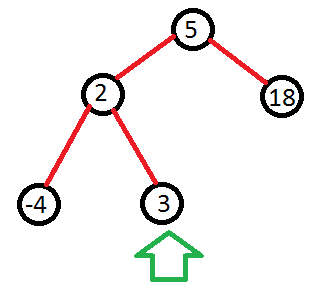
Árboles binarios de búsqueda:

* ¿Cómo se elimina un elemento?

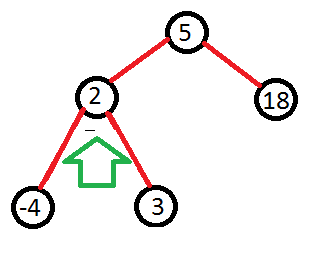
Hay 3 posibles escenarios.

* Si el nodo por eliminar es una hoja



En este casi simplemente se “corta” la referencia.

* Si el nodo por eliminar tiene un solo hijo.



Código:

Nodo remove(int e, nodo c){

If(c == null){

Return null;

}

If(e < c.valor){

c.left = remove(e, c.left);

}else if (c.left == null && c.right != null){

c.right = remove(e, c.right);

}else if(c.left != null && c.right != null){

c.value = findmin(c.right).value;

c.right = remove(c.value, c.right);

} else {

C= c.left != null ? c.left:c.right; // Operador ternario

}

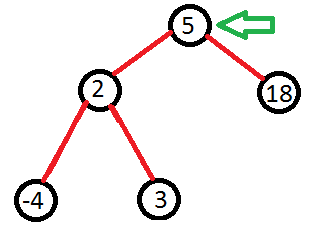
Return c;

}

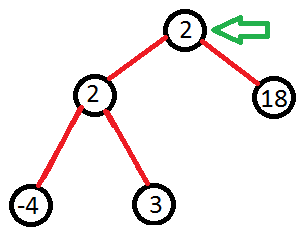
En este caso, se “salta” el nodo por eliminar y sin hijo lo reemplaza

* Si el nodo por eliminar tiene dos hojas. En este caso:

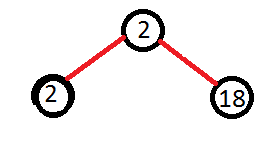
1. Se busca el elemento menor en el subárbol izquierdo.
2. Se copia el elemento encontrado en 1 en el nodo por eliminar.
3. Se ejecuta el proceso de eliminación a partir de la posición en la que se copió el elemento encontrado en 1.



1. El mayor a la izquierda es 2.
2. Se copia 2 a 5.

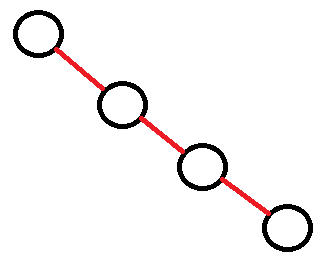


1. Se elimina 2 a partir del subárbol izquierdo (resalta en el escenario a 2).



Árboles AVL:

* Uno de los supuestos de los árboles es que proveen búsquedas rápidas.
* Árbol que siempre está ordenado.
* Dependiendo del orden de inserción, se puede obtener un árbol como:

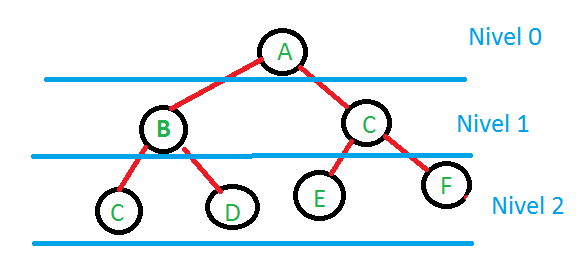


¿Qué sucede con esto?

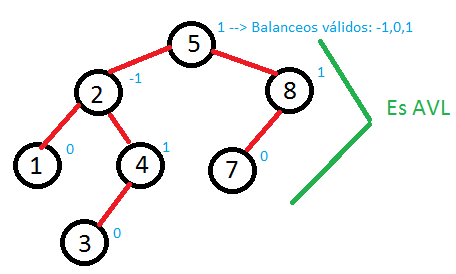
* Se pierde la rapidez para buscar y se reduce en una búsqueda secuencial.
* AVL significa Adelson- Velskii y Landes (autores).
* Es un árbol binario de búsqueda con una condición de balance para asegurar que la profundidad sea óptima:

🡪 La altura a la izquierda no puede diferir en más de 1 con respecto a la derecha

* Niveles de un árbol



* La altura de un árbol es el nivel máximo + 1.



Tarea 4: Implementar avl, Explicar el código con corridas anuales, 3 para cada caso.

16/10/2019