PROGRAMA PROFESIONAL

Ciencias de la computación

Tarea: Árboles

Curso:

Programación Competitiva

Integrantes:

- Angel Josue Loayza Huarachi
- Luis Fredy Huachaca Vargas

CCOMP 6 - 1.2

2022

1. **Link de GitHub:**

https://github.com/angel452/Competitiva

2. **Codigo:**

- **source.cpp**

```cpp
#include "tree.h"

void printTest1()
{
    TreeNode *root = new TreeNode(10);

    root->right = new TreeNode(11);
    root->left = new TreeNode(12);

    root->right->right = new TreeNode(13);
    root->right->left = new TreeNode(14);

    root->left->right = new TreeNode(15);
    root->left->left = new TreeNode(16);

    root->right->right->right = new TreeNode(17);
    root->right->right->left = new TreeNode(18);

    root->right->left->right = new TreeNode(19);
    root->right->left->left = new TreeNode(20);

    root->left->right->right = new TreeNode(21);
    root->left->right->left = new TreeNode(22);

    root->left->left->right = new TreeNode(23);
    root->left->left->left = new TreeNode(24);

    cout << endl << "------ Vertical 1 --------------- "<< endl;
    printNodesV1(root,0);

    cout << endl << "------ Vertical 2 --------------- "<< endl;
    printNodesV2(root,3); //3 = profundidad del arbol
    //cout << endl;

    cout << endl << "------ Horizontal 1 --------------- "<< endl;
    printNodesH1(root);

    cout << endl << "------ Horizontal 2 --------------- "<< endl;
    printNodesH2(root);

}

int main(void)
{
    //print_tree_test_1();
    printTest1();

    return 0;
}
```

- **tree.h**

```cpp
#include <iostream>
#include <string>
#include <vector>
#include "math.h"
using namespace std;

class TreeNode
{
public: // dodging encapsulation...
    int key;
    TreeNode *left;
    TreeNode *right;

    // constructor
    TreeNode(int _key = 0)
    {
        key = _key;
        left = NULL;
        right = NULL;
    }
};

void printNodesV1(TreeNode *head, int n)
{
    //cout << "Entra funcion" << endl;
    if(head == NULL)
    {
        return;
    }

    printNodesV1(head->right, n+1);

    for(int i = 0; i < n; i++)
    {
        cout << "|-----";
    }

    //cout << "a" << endl;
    cout << head->key << endl;

    printNodesV1(head->left, n+1);
}


void printNodesV2(TreeNode *head, int n)
{
    //cout << "Entra funcion" << endl;
    if(head == NULL)
    {
        return;
    }
    printNodesV2(head->left, n-1);

    for(int i = 0; i < n; i++)
    {
        cout << "-----|";
    }
    //cout << endl;

    //cout << "a" << endl;
    cout << head->key <<endl;

    printNodesV2(head->right, n-1);
}


void contarProfundidad(TreeNode *head) //falta
{
    TreeNode *puntero = new TreeNode();
    puntero = head;

    while(puntero != NULL)
    {
        cout << puntero->key << " ";
        puntero = puntero->left;
    }
}

void nodeToVector(TreeNode *head, vector<int> &vec1)
{
    if(head!=NULL)
    {
        nodeToVector(head->left, vec1);
        //cout << head->key << " - ";
        vec1.push_back(head->key);
        nodeToVector(head->right, vec1);
    }
}


void printH1Aux(vector<int> vec1, int n, int printPerRow)
{
    if(n < 1)
    {
        return;
    }

    int aux = n;
    int aux2;

    //cout << endl << "iteraciones: " << pow(2,printPerRow)-1 << endl;

    aux = aux/2;
    aux2 = aux;

    // -------------- Para el espaciado -------------------
    //cout << vec1[aux] << "  ";
    for (int i = 0; i < aux; i++)
    {
        cout << "  ";
    }
    cout << vec1[aux];
    // ----------------------------------------------------

    for(int i = 0; i < pow(2,printPerRow)-1; i++ )
    {
        aux2 = aux2 + ((aux*2)+2);
        if(aux2 < vec1.size())
        {
            // -------------- Para el espaciado --------------------
            //cout << vec1[aux2] << "  ";
            for (int i = 0; i < ((aux*2)+2)-1; i++)
            {
                cout << "  ";
            }
            cout << vec1[aux2];
            // ----------------------------------------------------
        }
    }
    cout << endl;

    printH1Aux(vec1,n/2,printPerRow+1);
}

void printNodesH1(TreeNode *head)
{
    vector<int> vec1;
    nodeToVector(head,vec1); //guardamos los nodos en un vector

    /*
    cout << "Vector out: " << endl;
    for(int i = 0; i < vec1.size(); i++ )
    {
        cout << vec1[i] <<  " - " ;
    }
    cout << endl;
    */

    printH1Aux( vec1, vec1.size(), 0);

}


void printH2Aux(vector<int> vec1, int n, int printPerRow)
{
    if(n == 4) //+1 de la profundidad
    {
        //cout << "fin" << endl;
        return;
    }

    int aux = pow(2,n)-1;
    int aux2;

    //cout << endl << "iteraciones: " << pow(2,printPerRow)-1 << endl;
```

```cpp
    //aux = aux*2;
    aux2 = aux;

    // -------------- Para el espaciado -------------------
    //cout << vec1[aux] << "  ";
    for (int i = 0; i < aux; i++)
    {
        cout << "  ";
    }
    cout << vec1[aux];
    // ----------------------------------------------------

    for(int i = 0; i < pow(2,printPerRow)-1; i++ )
    {
        aux2 = aux2 + ((aux*2)+2);
        if(aux2 < vec1.size())
        {
            // -------------- Para el espaciado -------------------
            //cout << vec1[aux2] << "  ";
            for (int i = 0; i < ((aux*2)+2)-1; i++)
            {
                cout << "  ";
            }
            cout << vec1[aux2];
            // ----------------------------------------------------
        }
    }
    cout << endl;

    printH2Aux(vec1,n+1,printPerRow-1);
}

void printNodesH2(TreeNode *head)
{
    vector<int> vec2;
    nodeToVector(head,vec2); //guardamos los nodos en un vector

    /*
    cout << "Vector out: " << endl;
    for(int i = 0; i < vec2.size(); i++ )
    {
        cout << vec2[i] <<  " - " ;
    }
    cout << endl;
    */

    printH2Aux( vec2, 0, 3);
}
```