Guía de ejercicios 2 - Árboles

1. Implementar el TAD ArbolBinarioEnteros (árbol binario de enteros) con sus constructores y proyectores. Considerar que existen los árboles vacíos como árboles binarios. Considerar una primera variante donde los nodos son números enteros, y otra variante donde son de un tipo genérico a (el TAD se llamará ArbolBinario(a)).

2. Definir una función recursiva cantidad, que dado un árbol binario, retorne la cantidad de nodos contenidos.

3. Definir una función recursiva altura, que dado un árbol binario, retorne su altura (la longitud de su rama más larga que va desde el nodo raíz hasta una hoja del árbol).

4. Definir una función recursiva =, que dados dos árboles binarios, retorna si ambos son idénticos. No solamente deberán tener la misma estructura, sino que deberán coincidir los contenidos de los nodos correspondientes.

5. Definir una función recursiva acceder, que dado un árbol binario y una lista de accesos de la forma “izquierda/derecha” (que conforma un “camino guiado” comenzando desde el nodo raíz), retorne el contenido del nodo del árbol binario que sea accesible utilizando el camino guiado dado comenzando desde la raíz del árbol.

6. Definir lo siguiente:

a) Una función recursiva pertenece, que dado un árbol binario de enteros y un entero dado, retorne si el entero dado es igual a algún elemento del árbol dado. Extender la función a que pueda ser aplicada sobre un árbol binario genérico.

b) Una función recursiva nivel, que dado un árbol binario y un valor dado, retorne en qué nivel se encuentra el nodo del árbol que es igual al valor dado. Si el valor no se encontrara en el árbol, retornar un valor superior a la altura del árbol.

c) Una función padre, que dado un árbol binario y un valor dado, retorne el contenido del nodo padre del nodo cuyo contenido es igual al valor dado.

d) Una función hijos, que dado un árbol binario y un valor dado, retorne una lista con el contenido de los hijos del nodo cuyo contenido es igual al valor dado.

e) Una función antecesores, que dado un árbol binario y un valor dado, retorne una lista con el contenido de todos los antecesores del nodo cuyo contenido es igual al valor dado (desde la raíz hasta el antecesor inmediato del nodo encontrado).

7. Definir una función recursiva sinHojas, que dado un árbol binario, retorne un árbol binario idéntico al original, pero que no contenga sus hojas.

8. Definir una función recursiva ramas, que dado un árbol binario, retorne la lista de todas sus ramas (listas que van desde el nodo raíz a cada una de sus hojas existentes).

9. Definir una función recursiva balanceado, que dado un árbol binario, retorne si se encuentra balanceado (la longitud de todas sus ramas no difieren en más de uno entre cualquiera de ellas).

10. Definir una función recursiva espejo, que dado un árbol binario, retorne otro que represente la versión estructural espejada del árbol original.

Por ejemplo:

11. Definir las funciones anteriores utilizando recursión de cola (con o sin estructuras de pila/cola explícitas) en lugar de utilizar recursión de pila.

12. Definir lo siguiente:

a) Funciones recursivas de recorrido lineal de un árbol binario con criterio depth-first (primero en profundidad) y con orden preorden, inorden y postorden, retornando los nodos recorridos en una lista en el mismo orden en que se van recorriendo.

b) Funciones equivalentes a las del punto anterior que sean recursivas de cola, en lugar de recursivas de pila.

13. Definir la función recursiva breadthFirst, que recorra un árbol binario con criterio breath-first (primero a lo ancho), retornando los nodos recorridos en una lista en el mismo orden en que se van recorriendo.

14. Definir la función recursiva insertar, que agrega un valor a un árbol binario, retornando el árbol binario resultante. Definir dos versiones para esta función que difiera su criterio y parámetros de inserción.

15. Definir la función eliminar, que elimina un valor de un árbol binario, retornando el árbol binario resultante. Definir dos versiones para esta función que difiera su criterio y parámetros de eliminación.

16. Implementar el TAD ArbolBinario(a) definido anteriormente, con sus operaciones más elementales, utilizando los tipos de datos adecuados para las siguientes variantes de implementación:

a) Arboles binarios doblemente encadenados

b) Arreglos unidimensionales estáticos (con una dimensión N grande) donde en las posiciones se guardan tanto valores de sus nodos como punteros/índices a sus hijos izquierdos y derechos.

c) Arboles binarios con nodos con un puntero adicional al siguiente según el recorrido inorden.

17. Implementar el procedimiento recursivo mostrar, que dado un árbol binario, muestre el contenido de todos sus nodos con un nivel de “indentación” propio del nivel en que se encuentra en el árbol binario original cada nodo mostrado.

18. Implementar el procedimiento hermano, que dado un árbol binario y una referencia a un nodo del árbol, retorne a su nodo hermano (posible hijo del mismo padre) ubicado dentro de la estructura del árbol.

19. Implementar el procedimiento primos, que dado un árbol binario y una referencia a un nodo del árbol, retorne a sus nodos primos (hijos de padres hermanos) ubicados dentro de la estructura del árbol.

20. Implementar el procedimiento o función ancestroComun, que dado un árbol binario y dos referencias a nodos del árbol, retorne una referencia al ancestro común más cercano a los dos nodos referenciados originalmente.

21. Definir e implementar la operación insertarOrdenado, que dado un árbol binario ordenado y un valor, inserta ese valor en el árbol binario ordenado, retornándolo como resultado.

22. Definir e implementar la operación borrarOrdenado, que dado un árbol binario ordenado y un valor, elimina ese valor en el árbol binario ordenado, retornándolo como resultado.

23.

a) Implementar el TAD ArbolBinarioHojas(a,b) (árbol binario con hojas sin árboles vacíos, donde el tipo de dato de los nodos hoja puede diferir del tipo de dato de los nodos no hoja) con sus constructores y proyectores.

b) Definir funciones similares a las ya conocidas sobre el tipo ArbolBinario(a) sobre este nuevo tipo definido.

c) Definir una función convertir, que dado un árbol binario de tipo ArbolBinario(a) lo convierta en un árbol con una estructura análoga de tipo ArbolBinarioHojas(a,a), en el caso de que sea posible.

24. Implementar las operaciones destructoras de todos los tipos de árboles binarios implementados.

25. Implementar el TAD ArbolN(a), de árboles n-arios genéricos. Definir sus constructores y sus operaciones a partir de los operaciones análogas de los árboles binarios.