Estructuras de Datos

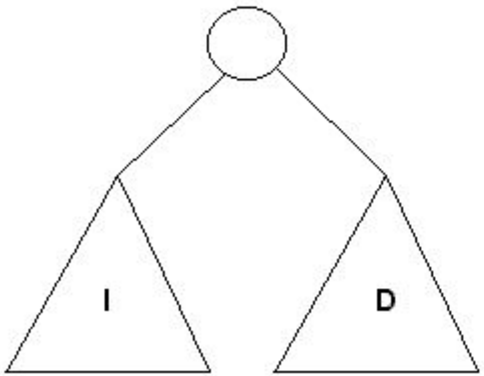
Árboles

Tópicos

- Árboles AVL
  - Inserción en un AVL
    - Rotación Simple
    - Rotación Doble
  - Eliminación en un AVL

Árboles AVL

Un ABB se dice que es un Árbol AVL si para todo nodo interno la diferencia de altura de sus dos árboles hijos es menor o igual que 1.



$$|altura(I) - altura(D)| \leq 1$$

Árboles AVL

Es decir que si definimos

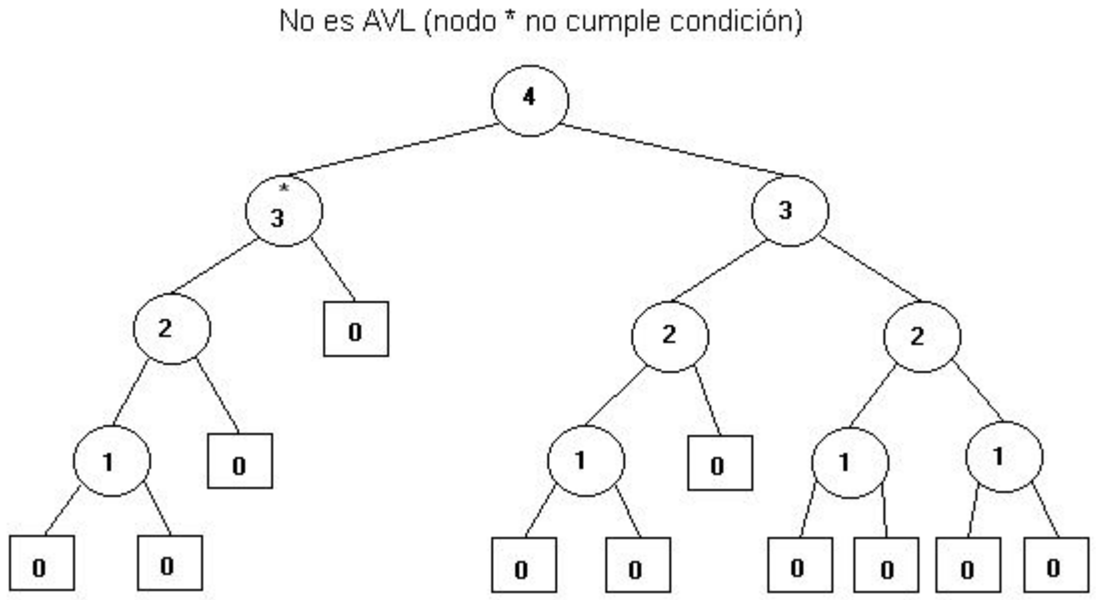
```
int btree_balance_factor(Btree N){
    return btree_altura(N->right)-btree_altura(N->left);
}
```

donde `btree_altura` nos devuelve la altura del árbol.

En el caso de que el argumento corresponda a un nodo en un Árbol AVL el valor de retorno va a estar en el conjunto `{-1, 0, 1}`.

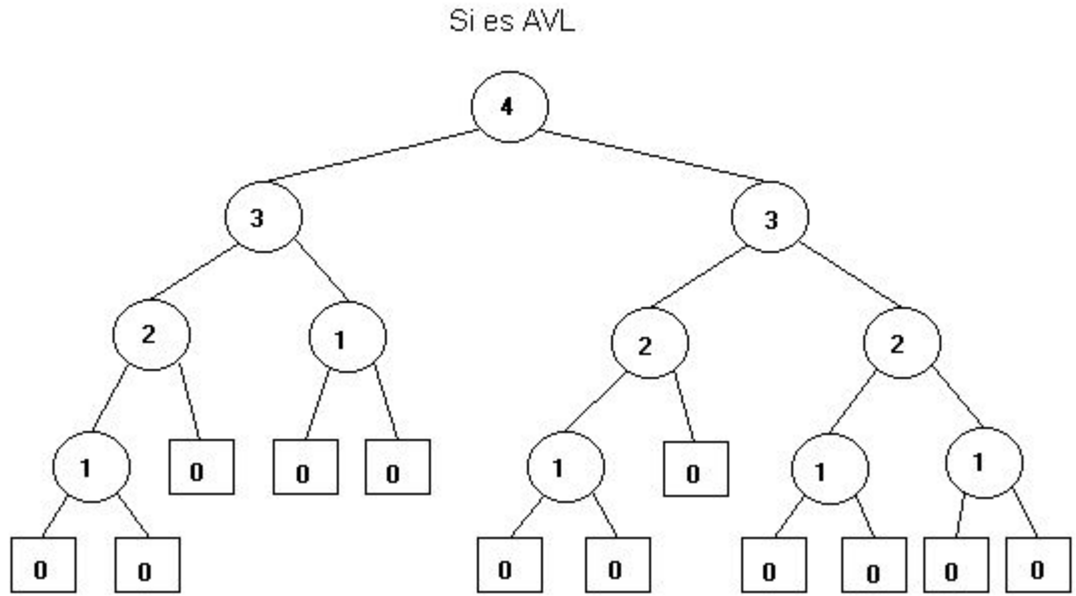
Árboles AVL

Por ejemplo: (el número dentro de cada nodo indica su altura)



Árboles AVL

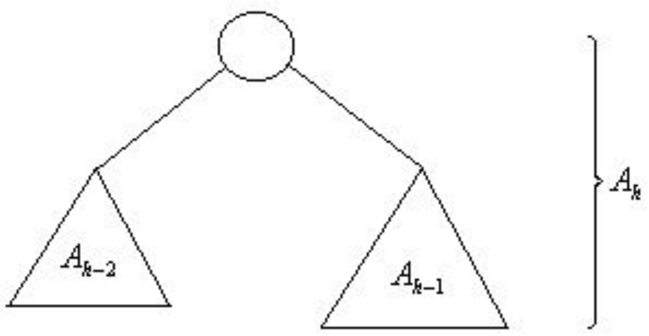
Por ejemplo: (el número dentro de cada nodo indica su altura)



Árboles AVL

Para una altura h dada, ¿cuál es la cantidad mínima de nodos que se necesitan para construir un Árbol AVL que alcance esa altura?

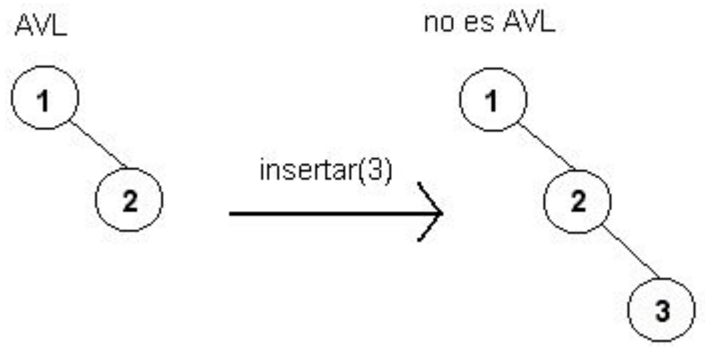
A partir de esta pregunta, y el hecho que en un Árbol AVL de estas características la diferencia de altura de sus hijos en todos los nodos tiene que ser 1 (para minimizar la cantidad de nodos), se puede demostrar que la altura de un Árbol AVL es del orden de `log n`, donde n es la cantidad de nodos.



Inserción en un AVL

La inserción en un Árbol AVL se realiza de la misma forma que en un ABB, con la salvedad que hay que modificar la información de la altura de los nodos que se encuentran en el camino entre el nodo insertado y la raíz del árbol.

El problema potencial que se puede producir después de una inserción es que el árbol con el nuevo nodo no mantenga la propiedad:



Inserción en un AVL

En el ejemplo de la figura, la condición de balance se pierde al insertar el número 3 en el árbol, por lo que es necesario restaurar de alguna forma dicha condición. Esto siempre es posible de hacer a través de una modificación simple en el árbol, conocida como rotación.

Inserción en un AVL

Supongamos que después de la inserción de un elemento X el nodo desbalanceado más profundo en el árbol es N.

Esto quiere decir que la diferencia de altura entre los dos hijos de N tiene que ser 2, puesto que antes de la inserción el árbol estaba balanceado.

Inserción en un AVL

El problema pudo ser ocasionado al insertar el elemento en una de estas cuatro posibles opciones:

1. El elemento X fue insertado en el subárbol izquierdo del hijo izquierdo de N.
2. El elemento X fue insertado en el subárbol derecho del hijo izquierdo de N.
3. El elemento X fue insertado en el subárbol izquierdo del hijo derecho de N.
4. El elemento X fue insertado en el subárbol derecho del hijo derecho de N.

Dado que el primer y último caso son simétricos, así como el segundo y el tercero, sólo hay que preocuparse de dos casos principales: una inserción "hacia afuera" con respecto a N (primer y último caso) o una inserción "hacia adentro" con respecto a N (segundo y tercer caso).

Inserción en un AVL

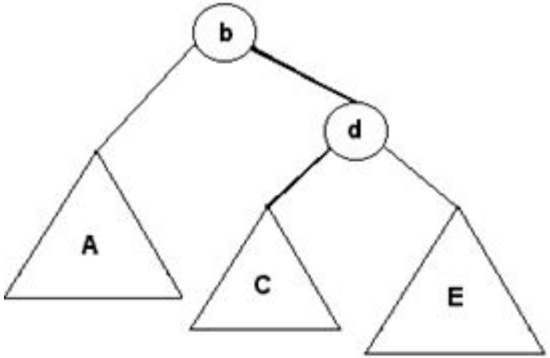
El desbalance por inserción "hacia afuera" con respecto a N se soluciona con una rotación simple que puede ser a izquierda o derecha dependiendo de dónde estaba el desbalance.

La inserción "hacia adentro" requiere de una rotación doble, es decir, dos rotaciones simples.

Comencemos viendo como serían las rotaciones simples y luego, las dobles.

Inserción en un AVL – Rotación simple

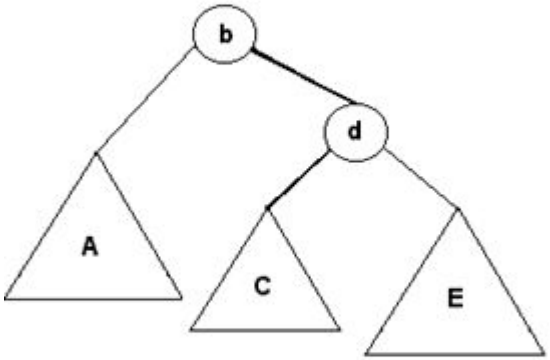
Supongamos que tenemos un Árbol AVL al que se le inserta un nodo de forma tal que deja de cumplir la condición. Es decir, tenemos este árbol, que representa nuestro caso 4.



El elemento X fue insertado en E y dejó de cumplir la condición, ya que hay 2 niveles de diferencia entre el subárbol izquierdo y el derecho de b.

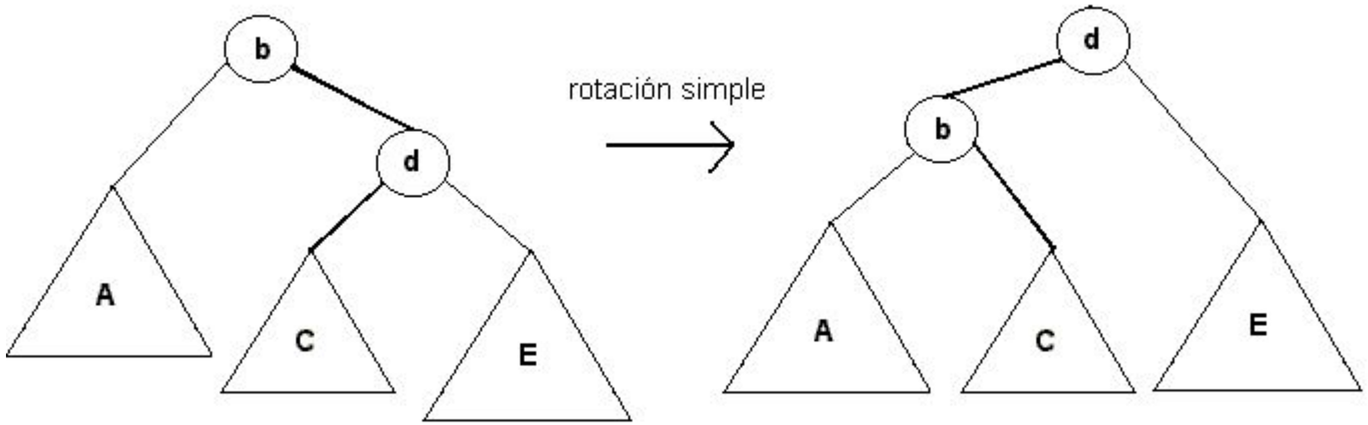
Inserción en un AVL – Rotación simple

Para recuperar la condición de balance se necesitaría bajar A en un nivel y subir E en un nivel. Esto lo vamos a lograr cambiando las referencias derecha de b e izquierda de d, quedando este último como nueva raíz del árbol.



Inserción en un AVL – Rotación simple

Es decir, nuestro árbol se transforma usando una rotación simple a izquierda. Podemos verlo así:



Inserción en un AVL – Rotación simple

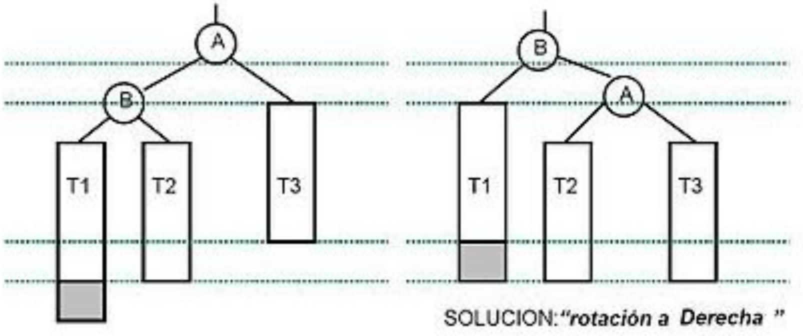
Resumiendo, supongamos que antes de la inserción, la altura de nuestro Árbol AVL era de C+1.

Con el movimiento que hicimos el nuevo árbol tiene la misma altura que antes de insertar el elemento, es decir, C+1. Esto implica que no puede haber nodos desbalanceados más arriba en el árbol, por lo que es necesaria una sola rotación simple para devolver la condición de balance al árbol.

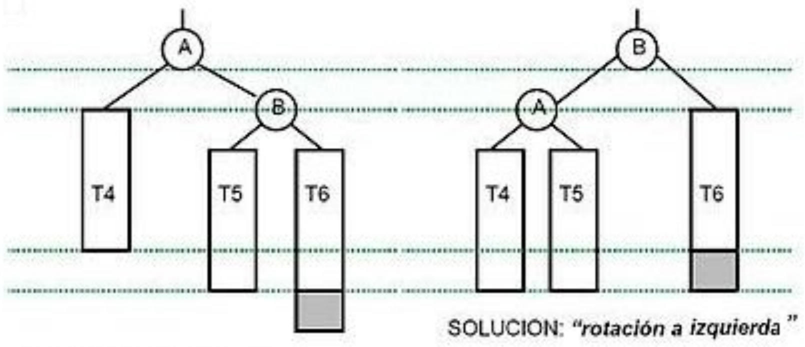
El árbol obtenido, de esta manera, es un Árbol AVL.

Inserción en un AVL – Rotación simple

Resumiendo, nuestras rotaciones simples a derecha e izquierda respectivamente serían:



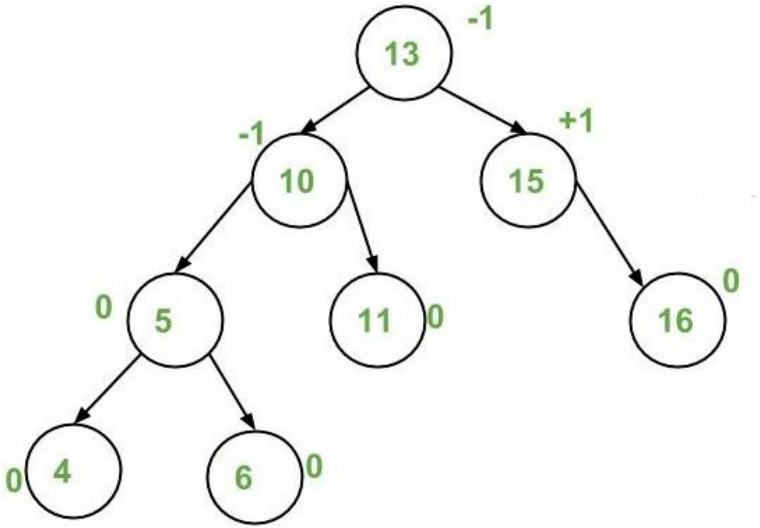
Caso 1: Izquierda-Izquierda



Caso 4: Derecha- Derecha

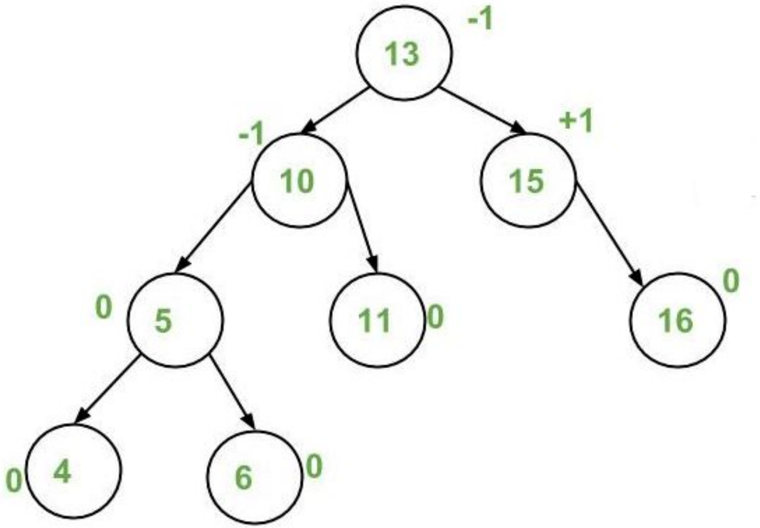
Inserción en un AVL – Rotación simple

Veamos un ejemplo práctico de esto. Supongamos que tenemos el siguiente Árbol AVL.



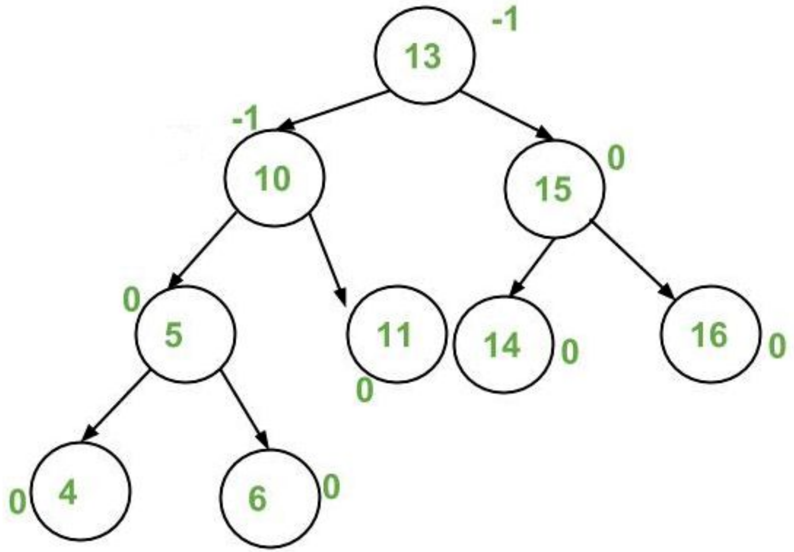
Inserción en un AVL – Rotación simple

En cada nodo se almacena un valor entero y, el número que aparece al costado es el resultado de la función `btree_balance_factor`.



Inserción en un AVL – Rotación simple

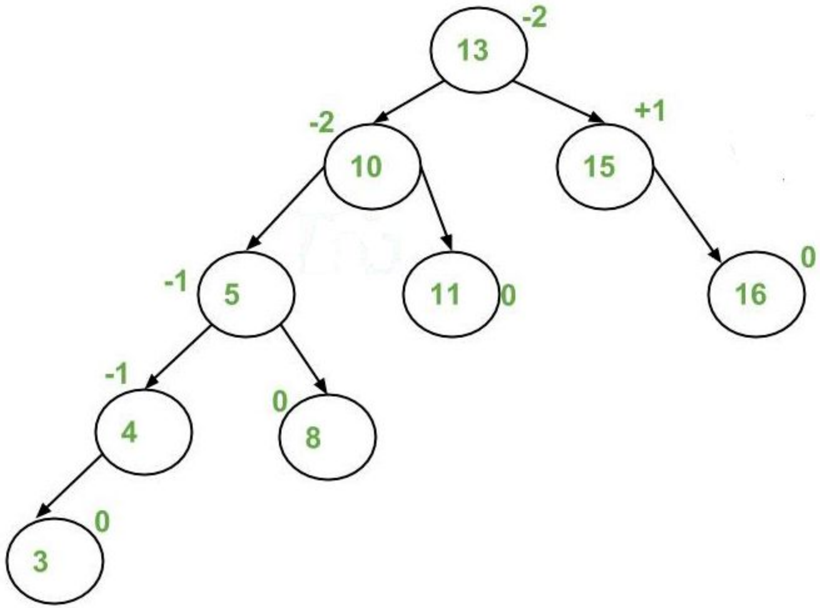
Supongamos que insertamos el valor 14 al árbol. Lo que obtenemos sería:



El cual sigue estando balanceado. O sea, no tenemos que aplicar ninguna rotación.

Inserción en un AVL – Rotación simple

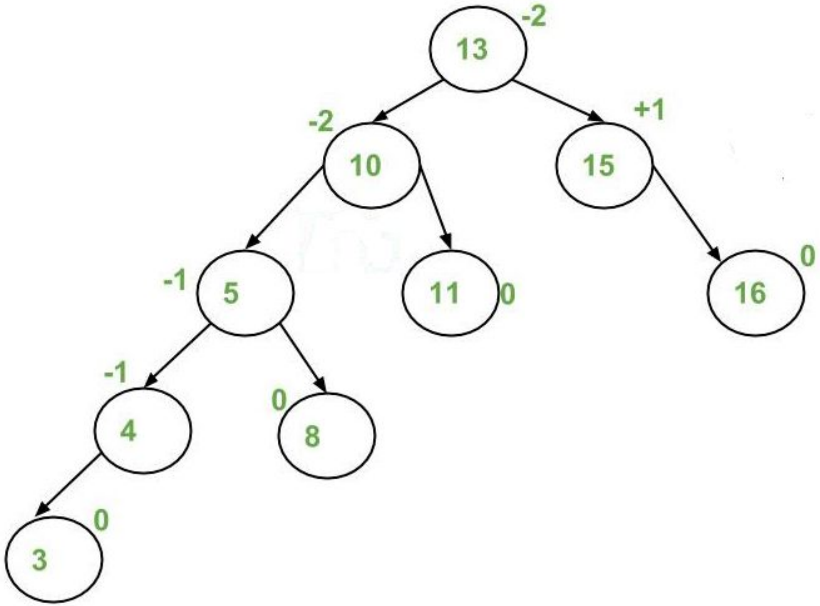
Pero, qué pasaría si, en lugar del 14 insertamos el valor 3. El árbol obtenido sería:



El cual NO está balanceado.

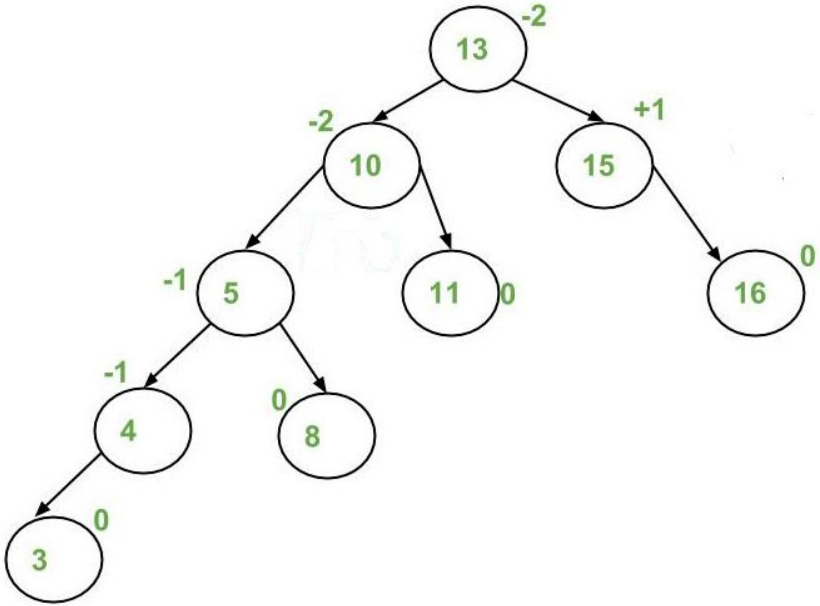
Inserción en un AVL – Rotación simple

Para poder balancearlo tenemos que identificar el nodo más profundo que está desbalanceado. Ese nodo es el que tiene el valor 10.



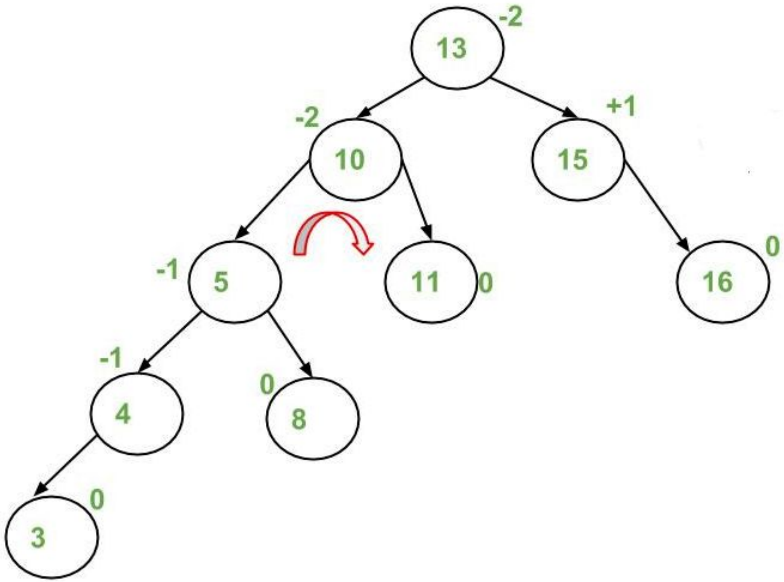
Inserción en un AVL – Rotación simple

Si lo analizamos podemos ver que estamos en el caso 1 (de los mencionados en el slide 11), ya que el desbalance está en el subárbol izquierdo del hijo izquierdo del nodo. O sea que nuestra solución debería ser una rotación a derecha.



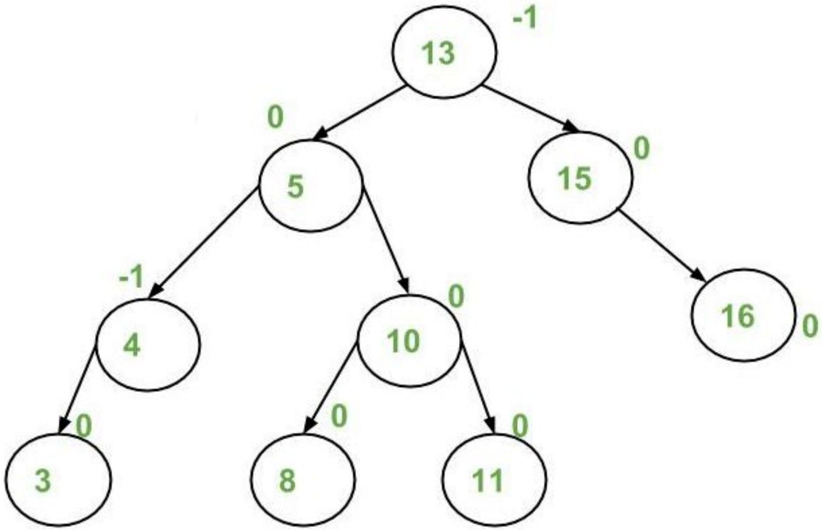
Inserción en un AVL – Rotación simple

Es decir, dado que hay dos niveles más a izquierda que a derecha, tenemos que hacer una rotación a derecha para poder subir el Nodo con el 5 y bajar el Nodo con el 10.



Inserción en un AVL – Rotación simple

Obteniéndose el siguiente árbol el cual es un Árbol AVL.



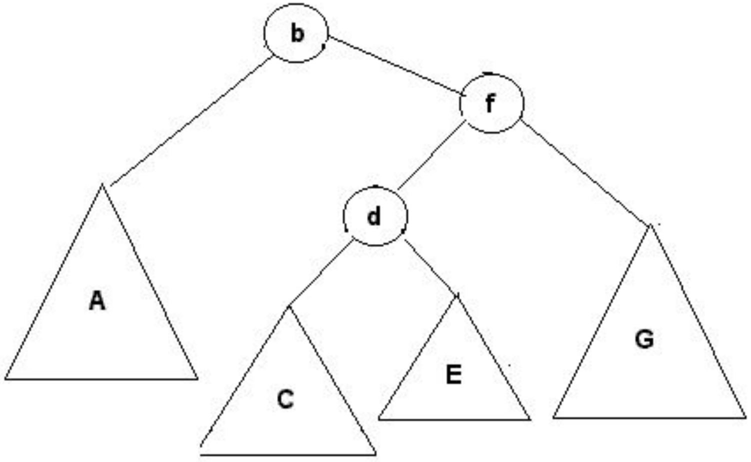
Inserción en un AVL – Rotación doble

Volviendo a nuestra motivación original, cuando insertamos nos quedan analizar estas dos situaciones:

2. El elemento X fue insertado en el subárbol derecho del hijo izquierdo de N.
3. El elemento X fue insertado en el subárbol izquierdo del hijo derecho de N.

Inserción en un AVL – Rotación doble

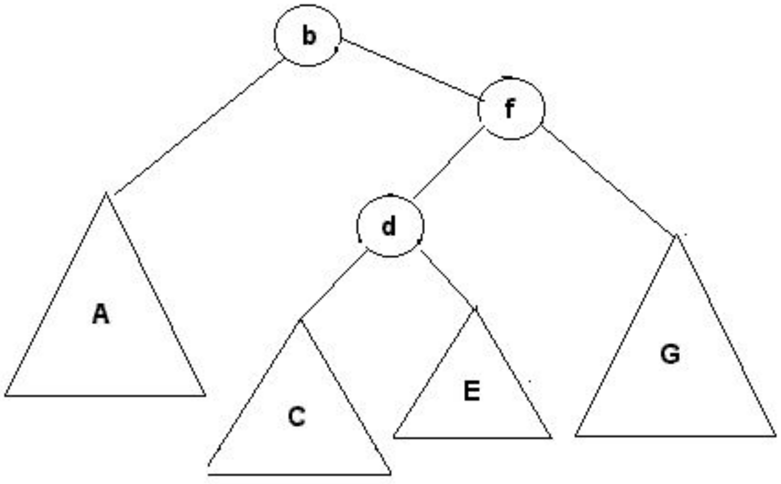
Veamos este ejemplo que corresponde al caso 3. Supongamos que nuestro elemento fue insertado en el subárbol C lo que produce el desbalance. Es decir, en el subárbol izquierdo del hijo derecho de b.



Inserción en un AVL – Rotación doble

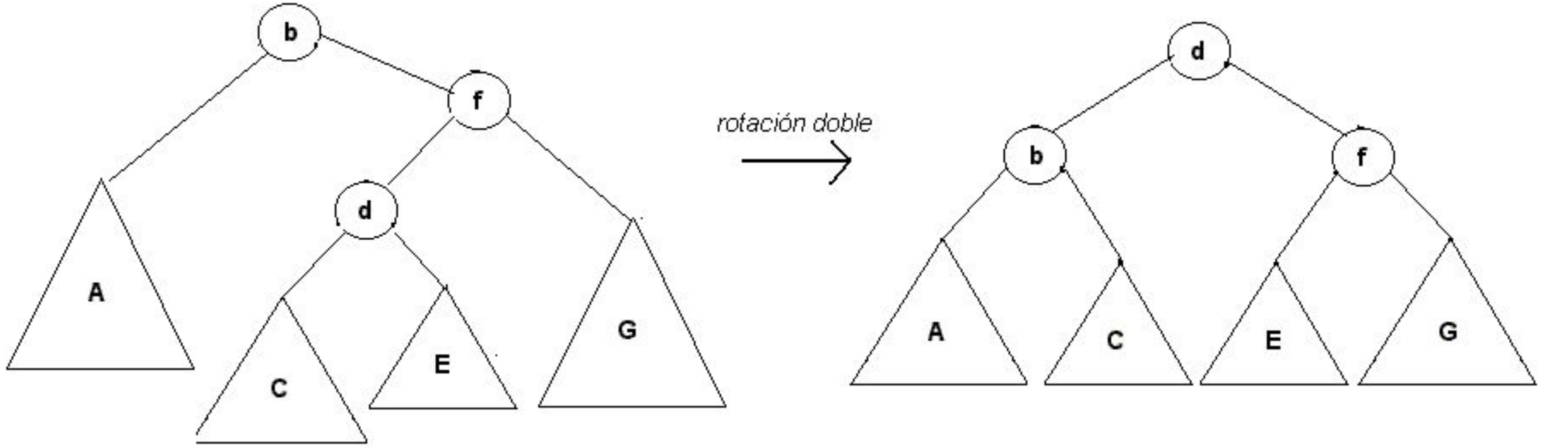
O sea, tenemos un desbalance producido por una inserción "hacia adentro" con respecto a b.

Para recuperar el balance del árbol es necesario subir C y bajar A pero, como están en diferentes ramas tenemos primero que subir a d para luego poder bajar a b.



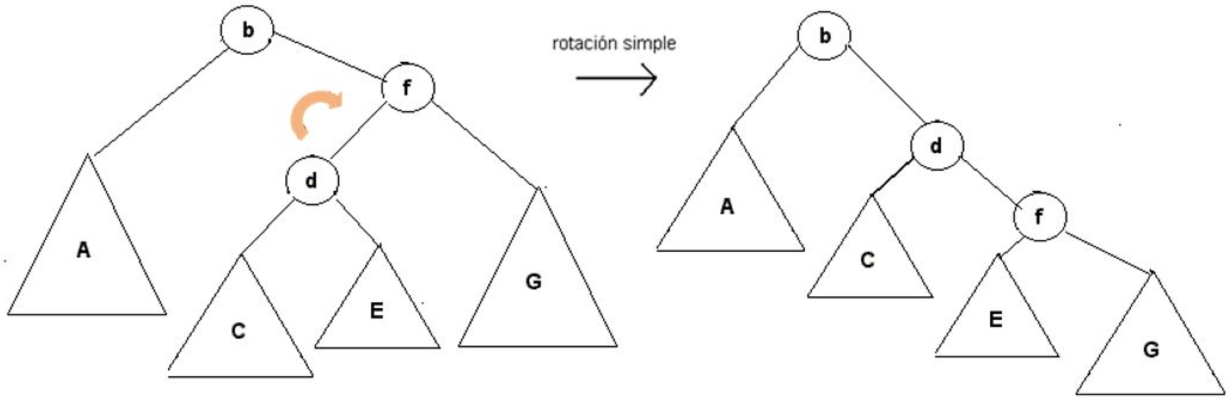
Inserción en un AVL – Rotación doble

Por este motivo necesitamos aplicar dos rotaciones simples: la primera entre d y f, y la segunda entre d, ya rotado, y b, obteniéndose el resultado de la figura.
Vamos a ver el paso a paso de las rotaciones para transformar nuestro árbol en un Árbol AVL.



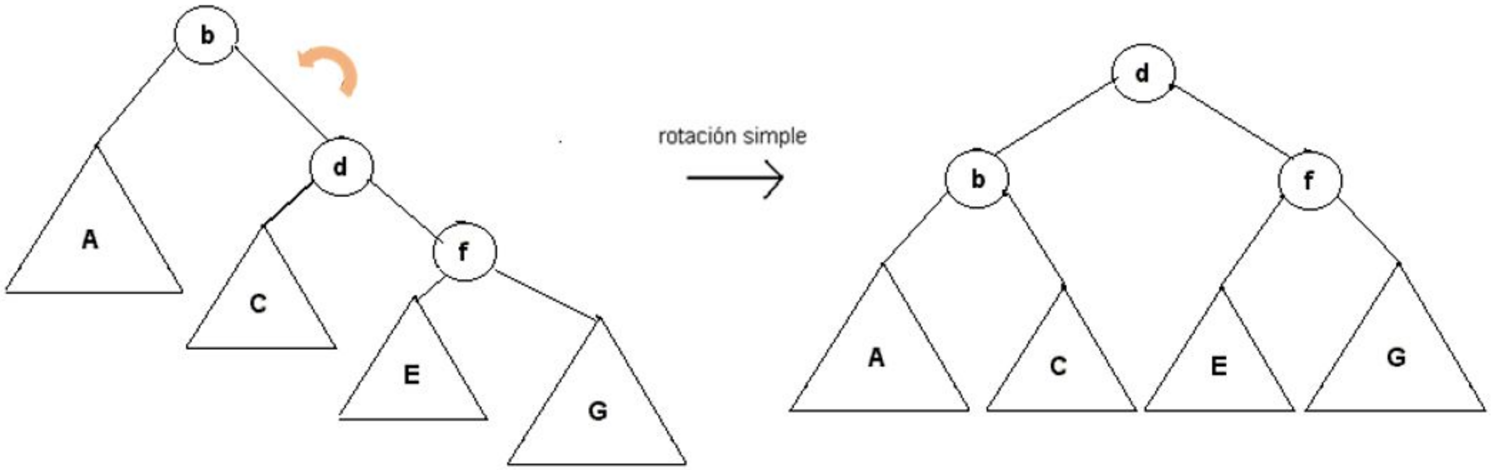
Inserción en un AVL – Rotación doble

Paso 1: aplicamos una rotación simple a derecha para llegar a un estado intermedio y subir C.



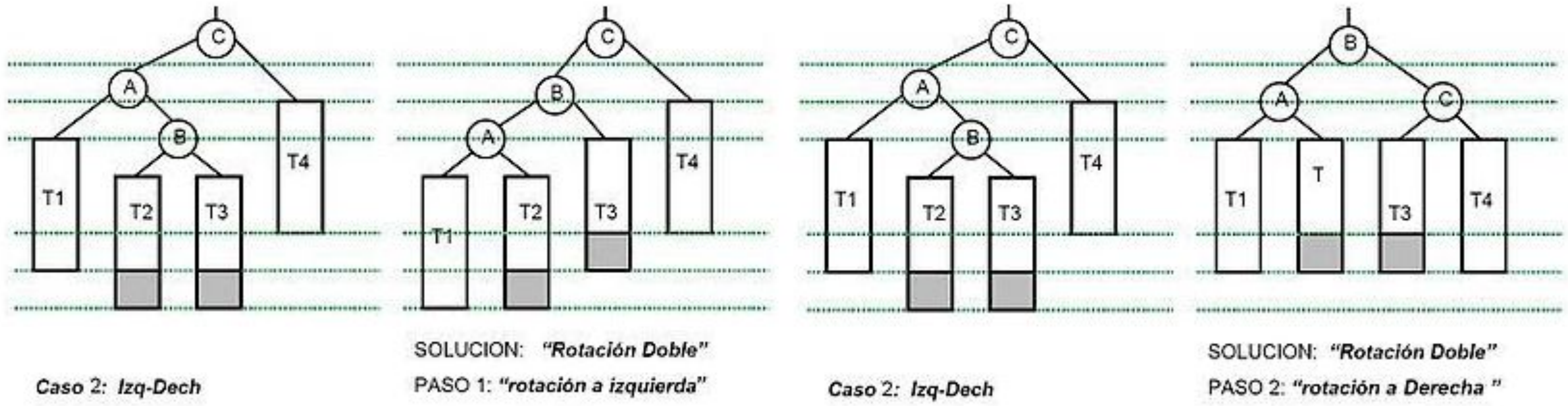
Inserción en un AVL – Rotación doble

Paso 2: volvemos a aplicar una rotación simple pero a izquierda para bajar A y finalmente alcanzar nuestro objetivo: tener un Árbol AVL.



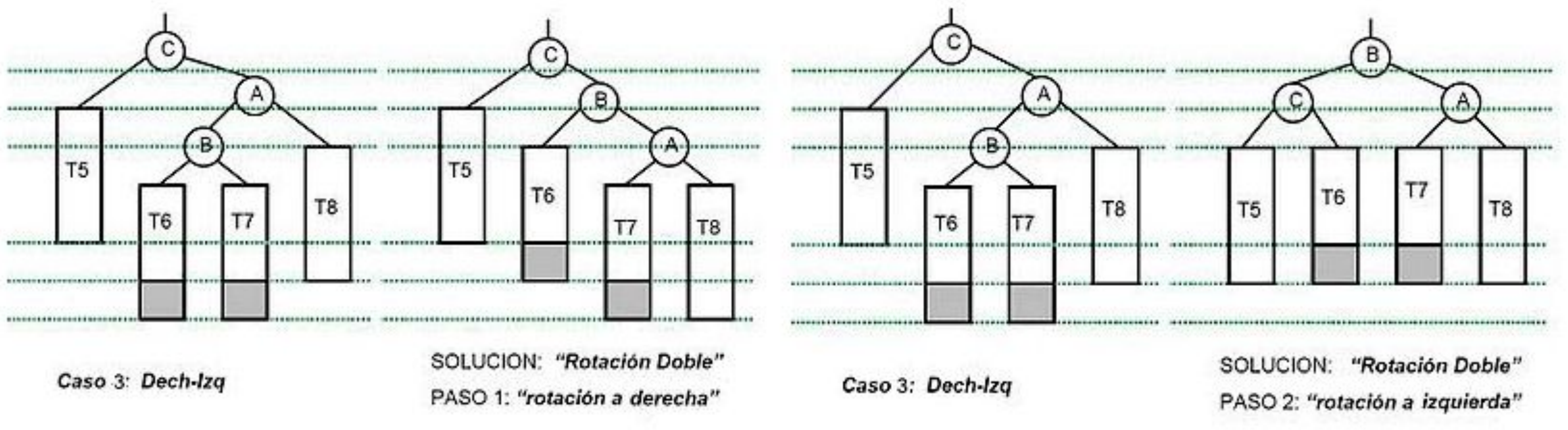
Inserción en un AVL – Rotación doble

Acá tenemos ejemplificados los casos



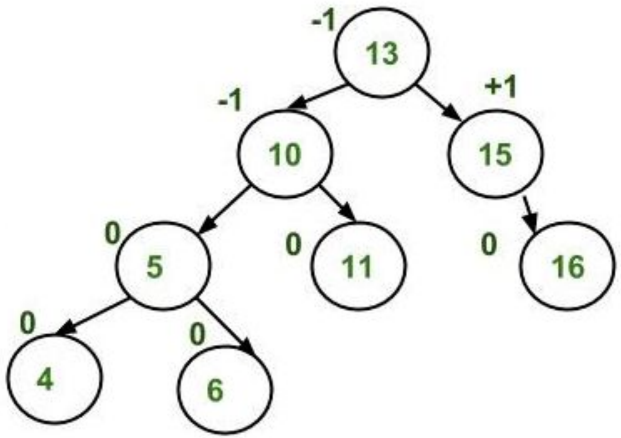
Inserción en un AVL – Rotación doble

También se puede hacer primero una rotación a izquierda y, luego una a derecha:



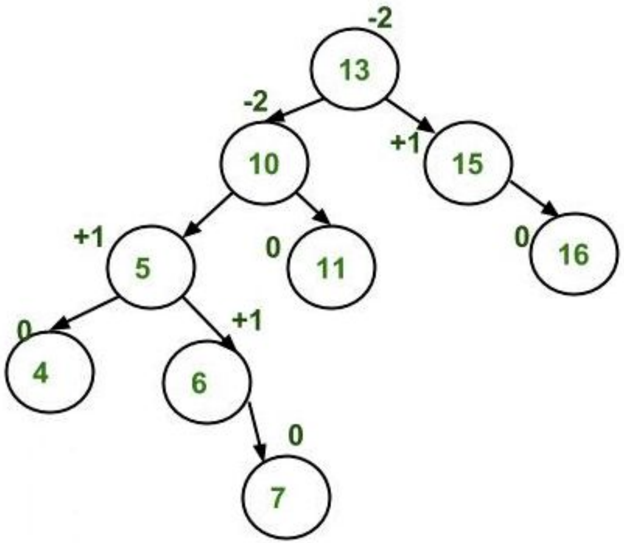
Inserción en un AVL – Rotación doble

Veamos un ejemplo práctico del uso de rotaciones dobles. Supongamos que tenemos el siguiente Árbol AVL.



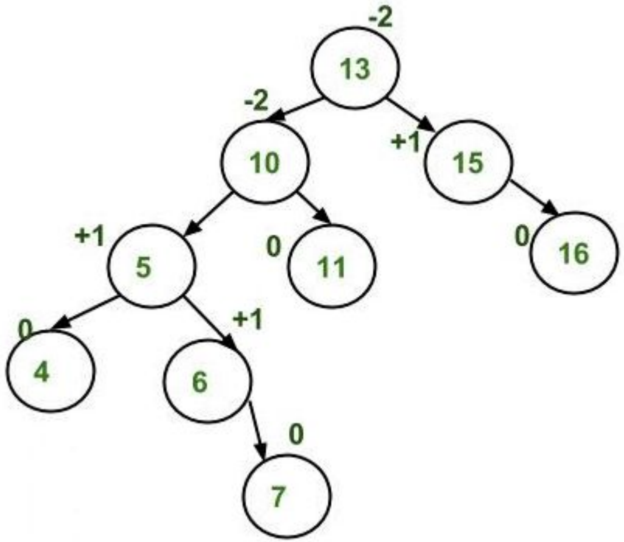
Inserción en un AVL – Rotación doble

Si insertamos el valor 7 obtenemos el siguiente árbol, el cuál está desbalanceado.



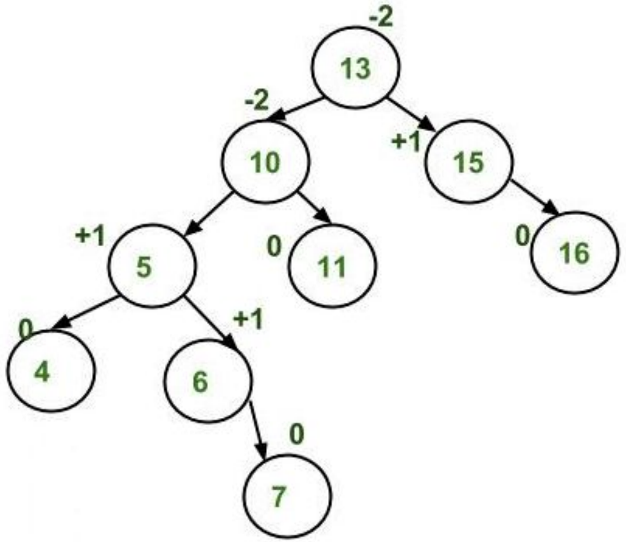
Inserción en un AVL – Rotación doble

El nodo más profundo desbalanceado es el que tiene el valor 10. Vemos que estamos en el caso 2 (slide 11), el nodo fue insertado en el subárbol derecho del hijo izquierdo del nodo 10.



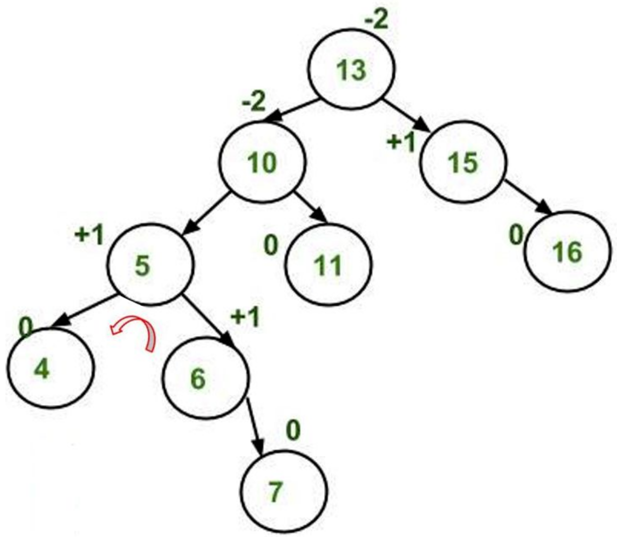
Inserción en un AVL – Rotación doble

Para resolverlo tenemos entonces que hacer una rotación doble.
En este caso sería, primero a izquierda para subir el nodo con el 7 (esto se realiza sobre el nodo 5) y, luego a derecha sobre el nodo 10.



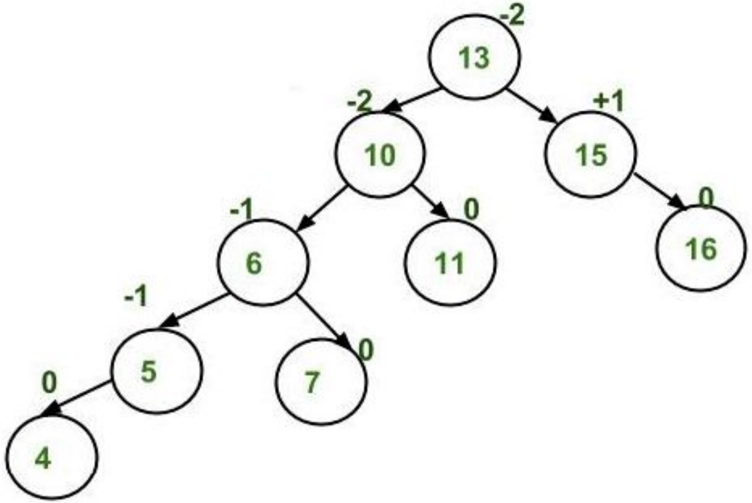
Inserción en un AVL – Rotación doble

Comencemos con la primera rotación simple a izquierda.



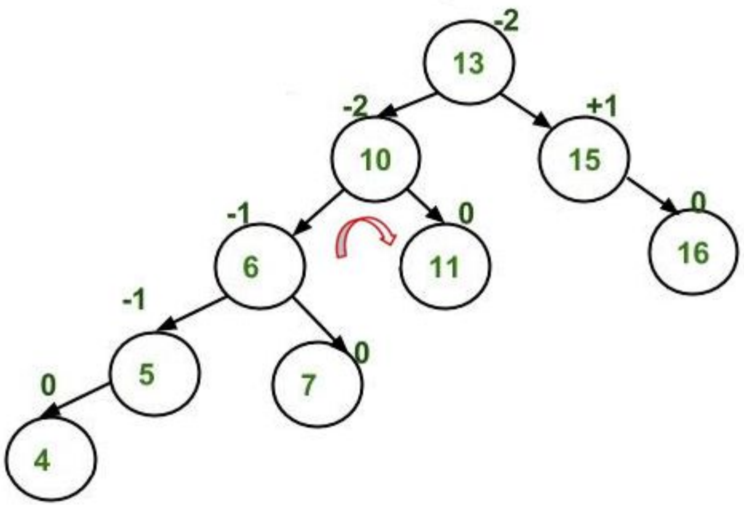
Inserción en un AVL – Rotación doble

El resultado de esta rotación a izquierda sería:



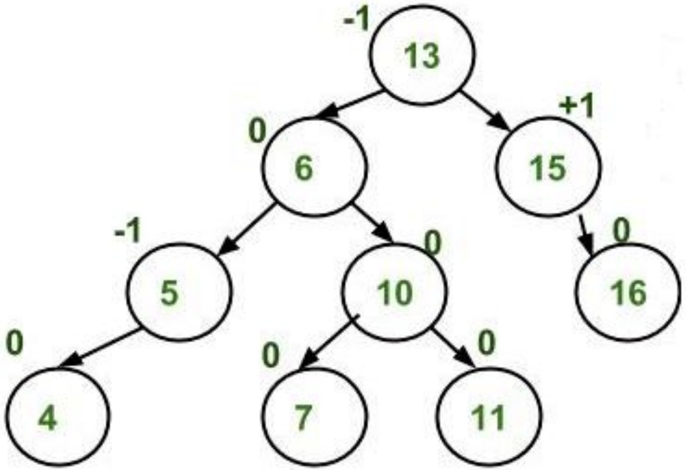
Inserción en un AVL – Rotación doble

Ahora, tenemos que hacer la rotación a derecha sobre el 10 ya que el desbalance está en el subárbol izquierdo del hijo izquierdo del nodo.



Inserción en un AVL – Rotación doble

Finalmente, luego de esta rotación, obtenemos el siguiente árbol que sí es un Árbol AVL.



Eliminación en un AVL

La eliminación en árbol AVL se realiza de manera análoga a un ABB, pero también es necesario verificar que la condición de balance se mantenga una vez eliminado el elemento.

En caso que dicha condición se pierda, será necesario realizar una rotación simple o doble dependiendo del caso, pero es posible que se requiera más de una rotación para reestablecer el balance del árbol.