CLASE 06 | ESTRUCTURA DE DATOS III

POR ALEJO BENGOECHEA

ESTRUCTURA DE DATOS III

▼ **ÁRBOLES**

Existen muchos tipos de árboles. Nos vamos a detener en los **Árboles Binarios**.

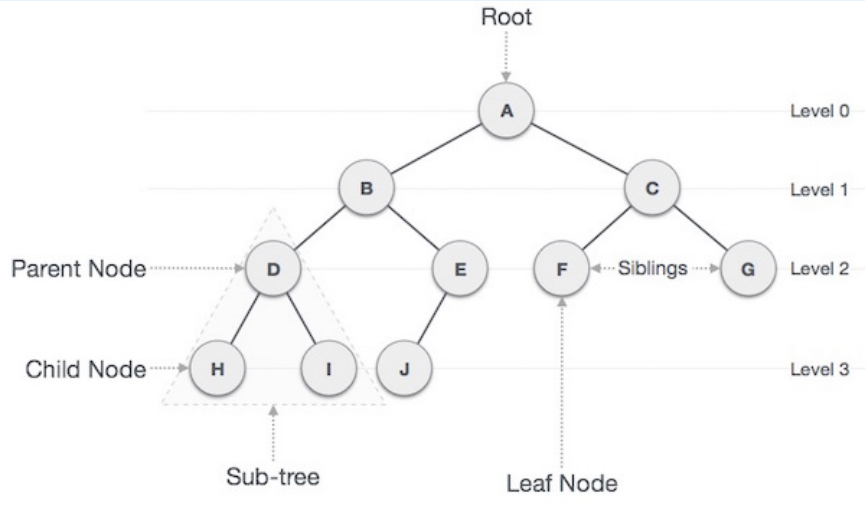
Los **Árboles** son listas simplemente enlazadas, con la diferencia que pueden conectar con ningún, 1 o 2 nodos.

Nodo padre/Nodo raíz: es el primer nodo que se crea.

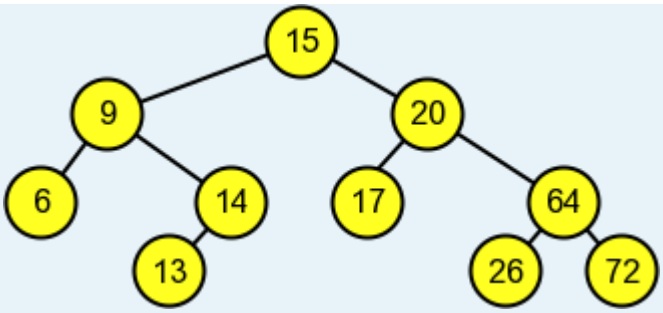
Nodo hijo: hace referencia a los nodos que descienden de otros.

Nodos hermanos: son aquellos que están en el mismo nivel.

Nodo hoja: son los nodos que no tienen hijos.



Dentro de los *árboles binarios* también existen muchos otros tipos de árboles. Nosotros vamos a ver los **Árboles Binarios de Búsqueda**.



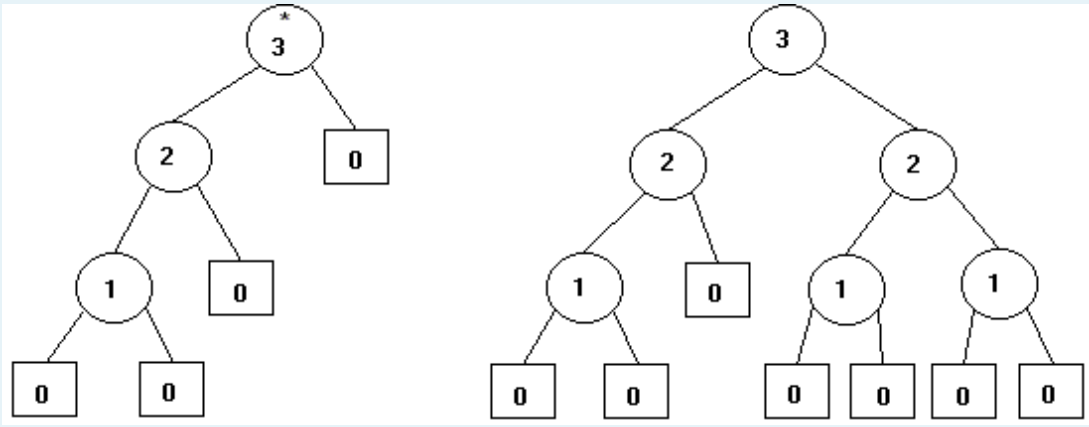
Este tipo de árbol tiene una diferencia con los árboles binarios normales. Esa diferencia es que estos están ordenados.

La forma de orden de un *árbol binario de búsqueda* es que, del lado izquierdo van a tener siempre un valor de referencia menor, y del lado derecho un valor mayor. En la imagen, por ejemplo, vemos que empieza con el número **15**. A su izquierda hay un número menor, y a su derecha uno mayor. A su vez, los *nodos hijos* cumplen con la misma regla. Y así sucesivamente.

▼ **AVL**

A su vez, dentro de los *árboles binarios de búsqueda*, existen los **árboles avl**. Estos árboles son auto-balanceados.

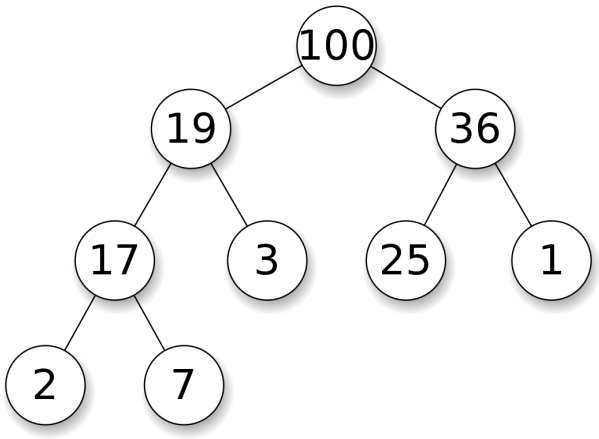
Que un árbol sea balanceado quiere decir que, entre los nodos de mayor nivel, difieren entre 0 y 1 nivel con otras ramificaciones.



El de la izquierda es **no balanceado**. El de la derecha es **balanceado**.

▼ **MAX HEAP**

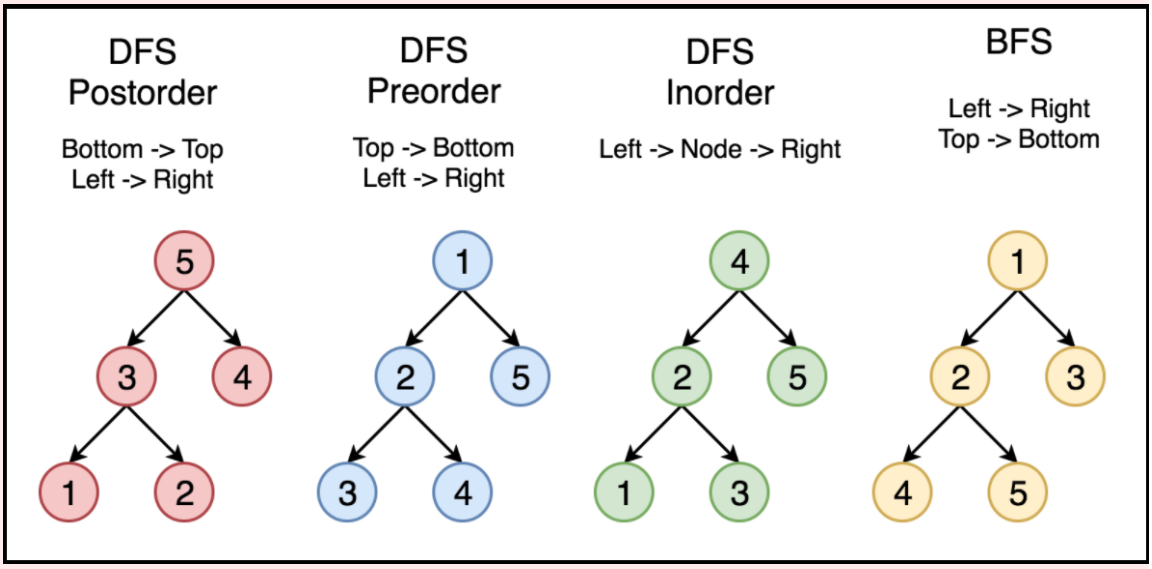
Estos árboles tienen la particularidad de que los valores se distribuyen de arriba (mayores) hacia abajo (menores).



En el caso de que esto fuera al revez (que los números menores van de arriba hacia abajo), este árbol se llamaría **MIN HEAP**.

▼ RECORRIDOS

Estos recorridos se aplican para cualquier tipo de árbol.



▼ BREADTH FIRST SEARCH (BFS)

En este recorrido se imprime de izquierda a derecha y de arriba a abajo.

▼ DEPTH FIRST SEARCH Inorder (DFS)

En este caso, el algoritmo buscará el nodo que esté más a la izquierda de todos. Luego, busca a su padre, y posteriormente al hermano de la derecha.

▼ DEPTH FIRST SEARCH Preorder (DFS)

En este caso, el algoritmo imprimirá cada nodo por el que pase. El recorrido es de arriba-abajo, priorizando la ramificación izquierda. Una vez que no hay un nodo hijo a la izquierda, vuelve un nodo y va para la derecha.

▼ **DEPTH FIRST SEARCH Postorder (DFS)**

Este algoritmo recorre de abajo-arriba. Primero imprimirá el nodo izquierdo, luego el derecho, y finalmente al padre.