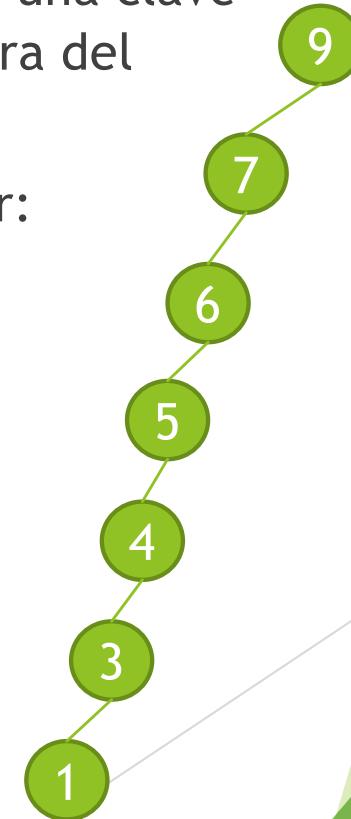
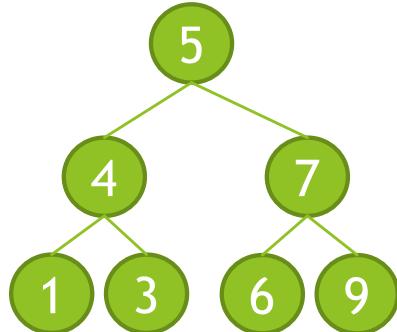


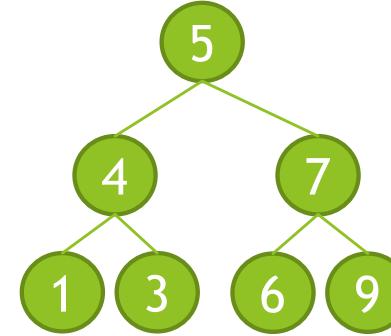
Balanceo de Arboles Binarios

Motivación

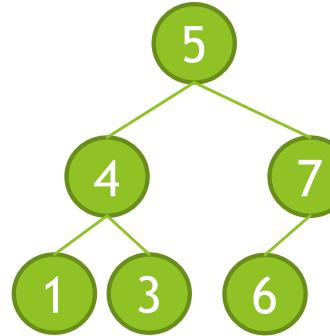
- ▶ Altura ideal $O(\log n)$
- ▶ Un ABB permite determinar la existencia de una clave en el árbol en tiempo proporcional a la altura del mismo.
- ▶ En un ABB de n nodos cualquiera, puede ser:
 - ▶ Peor de los casos, $O(n)$.
 - ▶ Mejor de los casos $O(\log n)$,



Árbol completo



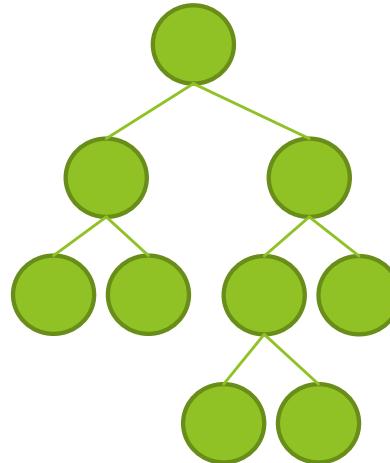
- ▶ Un árbol binario completo es también un árbol balanceado.
- ▶ Problema:
 - ▶ Extremadamente rígidos
Insertar o borrar un nodo genera desbalance.

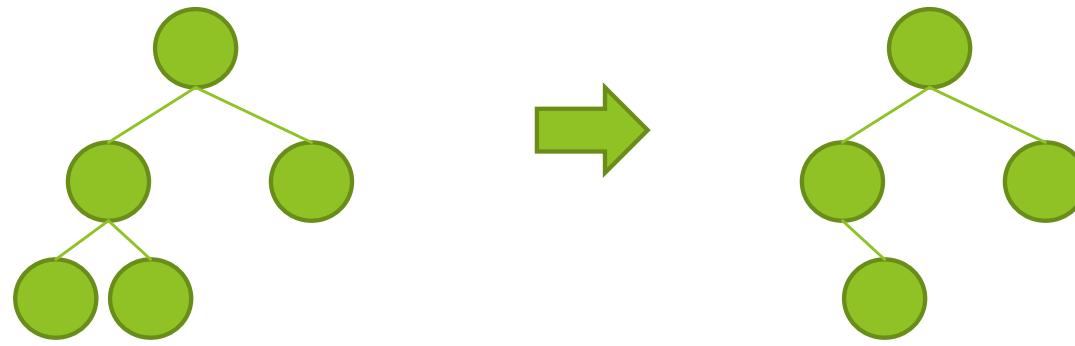


Balanceo perfecto

- ▶ Un árbol binario se dice *lleno* si todo tiene tiene 0 o 2 hijos
- ▶ Un árbol binario se dice perfectamente balanceado si:
 - ▶ Es un árbol lleno
 - ▶ Todas sus hojas se encuentran en el último o antepenúltimo nivel

Altura = $\log (n+1)$



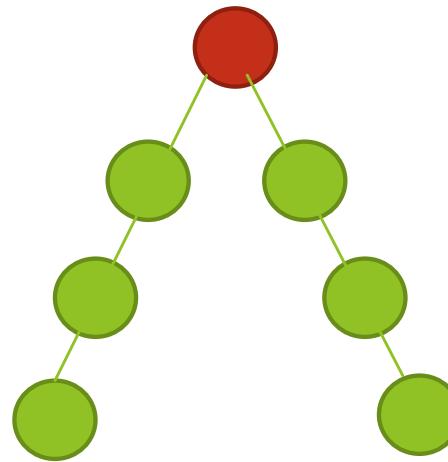


Un árbol perfectamente balanceado deja de serlo al eliminar un nodo

Balanceo en altura

- ▶ Un árbol binario se dice balanceado en altura si:
para cada nodo, la altura de su subárbol derecho e izquierdo
difieren a lo sumo en una unidad.

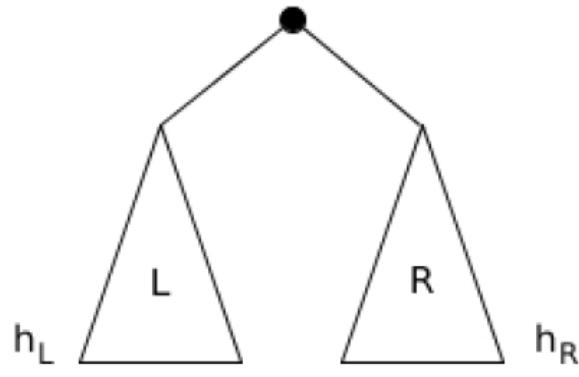
Balanceo en altura



Balanceo en peso

- ▶ Un árbol se dice balanceado en peso si para cada nodo, la *cantidad* de nodos en su subárbol derecho e izquierdo difieren a lo sumo en una unidad

$$\text{Altura} = \log(n+1)$$



$$| L - R | \leq 1$$

Problema

- ▶ Establecer las reglas para el rebalanceo.
- ▶ Estrategias:
 - ▶ Conocer la altura de cada subárbol
 - ▶ Conocer el peso de cada subárbol
 - ▶ ¿Re balancear al insertar?
 - ▶ ¿Re balancear al eliminar?
 - ▶ ¿Rebalanceo por requerimiento?

AVL

Adelson Velskii - Landis (1962)

An algorithm for the organization of information

- ▶ Los árboles AVL están *siempre* balanceados.
- ▶ La altura de sus ramas **no** difiere en más de una unidad

Costo se mantiene en
 $O(\log n)$

Factor de equilibrio

Puede ser:

Almacenado en cada nodo

Calculado a partir de la altura de los subárboles

$FE = \text{Altura del subárbol derecho} - \text{altura subárbol izquierdo}$

Para un AVL debe ser -1, 0, 1

0 : equilibrado

1: equilibrado subárbol derecho mas alto

-1: equilibrado subárbol izquierdo mas alto

$|FE| >= 2$ rebalancear

Rotación de un árbol



Operación que cambia la estructura de un arbol sin afectar el orden de sus elementos.



Se utiliza para balancear dos ramas de diferente profundidad.

Rotaciones



Pueden ser usadas a cualquier altura del árbol.

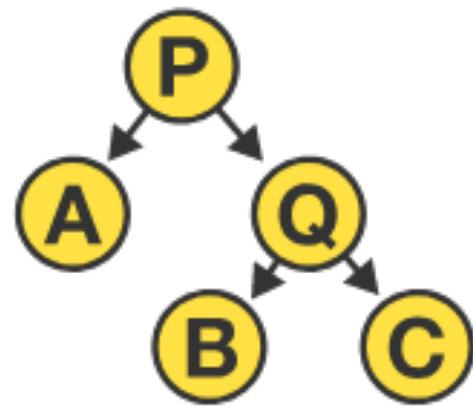
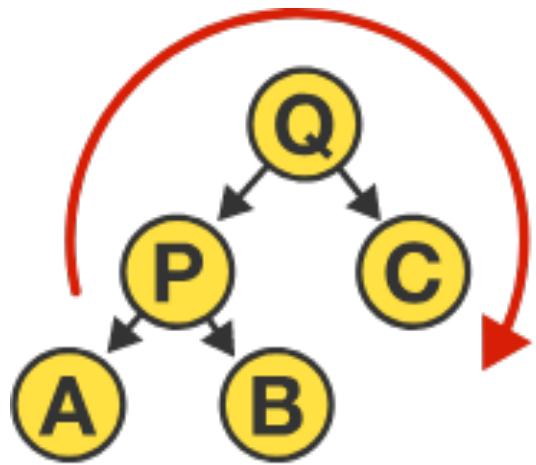


Regularmente se realizan de abajo hacia arriba sobre los nodos en que se produce el desequilibrio.

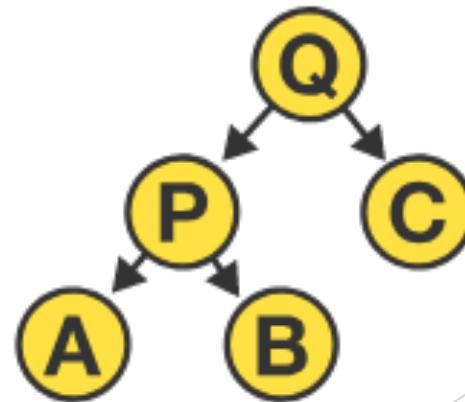
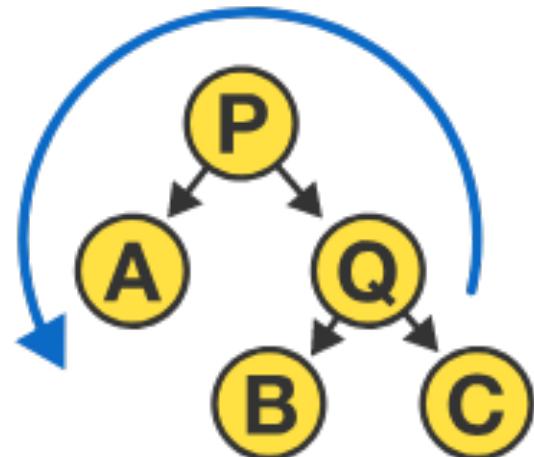


Pueden ser:

Rotación simple
(izquierda o derecha)
Rotación doble
(izquierda o derecha)

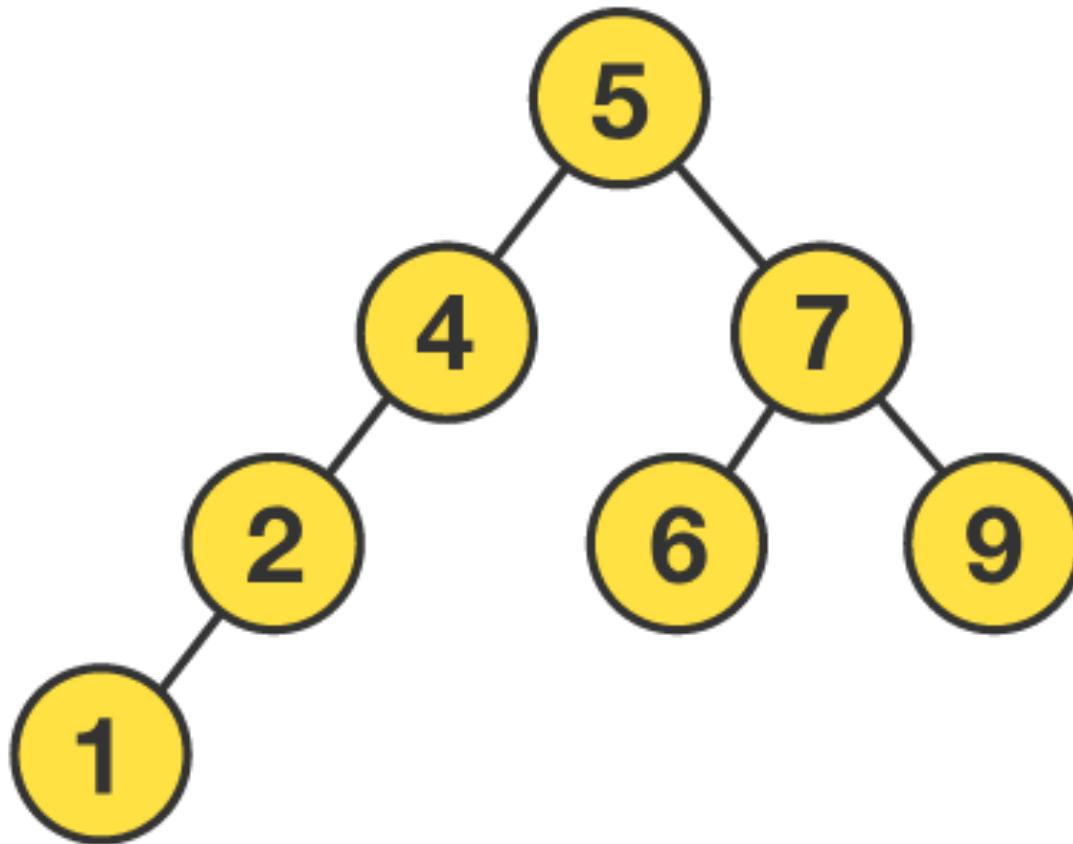


Right rotation

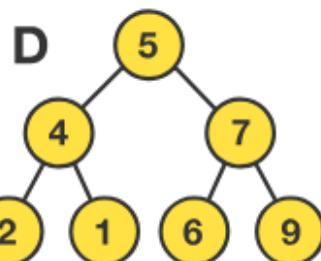
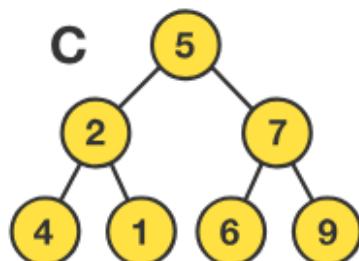
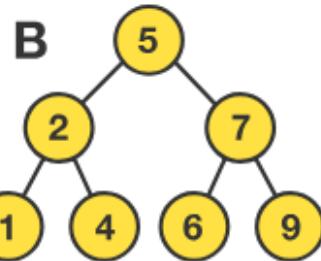
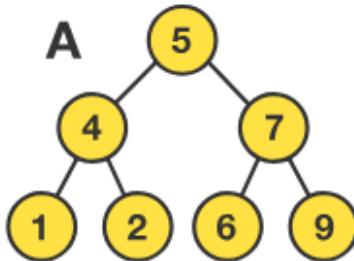
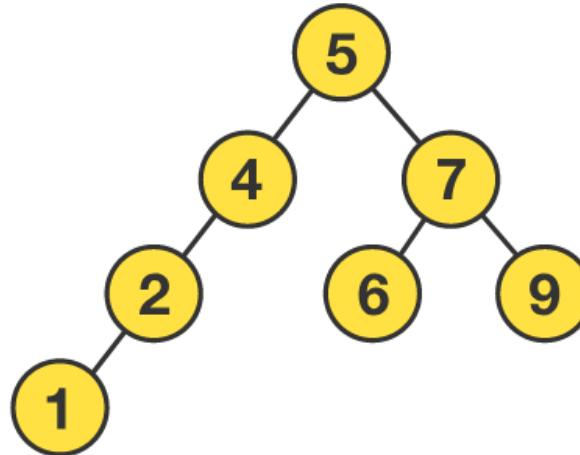


Left rotation

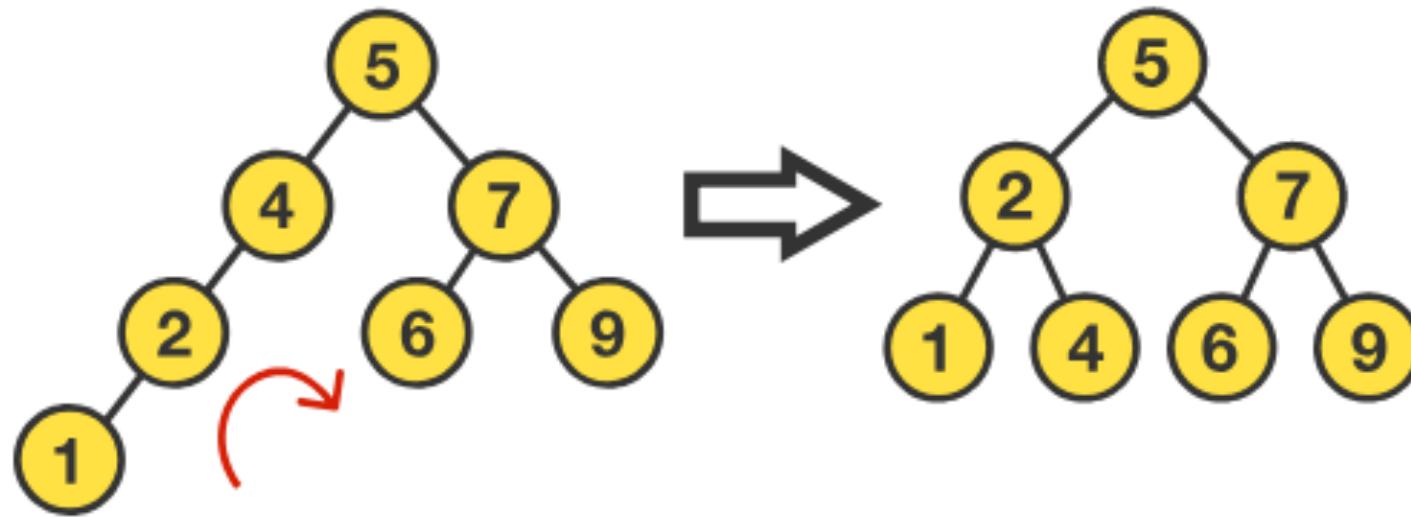
Ejercicio:



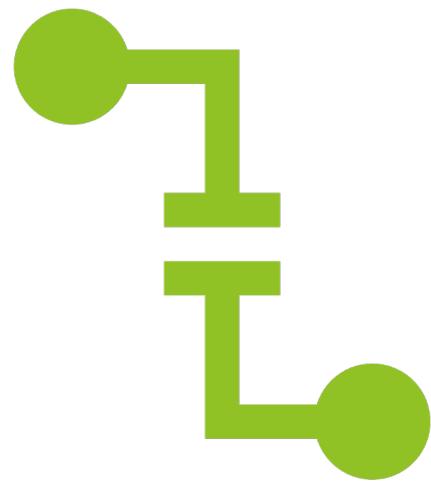
Ejercicio:



B

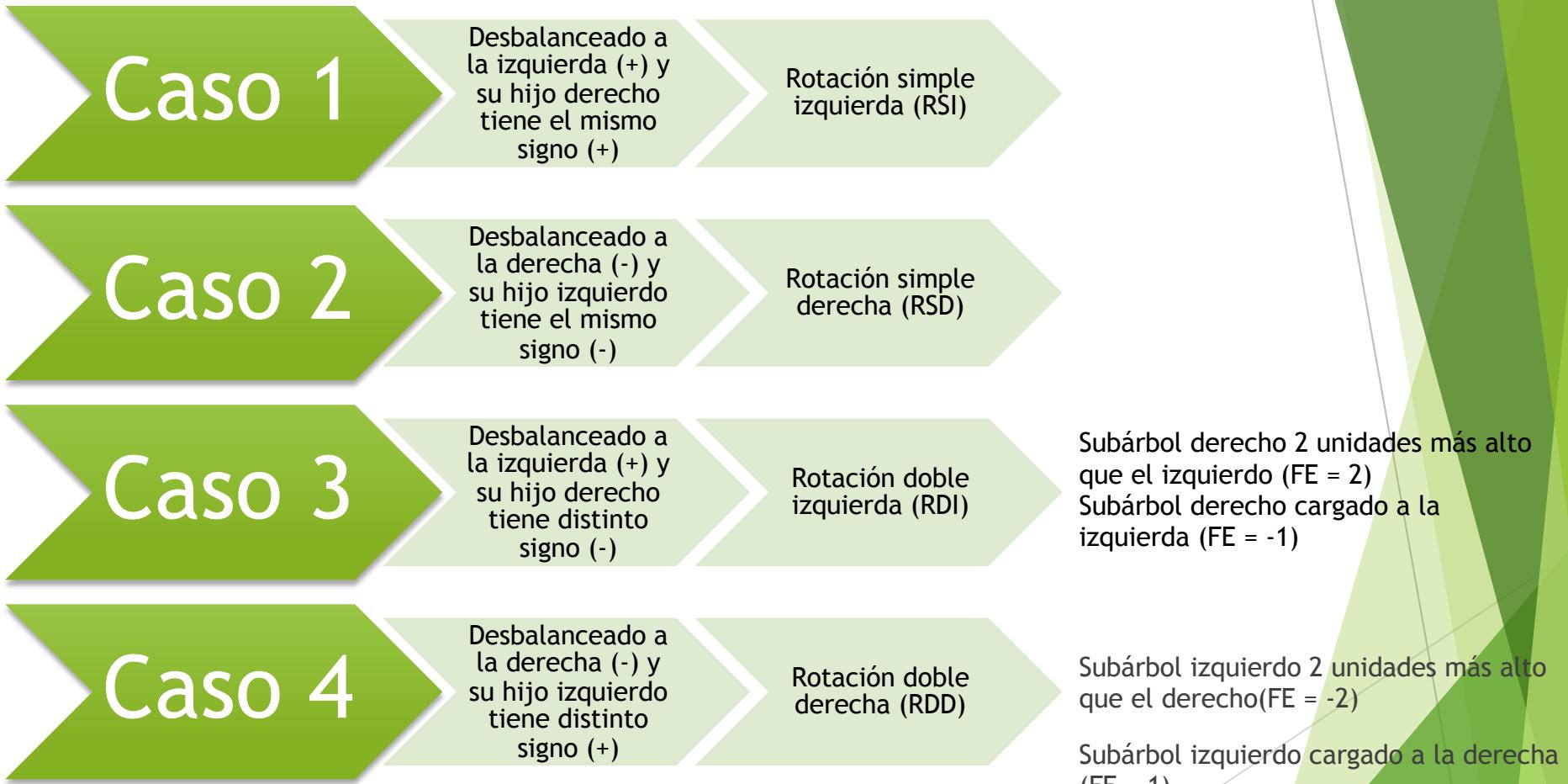


Procedimiento



- ▶ Después de *insertar* un nodo:
 - ▶ Inspeccionar la rama
 - ▶ Realizar las rotaciones.

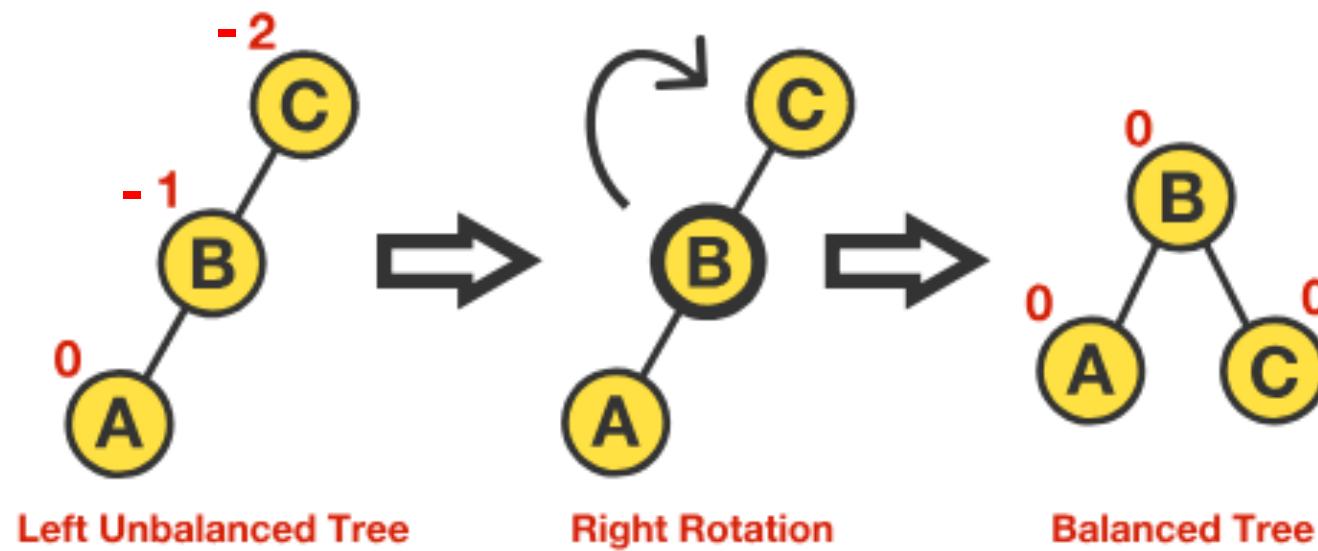
Elección de rotación adecuada



Elección de rotación adecuada

Factor de equilibrio	Hijo Izquierdo	Hijo Derecho	Rotación
-2	-		RSD
-2	+		RDD
+2		+	RSI
+2		-	RDI

Rotación simple a la derecha

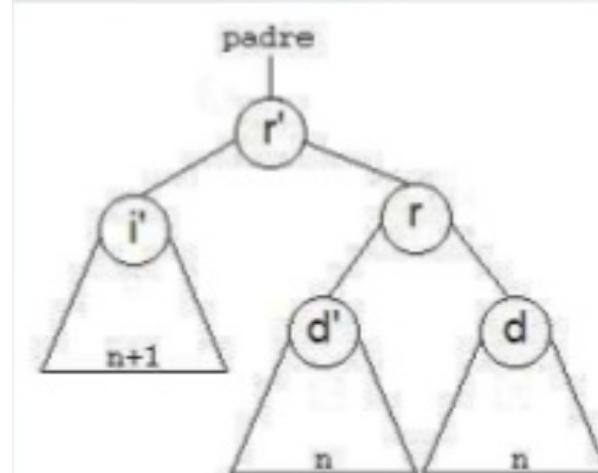
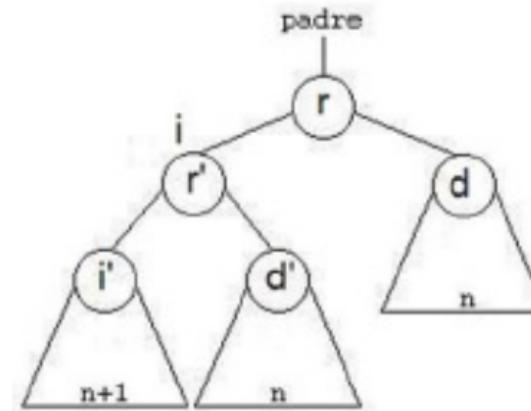


Rotación simple a la derecha

- ▶ De un árbol de raíz (r) y sus hijos izquierdo (i) y derecho (d), se formará un nuevo árbol que:
 - ▶ Raíz (r') : hijo izquierdo
 - ▶ Hijo izquierdo (i'): hijo izquierdo de i
 - ▶ Hijo derecho:
 - ▶ Raíz: raíz original (r)
 - ▶ Hijo izquierdo: hijo derecho de i
 - ▶ Hijo derecho: hijo derecho

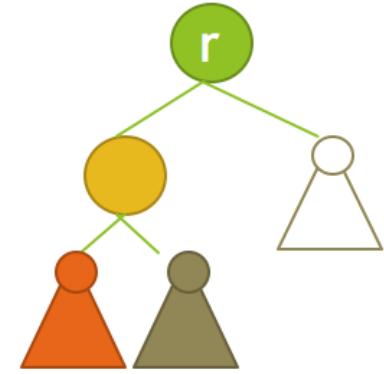
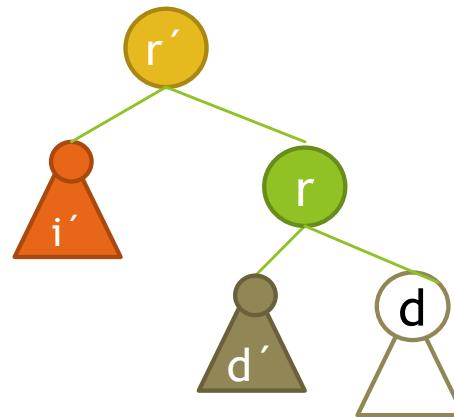
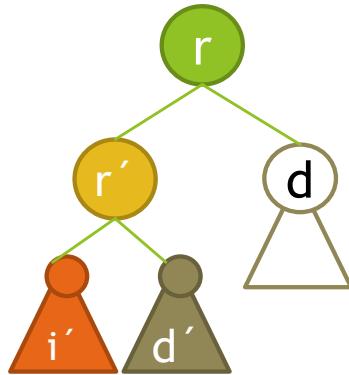
Rotación simple a la derecha

- ▶ De un árbol de raíz (r) y sus hijos izquierdo (i) y derecho (d), se formará un nuevo árbol que:
 - ▶ Raíz (r'): hijo izquierdo
 - ▶ Hijo izquierdo (i'): hijo izquierdo de i
 - ▶ Hijo derecho:
 - ▶ Raíz: raíz original (r)
 - ▶ Hijo izquierdo: hijo derecho de i
 - ▶ Hijo derecho: hijo derecho

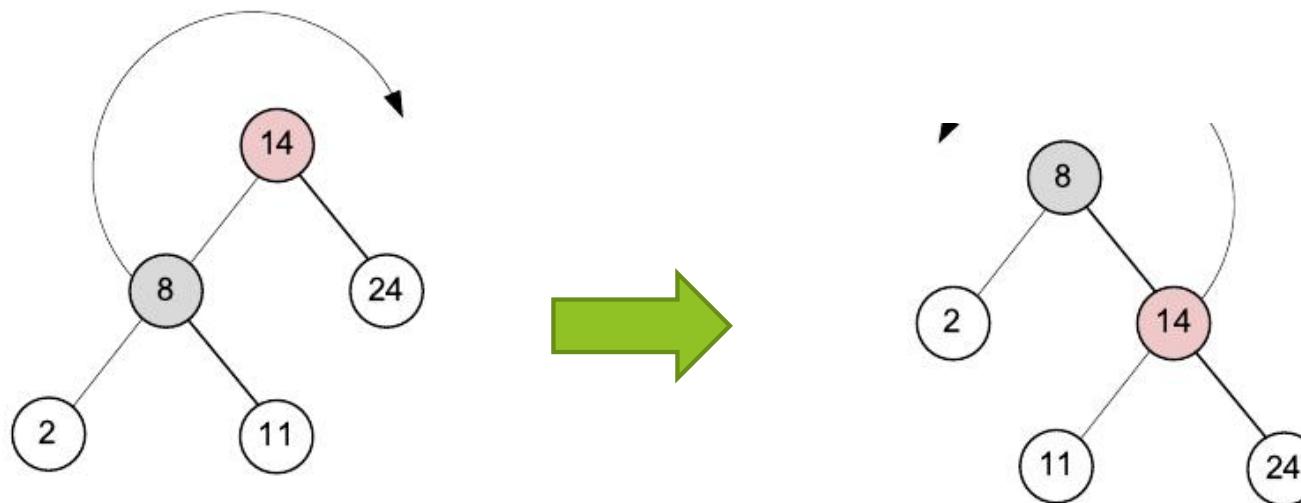


Rotación simple a la derecha

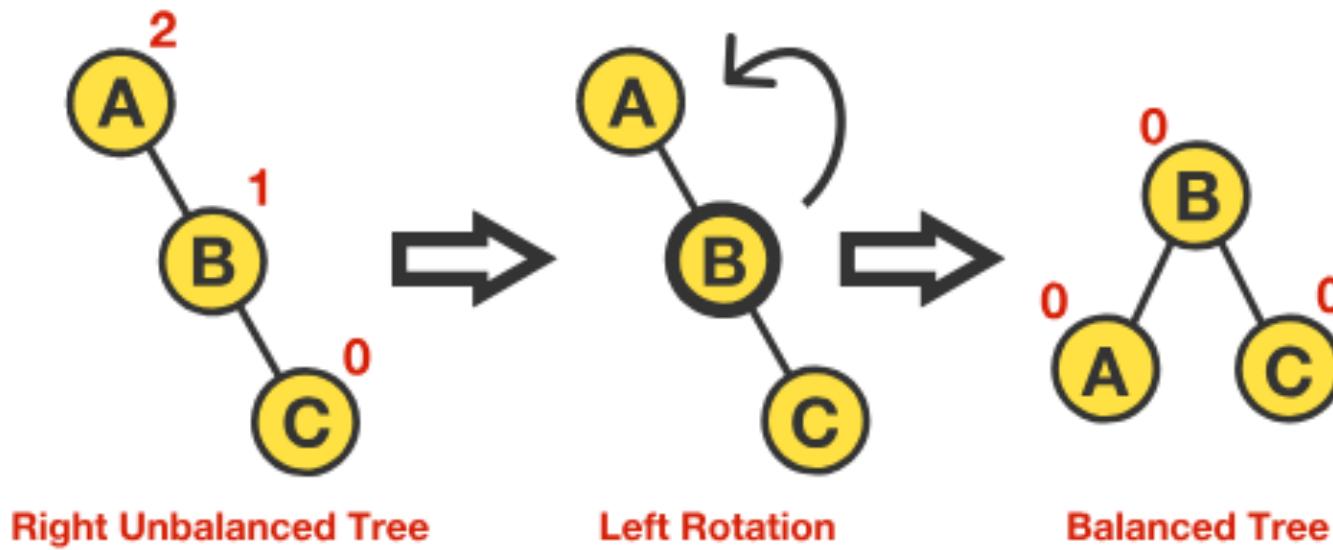
- ▶ De un árbol de raíz (r) y sus hijos izquierdo (i) y derecho (d), se formará un nuevo árbol que:
 - ▶ Raíz (r'): hijo izquierdo
 - ▶ Hijo izquierdo (i'): hijo izquierdo de i
 - ▶ Hijo derecho:
 - ▶ Raíz: raíz original (r)
 - ▶ Hijo izquierdo: hijo derecho de i
 - ▶ Hijo derecho: hijo derecho



Rotación simple a la derecha



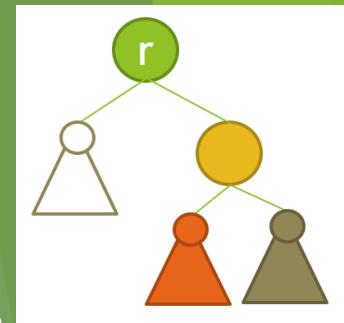
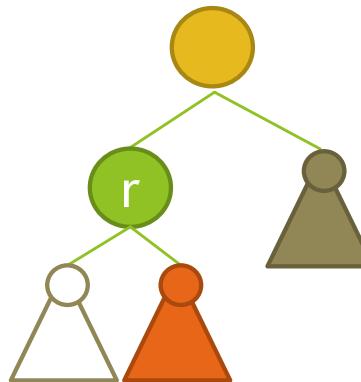
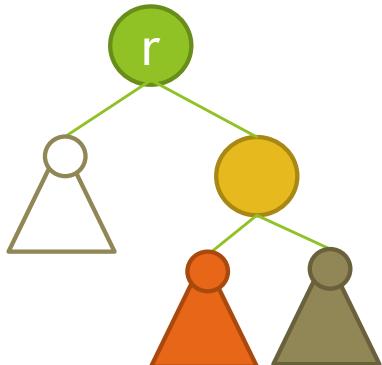
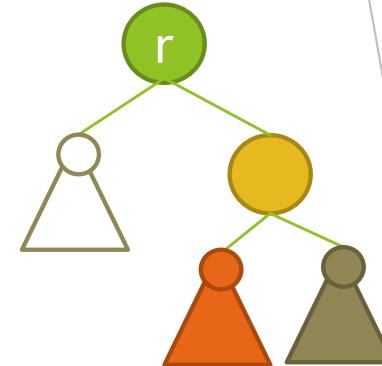
Rotación simple a la izquierda



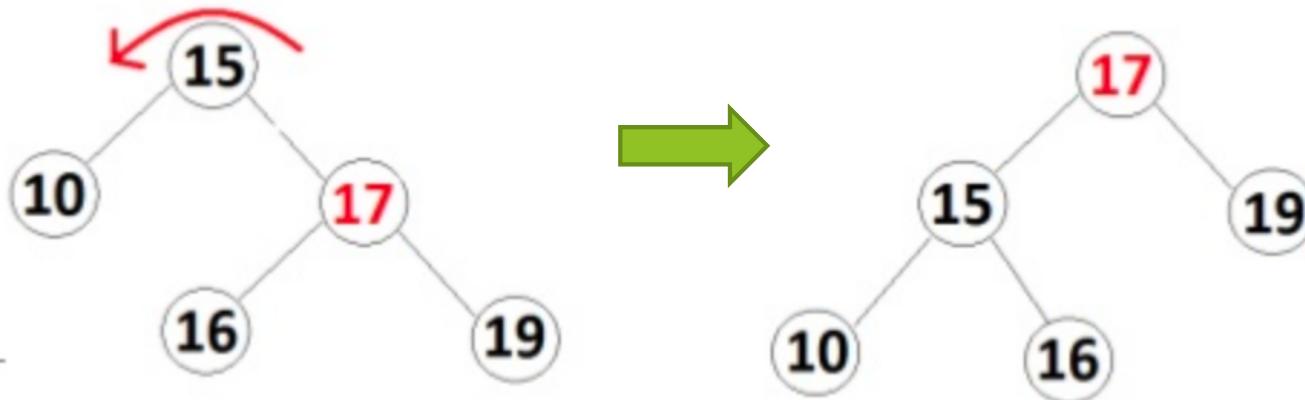
Rotación simple a la izquierda

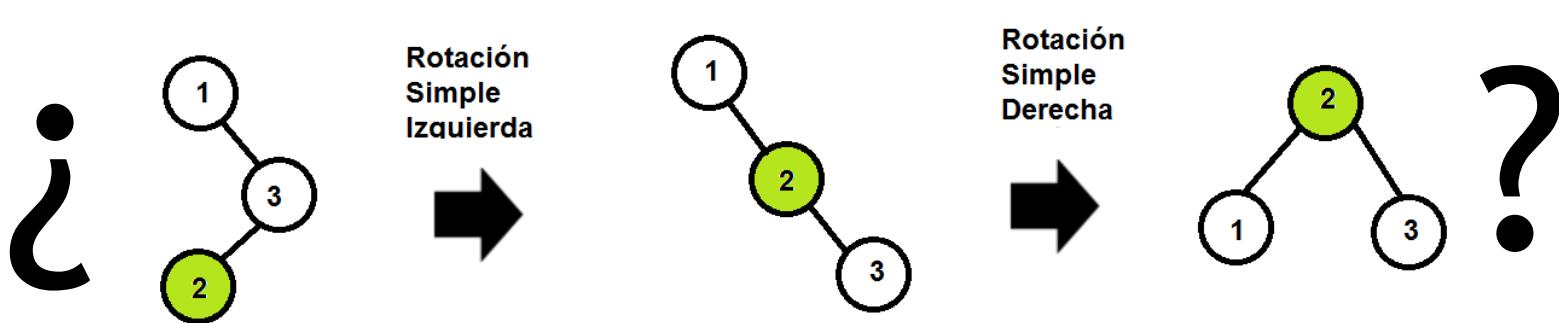
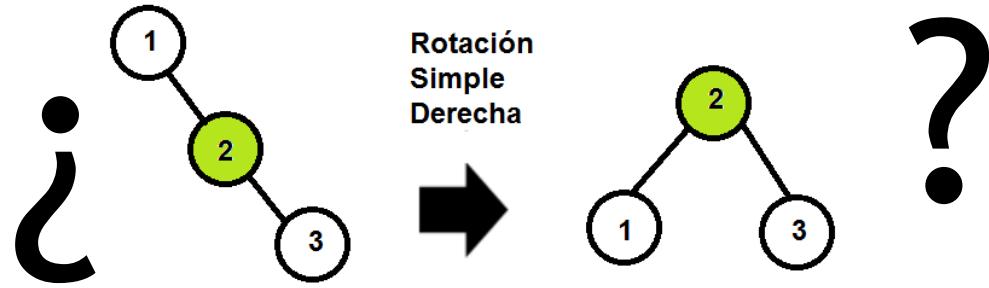
- ▶ De un árbol de raíz (r) y sus hijos izquierdo (i) y derecho (d), se formará un nuevo árbol que:

- ▶ Raíz (r') : hijo derecho
- ▶ Hijo derecho : hijo derecho del hijo derecho (d')
- ▶ Hijo izquierdo:
 - ▶ Raíz: raíz original (r)
 - ▶ Hijo derecho: hijo derecho de d'
 - ▶ Hijo izquierdo: hijo izquierdo del árbol

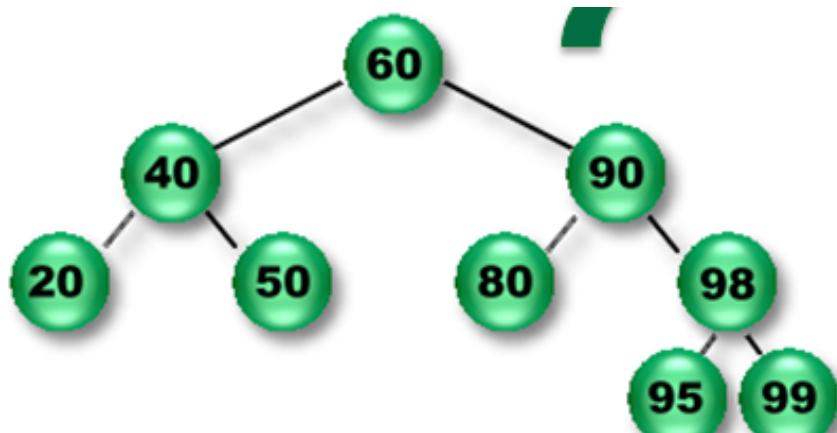


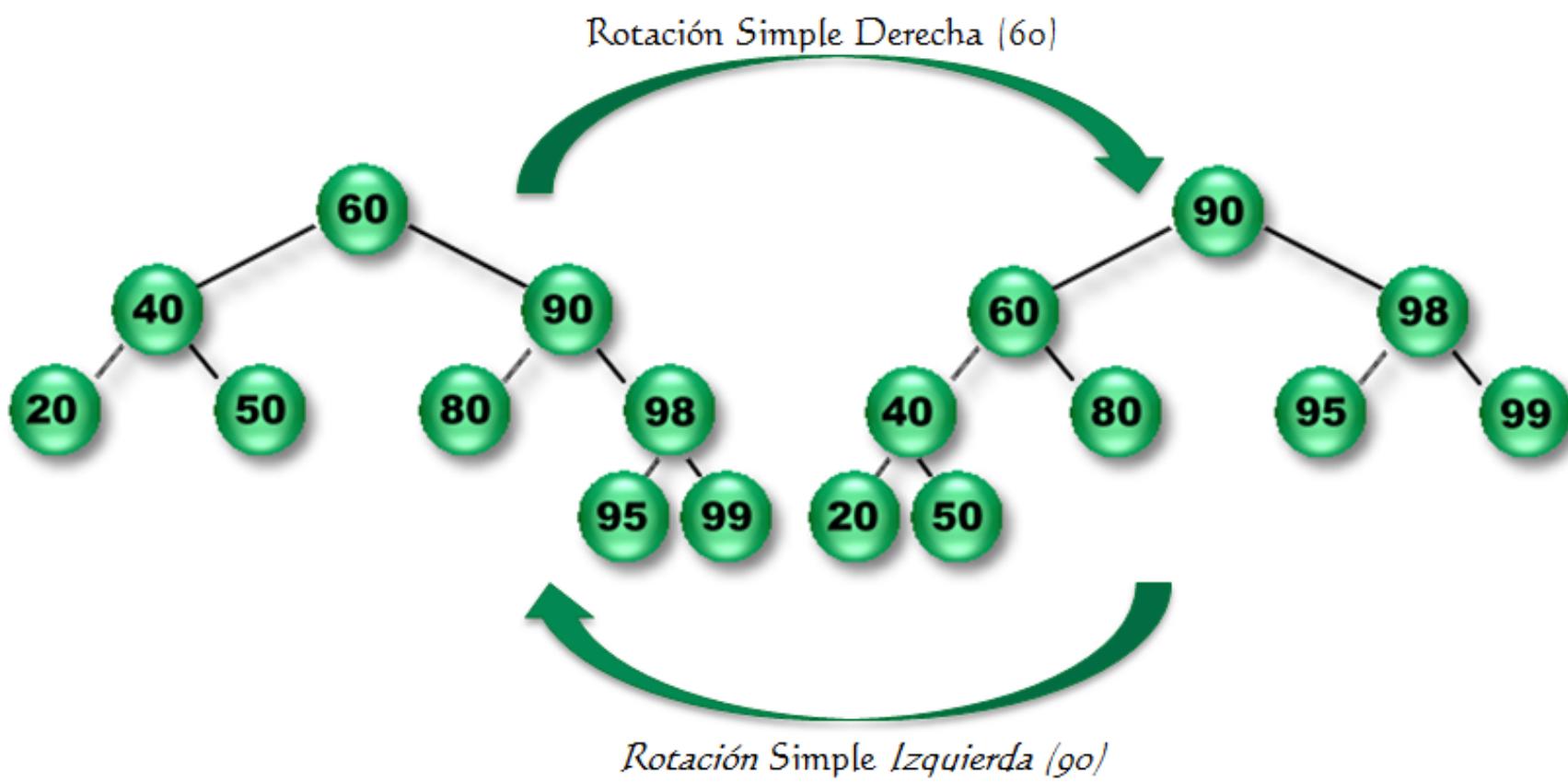
Rotación simple a la izquierda

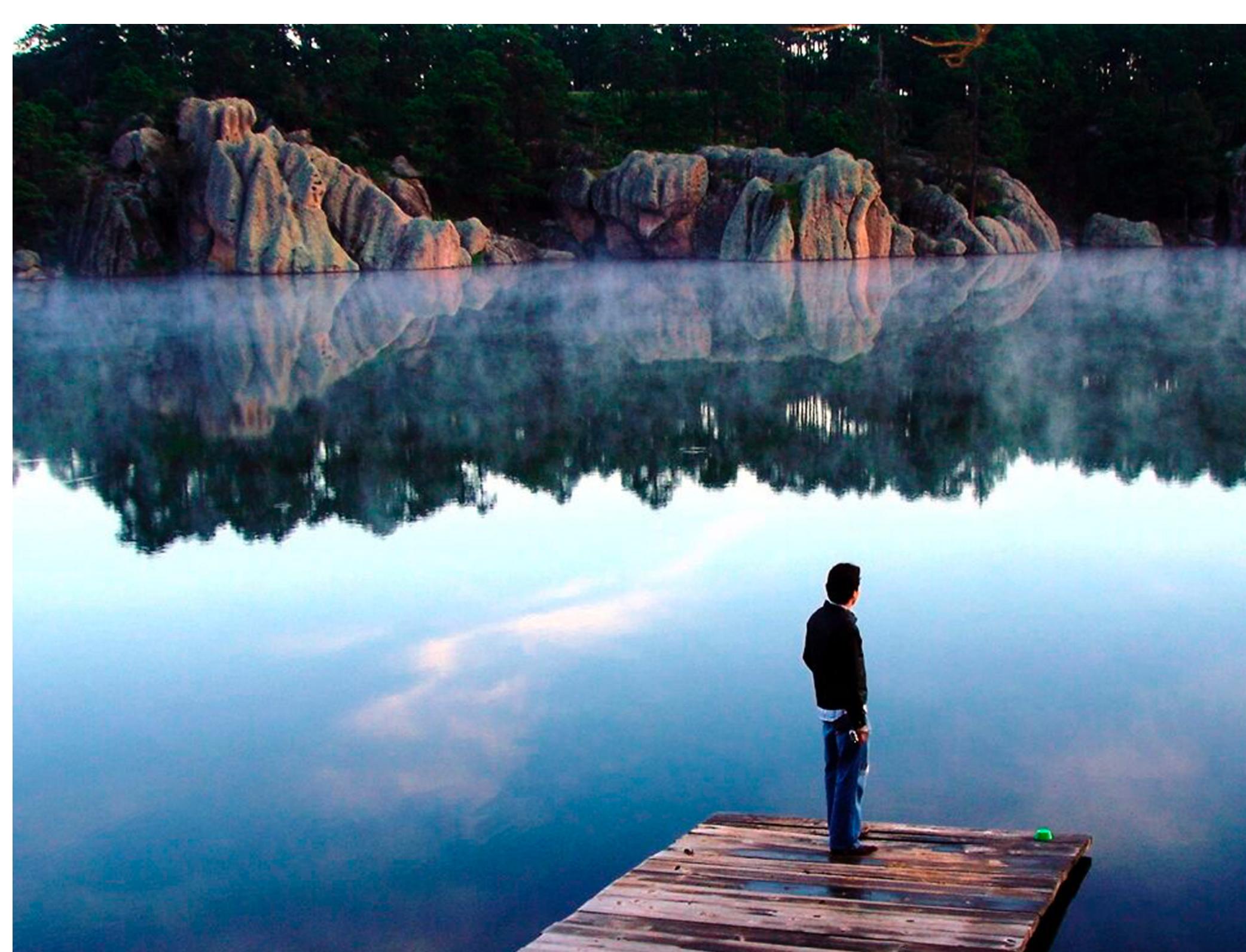




Rotación simple a la derecha (60)

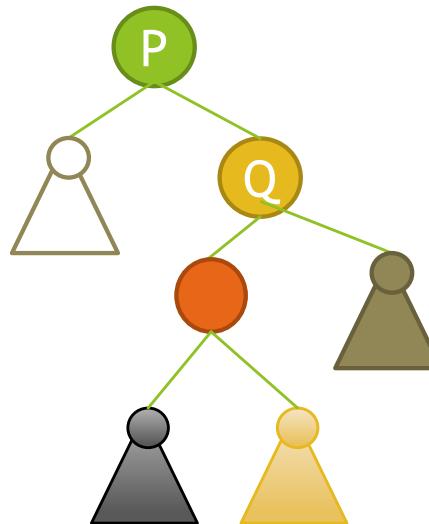






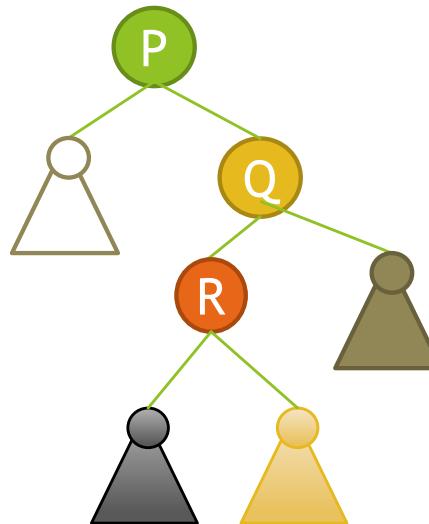
Rotación doble izquierda

- ▶ Subárbol derecho 2 unidades más alto que el izquierdo ($FE = 2$)
- ▶ Subárbol derecho cargado a la izquierda ($FE = -1$)



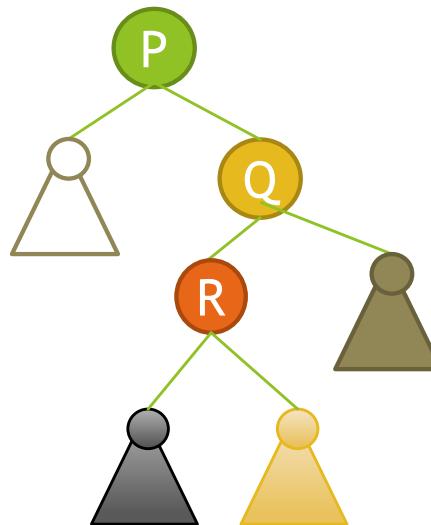
Rotación doble izquierda

- ▶ P -> nodo que muestra el desequilibrio
- ▶ Q -> raíz del subárbol derecho de P
- ▶ R -> nodo raíz del subárbol izquierdo de Q



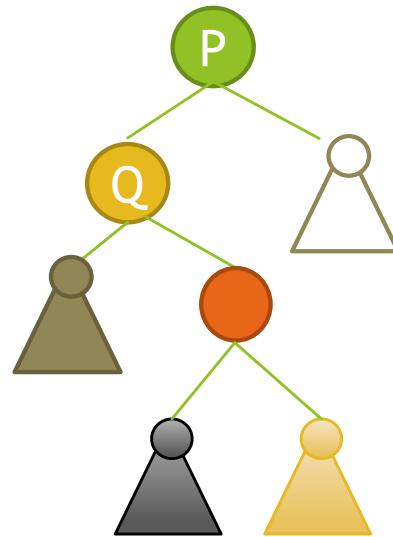
Rotación doble izquierda

- ▶ Rotación simple a la derecha en Q
- ▶ Rotación simple a la izquierda en P



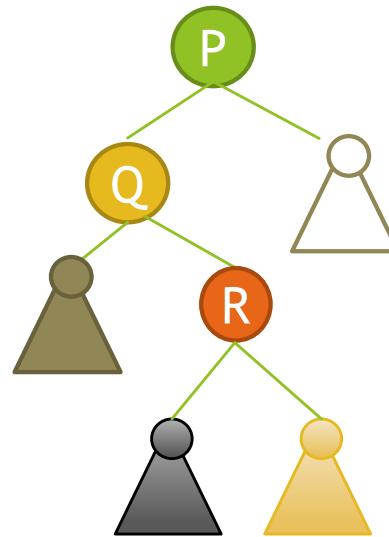
Rotación doble derecha

- ▶ Subárbol izquierdo 2 unidades más alto que el derecho($FE = -2$)
- ▶ Subárbol izquierdo cargado a la derecha ($FE = 1$)



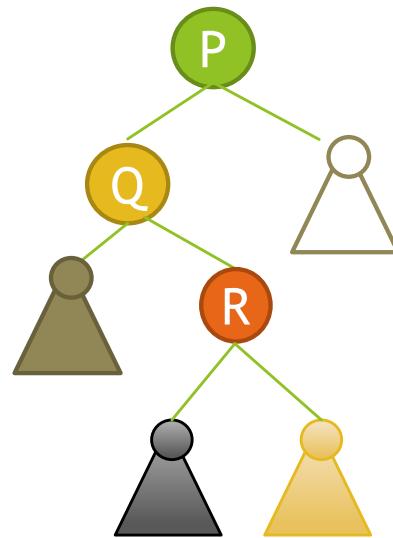
Rotación doble derecha

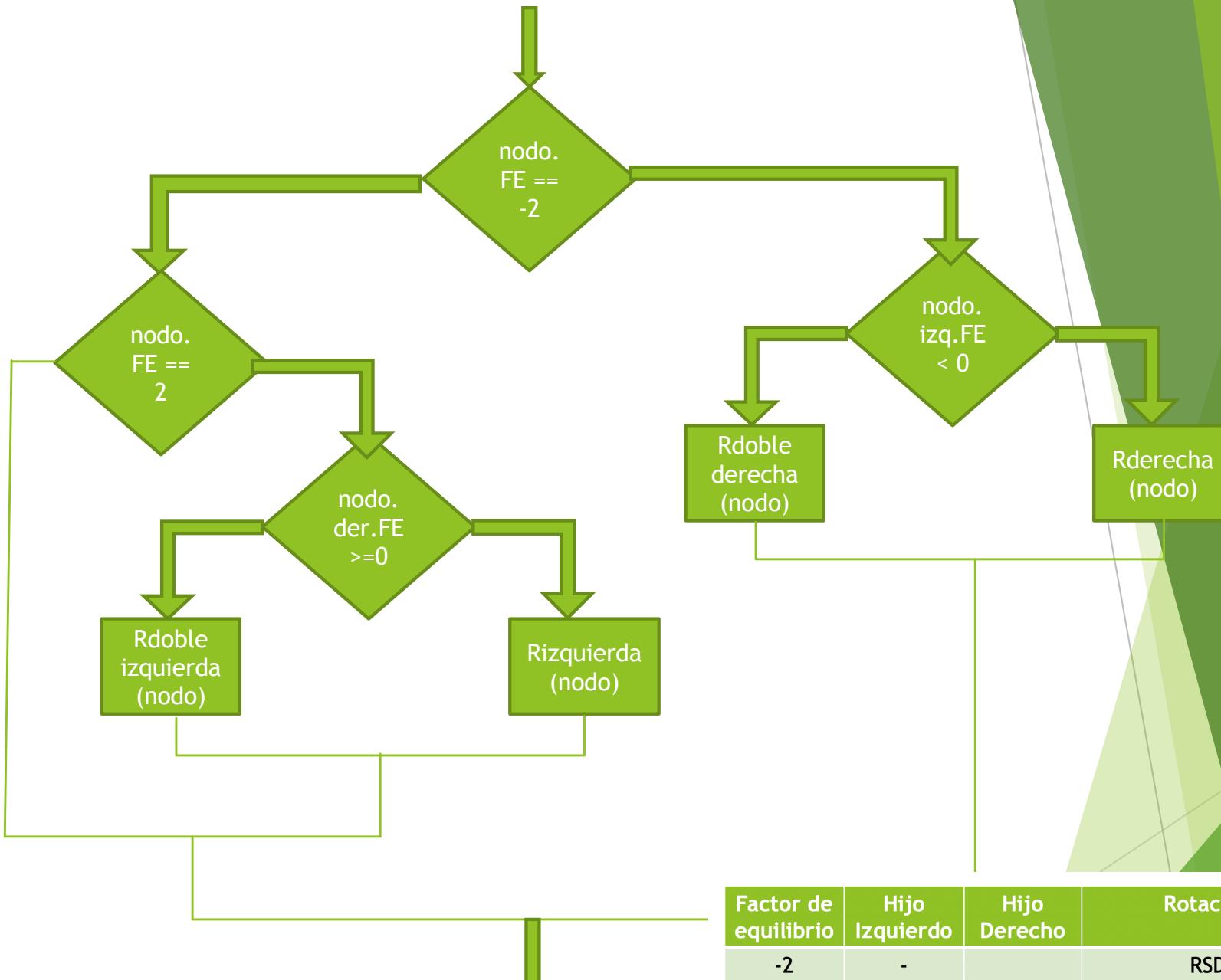
- ▶ P -> nodo que muestra el desequilibrio
- ▶ Q -> raíz del subárbol izquierdo de P
- ▶ R -> nodo raíz del subárbol derecho de Q



Rotación doble derecha

- ▶ Rotación simple a la izquierda en Q
- ▶ Rotación simple a la derecha en P





Factor de equilibrio	Hijo Izquierdo	Hijo Derecho	Rotación
-2	-		RSD
-2	+		RDD
+2		+	RSI
+2		-	RDI