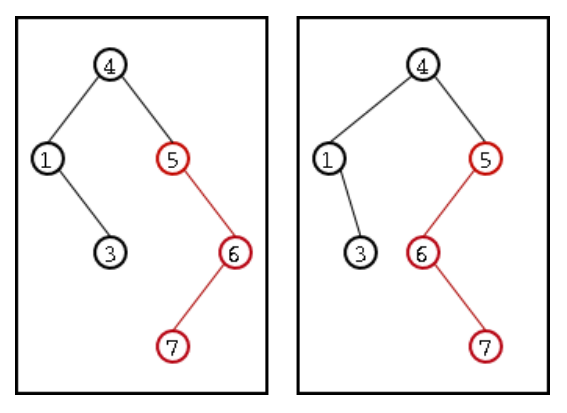
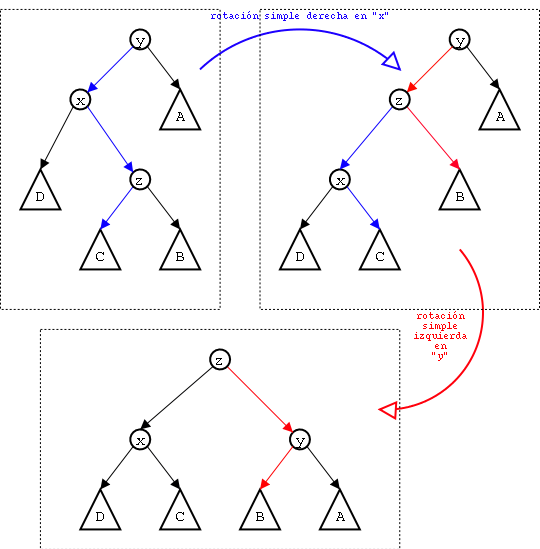
**Rotaciones dobles**

Al hacerse inserciones en un árbol AVL sabemos que se pueden presentar desequilibrios tanto hacia la derecha como hacia la izquierda, pero el problema sucede que al equilibrarlo pueden quedar desequilibrios que no son ni a la derecha ni a la izquierda “zig-zag” como lo podemos ver en la imagen debajo.



Cuando suceden estos casos entonces se aplica otro tipo de rotación llamado “Rotación Doble” que puede ser tanto a la derecha como a la izquierda.

Esta rotación consta de dos simple rotaciones. En el ejemplo siguiente lo podemos ver que está rotando del lado izquierdo. Si el proceso sucede del lado derecho hiciera el mismo efecto.



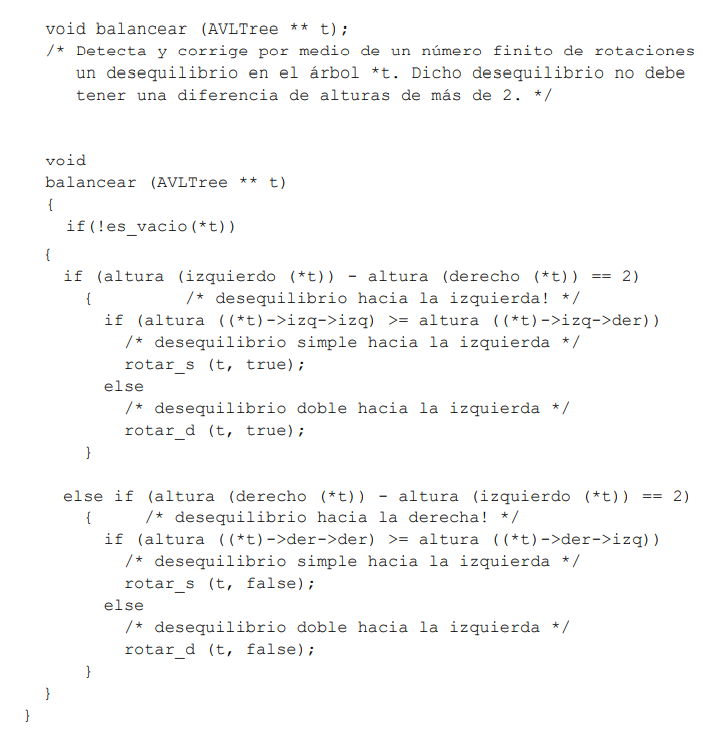
**Balance del árbol**

Como pudimos ver en el punto anterior, cada vez que se modifique el árbol tenemos un porciento de riesgo de que se desequilibre el algún o algunos de los nudos, la cual debe conservar su queremos obtener tiempos de ejecución de orden **O(log(n))** en el peor de los casos.

La idea que podemos obtener de la implementación de árboles AVL para la implementación de inserción y eliminación de nodos es la siguiente:

* Ejecutar los algoritmos de la misma manera que el árbol binario, pero en cada recursión se actualiza las alturas y se re balancea el árbol si es necesario.

Lo que debemos hacer es un método que detecte un desequilibrio en un nodo del arbol y por medio de una cantidad de rotaciones lo equilibre como podemos ver debajo.



Ese es la función para el equilibrio, pero veremos una imagen para ver eso más simple y explicando el código.

