Práctico 1 – Arboles

Ejercicio 1

Implemente las siguientes funciones sobre árboles binarios.

a) int **cantNodos**();
Pos: Retorna la cantidad de nodos del AB.

b) int **cantHojas**();
Pos: Retorna la cantidad de nodos hoja del AB.

c) int **peso**();
Pos: Retorna el peso del AB.

d) int **altura**();
Pos: Retorna la altura del AB.

e) boolean **todosPares** ();
Pos: Retorna true sii todos los elementos del AB son pares.

f) boolean **iguales** (AB a, AB b);
Pos: Retorna true sii el árbol binario a es igual al árbol binario b. Dos árboles son iguales si son vacíos o si tienen los mismos elementos y en el mismo orden.

g) AB **clon** (AB a);
Pos: Retorna un nuevo AB resultado de copiar todos los elementos del AB actual en el nuevo árbol.
Nota: Se debe crear un AB nuevo e independiente en memoria.

h) AB **espejo** (AB a);
Pos: Retorna un nuevo AB resultado de copiar espejados todos los elementos del AB actual en el nuevo árbol. Nota: Se debe crear un AB nuevo e independiente en memoria.

i) Implemente una función que dado un AB retorne true sii el árbol es equilibrado. e) Cite un nodo sin entradas.

Ejercicio 2

Dibuje el árbol resultante de insertar la siguiente secuencia de números en un árbol binario de búsqueda.
3, 1, 4, 6, 9, 2, 5 ,7

Ejercicio 3

Implemente las siguientes funciones para la estructura de **Árbol Binario de Búsqueda** de enteros.

a) void **insertar** (int x);
Pos: Inserta el dato pasado como parámetro en el árbol manteniéndolo ordenado.

b) void **borrarMínimo** ();
Pos: Elimina el menor elemento del ABB.

c) void **borrarElemento** (int x);
Pre: El elemento x pasado como parámetro pertenece al ABB.
Pos: Elimina el elemento x del ABB, manteniéndolo ordenado.

d) boolean **pertenece** (int x);
Pos: Retorna true sii el dato pasado como parámetro pertenece al ABB.

e) void **listarAscendente** ();
Pos: Lista en pantalla los elementos del ABB ordenados de menor a mayor.

f) void **listarDescendente** ();
Pos: Lista en pantalla los elementos del ABB ordenados de mayor a menor.

Ejercicio 4

Implemente las siguientes funciones para una estructura de **Árbol General** representada como árbol binario con la semántica primerHijo, siguienteHermano.

a) void **insertar**(String padre, String dato);
Pre: padre pertenece al AG.
Pos: Inserta el dato pasado como parámetro en el árbol dependiendo directamente de padre.

b) int **cantHojas**();
Pos: Retorna la cantidad de nodos hoja del AG.

c) int **altura**();
Pos: Retorna la altura del AG.

d) void **borrarElemento** (String dato);
Pre: El dato pasado como parámetro pertenece al AG y la ocurrencia es única.
Pos: Elimina el dato del AG y la subestructura jerárquica que de él dependa.

e) nodoAG **buscar** (String dato);
Pos: Retorna un "puntero" al nodo que contiene el dato pasado como parámetro, si éste pertenece al AG, sino retorna null.

Ejercicio 5

a) Desarrolle un algoritmo que dado un árbol binario de enteros no vacío y un entero k, retorne el número de nivel del árbol que tiene mayor cantidad de nodos, considerando sólo hasta nivel k. *Nota:* Considerar que el nivel de la raíz es 0. Recorrer el árbol una sola vez.

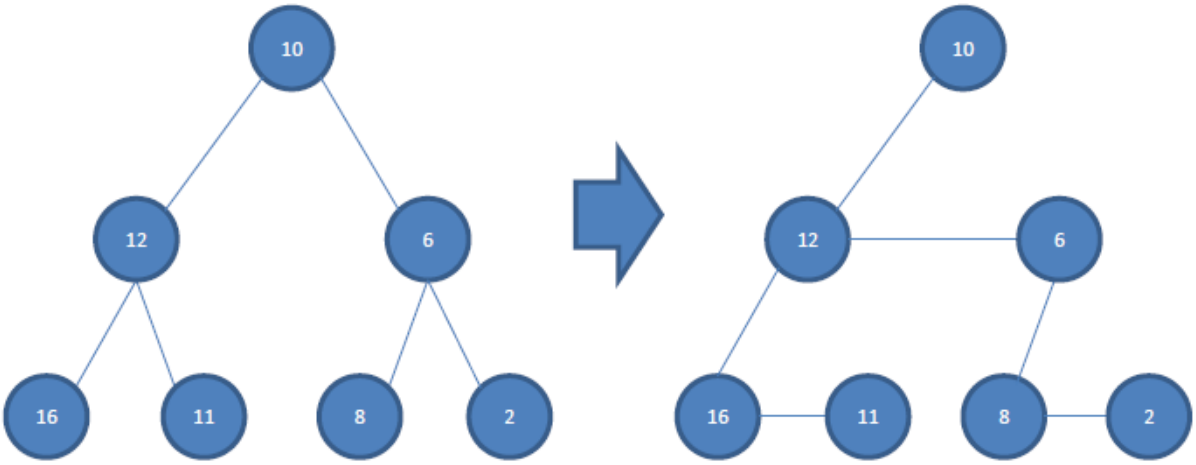
b) Desarrolle un algoritmo que dado un árbol binario de enteros retorne una lista con todos los elementos del árbol y el número del nivel en el que se encuentra cada uno. *Nota:* Considerar que el nivel de la raíz es 0.

c) Desarrolle un algoritmo que, recibiendo un árbol binario de enteros y un separador, lo imprima en orden por niveles, separando los niveles con el separador.

Ejercicio 6

Implemente un método en el TAD Arbol Binario que transforme un árbol binario en un árbol general.

Firma sugerida: ArbolG **transformar**()



Ejercicio 7

a) Implemente un método que dado un árbol binario y un entero n, retorne una lista con los nodos del árbol que se encuentran en el nivel n.
Firma sugerida: `public Lista getNodosNivel(int nivel)`

Puede suponer que cuenta con la una implementación del TAD Lista, y que la información almacenada en sus nodos es del mismo tipo que la almacenada en el árbol.

b) Implemente el mismo método que en la parte a), pero para un árbol general.

c) ¿Puede afirmar que los árboles binarios de búsqueda sin balancear SIEMPRE tienen mejor eficiencia que los árboles binarios? ¿Por qué?

Ejercicio 8 (examen mayo 2015)

Dado un árbol general de Strings que representa el árbol genealógico de una familia, retornar el camino desde la raiz hasta un integrante de la familia dado.

Defina la estructura del árbol general de Strings y cualquier método que utilice.