

REPASO DE ABB Y COSTES

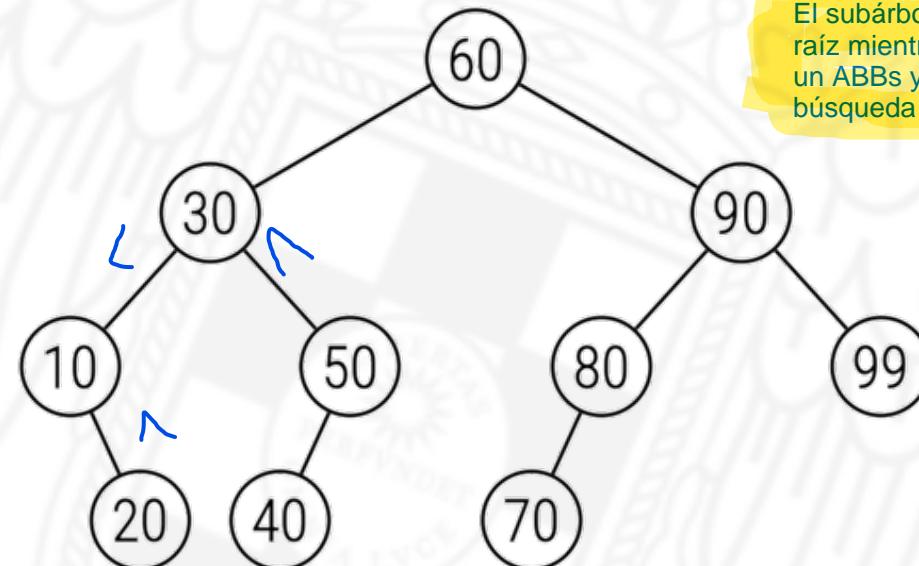


UNIVERSIDAD
COMPLUTENSE
MADRID

ALBERTO VERDEJO

Árboles binarios de búsqueda

- Los **árboles binarios de búsqueda** cumplen que, o bien el árbol es vacío, o bien el elemento en la raíz es mayor que los elementos del hijo izquierdo y menor que los elementos del hijo derecho, y recursivamente los dos hijos son a su vez árboles binarios de búsqueda.

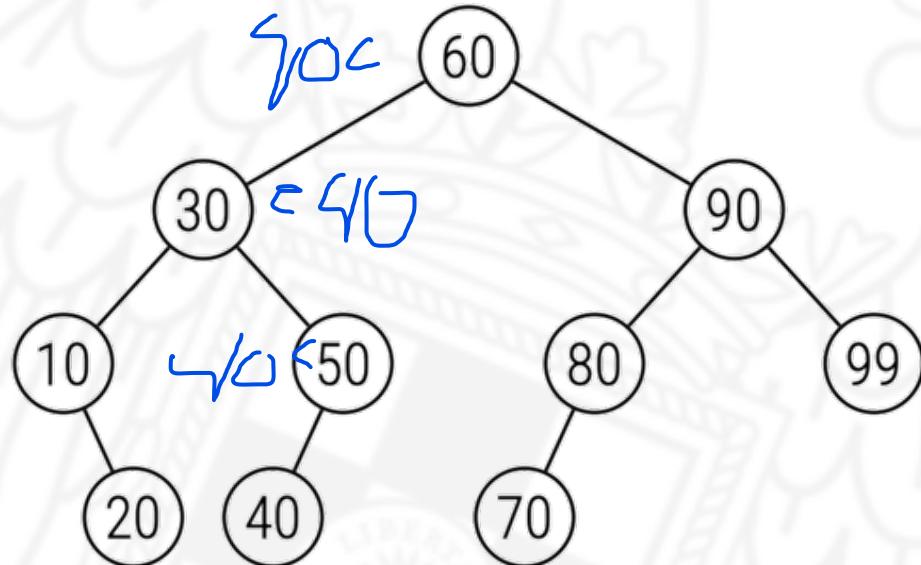


El subárbol izquierdo siempre tiene un valor menor que la raíz mientras que el derecho es mayor que la raíz. Eso es un ABBs y ambos hijos son a su vez, árboles binarios de búsqueda

Operación de búsqueda

- ▶ Buscamos el 40

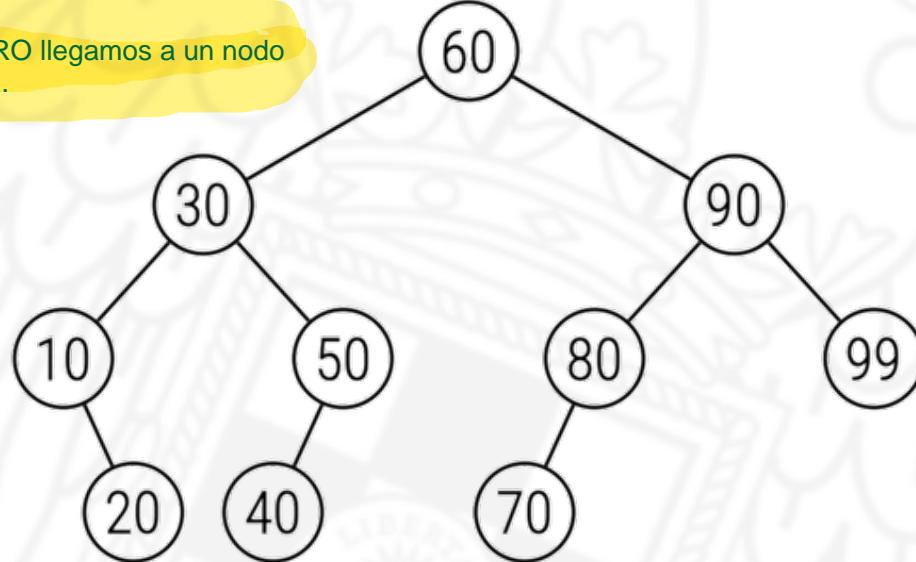
Para encontrarlo lo comparamos con la raíz. Como el 40 es menor que 60 buscamos en el hijo izquierdo. Lo comparamos con la raíz, que es 30. Como es mayor buscamos en el derecho. Así sucesivamente.



Operación de búsqueda

- ▶ Buscamos el 39

Seguiríamos el mismo camino PERO llegamos a un nodo vacío, lo que indica es que no está.

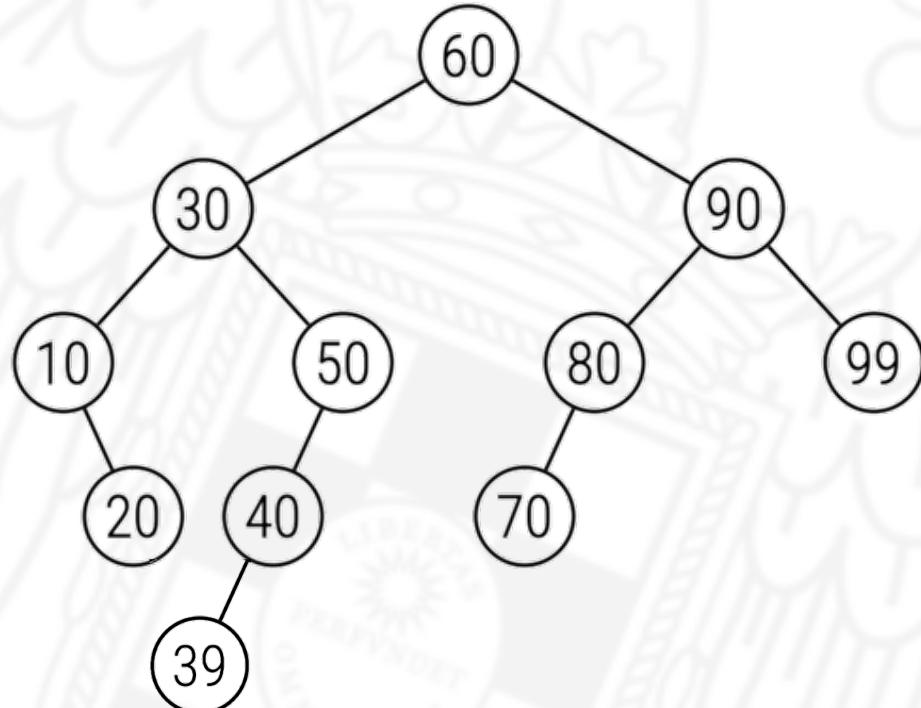


Como recorre todos los niveles del árbol, el coste en el caso peor va a ser lineal en la altura del árbol.

Operación de inserción

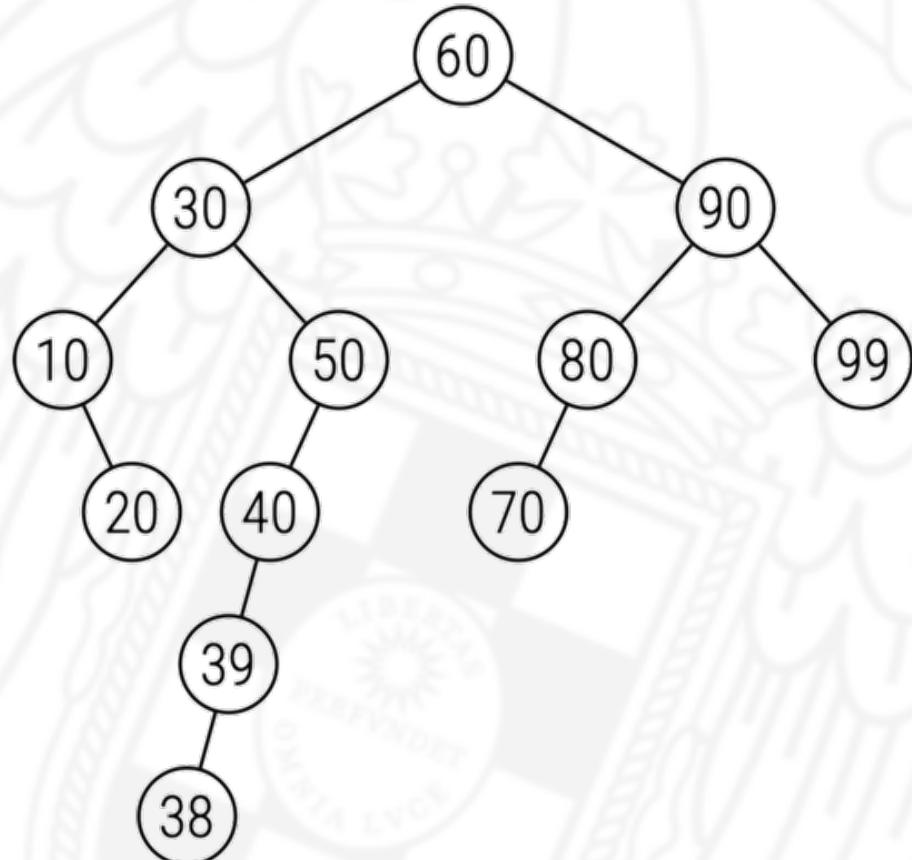
- ▶ Insertamos el 39

Para insertar hay que hacer algo parecido a una búsqueda. Tenemos que buscar donde insertarlo. Siempre y cuando aun NO esté.



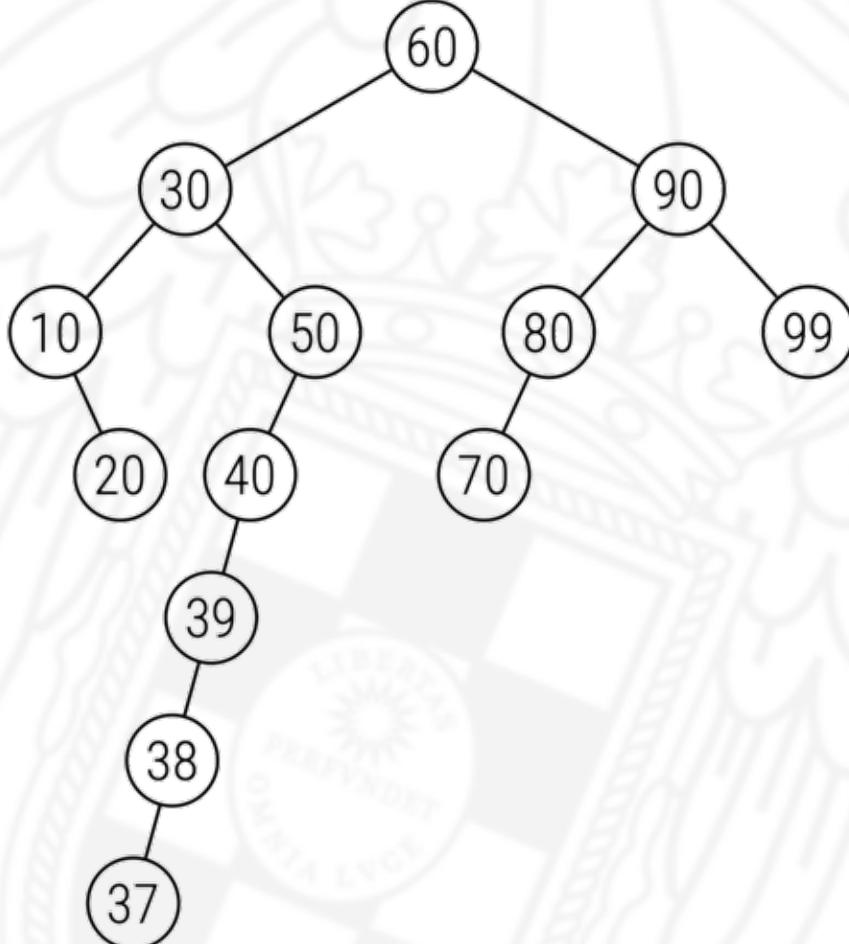
Operación de inserción

- ▶ Insertamos el 38



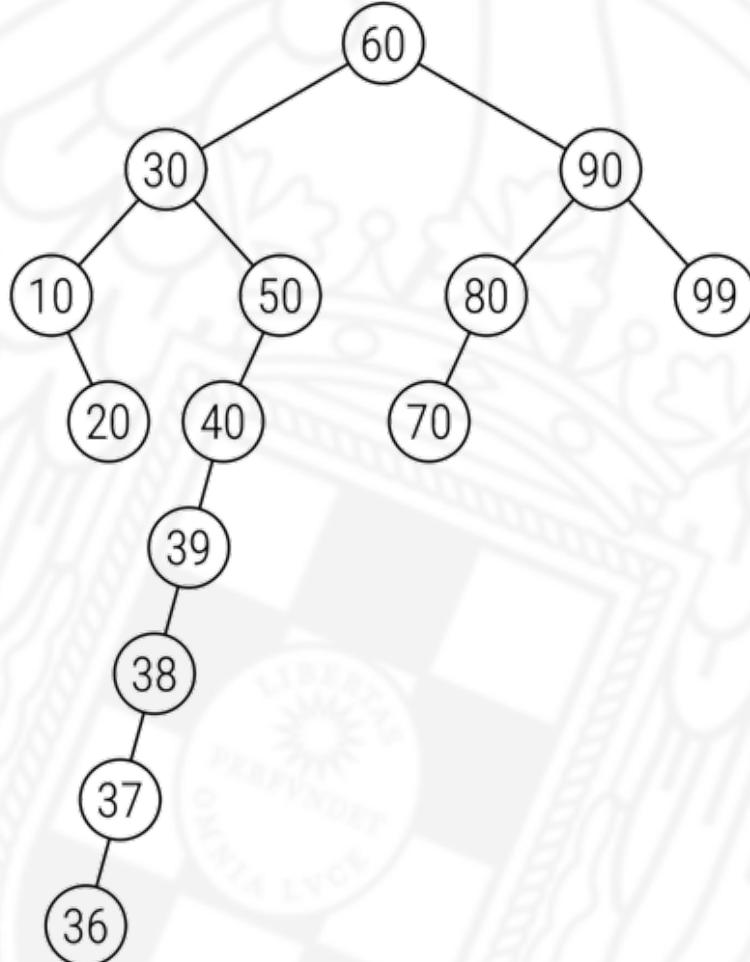
Operación de inserción

- ▶ Insertamos el 37



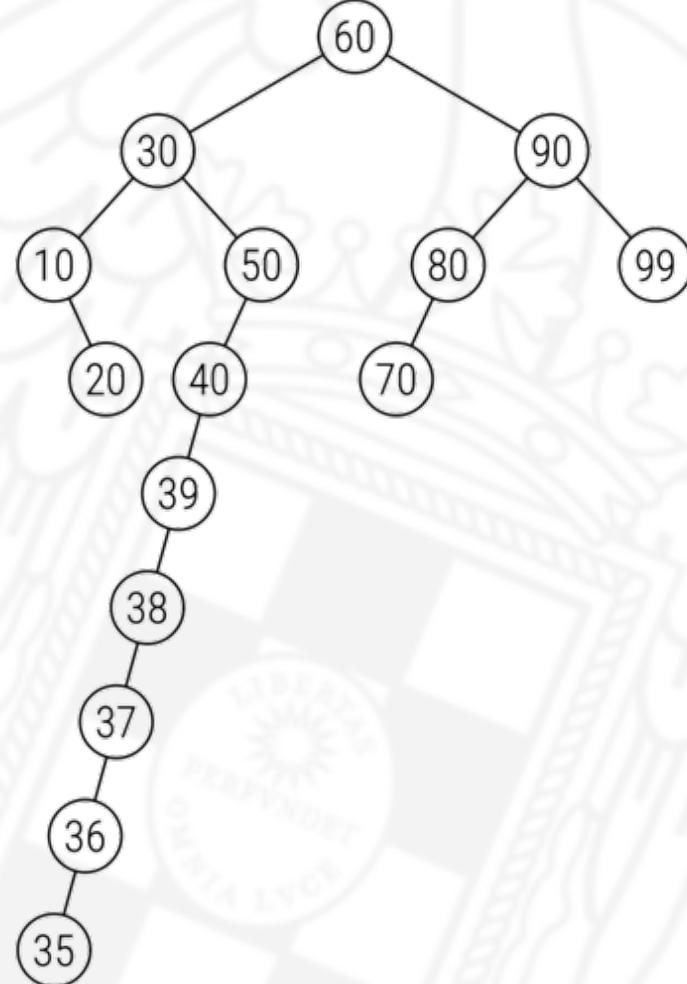
Operación de inserción

- ▶ Insertamos el 36



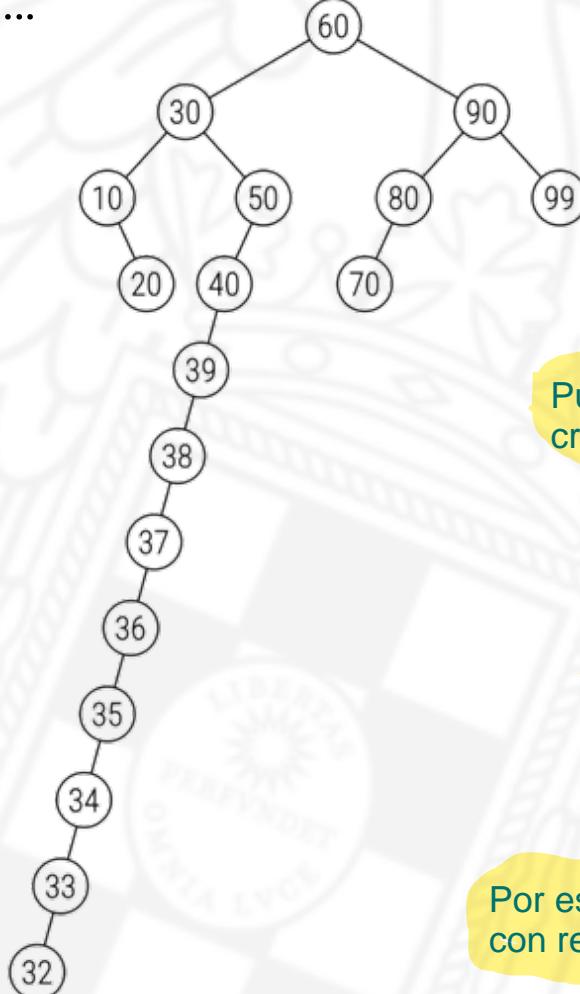
Operación de inserción

- ▶ Insertamos el 35



Operación de inserción

- ▶ Insertamos el 34, 33, 32, ...



Puede ser, por tanto, que al insertar un nodo, el árbol crezca en altura cada vez que insertamos un nodo.

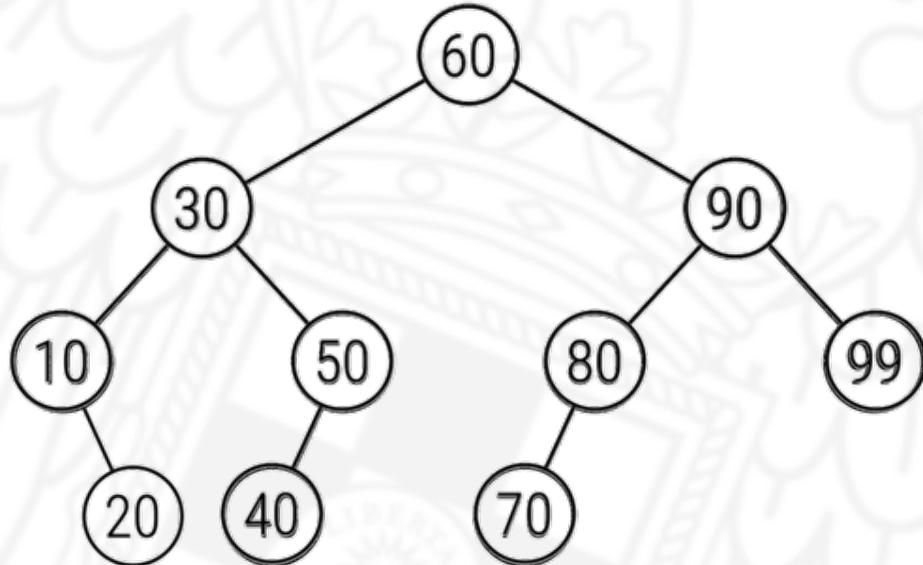
Por tanto, puede ser que el árbol tenga una altura proporcional al número de nodos.

Por eso, la inserción puede ser también de coste lineal con respecto a la altura del árbol.

Operación de eliminación

- Eliminamos el 20

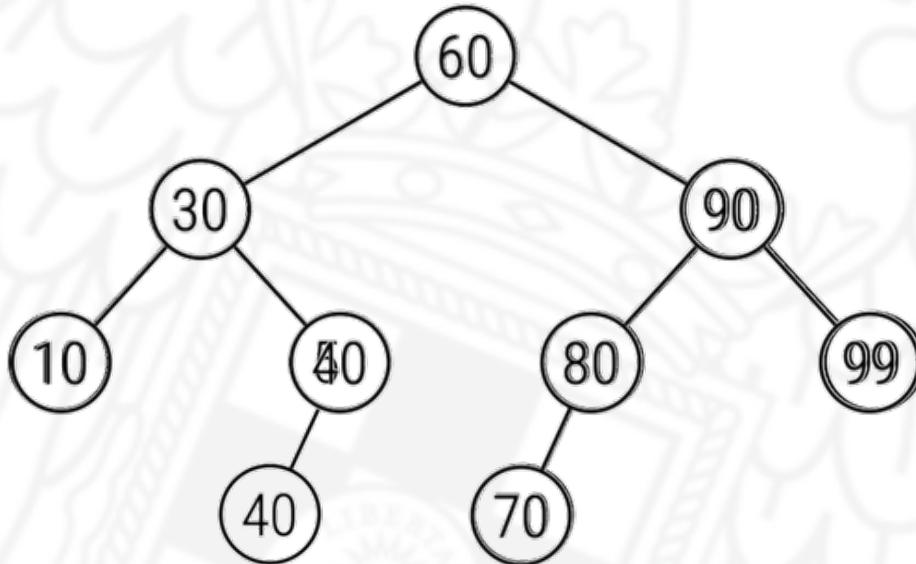
También tenemos que buscar el elemento, ya que si no está el árbol se queda igual.



Tenemos que distinguir casos: si el elemento que queremos borrar es una hoja, simplemente lo eliminamos;
En la siguiente diapo veremos el siguiente caso.

Operación de eliminación

- ▶ Eliminamos el 50



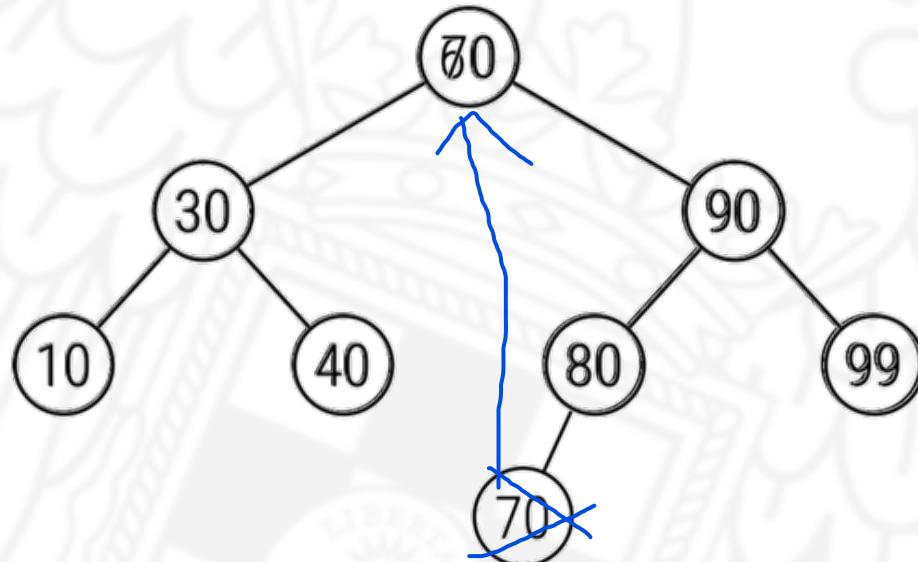
Si solamente tenemos un hijo, entonces el hijo sustituye al padre.

Si el 40 tuvieran también hijo izquierdo e hijo derecho, también se hubieran movido con el.

Operación de eliminación

- ▶ Eliminamos el 60

Coste lineal en la altura del árbol.

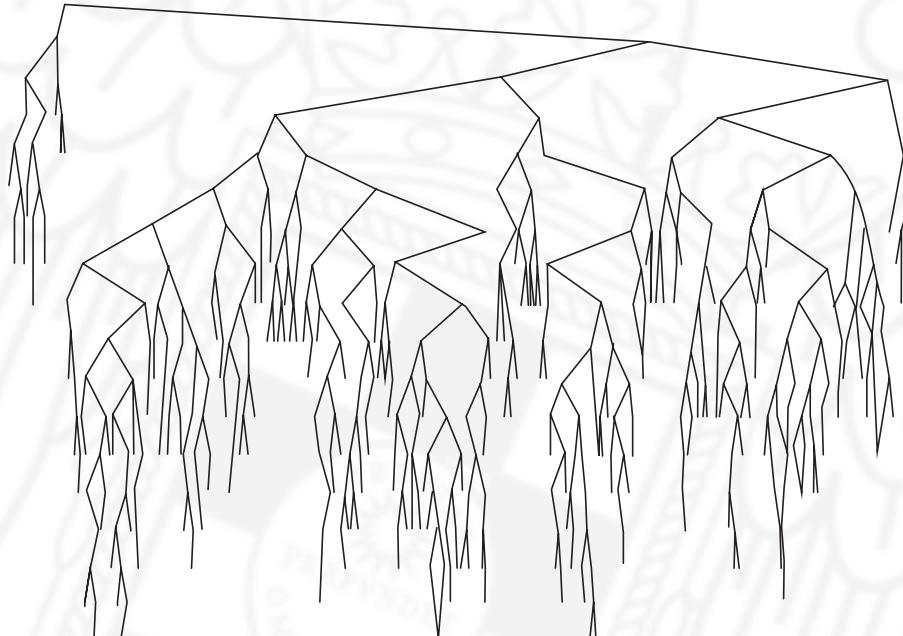


Puede ser sustituido por el anterior en un recorrido inorden, o también por el siguiente, también en el recorrido inorden.

- O coger el mayor del hijo izquierdo (en este caso el 40) o coger el menor del hijo derecho (el 70)

Análisis del caso medio

- Si todas las posibles ordenaciones de la entrada son igualmente probables, la profundidad **media** sobre todos los nodos está en $O(\log N)$.



La profundidad se mantiene logarítmica con respecto al número de nodos del árbol.

ABB generado aleatoriamente (500 hojas, profundidad media 9.98)

Análisis del caso medio

- ▶ Si también hay borrados, no está tan claro que todos los ABBs sean igual de probables. El efecto exacto de esta estrategia aún se desconoce.

Si siempre quitamos el hijo más pequeño del subárbol derecho, el árbol deja de tener una altura logarítmica con respecto a la altura del árbol.

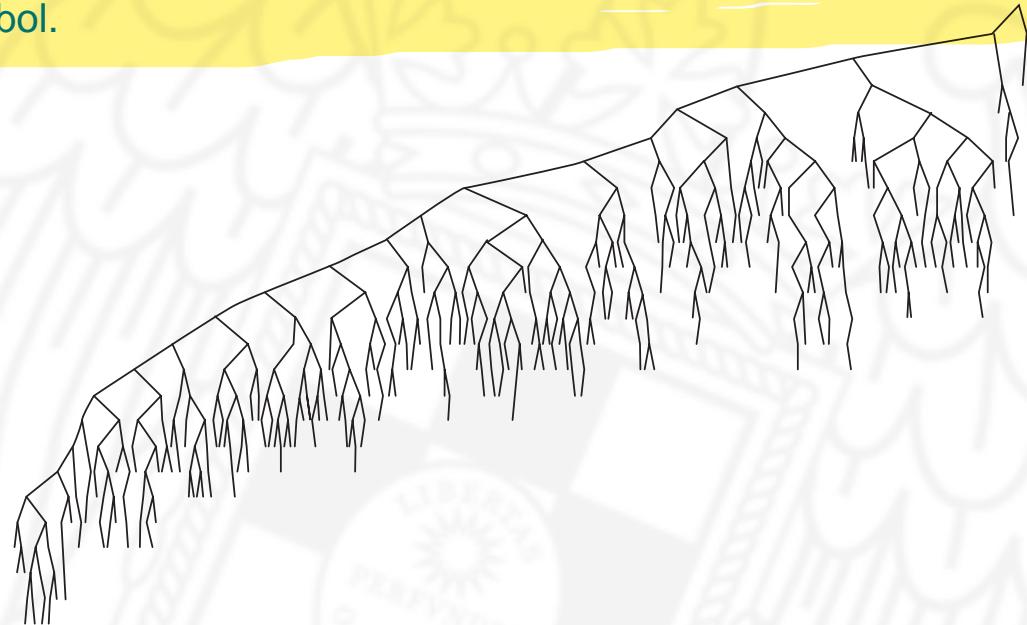


ABB después de $\Theta(N^2)$ pares inserción/borrado

En el siguiente vídeo vamos a ver un árbol que se mantiene equilibrado.