ESTRUCTURAS DE DATOS

TRABAJO PRÁCTICO N ° 6
Árboles

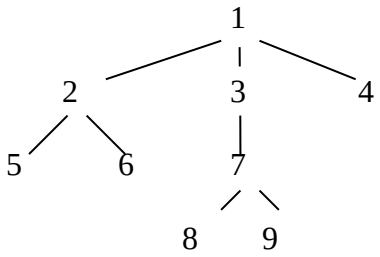
Departamento de Ciencias e Ingeniería de la Computación - U.N.S.
Primer cuatrimestre de 2019

Bibliografía:

[GT] Michael Goodrich & Roberto Tamassia. *Data Structures and Algorithms in Java. Fourth Edition*. John Wiley and Sons. 2006.
[W] Mark A. Weiss. *Estructuras de datos*. Addison-Wesley Iberoamericana España, S.A. 2000.
[W12] Mark A. Weiss. *Data Structures and Algorithm Anlysis in Java*. Third Edition. Addison-Wesley Pearson Education, Inc. 2012.

Ejercicio 1:
Defina recursiva y no recursivamente el concepto de árbol.

Ejercicio 2:
Sea A el siguiente árbol:



a) Indique el nodo raíz de A, los nodos internos de A y los nodos hoja de A.
b) Considere el subárbol con raíz en el nodo 3, indique sus nodos internos y sus nodos hojas.
c) ¿Cuántos nodos tiene el árbol A?
d) ¿Cuántos subárboles tiene el árbol A?
e) ¿Puede un nodo tener varios hijos? Indique los nodos de A con más de dos hijos. Indique además los hijos de los nodos: 1, 8 y 3. Indique los hermanos del nodo 4.
f) ¿Puede un nodo tener varios padres? ¿Por qué? Indique el padre de los nodos 8, 1, y 2.
g) Indique un camino desde el nodo 1 al nodo 9; y otro desde el nodo 2 al nodo 9.
h) Indique la longitud de los caminos desde 1 a 9 y desde 2 a 2.
i) Indique los descendientes del nodo 3, los ancestros del nodo 6, los descendientes propios del nodo 3, y los ancestros del nodo 1.
j) Indique la altura del nodo 1, del nodo 7, y del nodo 4, indicando los caminos con la cual se obtienen.
k) Indique la altura del árbol A y muestre un camino que la justifique.
l) Indique la altura de los nodos hoja.
m) Indique la profundidad de los nodos hoja.
n) Indique los nodos cuya profundidad sea 3.
o) Defina el concepto de nivel de un nodo. Indique los nodos de nivel 3.
p) Defina recursivamente los conceptos de camino, longitud de un camino, altura, y profundidad.
q) Liste los nodos del árbol A en orden previo (preorden), orden posterior (postorden) y por niveles.

Ejercicio 3:
Indique cuáles son los conceptos que aparecen en árboles y que se corresponden con los conceptos de posición, elemento de una lista, y nodo de una lista. Indique las similitudes y diferencias entre listas y árboles.

Ejercicio 4:
a) Implemente el TDA Arbol general como se propone en [GT] utilizando una estructura de datos con nodos que almacenan un rótulo de tipo genérico E, una colección de árboles de E (representando a los nodos hijos de dicho nodo) y una referencia al nodo padre. Además, considere agregar las siguientes operaciones para modificar el contenido del árbol (agregando las excepciones que considere necesarias):

- *CrearRaíz(rótulo)*: Crea un nodo con rótulo *rótulo* como raíz del árbol (considere implementar un constructor que crea un árbol vacío y otro que crea un árbol hoja).
- *AgregarNodoComoPrimerHijo(padre, rótulo)*: Agrega un nodo con rótulo *rótulo* que sea el primer hijo del nodo referenciado por *padre* (y retorna la posición del nodo agregado).
- *Agregar NodoComoÚltimoHijo(padre, rótulo)*: Agrega un nodo con rótulo *rótulo* que sea el último hijo del nodo referenciado por *padre* (y retorna la posición del nodo agregado).
- *Agregar NodoDelante(padre, hermanoDerecho, rótulo)*: Agrega un nodo con rótulo *rótulo* que sea hijo del nodo referenciado por *padre* y su hermano derecho sea el nodo referenciado por *hermanoDerecho* (y retorna la posición del nodo agregado).
- *Agregar NodoDetrás(padre, hermanoIzquierdo, rótulo):* Agrega un nodo con rótulo *rótulo* que sea hijo del nodo referenciado por *padre* y su hermano izquierdo sea el nodo referenciado por *hermanoIzquierdo* (y retorna la posición del nodo agregado).
- *EliminarNodoExterno(nodo)*: Elimina el nodo referenciado por *nodo*, cuando *nodo* es una hoja.
- *EliminarNodoInterno(nodo)*: Elimina el nodo referenciado por *nodo*, cuando *nodo* no es una hoja. Los hijos de *nodo* reemplazan a *nodo* en el mismo orden en el que aparecen.

b) Usando su implementación, escriba un programa que cree un árbol de enteros como el presentado en el ejercicio (2).

Ejercicio 5:
a) Analice de cuántas formas posibles se podría recorrer un árbol.
b) Escriba un comando tal que, para un árbol A y un recorrido (orden-previo, orden-posterior), imprima en pantalla un listado de los nodos de A en el recorrido indicado.
c) Escriba un comando como el del ejercicio anterior, pero en lugar de imprimir los elementos en pantalla, retorne una lista con los elementos en el orden en que fueron visitados.
d) Defina un iterador que permita recorrer un árbol en preorden. Utilice el iterador para resolver nuevamente el inciso (c).
e) Redefina el método clone() de tal manera que permita retornar una copia de la estructura del árbol receptor del mensaje.

Ejercicio 6:
a) Escriba en Java un método para imprimir un árbol por niveles, que muestre un nivel por cada línea de la pantalla, comenzando de la raíz en adelante.
b) Ídem al inciso anterior, pero comenzando del último nivel, y terminando con la raíz.

Ejercicio 7:
Realice los siguientes incisos considerando la representación de colección de hijos implementada como se propone en [GT]. Para realizar los mismos, suponga que los métodos (comandos o consultas) solicitados se implementan como operaciones agregadas al TDA Arbol. Calcule en cada caso el orden del tiempo de ejecución de los comandos o consultas implementados.

a) Escriba un método tal que, dado un árbol A, lo recorra en orden previo y devuelva una lista con referencias a las posiciones de A que sean hijos extremos izquierdos, y que no sean hojas.
b) Escriba un método tal que, dado un árbol A, elimine todo hijo extremo izquierdo N (que no sea hoja) y luego ubique a sus hijos como nuevos hijos del padre de N.
c) Escriba un método tal que, dado un árbol A, elimine todas las hojas de A (deben quedar solo los nodos interiores del árbol original).
d) Realice un método que elimine todos los nodos que tengan un rótulo R dado.
e) Realice un método que elimine los nodos con rótulo R que estén en el nivel 4.
f) Escriba un método tal que dados dos nodos N1 y N2 de un árbol A, indique si existe un camino entre N1 y N2. Indique qué diferencias existirán entre solución propuesta y una alternativa en la que no se contara con la referencia al nodo padre en la estructura de datos de los nodos del árbol.
g) Modifique el método anterior para que devuelva una lista con el camino hallado.
h) Escriba un método que indique la profundidad de un nodo N de un árbol A, e indique el camino que la justifica.
i) Escriba un método para hallar la altura de un nodo N de un árbol A e indique el camino que la justifica.
j) Escriba un método para hallar la altura de un nodo con rótulo R en un árbol A e indique el camino que la justifica. Suponga que el árbol A no tiene rótulos repetidos.

Ejercicio 8:
Agregue una operación a la clase Árbol definida en el ejercicio 4 que, dado un rótulo r y un número entero A>0, por cada nodo con altura igual a A del árbol receptor del mensaje inserta en dicho nodo un hijo extremo derecho con rótulo R. Esta operación no debe generar un nuevo árbol sino, modificar el árbol que recibe el mensaje.
Considere que no tiene acceso a la estructura de la clase lista, pero si puede utilizarla como un TDA.

Ejercicio 9:
Implemente en Java una operación para la clase Árbol definida en el ejercicio 4 que, recibiendo un rótulo R, modifique el Árbol A receptor del mensaje de forma tal que, por cada nodo de A que sea interno y tenga rótulo R, se invierta el orden de sus hijos. Esta operación no debe generar un nuevo árbol sino, modificar el árbol que recibe el mensaje. Recuerde que está agregando un método a la clase Árbol, por lo tanto tiene total acceso a su estructura.

Ejercicio 10 (extraído de un examen):

a) Defina todas las estructuras de datos necesarias para implementar un árbol general de rótulos de tipo E genérico. No defina signaturas de métodos. Considere que posee el tipo lista completamente implementado.
b) Programe el constructor del árbol y del nodo.
c) Implemente la operación removeExternalNode(p) que elimina una hoja del árbol receptor del mensaje.
d) Defina la clase excepción NoExisteTalNodoException y programe su constructor.

e) Escriba un método en Java que, dado un árbol general T y un rótulo R, halle el nodo N de T con rótulo R y elimine todas las hojas del subárbol que tiene como raíz a N. Para esto, utilice el TDA Árbol dado en la teoría y todo otro TDA que considere oportuno. Implemente todas las operaciones auxiliares de árbol que utilice; puede asumir la lista y otros TDA que no sean el árbol totalmente implementados.
f) Estime orden del tiempo de ejecución de su solución justificando adecuadamente.

Ejercicio 11 (extraído de un examen):

a) Programe un método en Java que, reciba un árbol general A (implementado como en la teoría), y retorne un mapeo M (implementado con una lista) donde para cada rótulo X de A, la operación M.get(X) devolverá la cantidad total de descendientes de X en A. Asuma que el árbol A no tiene rótulos repetidos. Asuma el árbol y el mapeo completamente implementados.
b) Estime orden del tiempo de ejecución de su solución justificando adecuadamente.