DEPARTAMENTO DE ELECTRONICA

Ayudantía 6: Árboles binario de búsqueda

Objetivo Principal: Conocer cómo funcionan e implementar árboles binario de búsqueda.
Autor: Kevin Pizarro Aguirre

---

1. Árbol binario de búsqueda

Los árboles binarios de búsqueda, también conocidos como binary search tree (BST), son estructuras de datos que tienen las siguientes características:

1. Los nodos del sub-árbol de la izquierda del nodo raíz (root) son de menor valor que el nodo raíz.
2. Los nodos del sub-árbol de la derecha del nodo raíz son de mayor valor que el nodo raíz.
3. Cada sub-árbol cumple con las características de un BST.

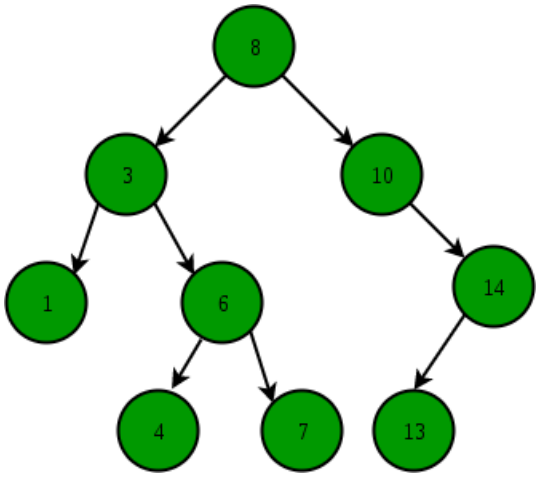

Figura 1: Esquema de árbol binario de búsqueda.

Esta estructura de datos, nos permite ordenar los nuevos datos de una manera conveniente, así si nuestro árbol es de 10000 nodos entonces a lo más deberemos realizar $log_2(10000) = 14$ iteraciones para encontrar el valor deseado, en vez de recorrer los 10000 nodos. El cálculo anterior es válido siempre y cuando el árbol esté **balanceado**, es decir, la cantidad de niveles del sub-árbol de la derecha no tenga una diferencia mayor a 1 en la cantidad de niveles del sub-árbol de la izquierda. En el peor caso, si el árbol no es balanceado, se tendrá que recorrer los 10000 nodos (poco probable). Algunos de los usos para los árboles en general (no sólo BST) es en implementación del sistema de archivos de un sietema operativo, bases de datos, filas de prioridad, entre otras.

2. Ejercicios

2.1. Árboles binarios de búsqueda

Se desea construir una base de datos para una **prestigiosa** empresa, la cual almacena la información de los trabajadores. Cada nodo (trabajador) tiene asociado un ID (identificador único), sueldo, nombre y fecha de contratación. Toda la información del trabajador debe estar almacenada en una estructura llamada *Empleado*, cada variable debe estar a un tipo de dato adecuado. El orden del árbol será según el ID de trabajador.

a) Cree las estructuras *Empleado* y *Nodo*, los cuales formarán el BST.

b) Programe una función que cree un nuevo nodo. Esta función recibe como parámetros el nombre del trabajador, su fecha de contratación, su sueldo y su ID; luego debe retornar el nodo.

c) Programe una función que le permita insertar su nuevo nodo en el BST, debe comparar los ID para ver que rama del árbol tomará.

d) Implemente una función que calcule el promedio de los sueldos de todos los trabajadores en el árbol y luego lo imprima por consola.

e) Si se asume que se ingresarán los trabajadores en orden (ID=1, ID=2...), entonces el BST ¿a qué otra estructura de datos es equivalente en dicho caso?.

f) ¿Cómo podría hacer que un BST no balanceado pase a estar balanceado o se "auto-balancee"? Puede buscar información al respecto.

Ya conociendo las grandes ventajas de los BST es conveniente en muchos casos pasar de un arreglo o una lista a un BST.

a) Se tiene un arreglo de números enteros distintos entre sí y ordenados ascendentemente. Programe una función que convierta el arreglo en un BST.

b) Se tiene una lista simplemente enlazada de número enteros distintos entre sí y ordenados ascendentemente. Programe una función que convierta la lista en un BST.

c) Implemente una función que busque el valor mínimo dentro del árbol.

d) Implemente una función que busque el valor máximo dentro del árbol.