Estructuras de Datos y algoritmos 1

Solución (Tema: Árboles) - Práctico 3

Tema: Punteros - Listas y árboles

Arboles

```
#include <iostream>
#include <assert.h>
typedef struct nodoArbolBinario NodoArbolBinario;
typedef NodoArbolBinario* ArbolBinario;
struct nodoArbolBinario
{
  int info;
  ArbolBinario izq;
  ArbolBinario der;
};

//1) Implemente una función que retorne la cantidad de
//nodos que tiene un árbol binario recibido como parámetro.

bool esVacio(ArbolBinario raiz)
{
  return raiz==NULL;
};

int nodos(ArbolBinario raiz)
{
  if(esVacio(raiz))
        return 0;
  else
        return (1+ nodos(raiz->izq)+ nodos(raiz->der));
};

//2) Implemente una función que retorne la altura que tiene un árbol binario recibido como parámetro.

int max(int i,int j)
{
  if(i>j)
        return i;
  else
        return j;
};
```

```
int altura(ArbolBinario raiz)
{
  if(esVacio(raiz))
        return 0;
  else
        return (1+ max(altura(raiz->izq), altura(raiz->der)));
};

//3) Implemente una función que retorne la cantidad de hojas
// que tiene un árbol binario recibido como parámetro.

bool esUnaHoja(ArbolBinario raiz)
{
  assert(!esVacio(raiz));
  return (raiz->izq==NULL && raiz->izq==NULL);
};

int hojas(ArbolBinario raiz)
{
  if(esVacio(raiz))
        return 0;
  else
        if (esUnaHoja(raiz))
              return 1;
        else
              return(hojas(raiz->izq)+hojas(raiz->der));
};

//5) Implemente una función que reciba dos árboles binarios y retorne true si ambos
//son iguales, false en caso contrario.

bool sonIguales (ArbolBinario raiz1, ArbolBinario raiz2)
{
  if(esVacio(raiz1) && esVacio(raiz2))
        return true;
  else
  {
        if (esVacio(raiz1) || esVacio(raiz2))
              return false;
        else
              return ((raiz1->info == raiz2->info)
                    && sonIguales(raiz1->izq, raiz2->izq)
                    && sonIguales(raiz1->der, raiz2->der));

  }

};
```

// 8) Implemente una función que inserte un elemento "e" en un árbol binario de búsqueda.

```
void Insertar(ArbolBinario &raiz, int e)
{
  if(esVacio(raiz))
  {
            raiz=new NodoArbolBinario;
            raiz->info=e;
            raiz->izq=NULL;
            raiz->der=NULL; // los arboles binarios de busqueda crecen a traves de sus hojas
  }
  else
  {
            if(raiz->info < e)
                      Insertar(raiz->der, e);
            else
                      if(raiz->info > e)
                                Insertar(raiz->izq, e);
  };
};

// Adicional –visto en el teórico-
ArbolBinario copioHastaUnNivel(ArbolBinario raiz, int nivel)
{
  if(esVacio(raiz) || nivel < 1)
            return NULL;
  else
  {
            ArbolBinario aux=new NodoArbolBinario;
            aux->info=raiz->info;
            aux->izq=copioHastaUnNivel(raiz->izq, nivel-1);
            aux->der=copioHastaUnNivel(raiz->der, nivel-1);
            return aux;

  };
};
```