Árboles Balanceados AVL.

El material sobre el tema presentado a continuación proviene del libro "Estructuras de Datos" de los autores Cairó y Guadarti.

Cuando se estudiaron los árboles binarios de búsqueda se mencionó que es una estructura sobre la cual se pueden realizar eficientemente las operaciones de búsqueda, inserción y eliminación. Sin embargo, si el árbol crece o decrece descontroladamente, el rendimiento puede disminuir considerablemente. El caso más desfavorable se produce cuando se inserta un conjunto de claves ordenadas en forma ascendente o descendente, como se muestra en la figura 6.26.

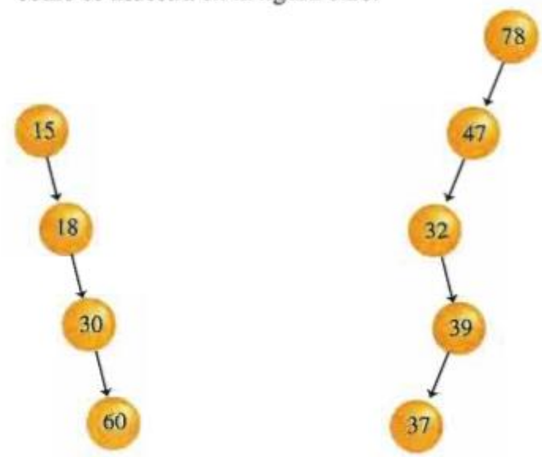


FIGURA 6.26
Árboles binarios de búsqueda con crecimiento descontrolado.

Es de notar que el número promedio de comparaciones que se deben realizar para localizar una determinada clave en un árbol binario de búsqueda con crecimiento descontrolado es *N*/2, cifra que muestra un rendimiento muy pobre en la estructura.

Con el objeto de mantener la eficiencia en la operación de búsqueda surgen los **árboles balanceados**. La principal característica de éstos es la de realizar reacomodos o balanceos, después de inserciones o eliminaciones de elementos. Estos árboles también reciben el nombre de **AVL** en honor a sus inventores, dos matemáticos rusos, G. M. Adelson-Velskii y E. M. Landis.

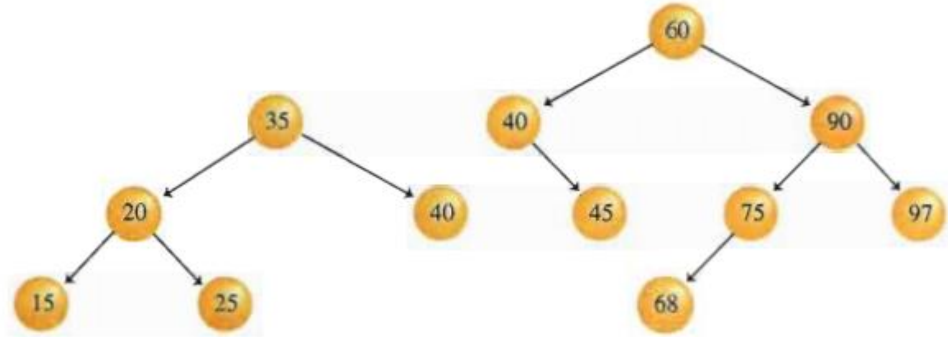
Formalmente se define un árbol balanceado como un árbol binario de búsqueda, en el cual se debe cumplir la siguiente condición: Para todo nodo *T* del árbol, la altura de los subárboles izquierdo y derecho no deben diferir en más de una unidad.

$$|H_{RI} - H_{RD}| \leq 1$$

donde H_{RI} es la altura de la rama o subárbol izquierdo y H_{RD} es la altura de la rama o subárbol derecho.

En la figura 6.27 se muestran dos ejemplos de árboles balanceados.

FIGURA 6.27
Dos árboles balanceados. a) Con altura 3. b) Con altura 4.



Observe el lector que si se insertan las claves 10, 18 o 27 en el árbol balanceado de la figura 6.27*a*, éste pierde el equilibrio. Sin embargo, si se insertan las claves 37 o 45 en el mismo árbol, éste mantiene el equilibrio; más aún, lo mejora.

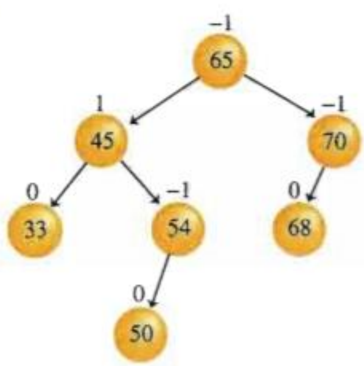
Ahora bien, con respecto a la eliminación, si a dicho árbol se le quitan las claves 15, 20 o 25 el árbol continúa siendo balanceado; incluso podrían quitársele las tres claves y el árbol no perdería el equilibrio. Sin embargo, si se elimina la clave 40 el árbol pierde el equilibrio y es necesario reestructurarlo.

Ahora bien, para poder determinar si un árbol está balanceado o no, se debe manejar información relativa al equilibrio de cada nodo del árbol. Surge así el concepto de **factor de equilibrio** de un nodo (***FE***) que se define como la altura del subárbol derecho menos la altura del subárbol izquierdo.

$$FE = H_{RD} - H_{RI}$$ **Fórmula 6.7**

Los valores que puede tomar *FE* son –1, 0, 1. Si *FE* llegara a tomar los valores de –2 o 2, entonces debería reestructurarse el árbol. En la figura 6.29 se presenta un árbol balanceado con el correspondiente *FE* para cada nodo del árbol.

FIGURA 6.29
Árbol balanceado con el correspondiente *FE*.



Observe el lector que en la figura 6.29 el *FE* del nodo que almacena el 65 es –1 puesto que la altura del subárbol derecho es igual a 2 y la altura del subárbol izquierdo es igual a 3.

$$FE_{65} = 2 - 3 = -1$$

El *FE* de 45 se calcula como:

$$FE_{45} = 2 - 1 = 1$$

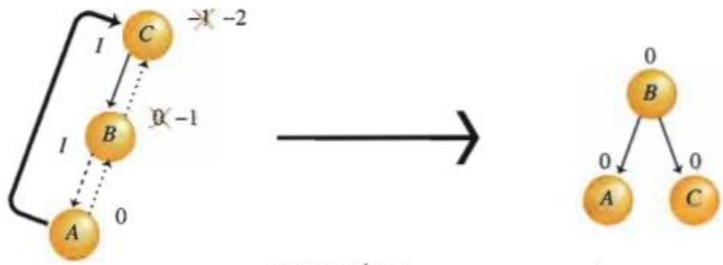
ROTACIONES EN AVL

Ejemplo 6.17

En la figura 6.30 se pueden observar gráficamente los diferentes tipos de rotaciones. En la figura 6.30*a* se presenta la rotación *II*, en la figura 6.30*b* la rotación *DD*, en la figura 6.30*c* la rotación *DI* y en la figura 6.30*d* la rotación *ID*.

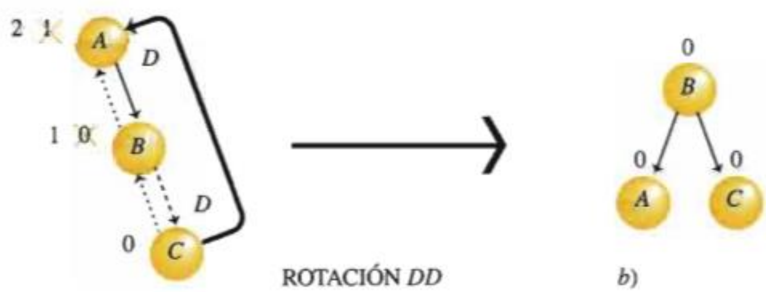
La línea continua (______) marca el estado de los nodos del árbol antes de realizar la inserción. La línea discontinua (_ _ _ _) indica el nuevo elemento insertado. La línea con puntos (.......) marca el camino de regreso hasta que se detecta el desequilibrio del árbol. La línea gruesa (______) indica el movimiento de los nodos en la rotación.

...RA 6.30
...es en árboles
...dos. a) Rotación *II*.
... *DD*. c) Rotación
...ción *ID*.

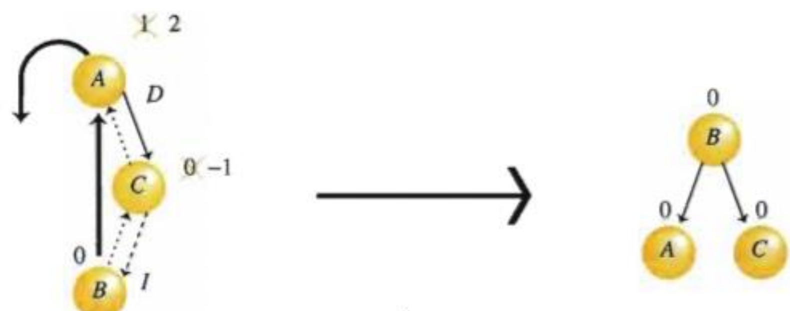


ROTACIÓN *II* *a*)

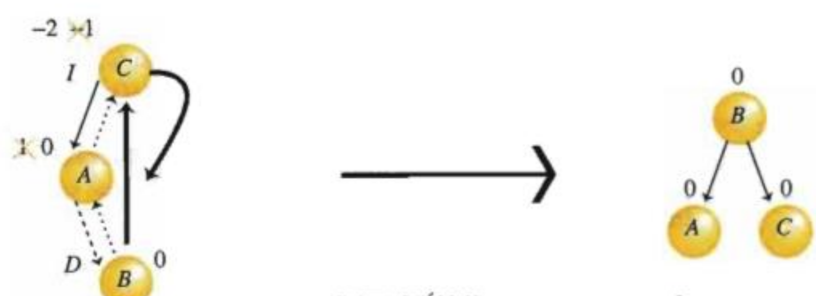
FIGURA 6.30
(*continuación*)



ROTACIÓN *DD* *b*)



ROTACIÓN *DI* *c*)



ROTACIÓN *ID* *d*)

Ejemplo 6.18 Supongamos que se desea insertar las siguientes claves en un árbol balanceado que se encuentra vacío:

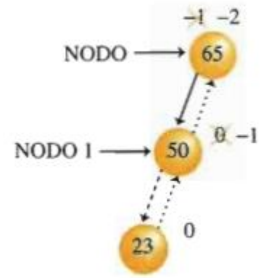
65 - 50 - 23 - 70 - 82 - 68 - 39

Las operaciones necesarias son las siguientes:

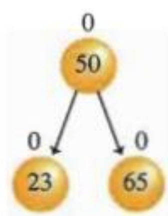
a) INSERCIÓN: CLAVE 65

b) INSERCIÓN: CLAVE 50

c) INSERCIÓN: CLAVE 23

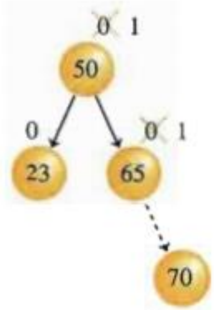


Luego de efectuar el reacomodo, el árbol queda así:

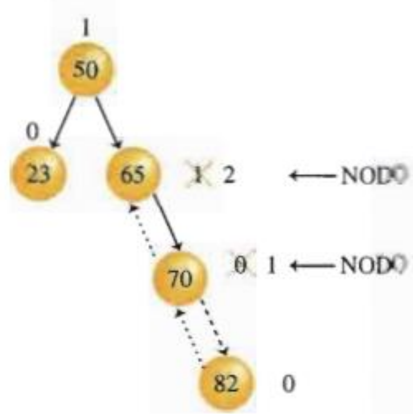


Al regresar siguiendo el camino de búsqueda se modifica el *FE* de los nodos 65 y 50, pero el equilibrio del árbol se mantiene.

d) INSERCIÓN: CLAVE 70



e) INSERCIÓN: CLAVE 82

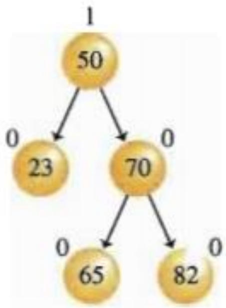


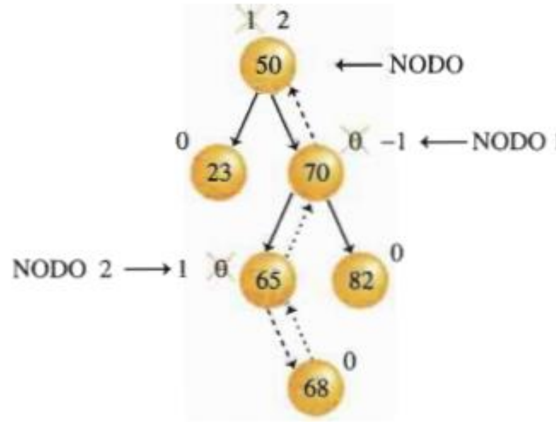
ROTACIÓN *DD*

Al regresar, luego de insertar el valor 82, siguiendo el camino de búsqueda se ob-serva una violación al criterio de equilibrio del árbol y se debe reestructurar. Se ap... con NODO la clave 65 y con NODO1 la rama derecha de dicha clave. Se verifica el *FE* de NODO1 y como en este caso es igual a 1 se puede realizar la rotación *DD*. El m...

Luego de volver a equilibrarlo, el árbol queda de esta forma:

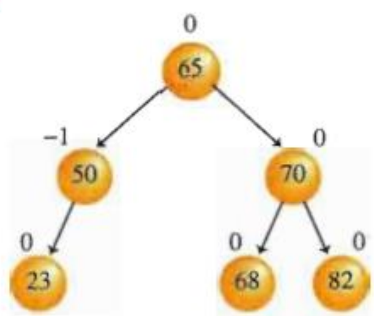
f) INSERCIÓN: CLAVE 68

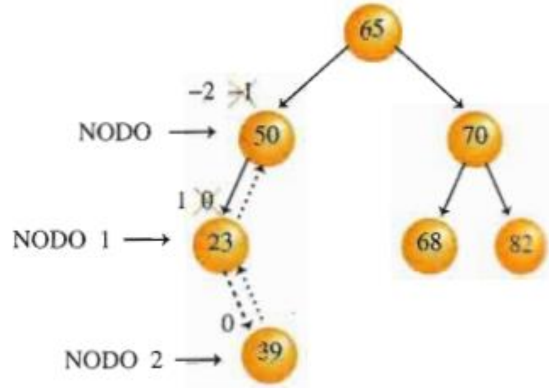




ROTACIÓN *DI*

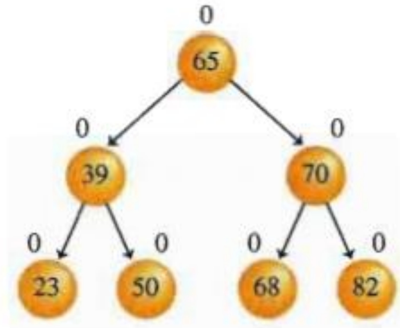
Luego de insertar la clave 68 y al regresar siguiendo el camino de búsqueda se advierte que en la clave 50 se rompe el equilibrio del árbol. Se apunta con NODO la clave 50 y con NODO1 su rama derecha. Se calcula el *FE* de NODO1 y como en este caso es igual a –1, se realiza la rotación *DI*. Se apunta entonces con NODO2 la rama izquierda





ROTACIÓN *ID*

Luego de volver a equilibrarlo, el árbol queda de la siguiente forma:



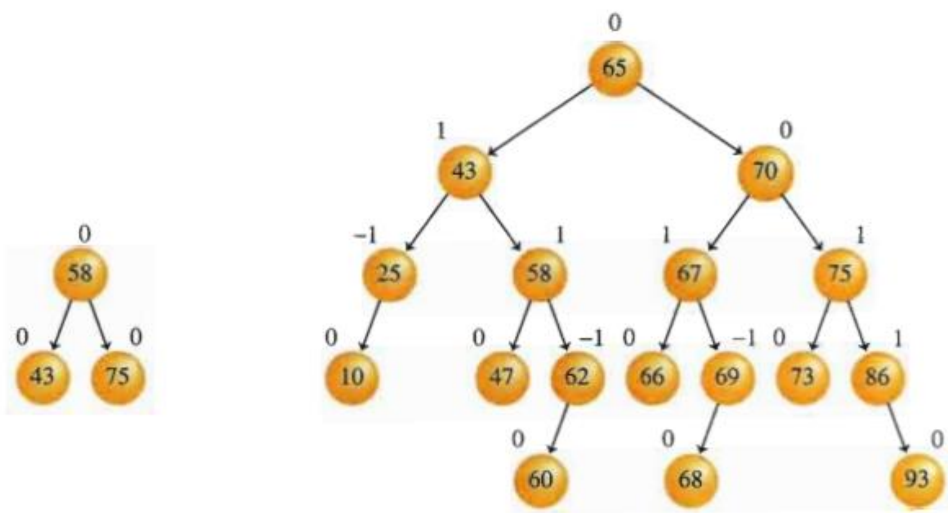
Nota: Observe que luego de realizar la inserción de un elemento y cuando se regresa por el camino de búsqueda, el *FE* del nodo visitado se incrementa en 1 si la inserción se hizo por su rama derecha y disminuye en 1 si la inserción se hizo por su rama izquierda.

Ejemplo 6.19

Dado como dato el árbol balanceado de la figura 6.31*a*, verifique si el mismo queda igual al de la figura 6.31*b* luego de insertar las siguientes claves:

86 - 65 - 70 – 67 - 73 - 93 - 69 - 25 - 66 - 68 - 47 - 62 - 10 - 60

FIGURA 6.31
Inserción en árboles balanceados. *a*) Antes de insertar las claves. *b*) Después de insertar las claves.



a) *b*)

Eliminación en árboles balanceados

La operación de **eliminación en árboles balanceados** es más compleja que la operación de inserción, como normalmente ocurre en casi todas las estructuras de datos. Consiste en quitar un nodo del árbol sin violar los principios que definen un árbol balanceado. Recuerde que se definió como una estructura en la cual, para todo nodo del árbol, se debe cumplir que: **la altura del subárbol izquierdo y la altura del subárbol derecho no deben diferir en más de una unidad.**

Eliminar nodos en un árbol balanceado resulta difícil a pesar de que se utiliza el mismo algoritmo de eliminación, idéntico en lógica pero diferente en implementación, que en los árboles binarios de búsqueda y las mismas operaciones de reacomodo que se utilizan en el algoritmo de inserción en árboles balanceados.

En la operación de eliminación en árboles balanceados se deben distinguir los siguientes casos:

1. Si el elemento a eliminar es terminal u hoja, simplemente se suprime.
2. Si el elemento a eliminar tiene un solo descendiente, entonces se tiene que sustituir por ese descendiente.
3. Si el elemento a eliminar tiene los dos descendientes, entonces se tiene que sustituir por el nodo que se encuentra más a la izquierda en el subárbol derecho o por el nodo que se encuentra más a la derecha en el subárbol izquierdo.

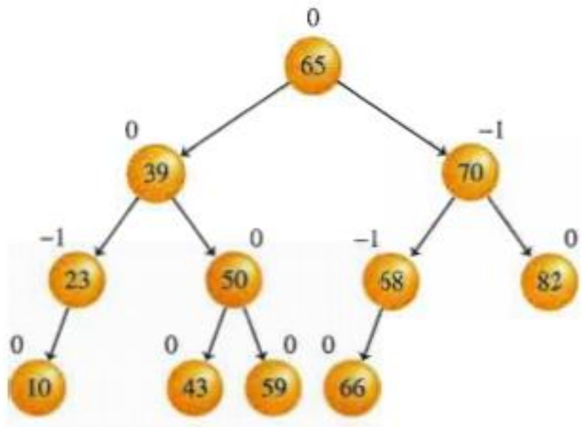
Para eliminar un nodo en un árbol balanceado lo primero que se debe hacer es localizar su posición en el árbol. Se elimina siguiendo los criterios establecidos anteriormente y se regresa por el camino de búsqueda calculando el *FE* de los nodos visitados. Si en alguno de los nodos se viola el criterio de equilibrio, entonces se debe reestructurar el árbol. El proceso termina cuando se llega a la raíz del árbol. Cabe aclarar que mientras que en el algoritmo de inserción una vez efectuada una rotación se podía detener el proceso, en este algoritmo se debe continuar puesto que se puede producir más de una rotación en el camino hacia atrás. Para comprender mejor la operación de eliminación en árboles balanceados, observe el siguiente ejemplo.

Ejemplo 6.20

Supongamos que se desea eliminar las siguientes claves del árbol balanceado de la figura 6.32:

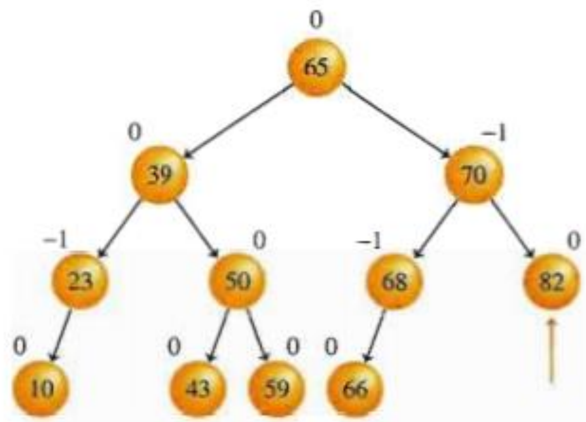
82 - 10 - 39 - 65 - 70 - 23 - 50 - 59

FIGURA 6.32
Árbol balanceado.

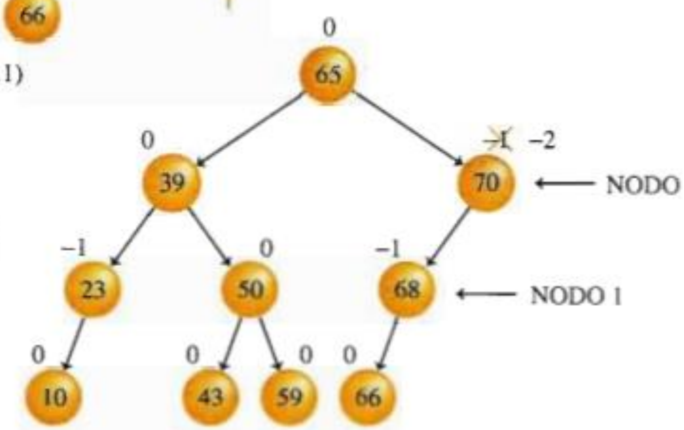


Las operaciones que se realizan son las siguientes:

a) ELIMINACIÓN: CLAVE 82

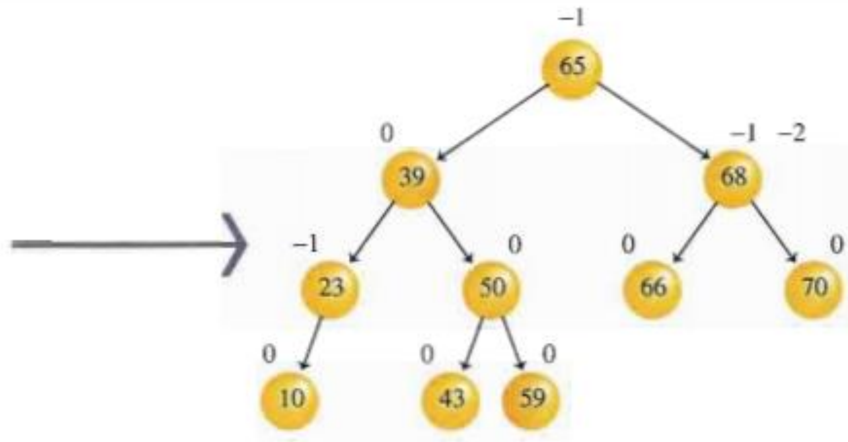


a.1)



a.2)

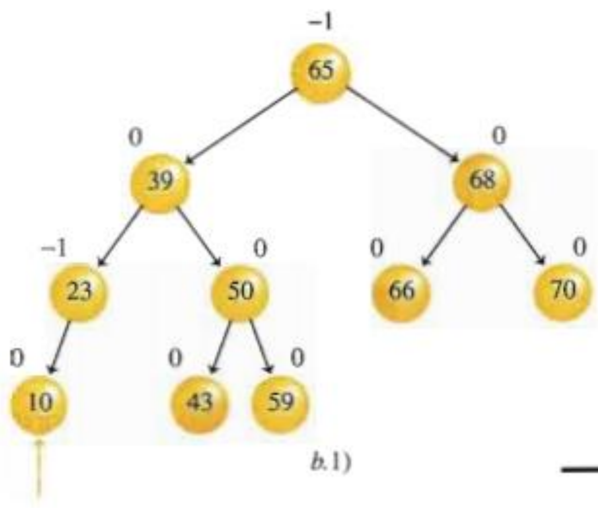
ROTACIÓN *II*



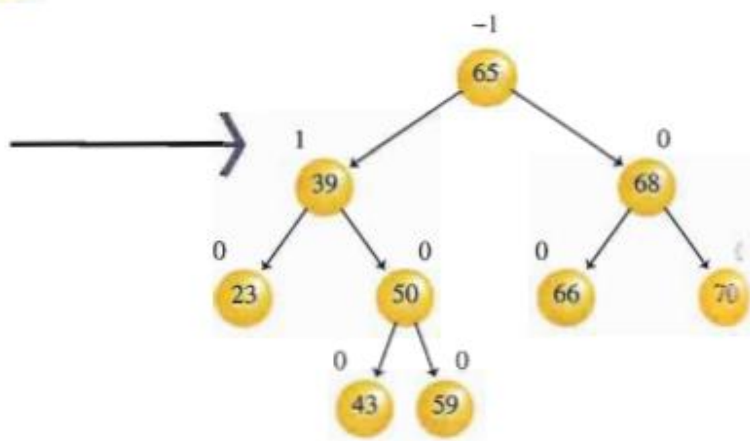
a.3)

La eliminación de la clave 82 es un proceso sencillo, ya que dicha clave no tiene descendientes (diagrama *a*.1). Al regresar siguiendo el camino de búsqueda es evidente que en la clave 70 se rompe el criterio de equilibrio y se debe reestructurar el árbol (diagrama *a*.2). Se apunta con NODO la clave 70 y con NODO1 la rama izquierda de NODO. Se verifica el *FE* de NODO1 y como éste es igual a −1, entonces se realiza la rotación *II*. Luego de la reestructuración, el árbol queda como en el diagrama *a*.3.

b) ELIMINACIÓN: CLAVE 10



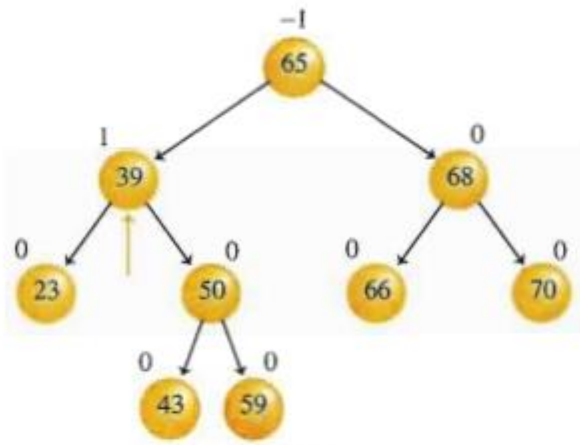
b.1)



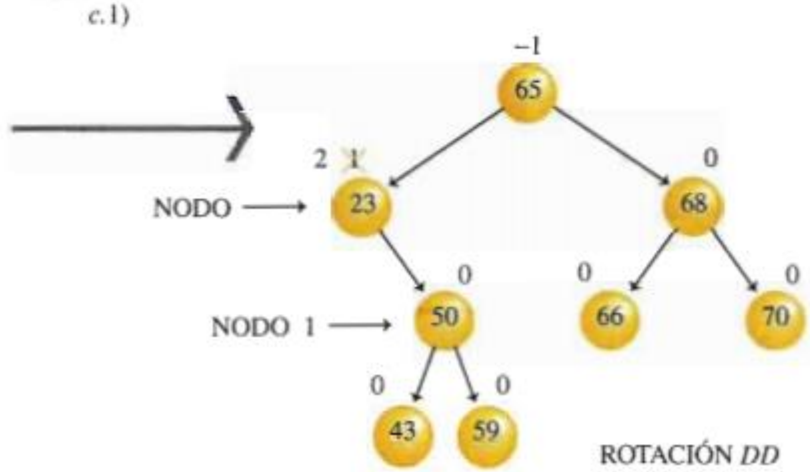
b.2)

La eliminación de la clave 10 es un proceso sencillo (diagrama *b*.1). No se debe reestructurar el árbol porque mantiene el equilibrio y sólo es necesario cambiar el *FE* de los nodos que almacenan al 23 y 39 (diagrama *b*.2).

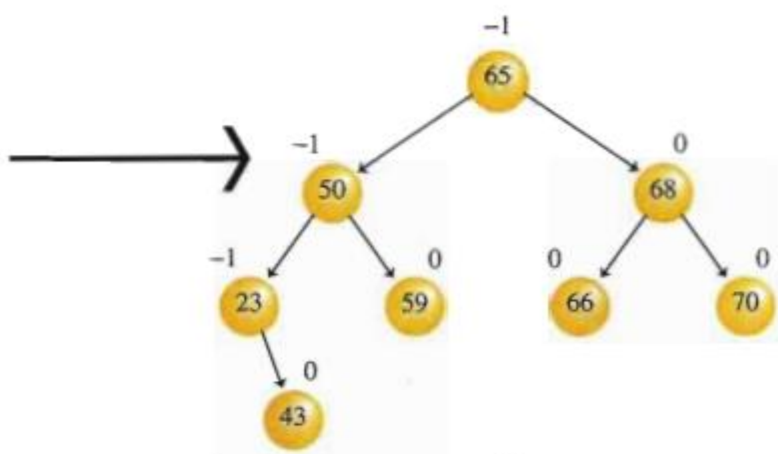
c) ELIMINACIÓN: CLAVE 39



c.1)



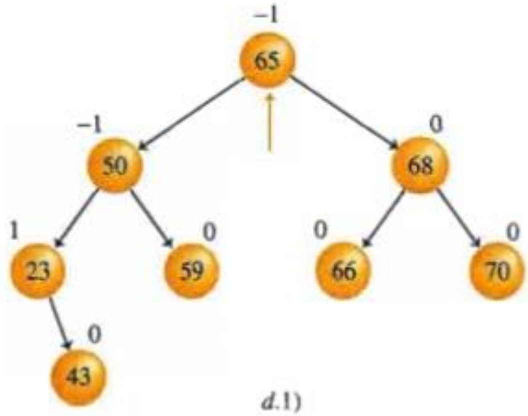
c.2)



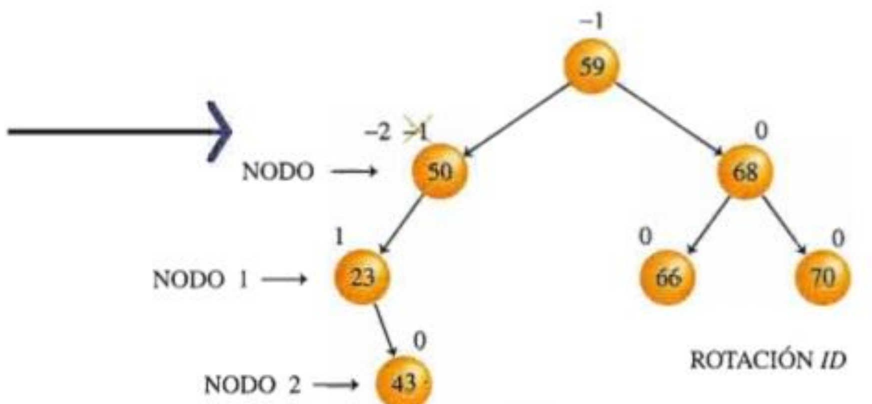
c.3)

Al eliminar la clave 39 se origina el caso más difícil de eliminación en árboles: la eliminación de una clave con dos descendientes (diagrama *c*.1). En este caso se opta por sustituir dicha clave por el nodo que se encuentra más a la derecha en el subárbol izquierdo (23). Luego de la sustitución, el árbol queda como se muestra en el diagrama *c*.2. Al realizar la sustitución se observa que en dicho nodo se viola el criterio de equilibrio y se debe reestructurar el árbol. Se apunta con NODO la clave 23 y con NODO1 la rama derecha de NODO. Se verifica el *FE* de NODO1 y como éste es igual a 0, se realiza la rotación *DD*. Luego del reacomodo, el árbol queda como en el diagrama *c*.3

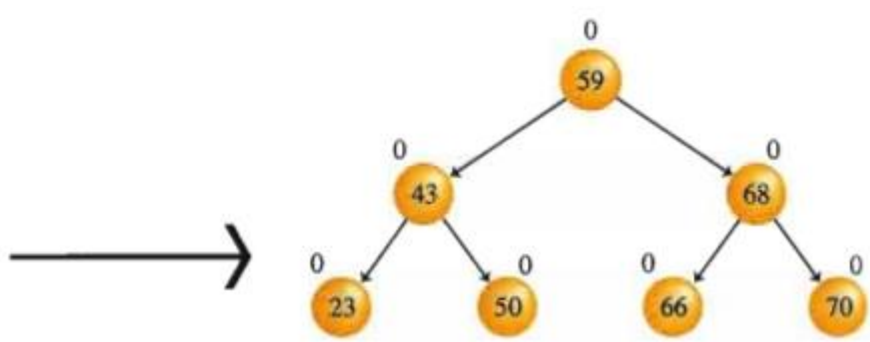
d) ELIMINACIÓN: CLAVE 65



d.1)



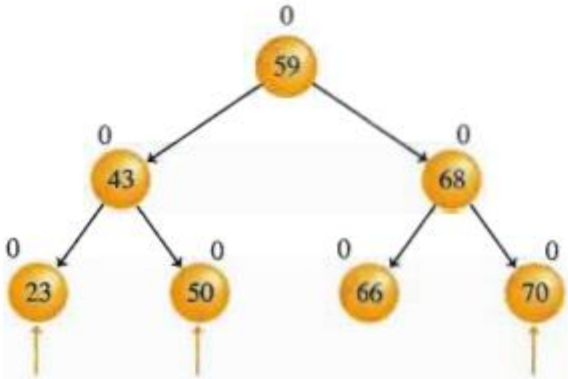
d.2)



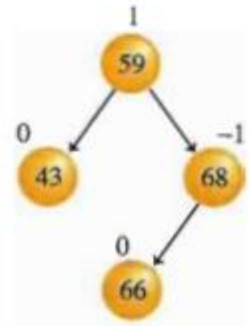
d.3)

Al eliminar la clave 65 surge nuevamente el tercer caso de eliminación, que corresponde a una clave con dos descendientes (diagrama *d*.1). Se sustituye dicha clave por el nodo que se encuentra más a la derecha en el subárbol izquierdo (59). Luego de la sustitución, el árbol queda como se presenta en el diagrama *d*.2. Es evidente que, después de la sustitución, en el nodo con la clave 50 se viola el criterio de equilibrio y se debe reestructurar el árbol. Se apunta con NODO la clave 50 y con NODO1 la rama izquierda de NODO y se verifica el *FE*. Como en este caso es igual a 1, se apunta con NODO2 la rama derecha de NODO1 y se realiza la rotación *ID*. Luego de la reestructuración, el árbol queda como el presentado en el diagrama (*d*.3).

e) ELIMINACIÓN: CLAVES 70, 23 Y 50



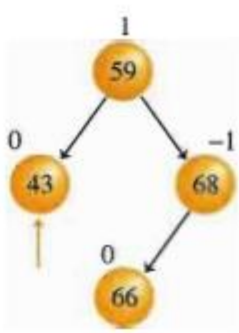
e.1)



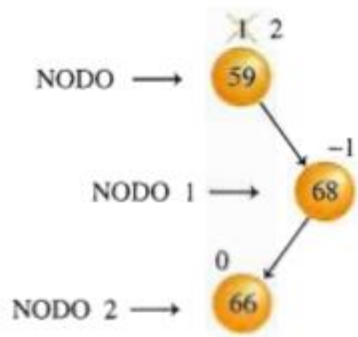
e.2)

Luego de la eliminación de las claves, el árbol queda como en el diagrama *e*.2.

f) ELIMINACIÓN: CLAVE 43

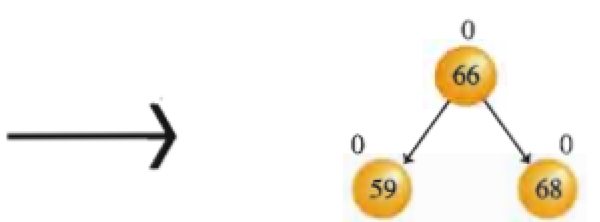


f.1)



ROTACIÓN *DI*

f.2)



f.3)

La eliminación de la clave 43 corresponde al primer caso de borrado en árboles, es el caso más simple (diagrama *f*.1). Sin embargo, al verificar el *FE* de la clave 59 se advierte que se rompe el equilibrio del árbol y se debe reestructurar (diagrama *f*.2). Se apunta con NODO la clave 59 y con NODO1 la rama derecha de NODO, y se verifica el *FE* de NODO1. Como éste es igual a –1, se apunta con NODO2 la rama izquierda de NODO1 y se realiza la rotación *DI*. Luego de la reestructuración, el árbol queda como en el diagrama *f*.3.

Observe cuidadosamente que luego de realizar la eliminación de un elemento y cuando se regresa por el camino de búsqueda, el *FE* del nodo visitado disminuye en 1 si la eliminación se hizo por su rama derecha y se incrementa en 1 si la eliminación se hizo por su rama izquierda.

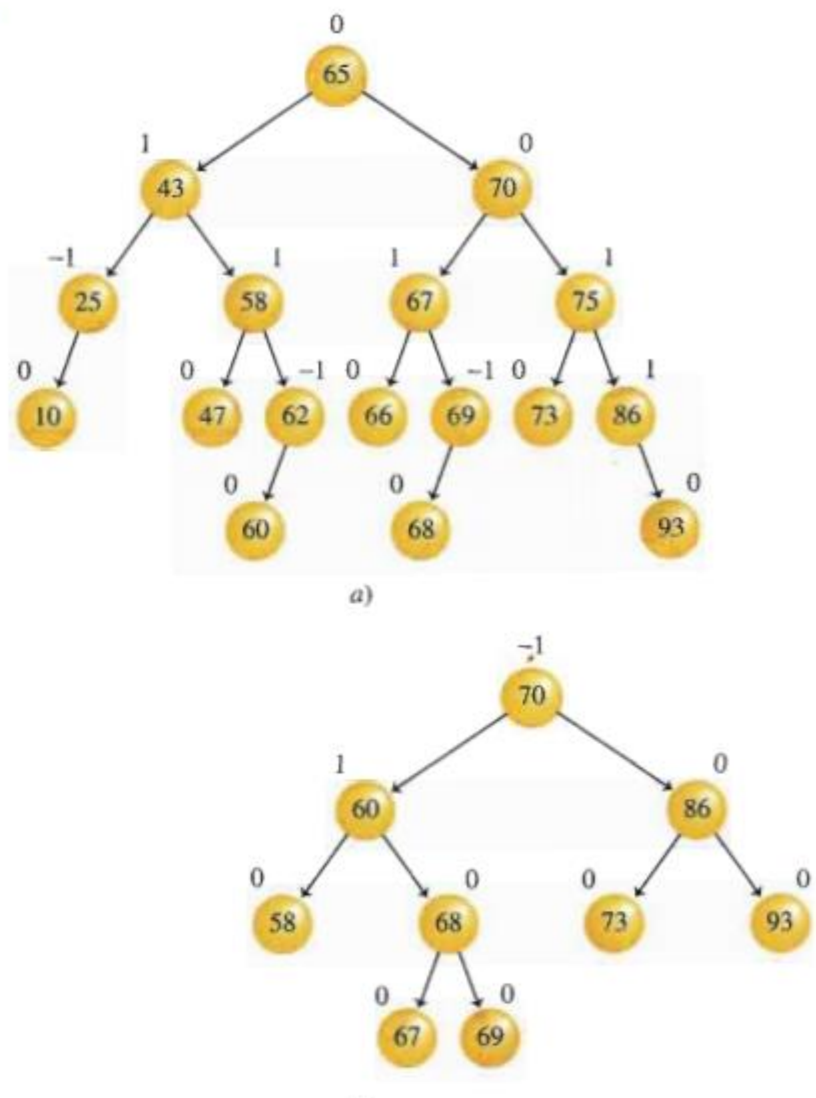
Ejemplo 6.21

Dado el árbol balanceado de la figura 6.33*a*, verifique si el mismo queda igual al de la figura 6.33*b* luego de eliminar las siguientes claves:

25 - 75 - 66 - 65 - 62 - 10 - 43 - 47

FIGURA 6.33

Eliminación en árboles balanceados. *a)* Antes de eliminar las claves. *b)* Después de eliminar las claves.



Ejemplo 6.22

Supongamos que se desea eliminar la clave 65 del árbol balanceado de la figura 6.34:

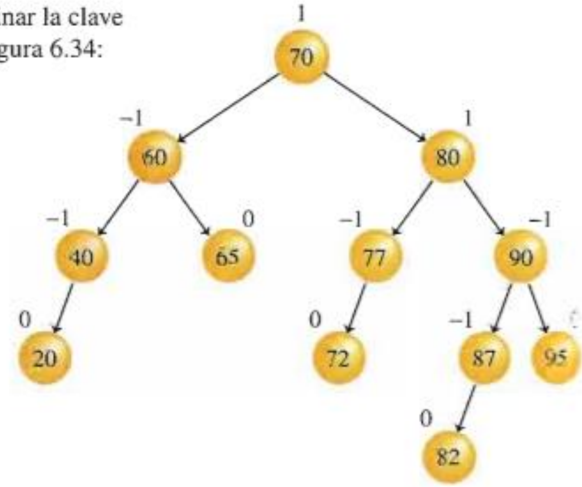
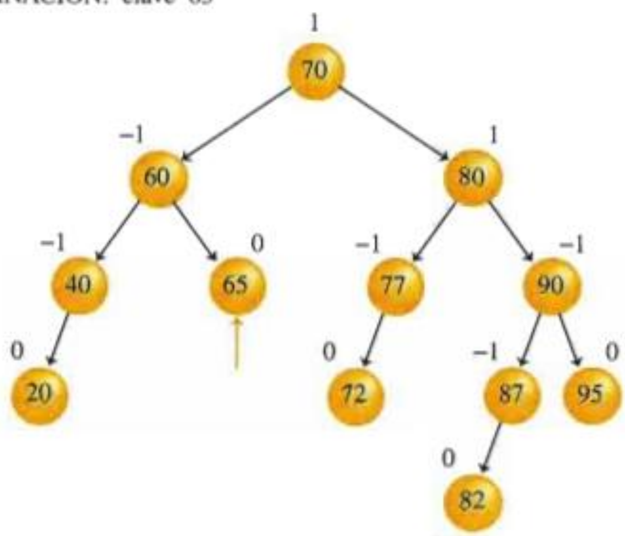


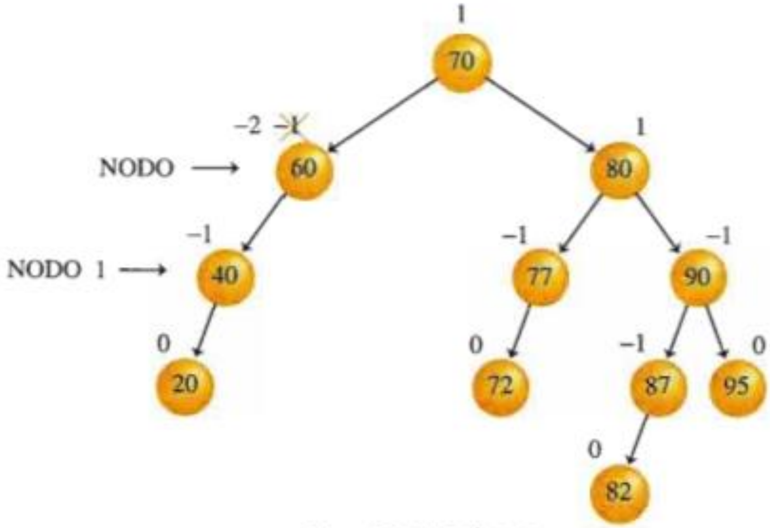
FIGURA 6.34
Árbol balanceado.

Las operaciones que se realizan son las siguientes:

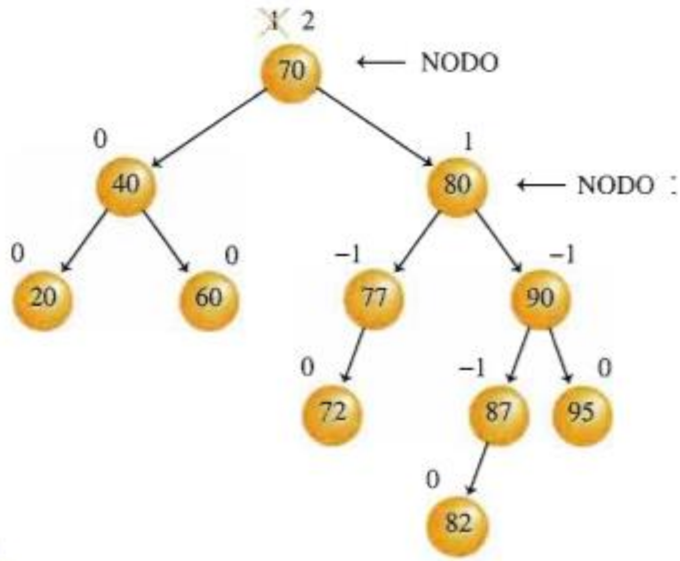
a) ELIMINACIÓN: clave 65



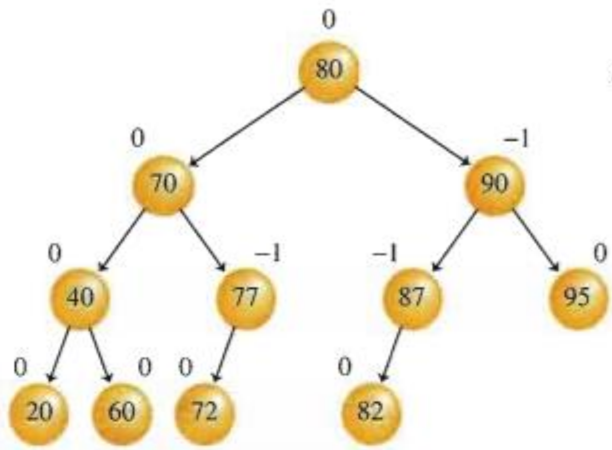
a.1)



a.2) ROTACIÓN *II*



a.3) ROTACIÓN *DD*



a.3)

Observe el lector que al eliminar la clave 65 se desbalancea el árbol y debemos efectuar la rotación *II*. Sin embargo, luego de balancear y modificar el factor de equilibrio del nodo que almacena la clave 70 nos damos cuenta que debemos efectuar un nuevo balanceo, ahora una rotación *DD*. Éste es un típico caso donde al eliminar una clave se produce una cadena de balanceos.