = expr.size() - 2;
    bintree<char>::node aux_node = resultado.root();
    if (i >= 0)
    {
        resultado.insert_right(aux_node, expr[i]);

        i--;
        resultado.insert_left(aux_node, expr[i]);

        i--;
        if (*aux_node.right() == '+' || *aux_node.right() == '-' || *aux_node.right() == '*' || *aux_node.right() == '/')
        {
            Expr_post(expr, i, resultado, aux_node.right());
            i--;
        }
        if (*aux_node.left() == '+' || *aux_node.left() == '-' || *aux_node.left() == '*' || *aux_node.left() == '/')
        {
            Expr_post(expr, i, resultado, aux_node.left());
        }
    }
    return resultado;
}

template <class T>
```

```cpp
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;
    }
}


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<char> aux = Expr_post("ab*b*d/e+");
    MostrarArbol(aux, aux.root());
    postorden(aux, aux.root());
    cout << endl;
}
```