**UNIVERSIDAD NACIONAL DE HURLINGHAM**

****

Materia: Estructura de Datos

Docente : González, Sergio

Tema: Informe, TP N°4 Listas Enlazadas y Arboles

Alumnos: Mariano A. Salto, Santiago Smith

Año: 2020, Primer Cuatrimestre virtual

**Introducción:**

En este informe vamos a explicar brevemente las estructuras de datos **árboles binarios y listas enlazadas**. Además de las funciones que implementamos para resolver el ejercicio que se nos presentó; y por último, una breve conclusión describiendo el proceso de resolución de cada uno de los requerimientos.

**Arboles Binarios**

**¿Qué son los árboles binarios?**

Un árbol binario es una estructura no lineal (a diferencia de las listas). Como tales no tienen una estructura lógica de tipo lineal o secuencial sino ramificada. Esta estructura se compone de un nodo raíz que cuenta con uno o dos punteros, uno hacia el hijo izquierdo y otro hacia el hijo derecho. El dato que se encuentra en la raíz es la clave, los datos que se ubican en el sub-árbol izquierdo son aquellos que son menores a la clave de la raíz y los que se ubican en el sub-árbol derecho son los datos mayores. Es importante destacar que cada nodo no puede tener más de dos hijos (al menos uno).

Las condiciones que se deben existir para que un árbol cumpla la función de tal son simples: cada nodo tiene un único padre y puede tener más de un hijo. Un nodo no puede tener múltples padres y no hay ciclos. Desde la raíz puedo llegar a cualquier nodo interno y hojas del mismo.

Sólo se puede acceder a los nodos desde su raíz e ir recorriendo hacia la derecha o hacia la izquierda dependiendo de la búsqueda que se realice.

**Algunas características de los nodos y los árboles son:**

**Trayectoria:** Es la secuencia de nodos que hay entre ellos, el largo de la trayectoria se mide por el numero de enlaces, es decir la cantidad de saltos hasta llegar a un nodo**.**

**Grado:** Es la cantidad de hijos que tiene un nodo (de sub-árboles hijos), si un nodo no tiene ningún hijo se dice que es una hoja. Si tiene un hijo ya sea izquierdo o derecho se dice que es de grado 1, y si tiene ambos es de grado 2.

**Peso:** Es la cantidad total de nodos que hay en el árbol.

**Altura de nodo:** Es el largo de trayectoria a la hoja más lejana, es decir la hoja que se encuentra a mayor cantidad de enlaces de la raíz

**Profundidad de nodo:** Es el largo de un nodo hoja hasta la raíz, es inverso a la altura. Los nodos que se encuentran en una misma profundidad se dice que están al mismo nivel.En resumen ,es la hoja más lejana a la raíz.

**Mínimo:** Es el primer nodo que no tiene hijo izquierdo

**Máximo:** Es el primer nodo que no tiene hijo derecho

**Predecesor de un nodo:** Es el nodo máximo que se encuentra en el sub-árbol izquierdo, es decir el más grande entre los más pequeños

**Sucesor de un nodo:** Es el mínimo del sub-árbol derecho, es decir, el más chico entre los más grandes.

Además, existen diferentes formas de recorrer un árbol, las cuales nos permiten acceder a los elementos de forma sistemática. Las mismas son operaciones recursivas.

Las formás de recorrer un árbol son:

**Pre-orden:** Apenas se llega al nodo, hacemos “algo” en él y luego se recorre el subárbol izquierdo (en pre-orden ) y por último el derecho (en pre-orden)

**In-orden:** al llegar a un nodo, no se hace nada y primero se recorre el sub-árbol izquierdo (en in-orden), luego, se regresa al nodo. Se realiza la operación que necesitamos en él y luego se recorre el sub-árbol derecho (en in-orden recursivamente)

**Pos-orden:** Se recorre primero el sub-árbol izquierdo (en pos-orden recursivamente), luego el derecho (en pos-orden recursivamente) y después de haber recorrido los dos sub-árboles, se hace lo que se realiza la operación que se requiera sobre el nodo raíz.

En cualquiera de estos recorridos es indistinto qué sub-árbol se quiere visitar primero, siempre y cuando se respete la visita al nodo raíz en el orden establecido.

**Ejemplos de uso**

Estas estructuras son usadas en páginas como Google y buscadores de internet (para nombrar solo una de las tantas aplicaciones de esta estructura de datos), partiendo de la idea de Adelson Velski y Landis (AVL), dos matemáticos rusos que plantearon esta idea publicándolo en un artículo en el año 1962 “An algorithm for the organization of information” (Un algoritmo para la organización de la información”).

El funcionamiento de esta idea, es que las palabras no se guardan como listado sino como un árbol previamente dicho. Los árboles balanceados facilitan la búsqueda de palabras ya “linkeadas” o referenciadas a páginas web y permiten que grandes caudales de información sean indexados en buscadores como Google,Yahoo o Bing. De esta manera la palabra que estemos buscando pueda ser encontrada en una pequeña cantidad de pasos, aún así cuando la cantidad de información almacenada es muy grande.

**Ejemplo Árbol binario:**

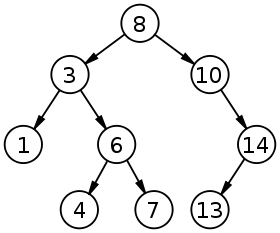


fig1. Ejemplo de árbol binario

En el ejemplo de la figura 1 , se puede observar un árbol y sus características principales:

* Nodo raíz = 8
* Profundidad = 3
* Peso = 9
* Altura = 3
* Nivel de 10 = 1
* Grado de 14 = 1
* Predecesor = 7
* Sucesor = 10

**Otras estructuras usadas para la realización del trabajo práctico**

**Listas Enlazadas**

**¿Qué son las listas enlazadas?**

Las listas enlazadas son una estructura de datos de complejidad computacional O(n) (a medida que aumentan los datos ingresados, el número de operaciones aumenta de forma proporcional). Dicha estructura cuenta internamente con nodos que contienen datos de cualquier tipo, y están referenciados uno con el otro, ya sea al anterior o al siguiente, en nuestra implementación usamos una lista enlazada simple que tiene solo una referencia o puntero al nodo que le sigue aunque también existen las doblemente enlazadas (pose dos punteros , uno al nodo anterior y otro al siguiente).

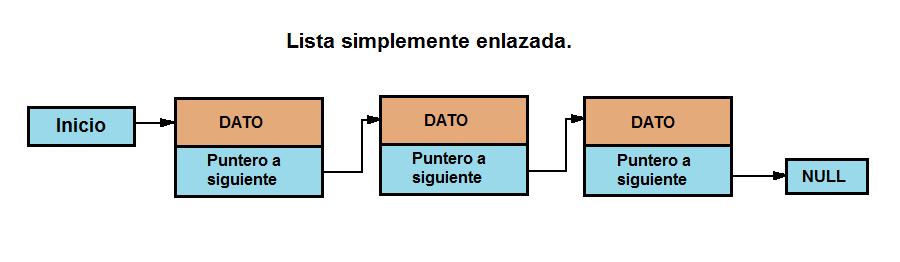
También podemos encontrar listas enlazadas circulares y listas enlazadas doblemente circulares, pero no serán explicadas en este informe.

En las listas enlazadas simples solo pueden acceder a sus datos de forma secuencial, ya que solo pueden recorrerse en una sola dirección accediendo desde el principio y uno a uno a todos los elementos de la misma.

Las listas enlazadas se guardan en memoria de forma no secuencial, a diferencia de un arreglo.

Para la inserción de los elementos , se inserta un nuevo nodo en la lista, si la lista se encuentra vacía, el elemento a insertar se convierte en nuestro primero, esto quiere decir, que vamos a tener un puntero apuntando hacía este.

Para la eliminación de un dato, al eliminar un nodo de la lista, hay que tener en cuenta que el enlace del nodo anterior al nodo que queremos eliminar, quede apuntando hacia el siguiente , ya que si eliminamos un nodo sin controlar los punteros podemos perder la mitad de nuestra lista.

 Fig2, ejemplo de lista enlazada simple.

**Conclusión:**

De la misma manera que el trabajo anterior , encaramos cada situación por separado para que los problemas sean más fáciles de resolver. Admitimos que la complejidad fue mayor y se duplicó el tiempo para la resolución de los requerimientos. A medida que nos acercamos al final del TP. Los problemas a resolver fueron reduciendo su complejidad y además porque los requerimientos con mayor problemática ya habían sido resueltos.

Tuvimos algunos problemas con la recursividad. Con un poco de prueba-error y entendiendo como funcionaba python para realizar esto, pudimos resolverlos. Para dar un ejemplo, como devolver una lista por parámetro luego de utilizar la recursividad y de esta manera retornarla correctamente desde la función llamada desde el árbol.

Logramos adaptar el archivo de prueba que nos enviaron para verificar los resultados y ejecutarlo, al comparar , los resultados fueron similares, de todas maneras quedamos a la espera de sus comentarios. Sabemos que algunas funciones podrían haber quedado aplicadas de forma más “elegante”.

Cada requerimiento que implementamos, se encuentra explicado en el código, tanto las auxiliares como las requeridas en el TP.

Cabe destacar que el aprender estas estructuras , como las anteriormente presentadas fueron todo un desafío de nuestra parte. Queremos aprovechar agradecerle al docente por la paciencia, el tiempo y la predisposición al momento de explicar y grabar los video-tutoriales para poder encarar la materia, entendemos que no es fácil el dictar una cátedra de forma virtual, pero siempre fue claro y conciso a la hora de trasmitir conocimientos. ¡Muchas Gracias!

Santiago Smith

Mariano A. Salto

**Bibliografía utilizada:**

Hhmosquera. ( f). Estructura de datos. Espacio de formación y aprendizaje virtual.

<https://hhmosquera.wordpress.com/arbolesbinarios/>

(20 de Junio de 2020). *Recorrido de árboles.* Wikipedia .

https://es.wikipedia.org/wiki/Recorrido\_de\_%C3%A1rboles

(16 de abril de 2020). *Arbol binario de busqueda.* Wikipedia.

https://es.wikipedia.org/wiki/%C3%81rbol\_binario\_de\_b%C3%BAsqueda

(s f). *Recorrido de un árbol*. Matemáticas para computadora.

https://moodle2.unid.edu.mx/dts\_cursos\_mdl/lic/TI/MC/AM/11/Recorrido\_arbol.pdf

[Fernando Schapachnik](https://elgatoylacaja.com/author/fschapachnik/). (08 de abril de 2015). Destripando Google. El gato y la caja. <https://elgatoylacaja.com/destripando-google/>