2.1.8. Árboles AVL.

Lo presentado en esta sección está tomado del material que sobre el tema publicó el Maestro en Computación José Andrés Vázquez Flores, Docente de la Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, México.

www.cs.buap.mx/~andrex/estructuras/ArbolesAVL.ppt

¿Por qué es importante el balanceo en un árbol de búsqueda?

La manera en que los elementos estén distribuidos en un árbol de búsqueda determinará su altura y, en consecuencia, la cantidad de comparaciones a realizar al buscar un elemento (eficiencia).

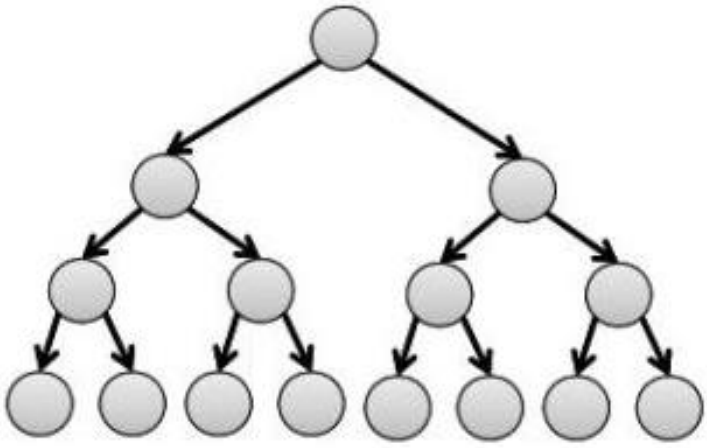
Análisis general de la eficiencia de búsqueda en un ABB.

La cantidad máxima de comparaciones al realizar una búsqueda en un ABB está determinada por la altura del árbol. Si un árbol degenera en una lista, se tiene un árbol cuya altura es igual a la cantidad de nodos en el árbol, y el peor caso corresponderá a realizar tantas comparaciones como nodos tenga el árbol.

¿Qué pasa si el árbol está balanceado?

Si la altura de un ABB determina la cantidad máxima de comparaciones en una búsqueda, lo ideal sería tener la altura mínima que puede tener un ABB para n elementos.

La altura mínima en un ABB con **n** elementos se dará en la medida que cada nivel del árbol esté integrado a su máxima capacidad.

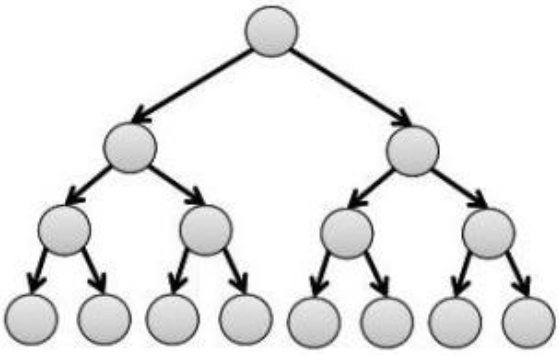


Nivel 0: 1 elemento
Nivel 1: 2 elementos
Nivel 2: 4 elementos
Nivel 3: 8 elementos
Nivel 4: 16 elementos
.
.
.

Figura 30

En general, se tiene que el número máximo de nodos en el nivel k en un árbol binario es $2^{\mathbf{k}}$.

Con base en esto, se puede encontrar la cantidad total de elementos que puede guardar un árbol binario de altura k.



Altura 0:	1 elemento
Altura 1:	3 elementos
Altura 2:	7 elementos
Altura 3:	15 elementos
Altura 4:	31 elementos
	.
	.
	.

Figura 31

Por lo tanto se tiene que el número máximo de nodos en un árbol binario de altura k es $2^{\mathbf{k}} - 1$.

Ahora bien, si la altura de un ABB determina el número máximo de comparaciones al buscar un elemento, ¿cuántas comparaciones se harán al buscar un elemento en un ABB ideal que tiene n elementos?

De acuerdo con el análisis que hemos hecho, la respuesta se obtiene al encontrar el valor de k, dado el valor de **n** para la fórmula:

$$n = 2^{\mathbf{k}} - 1$$

Por lo tanto, despejando k, aplicando las leyes de los logaritmos, se obtiene

$$k = \log_2 (n+1)$$

Entonces, la cantidad máxima de comparaciones a realizar en un ABB ideal de n elementos es:

$$\log_2 (n+1)$$

En este caso, se puede comprobar que la fórmula coincide con la de la búsqueda binaria en un arreglo, si se complementa con la función techo:

$$\lceil \log_2 (n+1) \rceil$$

Lo anterior nos lleva a concluir que, para que el peor caso de una búsqueda en un ABB se cumpla con la mayor eficiencia posible, el ABB debe estar balanceado.

Sin embargo, las operaciones de un ABB no aseguran que se cumpla el balanceo, por lo que se necesita una mejora en las operaciones, sin demeritar su eficiencia. Hasta ahora, no se ha encontrado la forma eficiente de mantener un ABB totalmente balanceado.

¿Qué es un árbol balanceado?

Se considera que un árbol binario está balanceado cuando todos sus niveles, excepto el último, están integrados a la máxima capacidad de nodos.

Las investigaciones respecto a esta estructura de datos no han logrado encontrar una técnica eficiente para manejar árboles de búsqueda completamente balanceados; las propuestas han llegado solo a presentar árboles parcialmente balanceados, sin repercutir la eficiencia de las operaciones de inserción y eliminación de nodos. La más común y usada de las técnicas es la de los árboles AVL.

¿Qué es un árbol AVL?

Un árbol AVL es un árbol binario de búsqueda que trata de mantenerse lo más balanceado posible, conforme se realizan operaciones de inserción y eliminación. Fueron propuestos en 1962 por los matemáticos ***Gueorgui Maksímovich ADELSÓN-VELSKI*** (Ruso) y ***Yevgueni Mijáilovich LANDIS*** (Ucraniano), de donde surge su nombre.

Su contribución principal consistió en presentar algoritmos eficientes de inserción y eliminación de elementos considerando un balanceo en el árbol que, a su vez, repercute en la eficiencia de las búsquedas. Formalmente, en los árboles AVL se debe cumplir el hecho de que para cualquier nodo del árbol, la diferencia entre las alturas de sus subárboles no exceda una unidad.

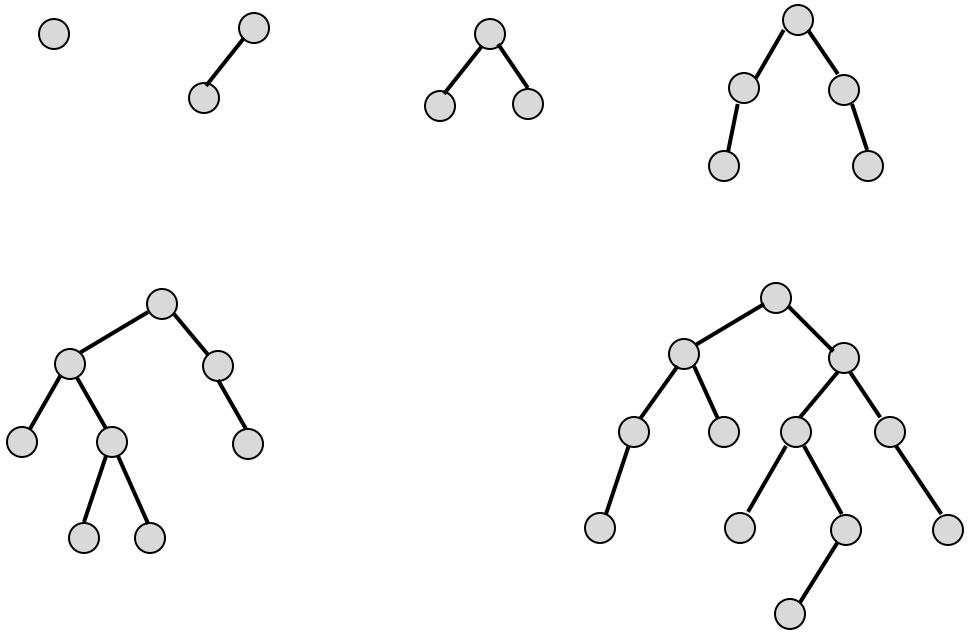


Figura 32. Ejemplos de árboles AVL

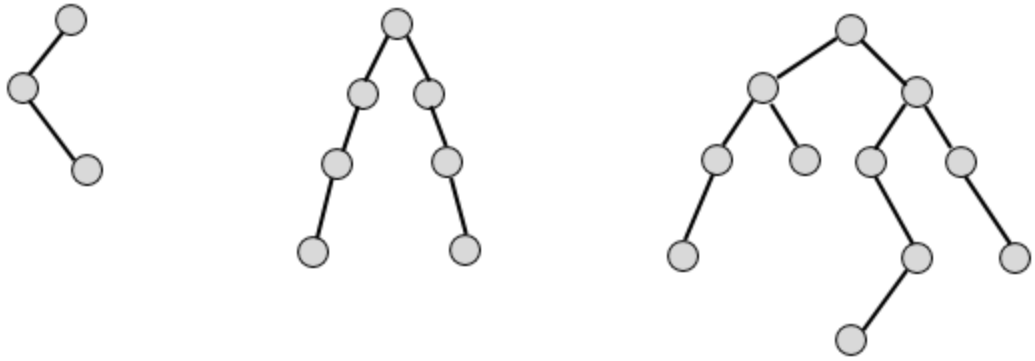


Figura 33. Ejemplos de árboles que NO son AVL

¿Qué es el factor de balance (FB) de un nodo?

Los nodos de un árbol AVL guardan un valor entre 1 y -1, lo que se conoce como Factor de Balance (FB), y representa la diferencia entre las alturas de sus subárboles.

Un FB igual a cero en un nodo significa que las alturas de sus subárboles son iguales;
un FB positivo significa que el subárbol derecho es más grande que el izquierdo, y
un FB negativo que el subárbol izquierdo es más grande que el derecho.

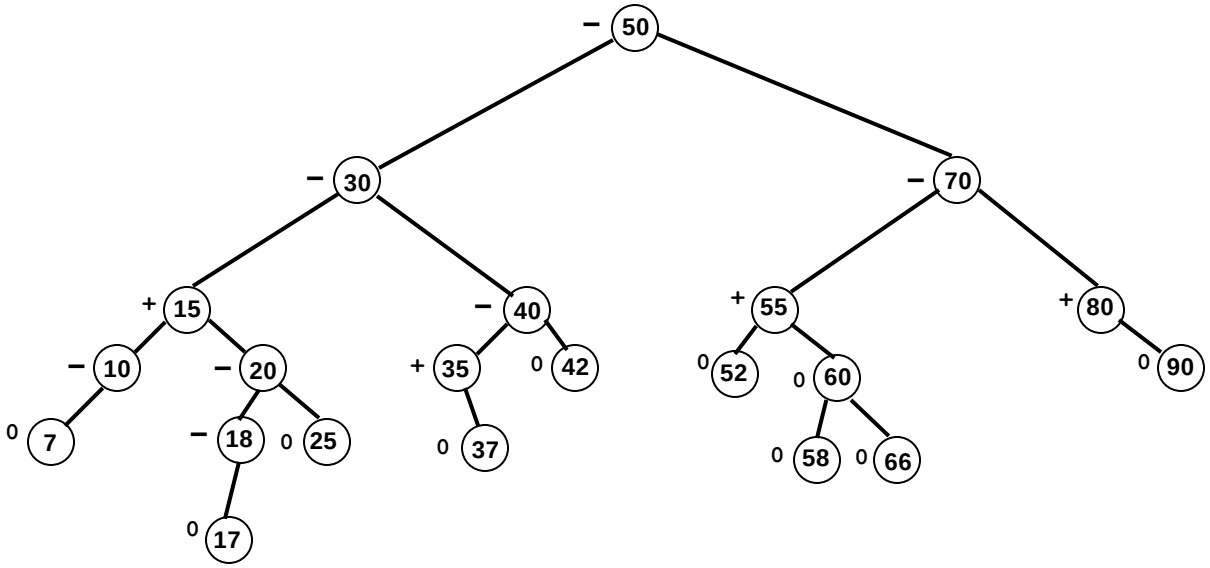


Figura 34. Ejemplo de cálculos de factor de balance

¿Y cómo balanceamos?

Por medio de las ROTACIONES, de las cuales hay los siguientes casos:

[1] Rotación Simple a la Derecha (RSD)

[2] Rotación Simple a la Izquierda (RSI)

[3] Rotación Doble a la Derecha (RDD)

[4] Rotación Doble a la Izquierda (RDI)

[1] Rotación Simple a la Derecha (RSD)

Esta rotación se usará cuando:

El subárbol izquierdo de un nodo sea 2 unidades más alto que el derecho, es decir:

cuando su FB = -2 y además, la raíz del subárbol izquierdo tenga una FB de -1 ó 0, es decir, que esté cargado a la izquierda (CASO A) o equilibrado.

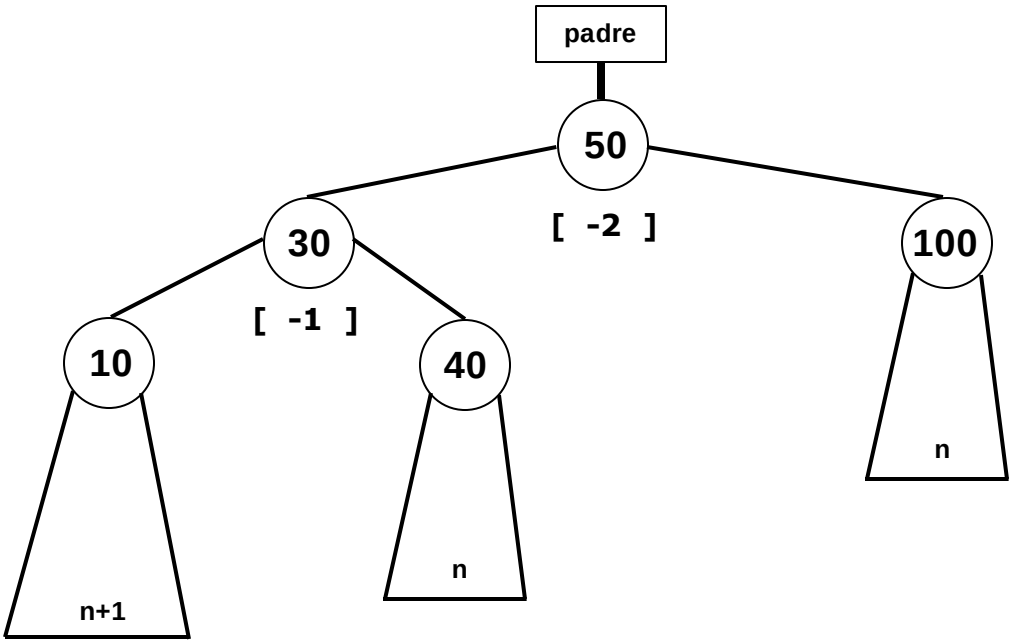


Figura 35. Árbol desequilibrado a la izquierda (caso A)

Se procede de la siguiente manera:

Llamaremos P al nodo que muestra el desequilibrio, el que tiene un FB de -2.

Y llamaremos Q al nodo raíz del subárbol izquierdo de P.

Además, llamaremos:

A al subárbol izquierdo de Q,

B al subárbol derecho de Q,

C al subárbol derecho de P.

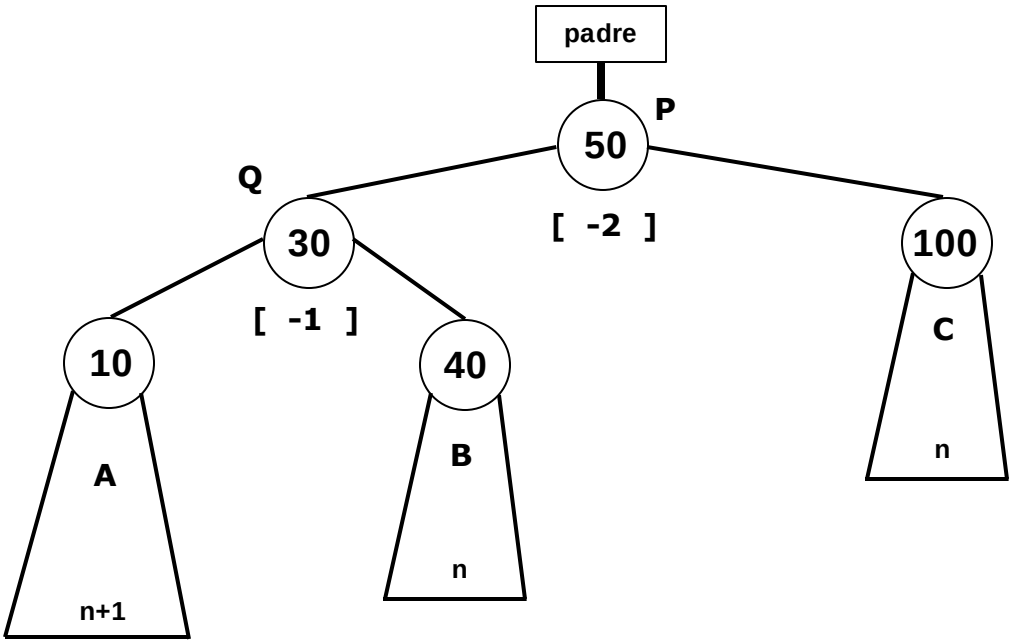


Figura 35 –A. Árbol etiquetado.

En la Figura 35-A se puede observar que:

tanto B como C tienen la misma altura (n) y A es una unidad mayor (n+1).

Esto hace que el FB de Q sea -1, la altura del subárbol que tiene Q como raíz es (n+2) y por lo tanto el FB de P es -2.

Y la rotación se efectúa de la siguiente manera:

1. **Pasamos el subárbol derecho del nodo Q como subárbol izquierdo de P.** Esto mantiene el árbol como ABB, ya que todos los valores a la derecha de Q siguen estando a la izquierda de P.

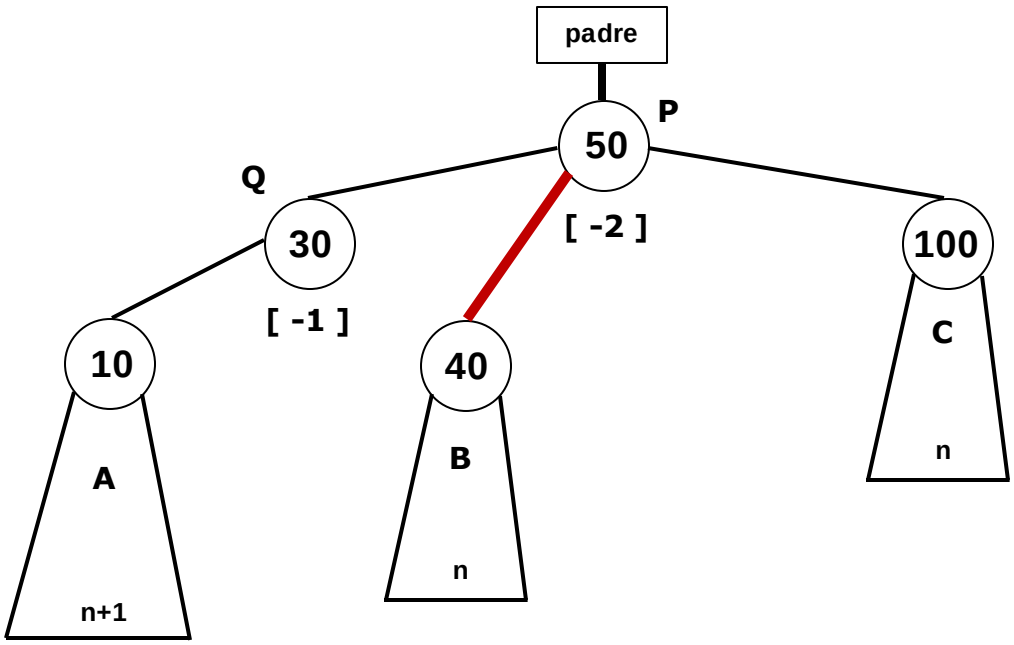


Figura 36. RSD CASO A Paso 1.

2. **El árbol P pasa a ser el subárbol derecho del nodo Q.** Aquí sucede la rotación propiamente dicha.

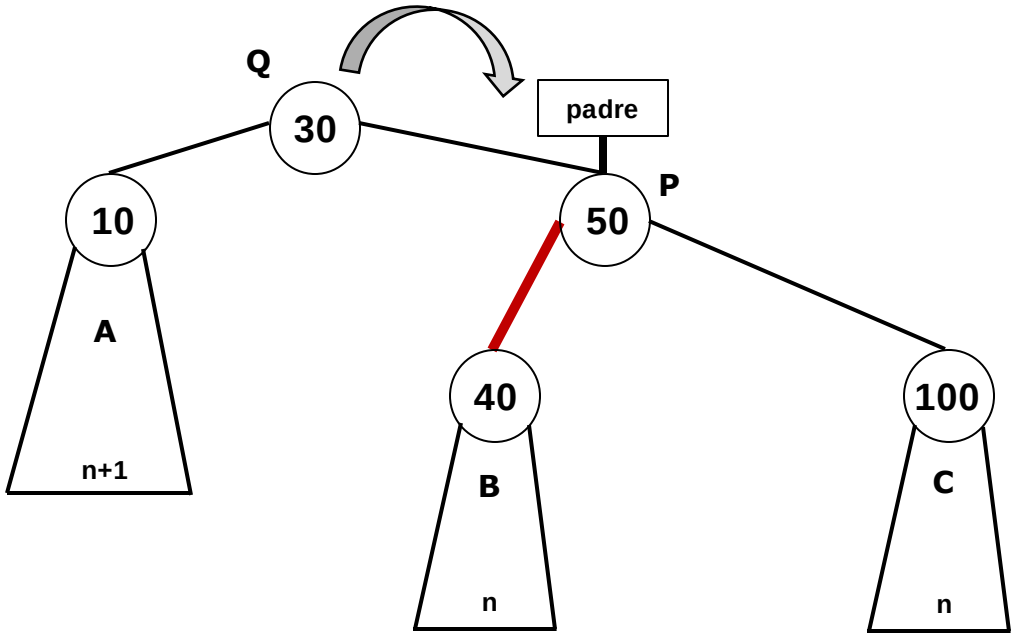


Figura 37. RSD CASO A Paso 2.

3. **Ahora, el nodo Q pasa a tomar la posición del nodo P, es decir, hacemos que la entrada al árbol sea el nodo Q, en lugar del nodo P.** Previamente, P puede que fuese un árbol completo o un subárbol de otro nodo de menor altura.

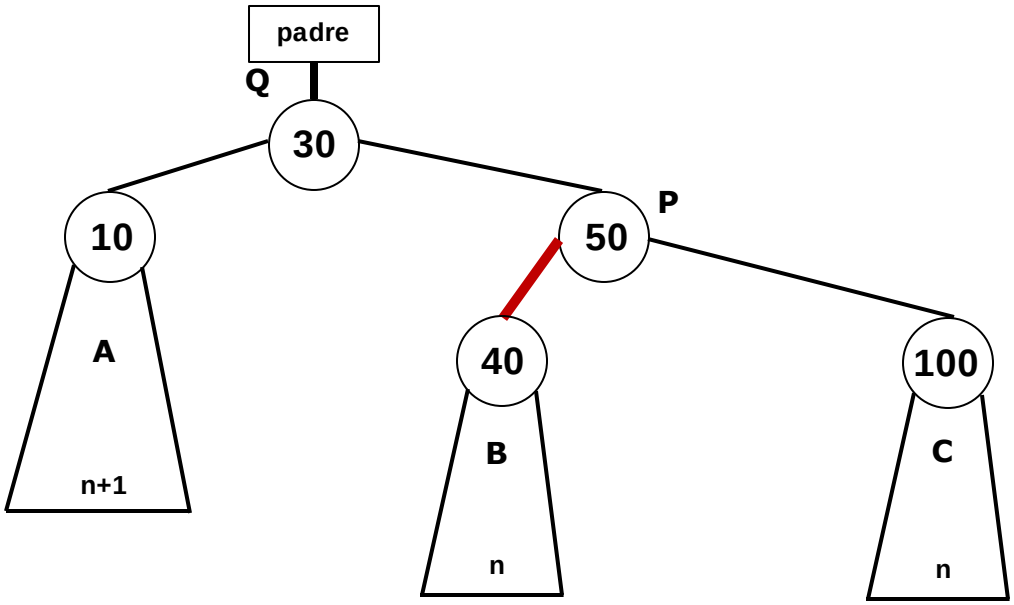


Figura 38. RSD CASO A Paso 3.

En el árbol resultante (Figura 39) se puede ver que tanto P como Q quedan equilibrados en cuanto altura.
En el caso de P porque sus dos subárboles tienen la misma altura (n),
en el caso de Q, porque su subárbol izquierdo A tiene una altura (n+1) y su subárbol derecho también, ya que a P se añade la altura de cualquiera de sus subárboles.

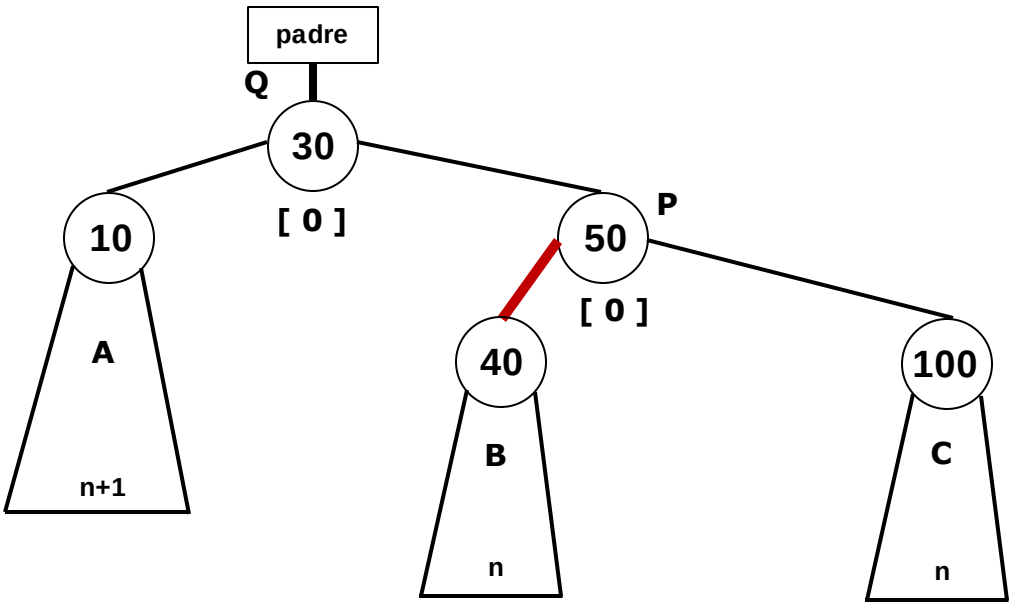


Figura 39. Árbol resultante de la rotación RSD CASO A.

CASO B: **En el caso de que el subárbol izquierdo esté equilibrado,** el procedimiento es similar, pero los FB de los nodos P y Q en el árbol resultante son diferentes. En principio, parece poco probable que nos encontremos un árbol con esta estructura, pero es posible encontrarlos cuando se borran nodos.

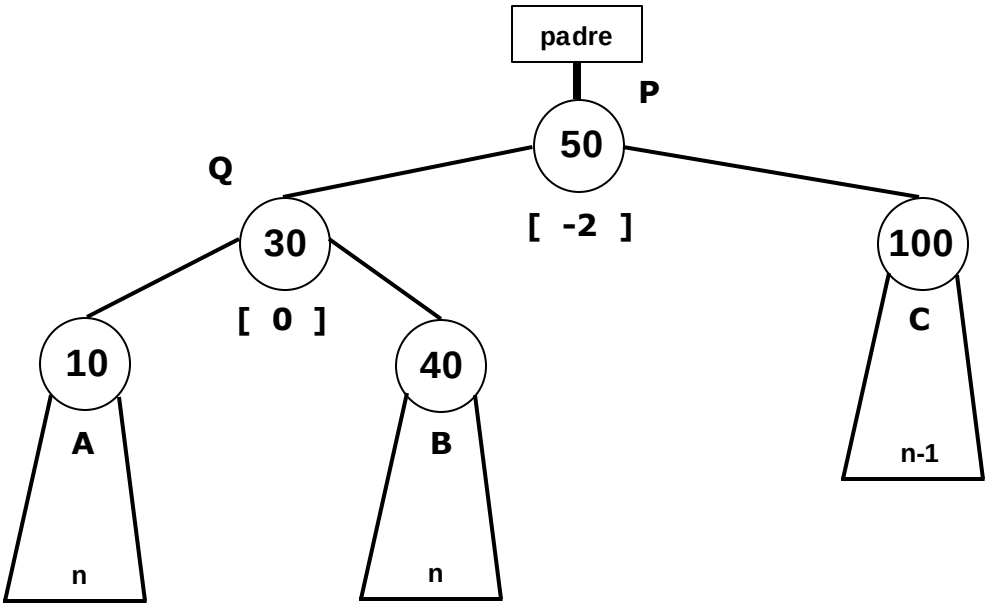


Figura 40. Árbol equilibrado a la izquierda (caso B).

Aplicamos el mismo algoritmo para la rotación:

1. **Pasamos el subárbol derecho del nodo Q como subárbol izquierdo de P.** Esto mantiene el árbol como ABB, ya que todos los valores a la derecha de Q siguen estando a la izquierda de P.

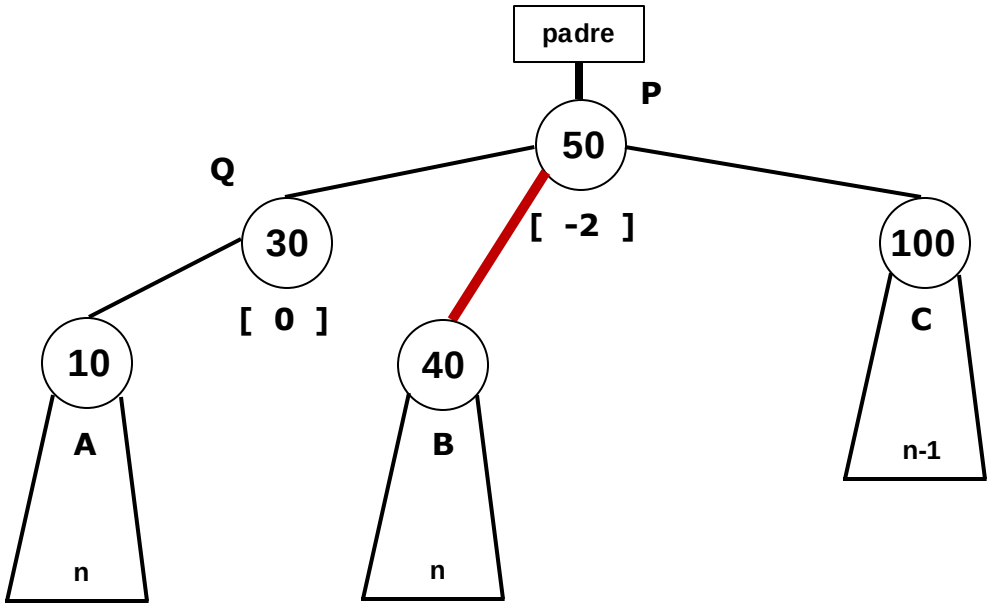


Figura 41. RSD CASO B Paso 1.

2. **El árbol P pasa a ser el subárbol derecho del nodo Q.** Aquí sucede la rotación propiamente dicha.

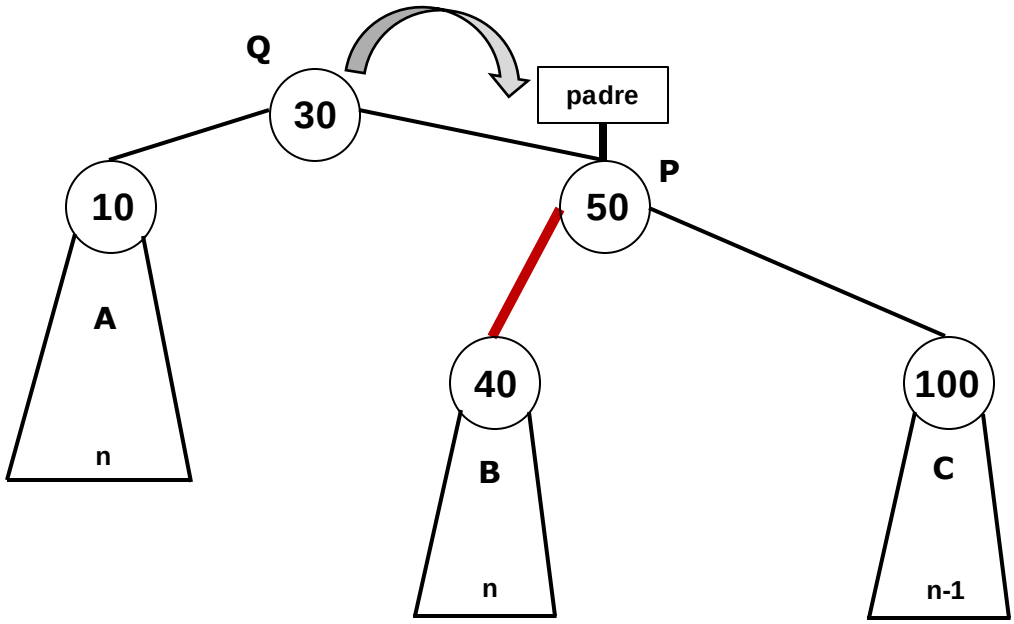


Figura 42. RSD CASO B Paso 2.

3. **Ahora, el nodo Q pasa a tomar la posición del nodo P, es decir, hacemos que la entrada al árbol sea el nodo Q, en lugar del nodo P.** Previamente, P puede que fuese un árbol completo o un subárbol de otro nodo de menor altura.

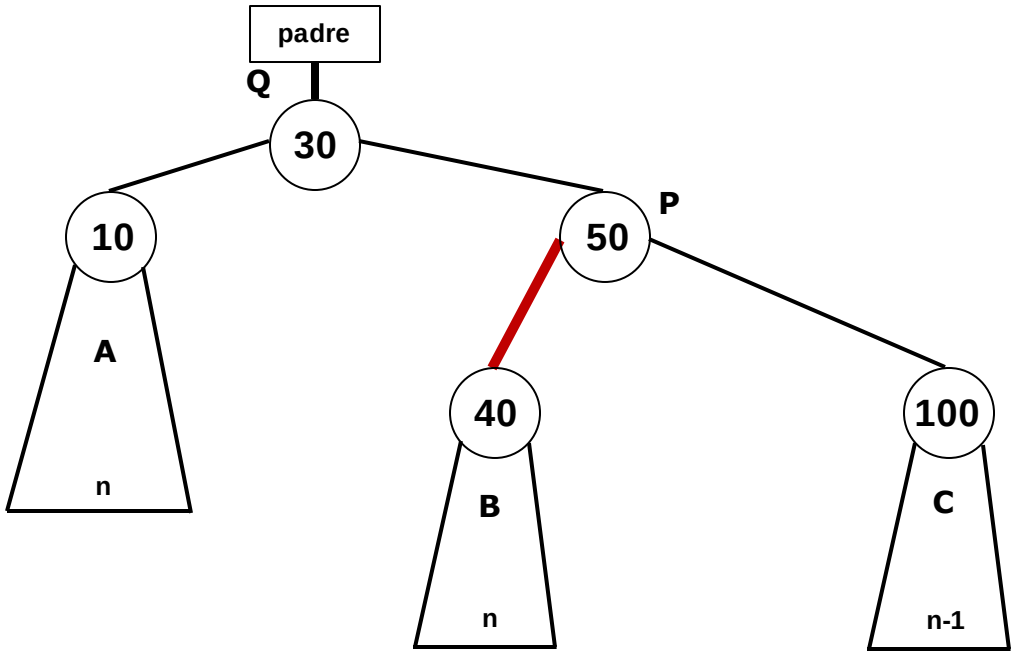


Figura 43. RSD CASO B Paso 3.

En el árbol resultante se puede ver que tanto P como Q quedan equilibrados en cuanto altura.

En el caso de P porque su subárbol izquierdo es una unidad más alto que el derecho, quedando su FB en -1.

En el caso de Q, porque su subárbol derecho tiene una altura (n+1) y su subárbol izquierdo, una altura de n.

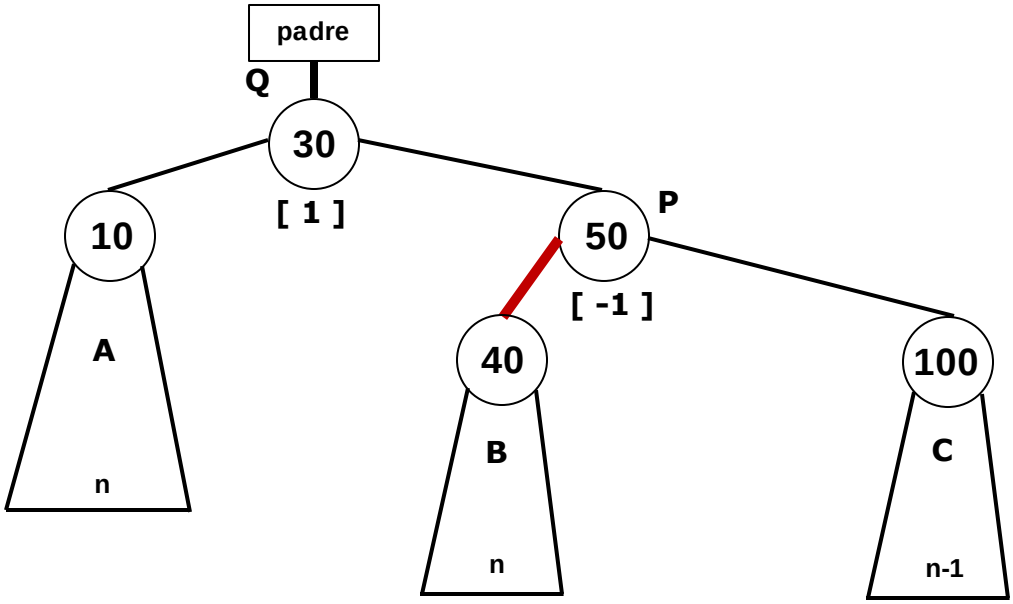


Figura 44. Árbol resultante de la rotación RSD CASO B.

De modo que, aunque aplicamos el mismo algoritmo, ya que en ambos casos se trata de una rotación simple, deberemos tener en cuenta estos detalles a la hora de ajustar los nuevos valores de FB a la hora de programarlo.

[2] Rotación Simple a la Izquierda (RSI)

Se trata del caso simétrico del anterior. Esta rotación se usará cuando el subárbol derecho de un nodo sea 2 unidades más alto que el izquierdo, es decir: cuando su FB = 2 y además, la raíz del subárbol derecho tenga una FB de 1 ó 0, es decir, que esté cargado a la derecha o esté equilibrado.

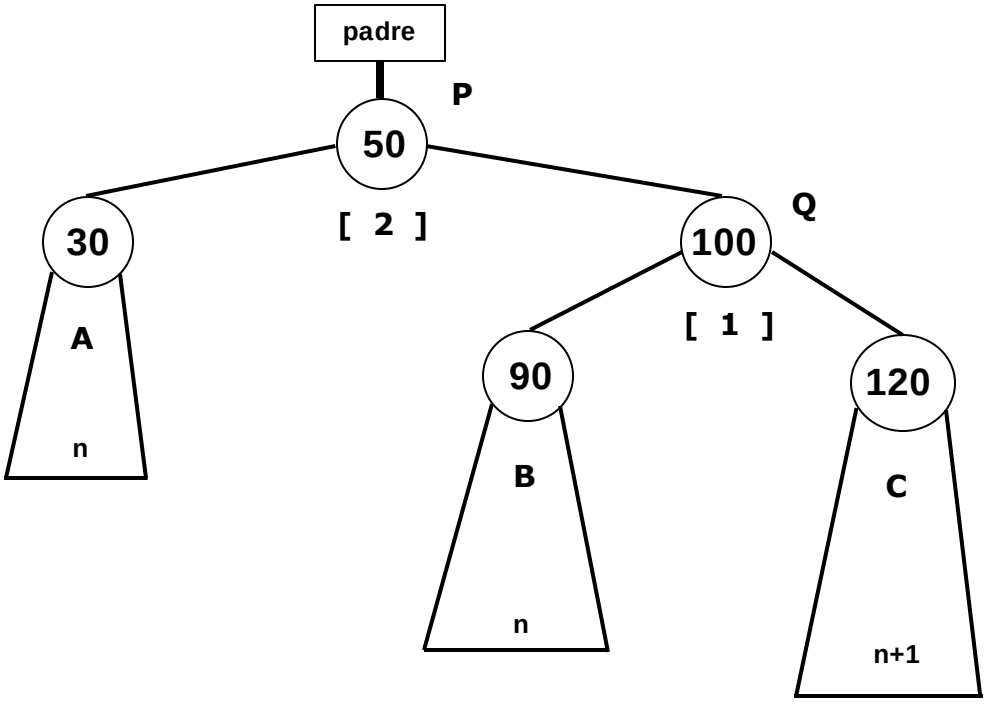


Figura 45. Árbol desequilibrado a la derecha

Se procede de la siguiente manera:

Llamaremos P al nodo que muestra el desequilibrio, el que tiene una FB de 2.

Y llamaremos Q al nodo raíz del subárbol derecho de P.

Además, llamaremos:

A al subárbol izquierdo de P,

B al subárbol izquierdo de Q,

C al subárbol derecho de Q.

En la Figura 45 se puede observar que:

tanto A como B tienen la misma altura (n), y C es una unidad mayor (n+1).

Esto hace que el FB de Q sea 1, la altura del subárbol que tiene Q como raíz es (n+2)

y por lo tanto el FB de P es 2.

Y la rotación se efectúa de la siguiente manera:

1. **Pasamos el subárbol izquierdo del nodo Q como subárbol derecho de P.** Esto mantiene el árbol como ABB, ya que todos los valores a la izquierda de Q siguen estando a la derecha de P.

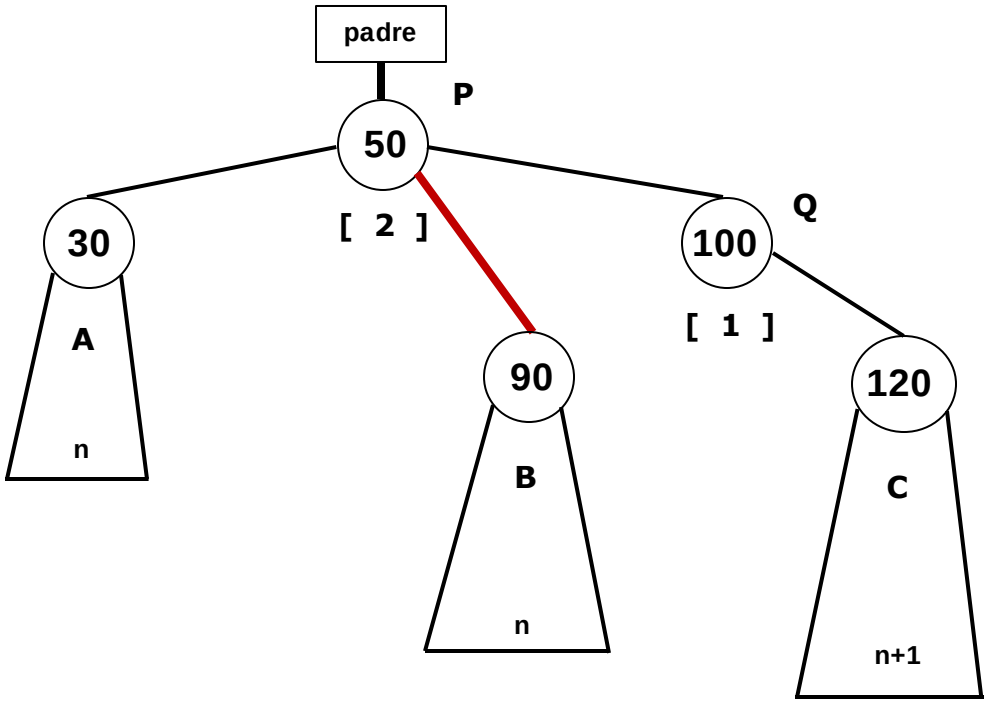


Figura 46. RSI Paso 1.

2. **El árbol P pasa a ser el subárbol izquierdo del nodo Q.** Aquí sucede la rotación propiamente dicha.

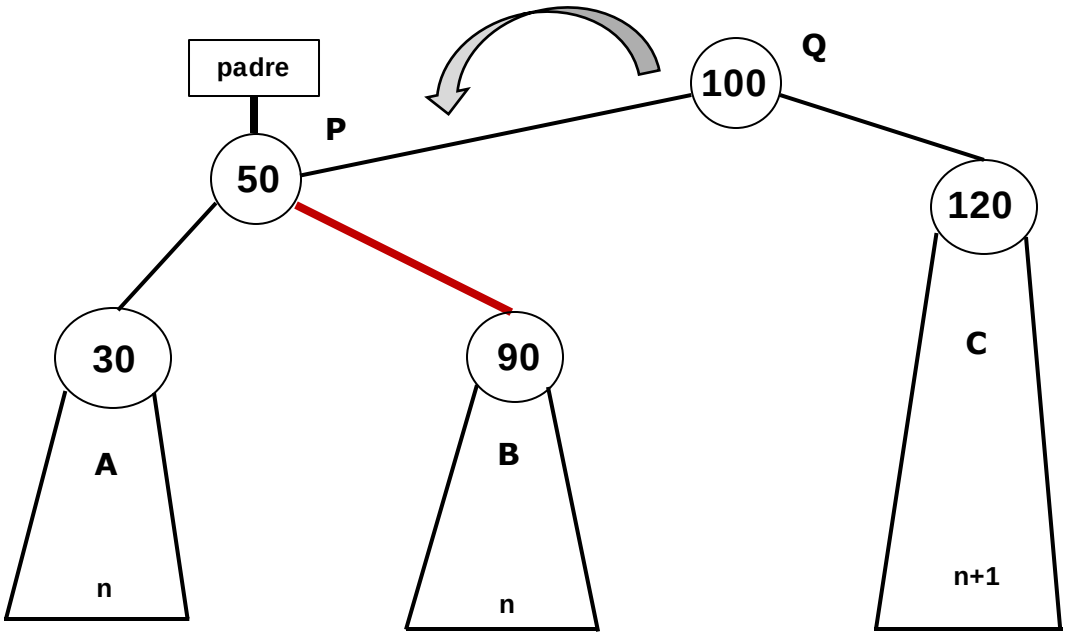


Figura 47. RSI Paso 2.

3. **Ahora, el nodo Q pasa a tomar la posición del nodo P, es decir, hacemos que la entrada al árbol sea el nodo Q, en lugar del nodo P.** Previamente, P puede que fuese un árbol completo o un subárbol de otro nodo de menor altura.

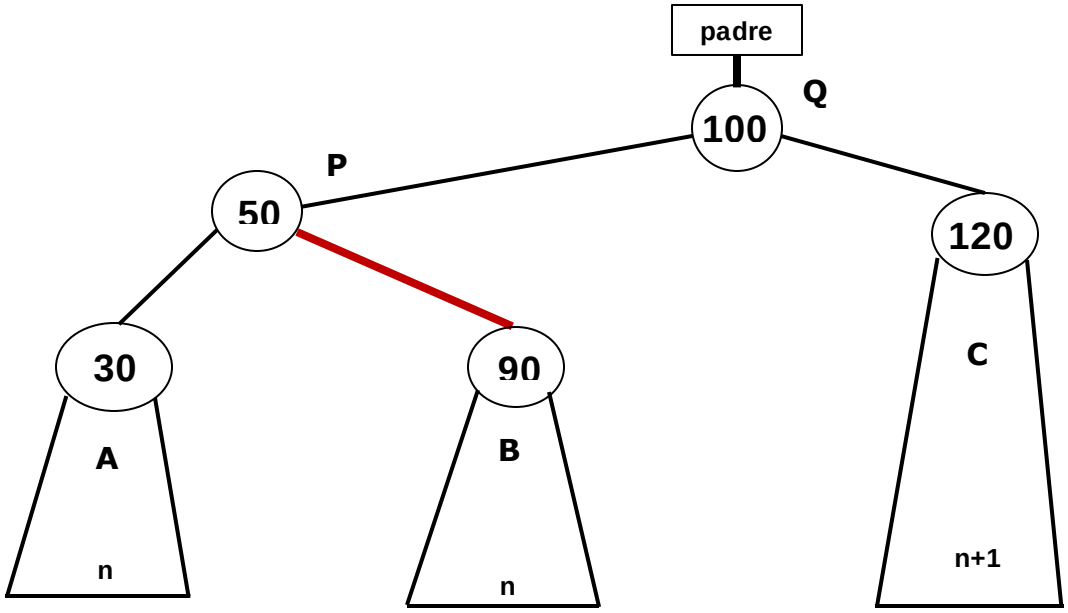


Figura 48. RSI Paso 3.

En el árbol resultante se puede ver que tanto P como Q quedan equilibrados en cuanto altura.

En el caso de P porque sus dos subárboles tienen la misma altura (n),

en el caso de Q, porque su subárbol izquierdo A tiene una altura (n+1) y su subárbol derecho también, ya que a P se añade la altura de cualquiera de sus subárboles.

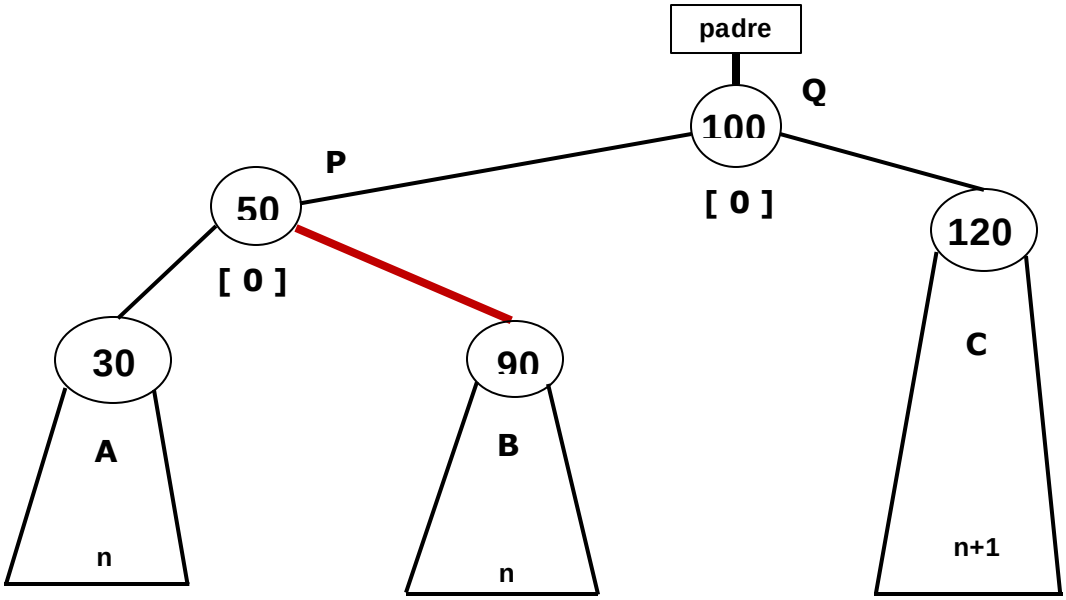


Figura 49. Árbol resultante de la rotación RSI

[3] Rotación Doble a la Derecha (RDD)

Esta rotación se usa cuando el subárbol izquierdo de un nodo sea 2 unidades más alto que el derecho, es decir: cuando su FB sea de -2.

Y además, la raíz del subárbol izquierdo tenga una FB de 1, es decir, que esté cargado a la derecha.

En la Figura 50 se muestra uno de los posibles árboles que pueden presentar esta estructura.
El nodo R puede tener una FB de -1, 0 ó 1.
En cada uno de esos casos los árboles izquierdo y derecho de R (B y C),

pueden tener alturas de: **n** y **n-1** , **n** y **n** , **n-1** y **n** , respectivamente.

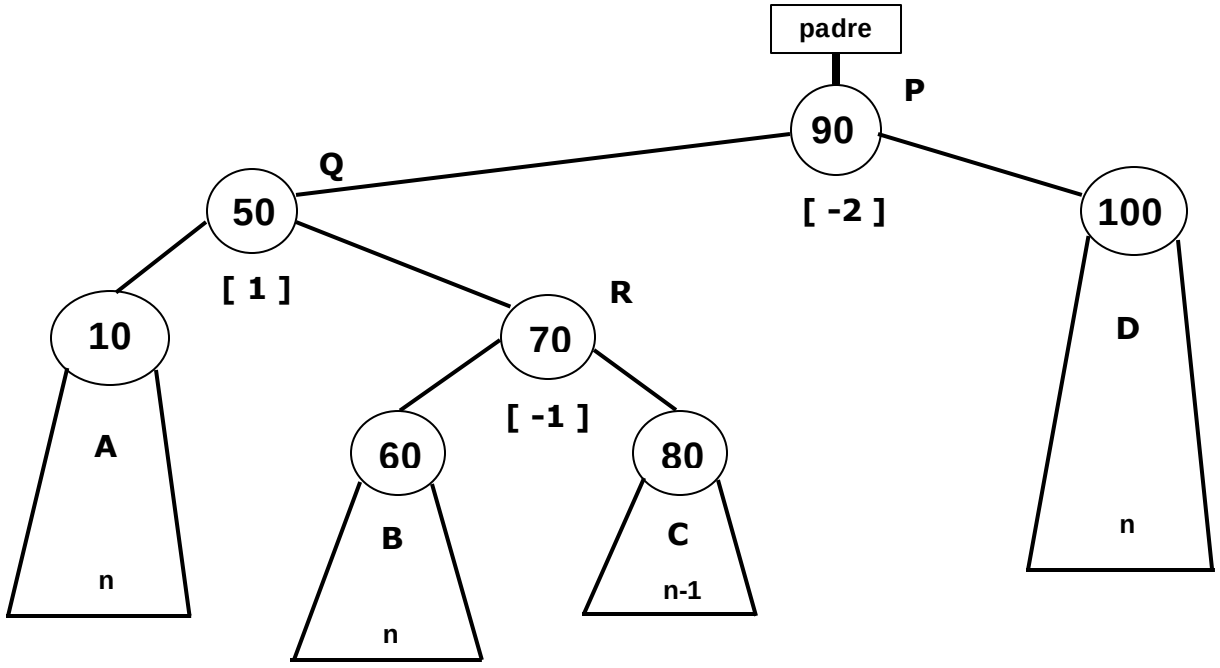


Figura 50. Árbol P desequilibrado a la izquierda.

El modo de realizar la rotación es independiente de la estructura del árbol **R**, cualquiera de las tres produce resultados equivalentes. Haremos el análisis para el caso en que FB sea -1.

En este caso tendremos que realizar dos rotaciones:

Llamaremos P al nodo que muestra el desequilibrio, el que tiene una FB de -2,
Llamaremos Q al nodo raíz del subárbol izquierdo de P,
Llamaremos R al nodo raíz del subárbol derecho de Q.

1. Haremos una ***rotación simple de Q a la izquierda***.
2. Después, haremos una ***rotación simple de P a la derecha***.

Con más detalle, procederemos del siguiente modo:

1. **Pasamos el subárbol izquierdo del nodo R como subárbol derecho de Q.** Esto mantiene el árbol como ABB, ya que todos los valores a la izquierda de R siguen estando a la derecha de Q.
2. **Ahora, el nodo R pasa a tomar la posición del nodo Q,** es decir, hacemos que la raíz del subárbol izquierdo de P sea el nodo R en lugar de Q.
3. **Paso 3-A. El árbol Q pasa a ser el subárbol izquierdo del nodo R.**

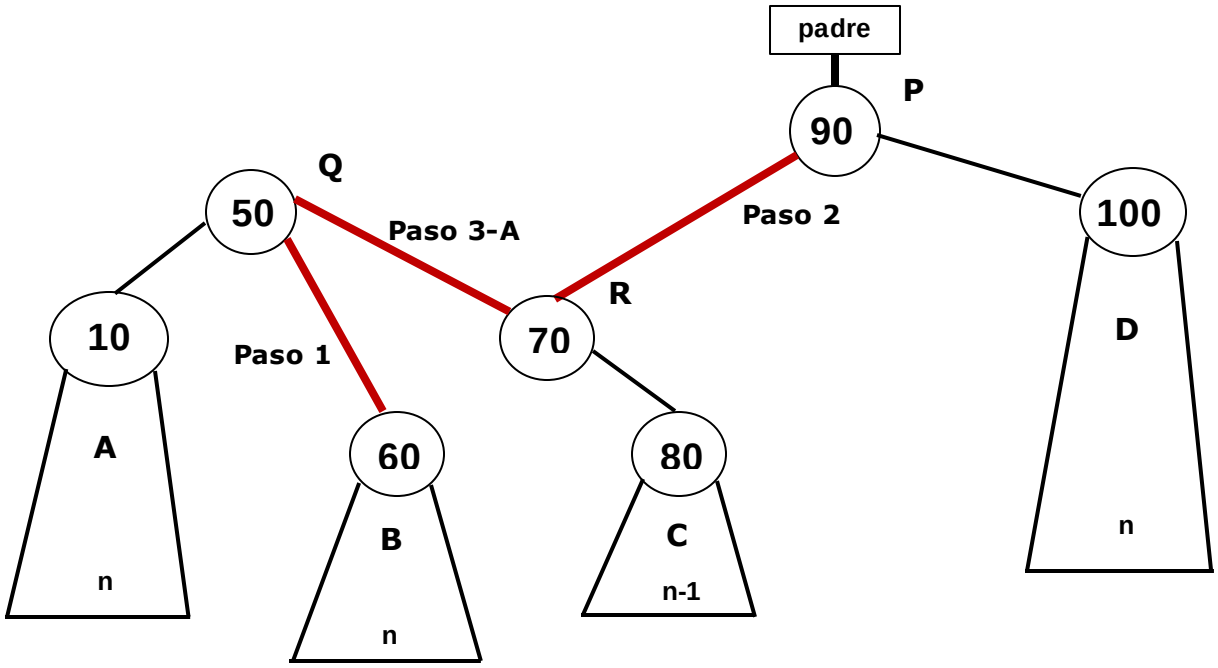


Figura 51-A. RDD pasos 1 al 3-A.

Paso 3-B. Aquí ocurre la rotación como tal.

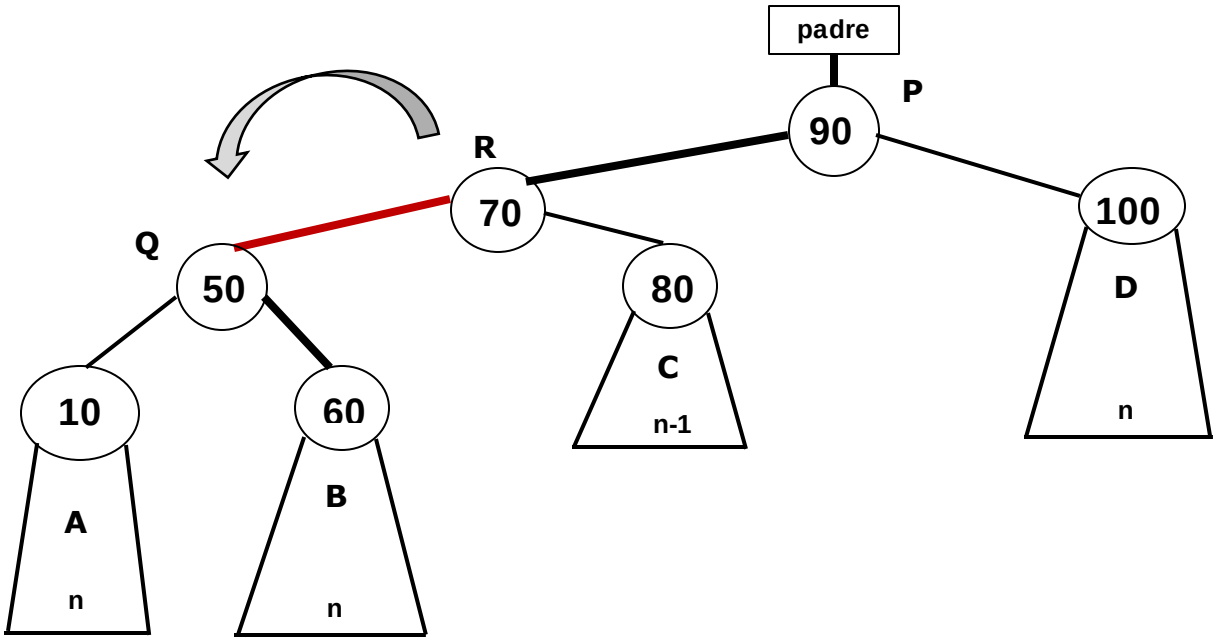


Figura 51-B. RDD paso 3-B (Rotación simple de Q a la Izquierda).

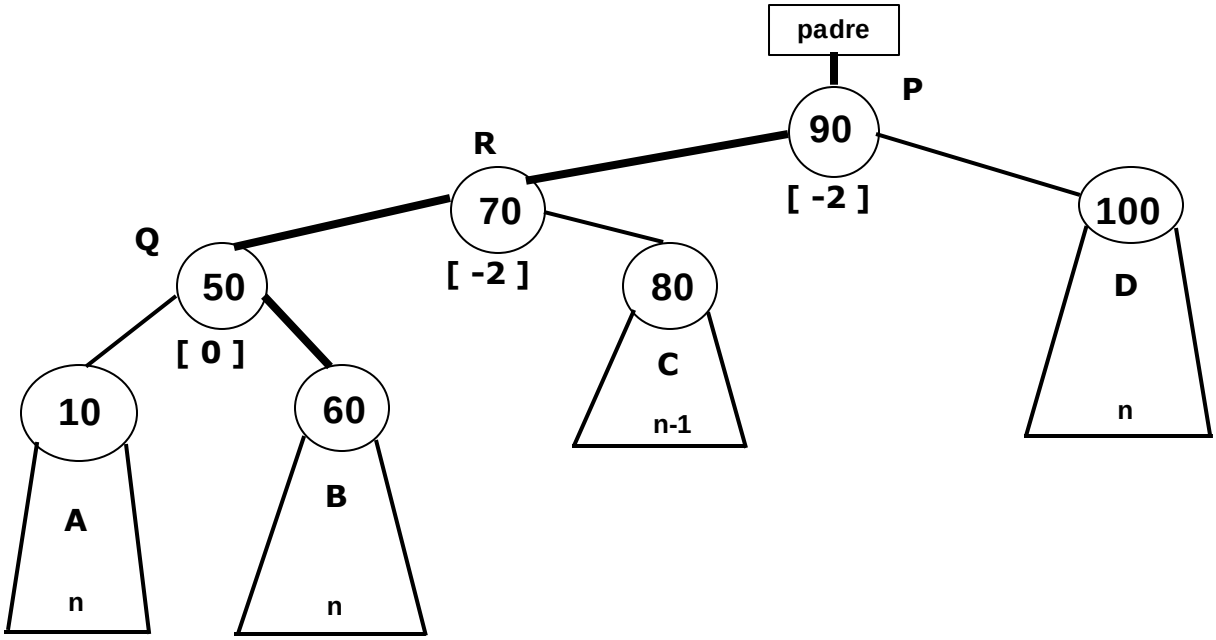


Figura 52-A. Árbol resultante de la primera rotación (simple de Q a la izquierda).

4. **Pasamos el subárbol derecho del nodo R como subárbol izquierdo de P.** Esto mantiene el árbol como ABB, ya que todos los valores a la derecha de R siguen estando a la izquierda de P.
5. **Ahora, el nodo R pasa a tomar la posición del nodo P, es decir, hacemos que la entrada al árbol sea el nodo R, en lugar del nodo P.** Como en los casos anteriores, previamente, P puede que fuese un árbol completo o un subárbol de otro nodo de menor altura.

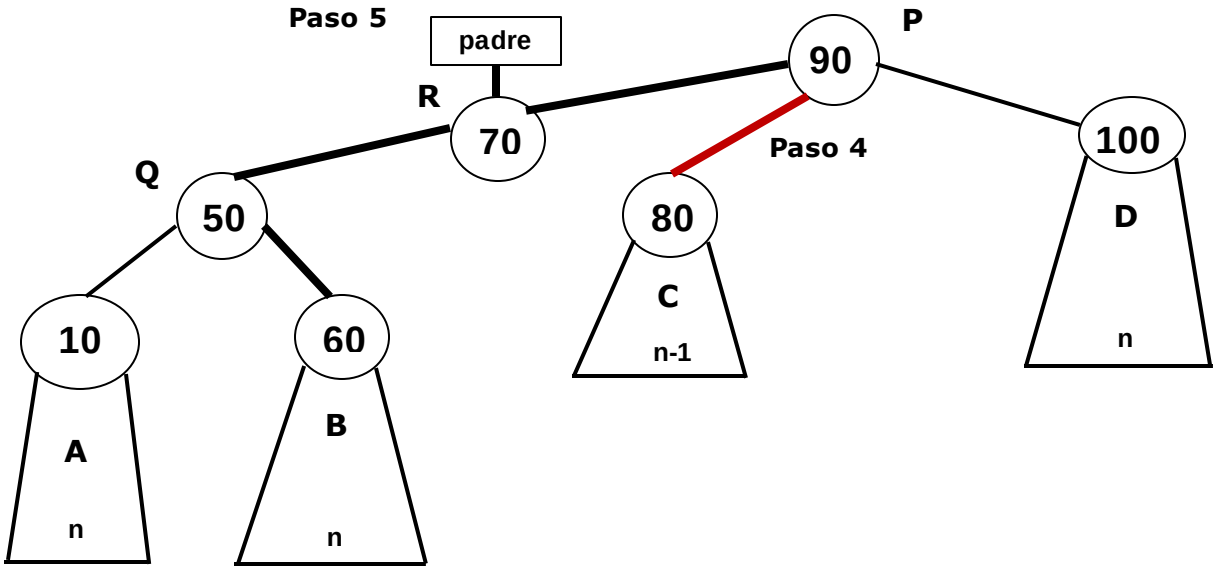


Figura 52-B. RDD pasos 4 y 5.

6. **El árbol P pasa a ser el subárbol derecho del nodo R.** Aquí ocurre la rotación como tal.

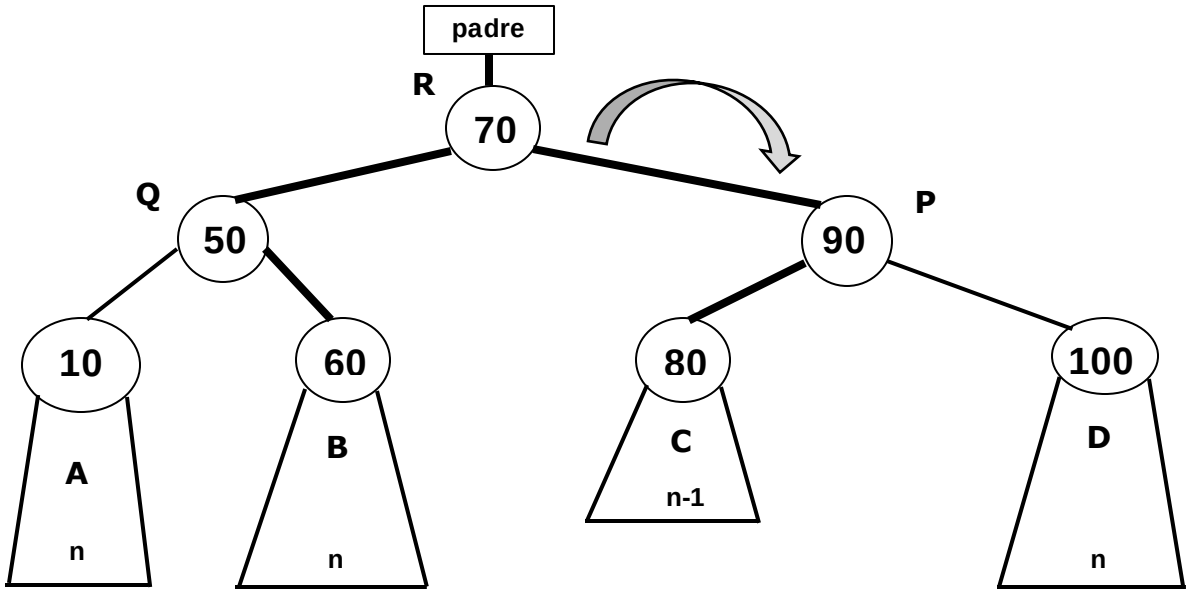


Figura 53. RDD paso 6 (Rotación simple de P a la Derecha).

La Figura 54 muestra el árbol resultante.

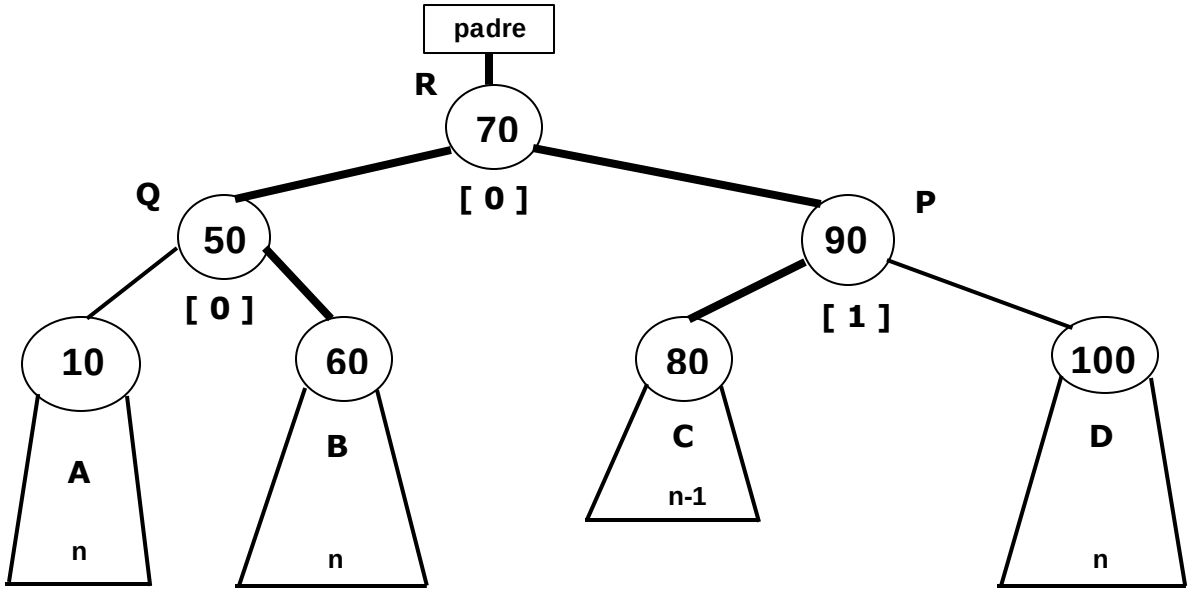


Figura 54. Árbol resultante

[4] Rotación Doble a la Izquierda (RDI)

Esta rotación se usa cuando el subárbol derecho de un nodo sea 2 unidades más alto que el izquierdo, es decir: cuando su FB sea de 2. Y además, la raíz del subárbol derecho tenga una FB de -1, es decir, que esté cargado a la izquierda. Se trata del caso simétrico del anterior.

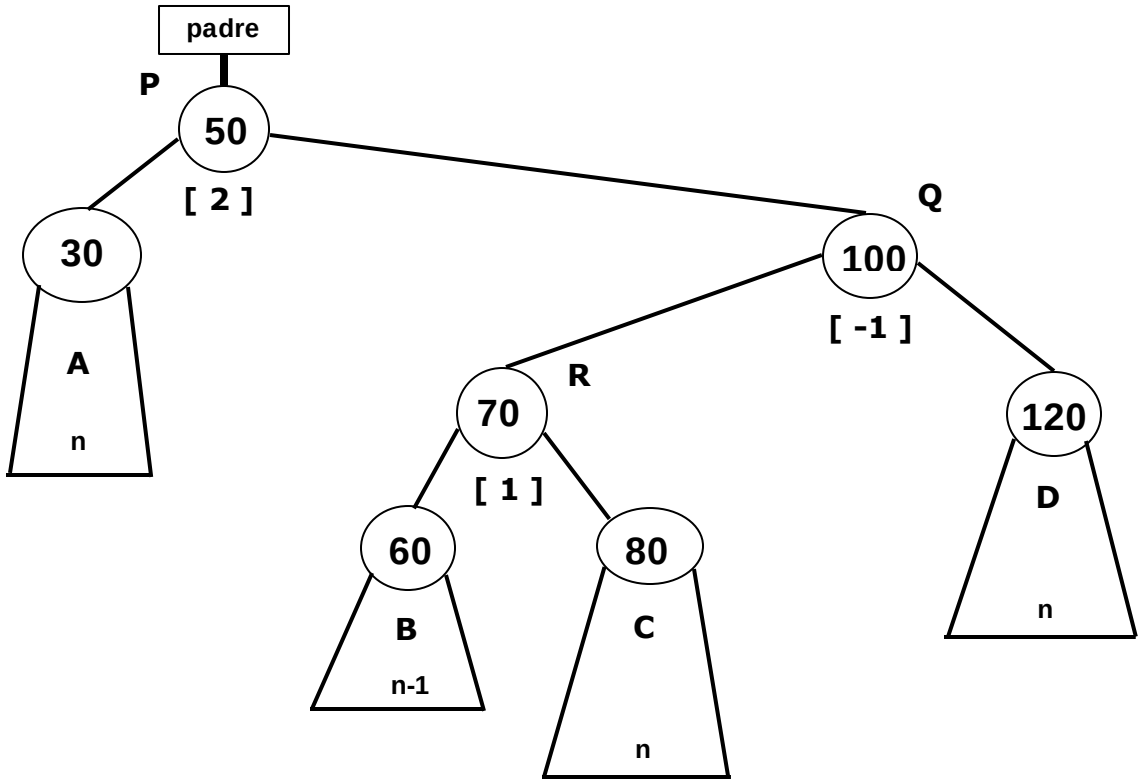


Figura 55. Árbol P desequilibrado a la derecha

En este caso también tendremos que realizar dos rotaciones:

Llamaremos P al nodo que muestra el desequilibrio, el que tiene una FB de 2.

Llamaremos Q al nodo raíz del subárbol derecho de P.

Llamaremos R al nodo raíz del subárbol izquierdo de Q.

1. Haremos una ***rotación simple de Q a la derecha***.
2. Después, haremos una ***rotación simple de P a la izquierda***.

Con más detalle, procederemos del siguiente modo:

1. **Pasamos el subárbol derecho del nodo R como subárbol izquierdo de Q.** Esto mantiene el árbol como ABB, ya que todos los valores a la derecha de R siguen estando a la izquierda de Q.
2. **Ahora, el nodo R pasa a tomar la posición del nodo Q,** es decir, hacemos que la raíz del subárbol derecho de P sea el nodo R en lugar de Q.
3. **Paso 3-A. El árbol Q pasa a ser el subárbol derecho del nodo R.**

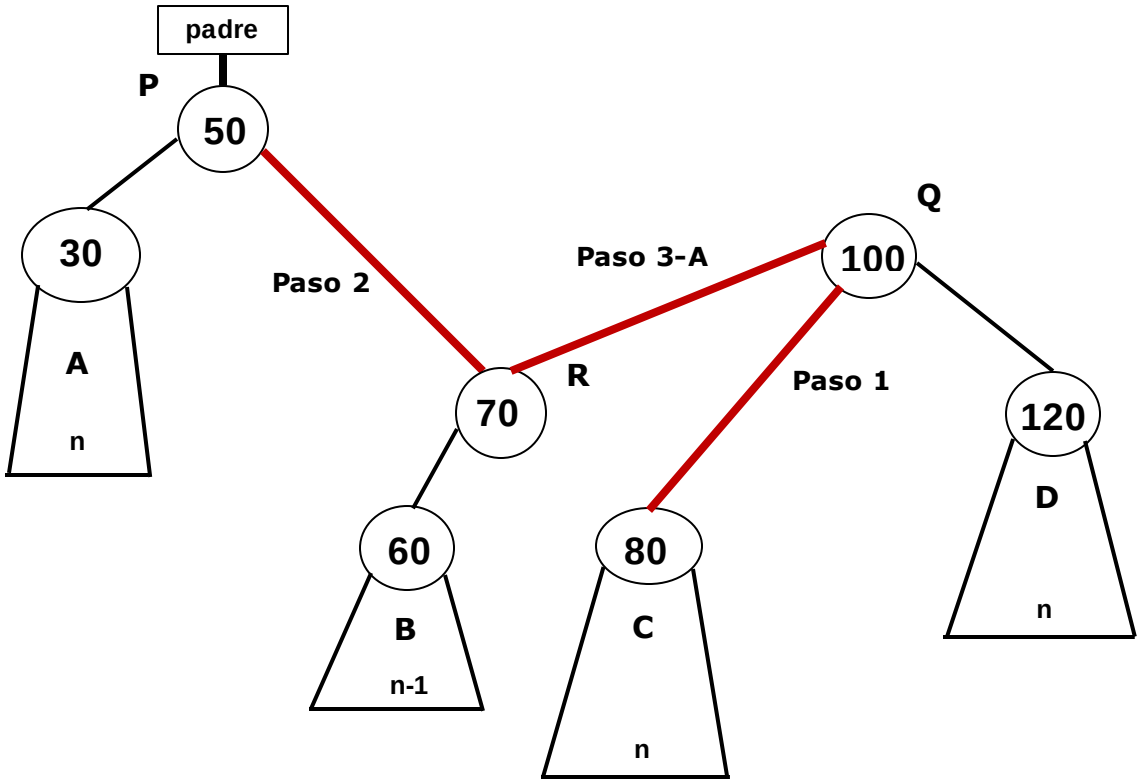


Figura 56. RDI pasos 1 a 3-A.

Paso 3-B. Aquí ocurre la rotación como tal.

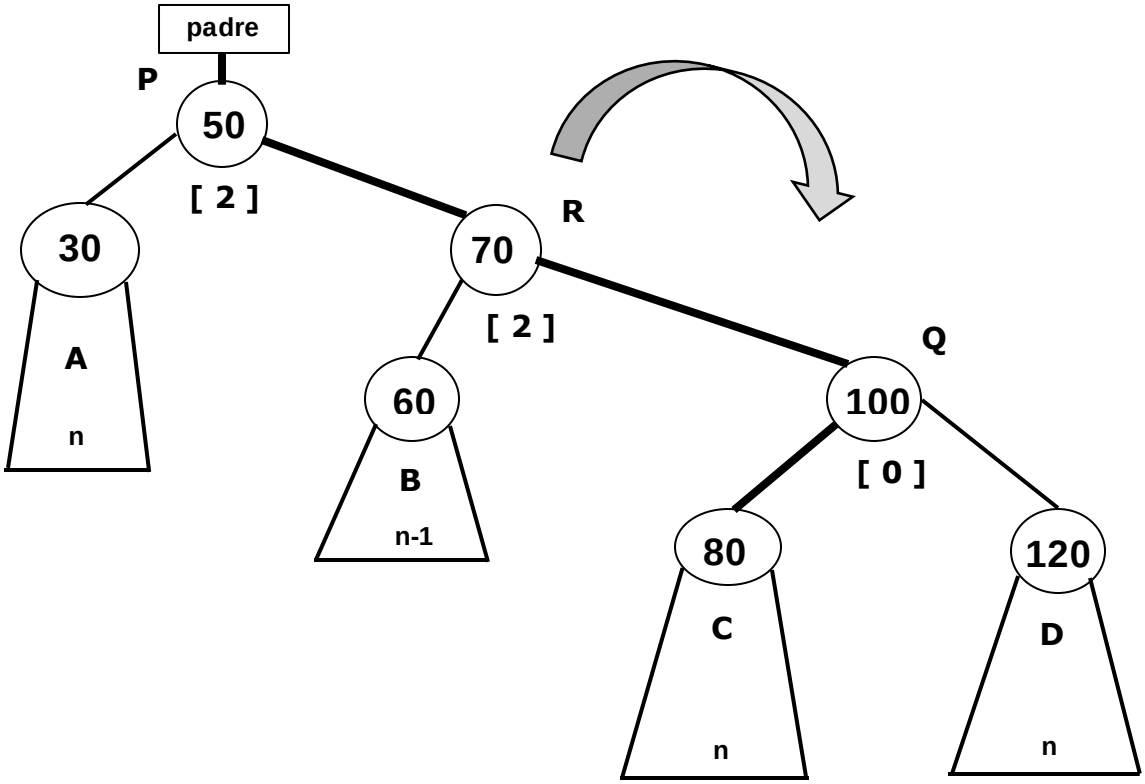


Figura 56-A. Árbol resultante de la primera rotación (simple de Q a la derecha).

4. **Pasamos el subárbol izquierdo del nodo R como subárbol derecho de P.** Esto mantiene el árbol como ABB, ya que todos los valores a la izquierda de R siguen estando a la derecha de P.
5. **Ahora, el nodo R pasa a tomar la posición del nodo P, es decir, hacemos que la entrada al árbol sea el nodo R, en lugar del nodo P.** Como en los casos anteriores, previamente, P puede que fuese un árbol completo o un subárbol de otro nodo de menor altura.

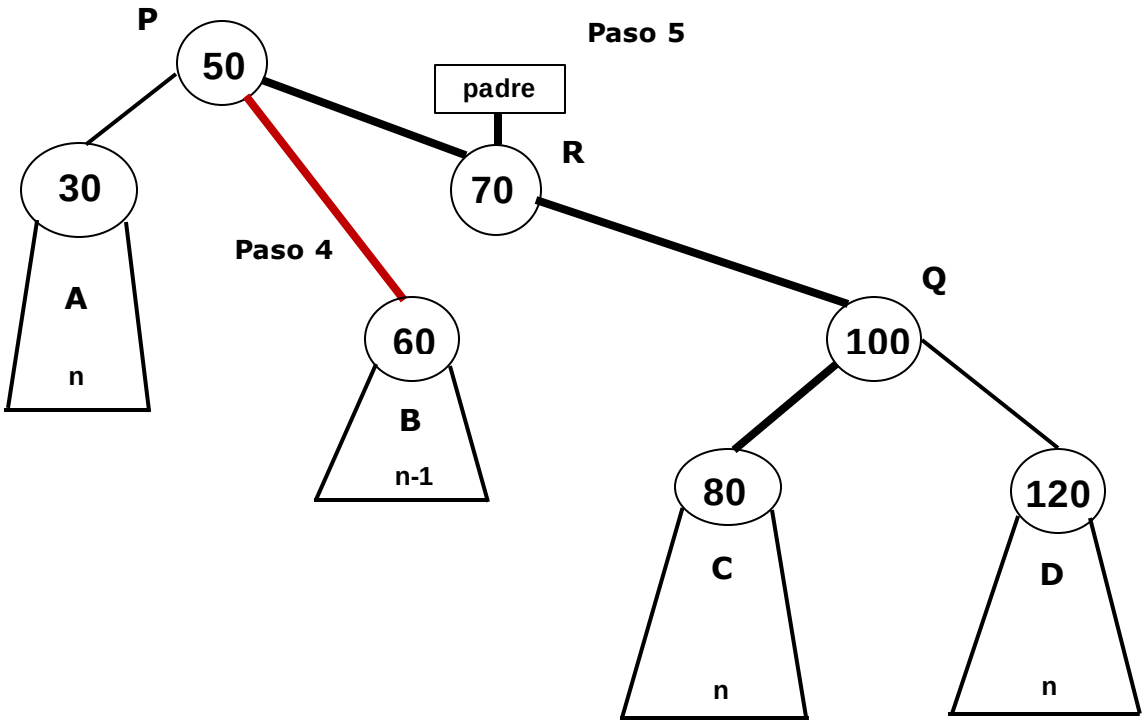


Figura 57. RDI pasos 4 y 5.

6. El árbol P pasa a ser el subárbol izquierdo del nodo R. Aquí ocurre la rotación como tal.

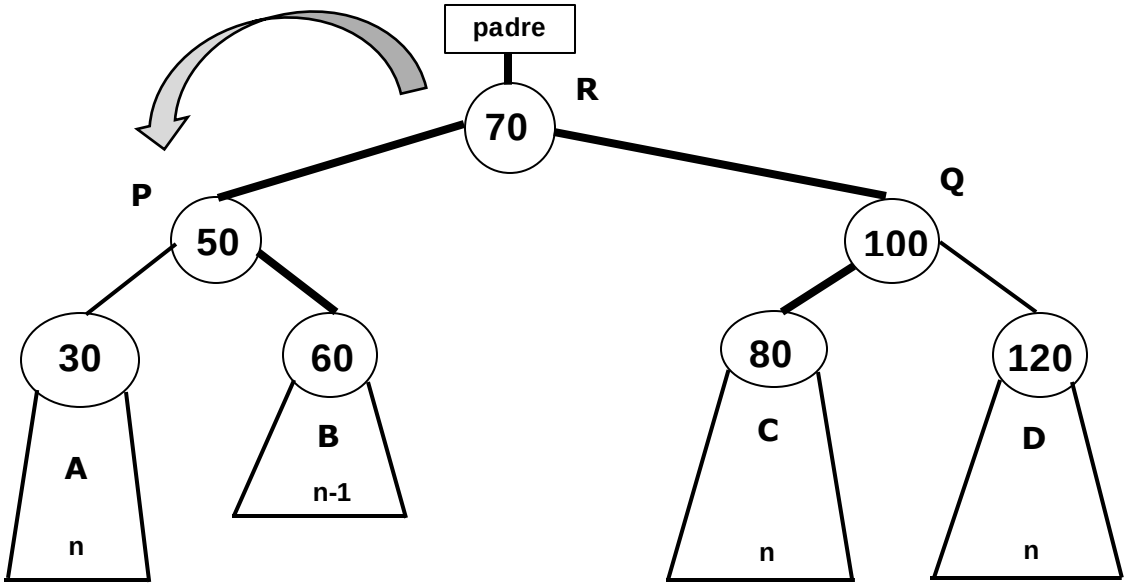


Figura 58. RDI paso 6 (Rotación simple de P a la Izquierda).

La Figura 59 muestra el árbol resultante.

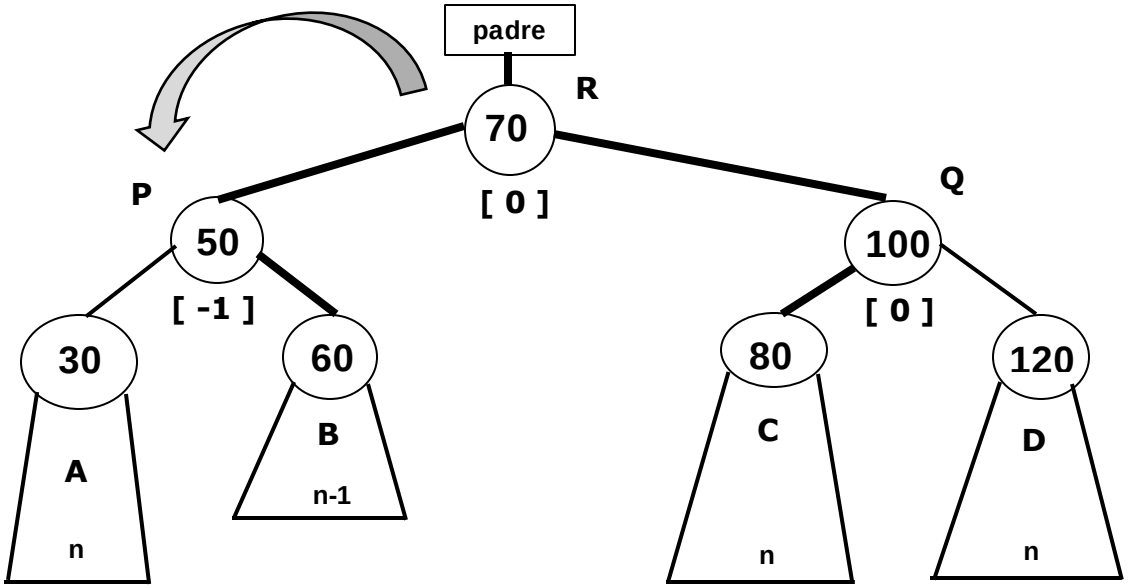


Figura 59. Árbol resultante

RE-EQUILIBRIO EN ÁRBOLES AVL POR INSERCIÓN DE UN NODO

Cuando el valor de FB de un nodo tome el valor -2 ó 2, no seguiremos el camino, sino que, con el valor de FB del nodo actual y el del nodo derecho si FB es 2 o el del nodo izquierdo si es -2, determinaremos qué tipo de rotación debemos hacer:

FB Nodo Actual	FB Nodo Derecho	FB Nodo Izquierdo	ROTACIÓN
-2	No importa	-1 ó 0	**RSD**
-2	No importa	1	**RDD**
2	-1	No importa	**RDI**
2	1 ó 0	No importa	**RSI**

El resto de los casos no interesan porque:

en nodos desequilibrados hacia la derecha, con valores de FB positivos, siempre buscaremos el equilibrio mediante rotaciones a la izquierda,

y viceversa, con nodos desequilibrados hacia la izquierda, con valores de FB negativos, buscaremos el equilibrio mediante rotaciones a la derecha.

Supongamos que el valor de FB del nodo ha pasado de -1 a -2, debido a que se ha añadido un nodo. Esto implica que el nodo añadido lo ha sido en la rama izquierda, si lo hubiéramos añadido en la derecha el valor de FB nunca podría decrecer.

RE-EQUILIBRIO EN ÁRBOLES AVL POR BORRADO DE UN NODO

Cuando el desequilibrio se debe a la eliminación de un nodo la cosa puede ser algo diferente, pero veremos que siempre se puede llegar a uno de los casos anteriores.

Supongamos el siguiente ejemplo, en el árbol AVL eliminaremos el nodo de valor 3:

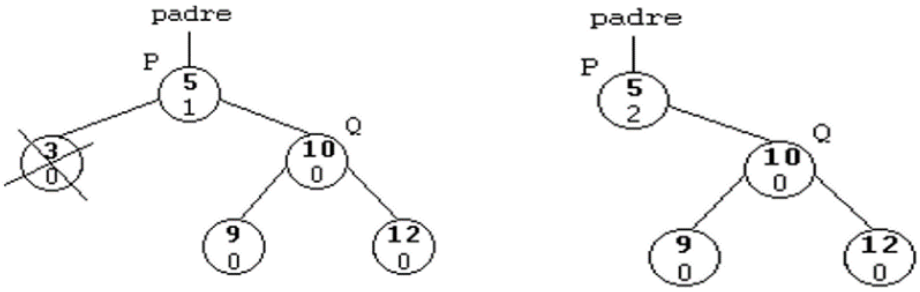
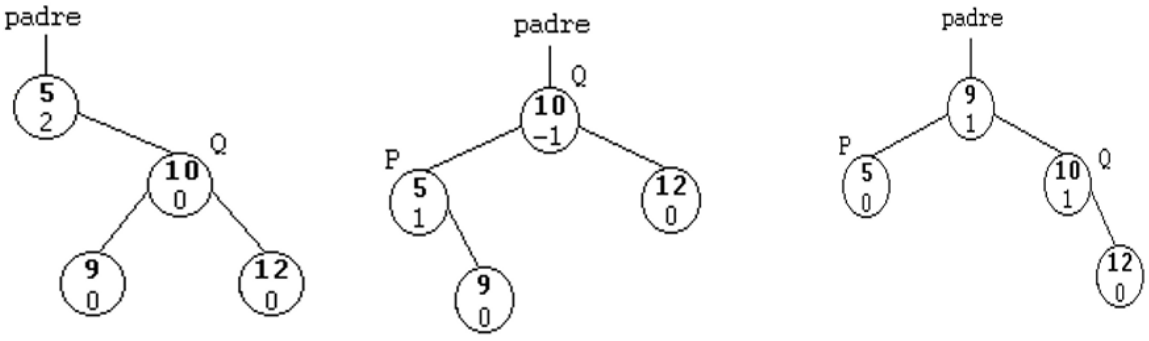


Figura 60. Borrado en un AVL que deja el árbol desequilibrado hacia la derecha

El valor de FB del nodo P pasa de 1 a 2, sabemos que cuando el valor de FB de un nodo es 2 siempre tenemos que aplicar una rotación a izquierdas.
Para saber cuál de las dos rotaciones debemos aplicar, miramos el valor de FB del nodo derecho.
Pero en este caso, el valor de FB de ese nodo es 0.
Esto no quiere decir que no podamos aplicar ninguna de las rotaciones, por el contrario, podremos aplicar cualquiera de ellas. *Aunque por economía, lo razonable es aplicar la rotación simple*.



(a) Árbol desequilibrado hacia la derecha (b) Aplicando rotación simple (c) Aplicando rotación doble

Figura 61.

Del mismo modo, el valor de FB del nodo derecho podría haber sido 1 ó - 1, en ese caso sí está determinado el tipo de rotación a realizar.

> El razonamiento es similar al eliminar nodos y el resultado es un nodo con FB de -2,
> en este caso se realizará una **rotación a derechas**,
> y la rotación dependerá del valor de FB del nodo izquierdo al que muestra el desequilibrio.
> **Si es 0 ó -1 haremos una rotación simple, si es 1, haremos una rotación doble.**

ALGORITMOS DE LAS ROTACIONES EN ÁRBOLES AVL.

Algoritmo para la rotación simple

Suponga que los apuntadores P1,P2 y P3 ya están colocados según los esquemas:
P1= apuntador al nodo padre del nodo pivote.
P2= apuntador al nodo pivote.
P3= apuntador al nodo hijo del nodo pivote, que es la raíz del subárbol más grande.

```
[1]   Si P1 no apunta a vacío
          Si la información del nuevo nodo es menor que la información apuntada por P1:
                Hijo izquierdo de P1=P3
          si no
                Hijo derecho de P1=P3
          Fin_Si
      si no
          P3 es la nueva raíz del árbol
      Fin_Si

[2]   Si la información del nuevo nodo es menor que la información apuntada por P2:
          Hijo izquierdo  de P2= Hijo derecho de P3
          Hijo derecho de P3 = P2
          Modificar FB desde hijo izquierdo de P3 hasta el padre del nuevo nodo
      si no
          Hijo derecho de P2 = Hijo izquierdo de P3
          Hijo izquierdo de P3 = P2
          Modificar FB desde el hijo derecho de P3 hasta el padre del nuevo nodo.
      Fin_Si

[3]    El FB del nodo señalado por P2 = 0
```

Algoritmo para la rotación doble

Suponga que los apuntadores P1,P2 y P3 ya están colocados según los esquemas:
P1= apuntador al nodo padre del nodo pivote.
P2= apuntador al nodo pivote.
P3= apuntador al nodo hijo del nodo pivote, que es la raíz del subárbol más grande.
P4= apuntador al nodo hijo del nodo apuntado por P3, que sigue en la ruta de búsqueda del nuevo nodo

```
[1]   Si P1 no apunta a vacío
          Si la información del nuevo nodo es menor que la información apuntada por P1:
                Hijo izquierdo de P1=P4
          si no
                Hijo derecho de P1=P4
          Fin_Si
      si no
          P4 es la nueva raíz del árbol
      Fin_Si

[2]   Si la información del nuevo nodo es menor que la información apuntada por P2:
          Hijo derecho de P3 = hijo izquierdo de P4
          Hijo izquierdo de P2 = hijo derecho de P4
          Hijo izquierdo de P4 = P3
          Hijo derecho de P4 = P2

          ---- seguir en [3.a]
          [3.a]
          Si la información del nuevo nodo es menor que la información de P4:
                modificar el FB desde el hijo derecho de P3 hasta el padre del nuevo nodo
                modificar el FB del nodo señalado por P2 (ahora vale +1)
          si no
                Si la información del nuevo nodo es mayor a la información de P4:
                      modificar FB desde hijo izquierdo de P2 hasta padre del nuevo nodo
                      modificar FB del nodo señalado por P3 (ahora vale -1)
                Fin_Si
          Fin_Si

          modificar FB del nodo señalado por P2 (ahora vale 0)


      si no
          Hijo izquierdo de P3= hijo derecho de P4
          Hijo derecho de P2= hijo izquierdo de P4
          Hijo derecho de P4 = P3
          Hijo izquierdo de P4 = P2

          ---- seguir en [3.b]
          [3.b]
          Si la información del nuevo nodo es mayor que la información de P4:
                modificar FB desde el hijo izquierdo de P3 hasta el padre del nuevo nodo
                modificar FB del nodo señalado por P2 (ahora vale -1)
          si no
                Si la información del nuevo nodo es menor a la información de P4:
                      modificar FB desde hijo derecho de P2 hasta el padre del nuevo nodo
                      modificar FB del nodo señalado por P3 (ahora vale +1)
                Fin_Si
          Fin_Si

          modificar FB del nodo señalado por P2 (ahora vale 0)
      Fin_Si
```

---------- ***FIN DEL DOCUMENTO***