Estructuras para Árboles

Unidad 4

Algoritmos y Estructuras de Datos

Árboles

Definiciones

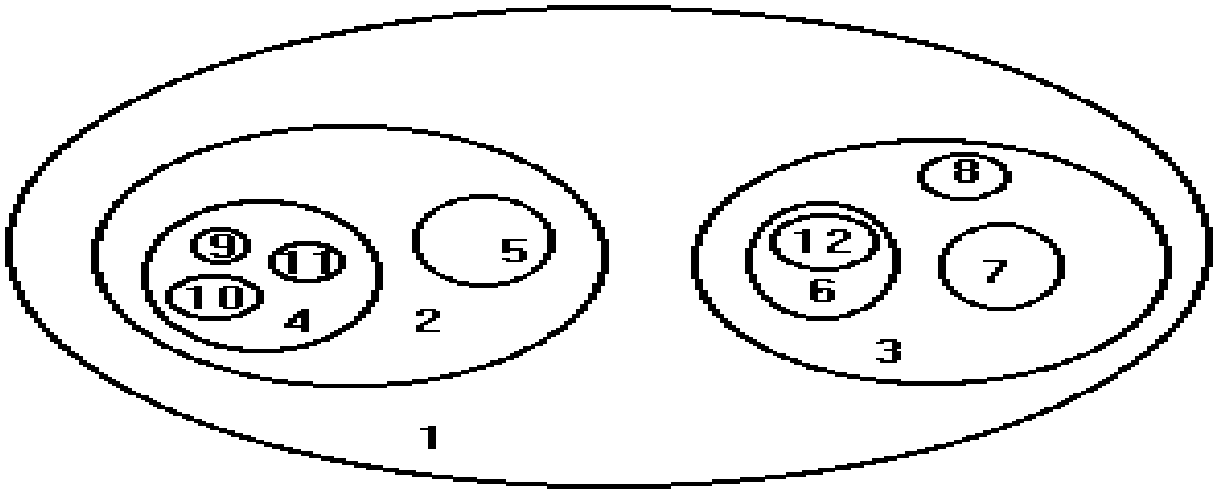
- Son de una de las estructuras de datos mas utilizadas
- Dentro de la computación tienen, entre otras las siguientes aplicaciones:
  - Organizar tablas
  - Asignación de bloques de memoria
  - Búsqueda
  - Ordenamiento

Definición

- Un árbol es un conjunto finito T de uno o mas nodos tal que:
  - Existe un nodo especial llamado raíz
  - Los nodos restantes están particionados en $m>0$ conjuntos disjuntos $T_1, \ldots T_m$ y cada uno de estos conjuntos es un árbol

Representación grafica

- Conjuntos anidados

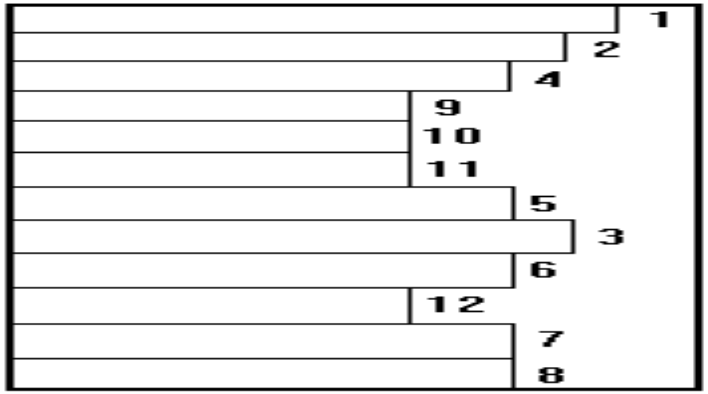


Representación grafica

- Paréntesis anidados:

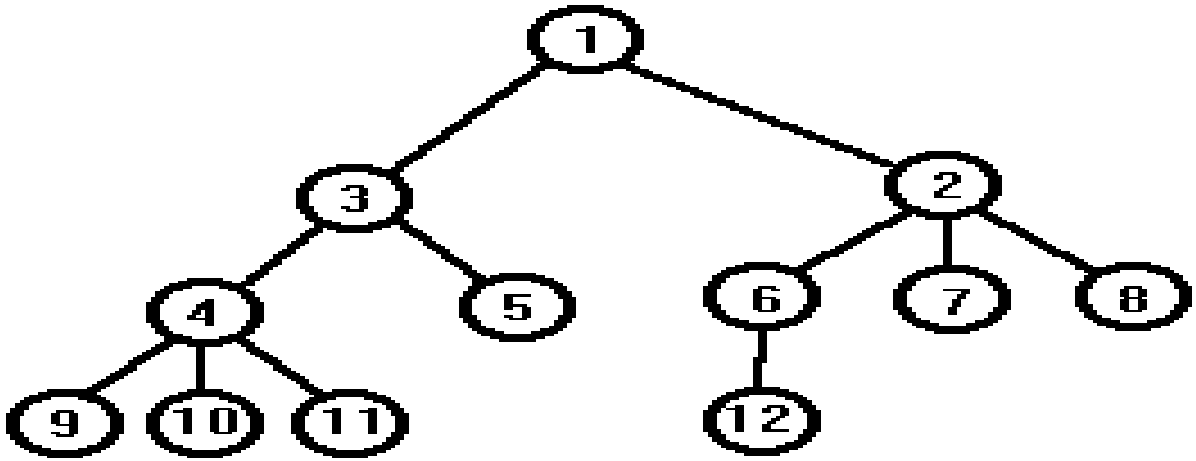
(1(2(4(9,10,11),5),3(6(12),7,8)))

- Indentacion:



Representación grafica

- La forma mas común es un grafo con la raíz hacia arriba:



Propiedades

- Cada vértice o nodo puede tener un nombre y una información asociada
- Un camino es una lista de vértices diferentes en los que vértices sucesivos están conectados por arcos en el árbol
- La longitud de un camino es el número de nodos del camino menos uno. Por tanto, hay un camino de longitud cero de cualquier nodo a si mismo

Propiedades

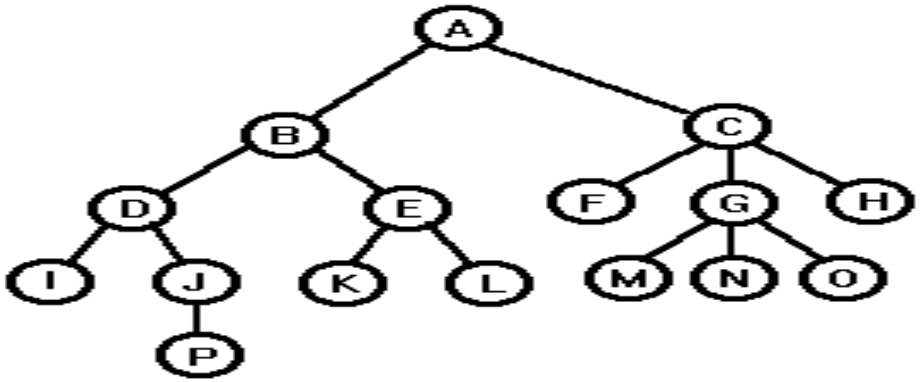
- Un nodo Y está abajo de un nodo X, si X está en el camino de Y a la raíz
- Cada nodo, excepto la raíz, tiene exactamente un nodo arriba de él, que es llamado su *padre*
- Los nodos que están exactamente abajo de él son llamados sus *hijos*
- El número de hijos que cuelgan de un nodo es llamado el *grado* del nodo

Propiedades

- El grado de un árbol es el grado máximo de los nodos del árbol

- Un nodo de grado cero es llamado hoja, es decir, no tiene hijos

Ejemplo



- Muestre el camino del nodo A al nodo P e indique la longitud del camino
- ¿Cuáles son las hojas del árbol?
- Muestre los nodos de grado 1, 2 y 3
- ¿Cuál es el grado del árbol?

SOLUCIÓN

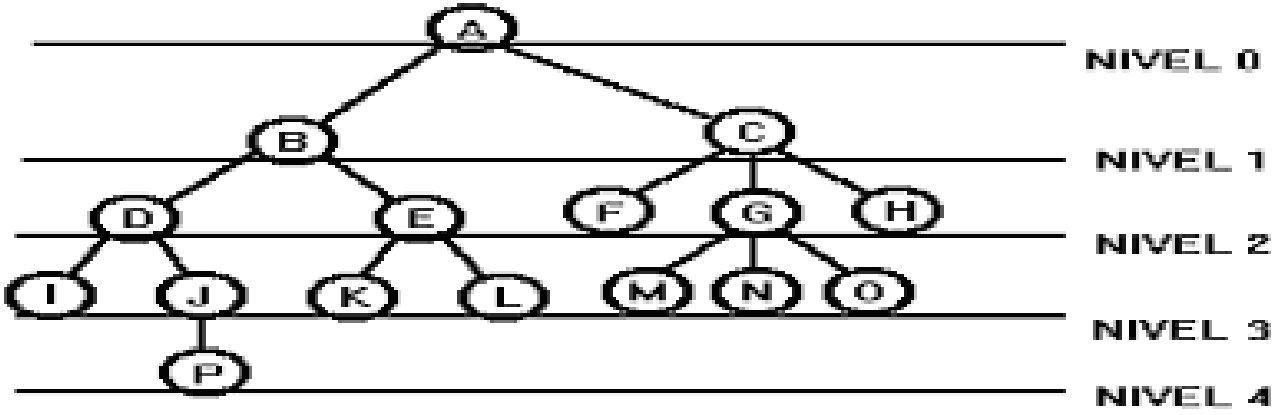
- El camino del nodo A al nodo P es (*A, B, D, J, P*), cuya longitud de camino es 4
- Las hojas son: *I, P, K, L, F, M, N, O* y *H*
- El único nodo de grado 1 es *J*.
- Los nodos de grado 2 son *A, B, D* y *E*.
- Los nodos de grado 3 son *C* y *G*.
- No existen nodos de mayor grado, por tanto, el grado del árbol es 3.

Propiedades

- El nivel de un nodo es el número de nodos en el camino de él hacia la raíz

- La altura de un árbol es el nivel máximo de todos los nodos en el árbol

Ejemplo



- La altura del árbol es 4

Bosques

- Un conjunto de árboles es llamado bosque
- Si se elimina la raíz de un árbol, se convierte en un bosque
- Si se aumenta un nodo a un bosque, se convierte en árbol

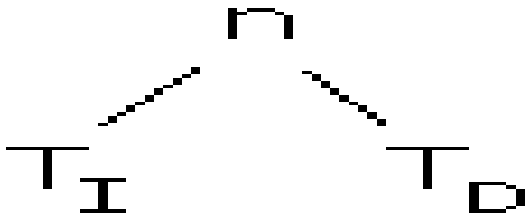
Árboles Binarios de Búsqueda (ABB)

Definiciones

- Son el caso particular mas simple
- Definición:
- Un conjunto T de elementos es un árbol binario de búsqueda si:
  - T es vació
  - T esta particionado en 3 conjuntos disjuntos

Definiciones

- Esos tres conjuntos son:
- Un solo elemento llamado la raíz
- 2 conjuntos que son ABB, llamados subárbol izquierdo y subárbol derecho

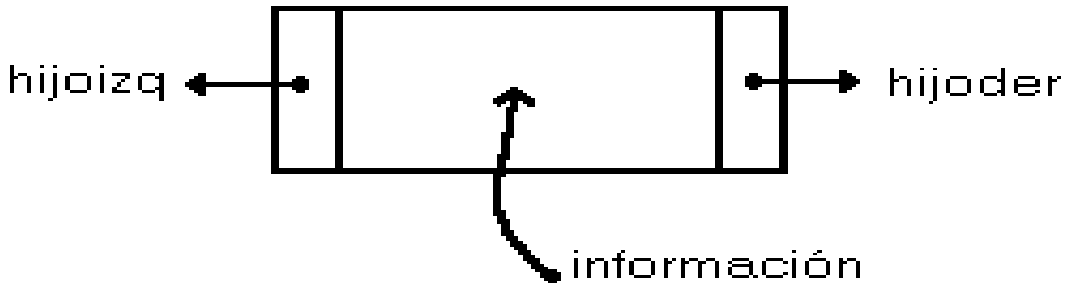


Definiciones

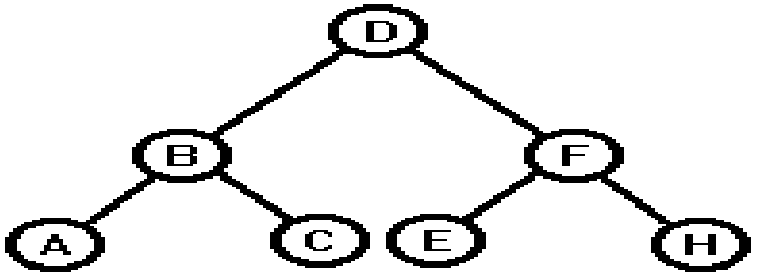
- Donde *n* es un nodo y *TD* y *TI* son árboles binarios, además de cumplir con las siguientes propiedades:
  - La llave de búsqueda (el valor) de *n* es MAYOR que todas las llaves de búsqueda de *TI*
  - La llave de búsqueda (el valor) de *n* es MENOR que todas las llaves de búsqueda de *TN*

Representación

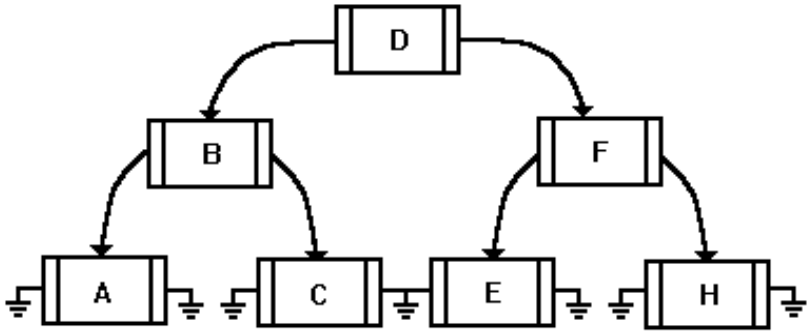
- La forma mas simple de representar un árbol binario es la ligada:



Ejemplo



- Representación ligada



Operaciones con un ABB

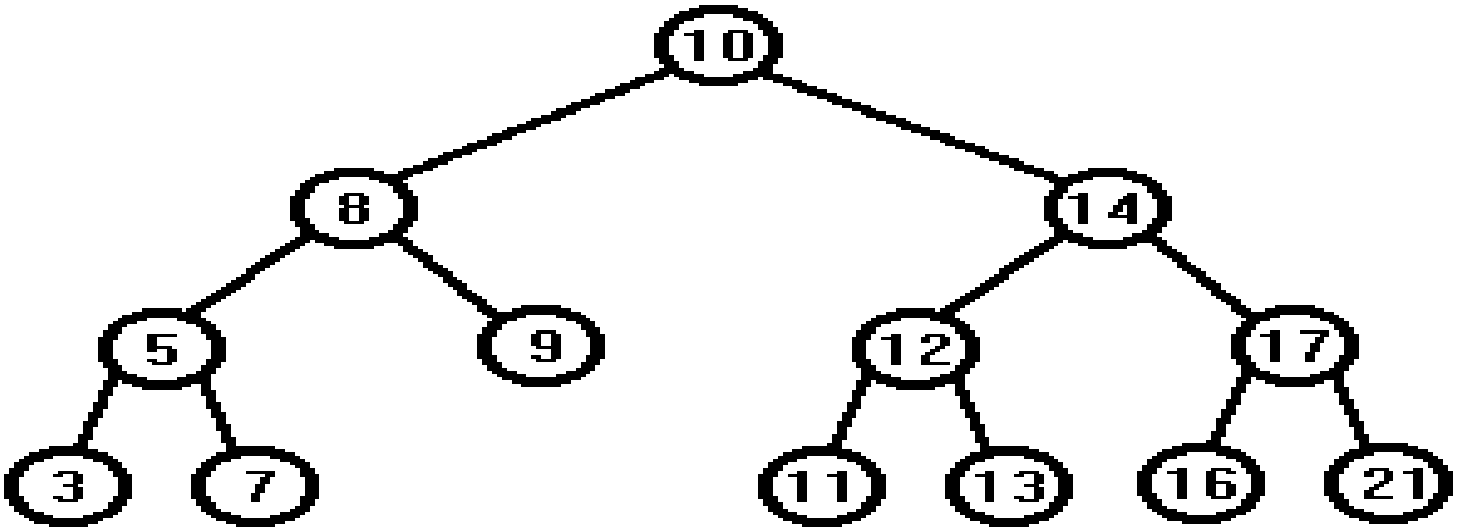
- Inserción
- Búsqueda
- Eliminación
- Recorridos

Inserción

- Insertar el elemento X
- Si el árbol esta vació, X será la raíz del árbol
- Si no esta vació se compara el valor de X con el del nodo padre:
  - Si es menor se inserta como un hijo izquierdo
  - Si es mayor se inserta como hijo derecho

Inserción, Ejemplo

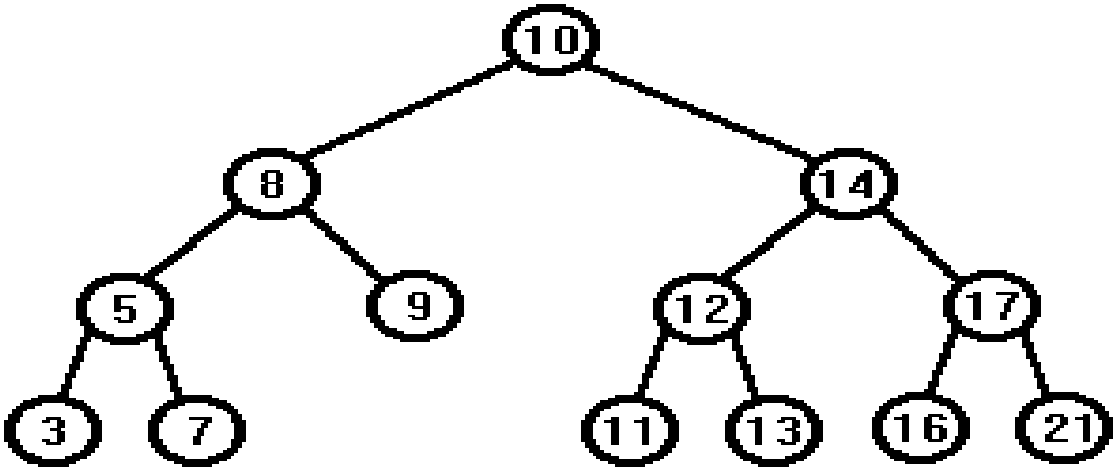
- Crear un ABB insertando los siguientes elementos: 10, 8, 14, 12, 9, 17, 5, 7, 11, 16, 13, 3, 21



Búsqueda

- Para buscar al elemento X:
  - Si X es la llave de la raíz de un ABB se ha terminado.
  - Si X es menor que el padre, se busca a la izquierda
  - Si X es mayor que el padre, se busca a la derecha

Búsqueda, Ejemplo



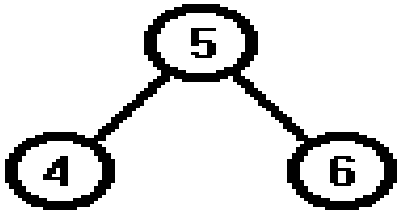
- Si se busca el elemento "7":
- Se compara 7 con 10 y se desciende por la izquierda
- Se compara 7 con 8 y se desciende por la izquierda
- Se compara 7 con 5 y se desciende por la derecha
- Se encuentra la llave deseada

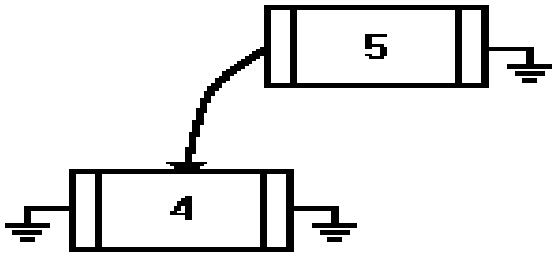
Eliminación

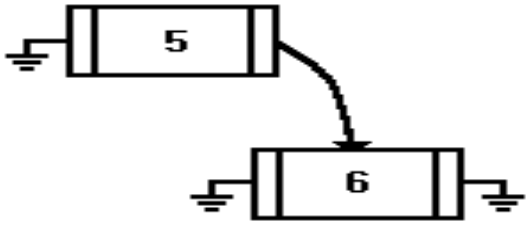
- Eliminación de un elemento X
- Si el elemento se encuentra, existen tres posibilidades:
  - El elemento es una hoja
  - Tiene un hijo único
  - Tiene dos hijos

Eliminación de una hoja

- Caso mas sencillo, en donde lo único que hay que hacer es que la liga que lo referencia debe ser *nil*

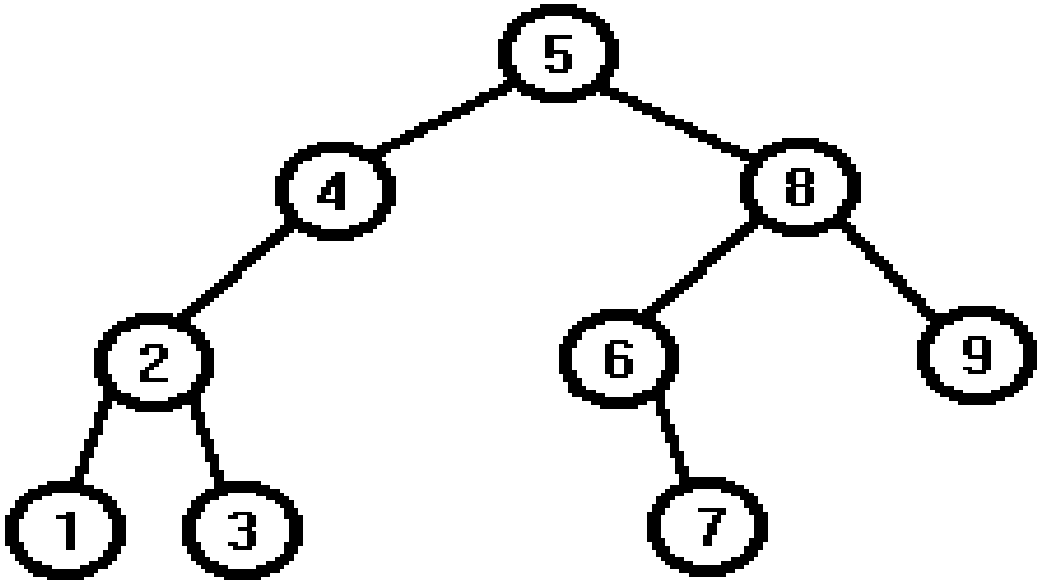






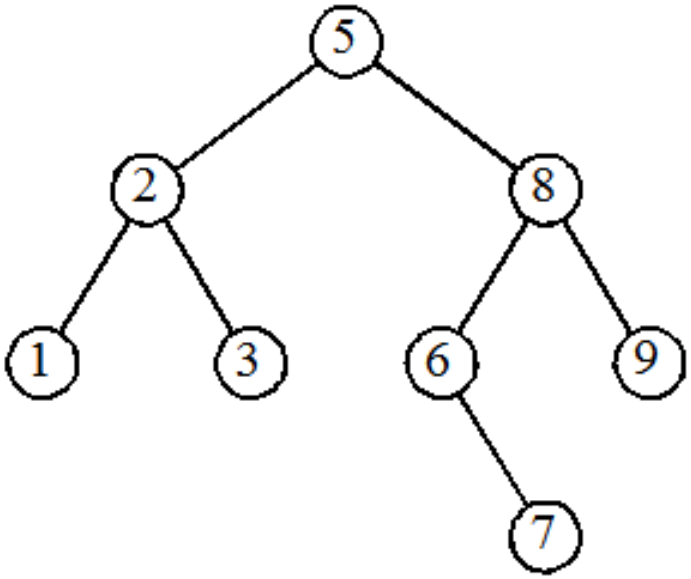
Con un hijo único

- Para eliminar un nodo, su padre deberá referenciar a su hijo



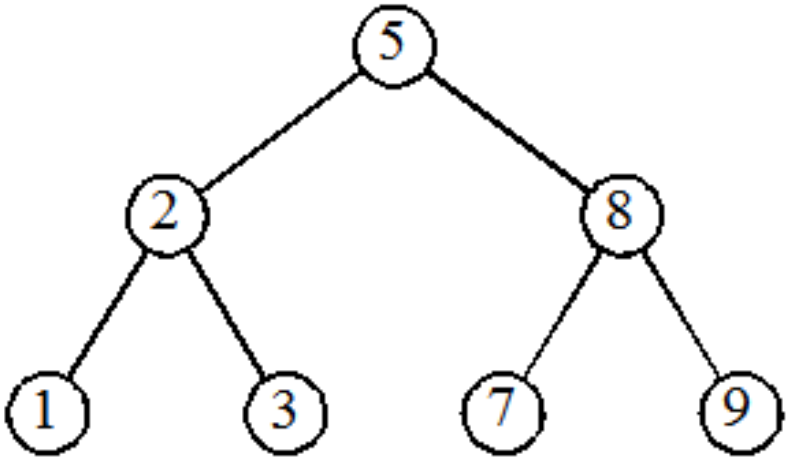
Con un hijo único

- Al eliminar la llave 4, se debe cambiar el subárbol izquierdo de 5 por el que contiene a 2 como raíz



Con un hijo único

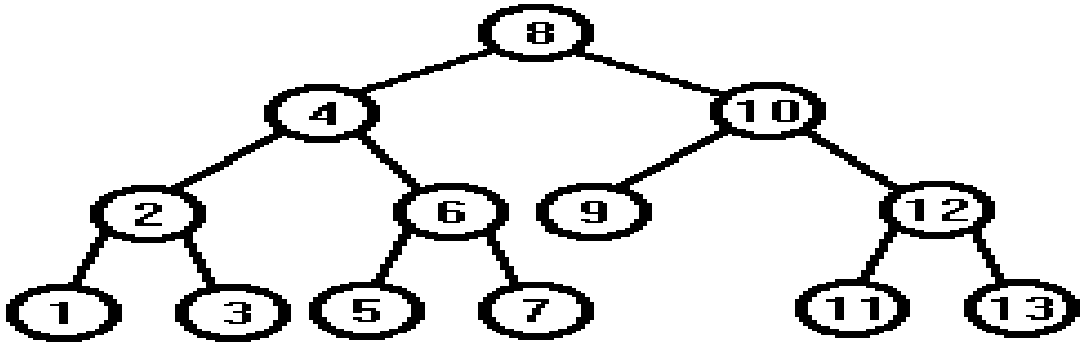
- Es el mismo procedimiento para eliminar un nodo con un único hijo izquierdo
- Al eliminar el nodo cuya información es 6, el subárbol izquierdo de 8 referenciará al subárbol derecho del 6 (nodo 7)



Con dos hijos

- Existen dos opciones:
  - Sustituir el valor de la información del nodo a eliminar por el nodo que contiene la menor llave del subárbol derecho (Menor de los mayores)
  - Sustituir el valor de la información del nodo a eliminar por el nodo que contiene la mayor llave del subárbol izquierdo (Mayor de los menores)

Eliminación con dos hijos

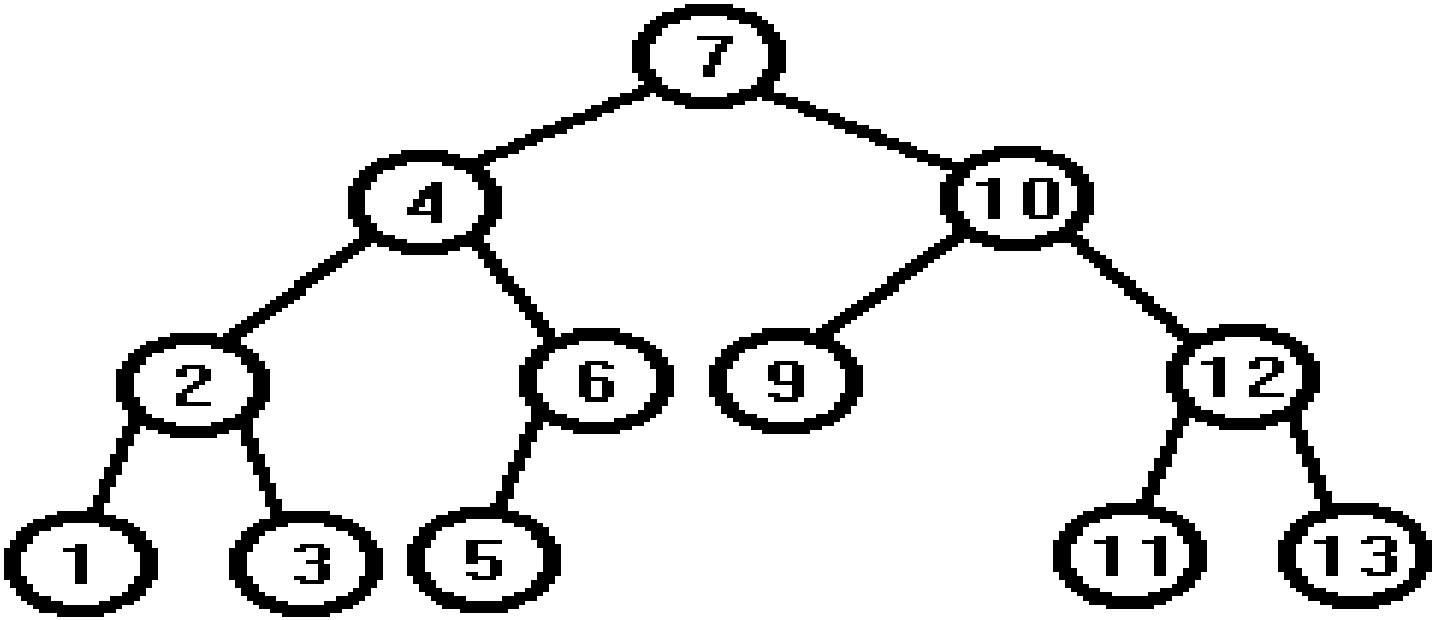


- Si se desea eliminar la raíz, se substituye el valor por la llave con el mayor de los menores y luego eliminar ese valor

- O sustituir el valor por la llave con el menor de los mayores y luego eliminar ese valor

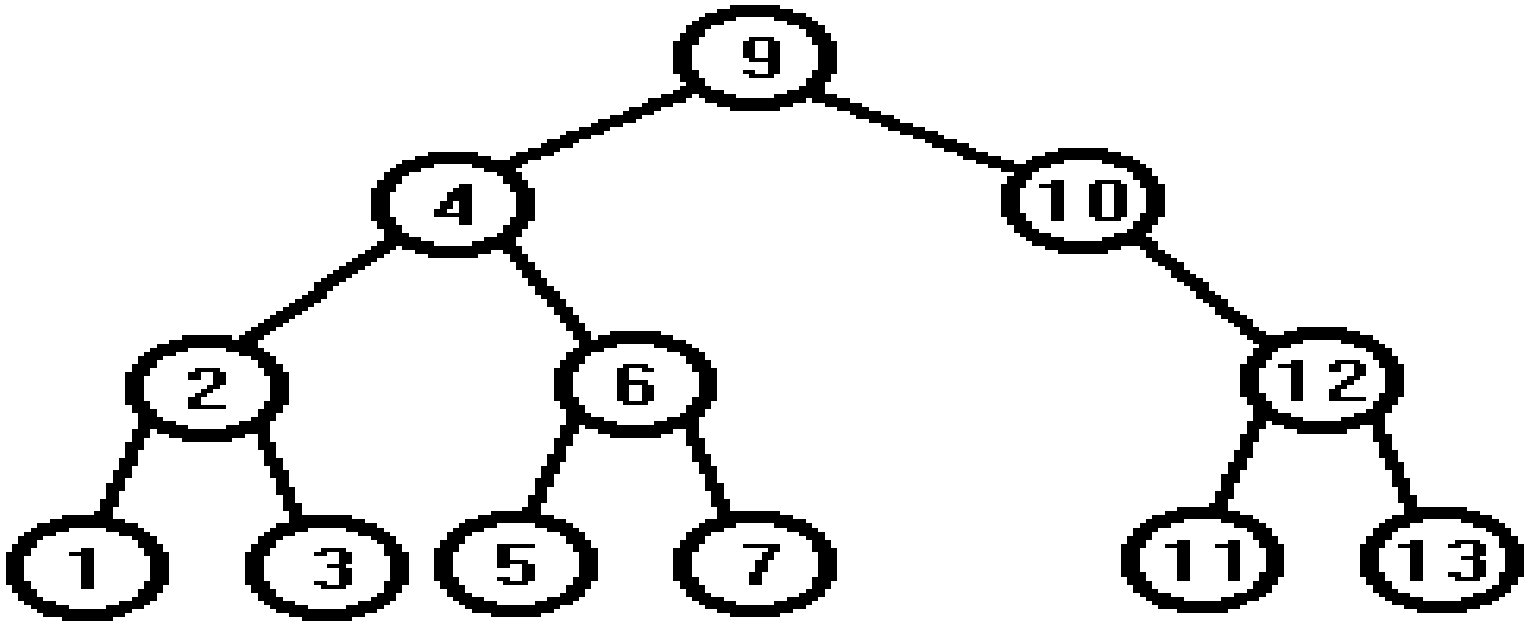
Eliminación con dos hijos

- Utilizando el mayor de los menores



Eliminación con dos hijos

- Utilizando el menor de los mayores



Recorridos en un ABB

- Existen tres maneras de recorrer un árbol binario de búsqueda:
  - Preorden o previo
  - Inorden o simétrico
  - Postorden o posterior

- La diferencia entre ellas es el orden en que se visitan los nodos

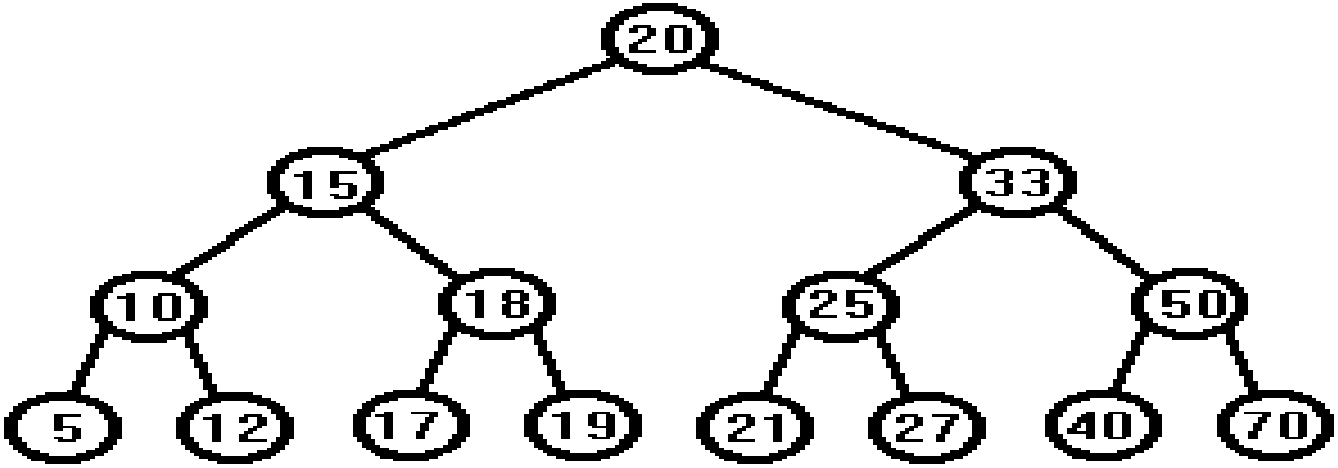
Recorridos

- Pre orden o previo
  - Visitar la raíz
  - Recorrer recursivamente el subárbol izquierdo
  - Recorrer recursivamente el subárbol derecho

- Inorden o simetrico
  - Recorrer recursivamente el subárbol izquierdo
  - Visitar la raíz
  - Recorrer recursivamente el subárbol derecho

- Post orden o posterior
  - Recorrer recursivamente el subárbol izquierdo
  - Recorrer recursivamente el subárbol derecho
  - Visitar la raíz

Recorridos, Ejemplo



- **preorden:**
- 20, 15 10, 5, 12, 18, 17, 19, 33, 25, 21, 27, 50, 40, 70
- **inorden:**
- 5, 10, 12, 15, 17, 18, 19, 20, 21, 25, 27, 33, 40, 50, 70
- **postorden:**
- 5, 12, 10, 17, 19, 18, 15, 21, 27, 25, 40, 70, 50, 33, 20

Desempeño

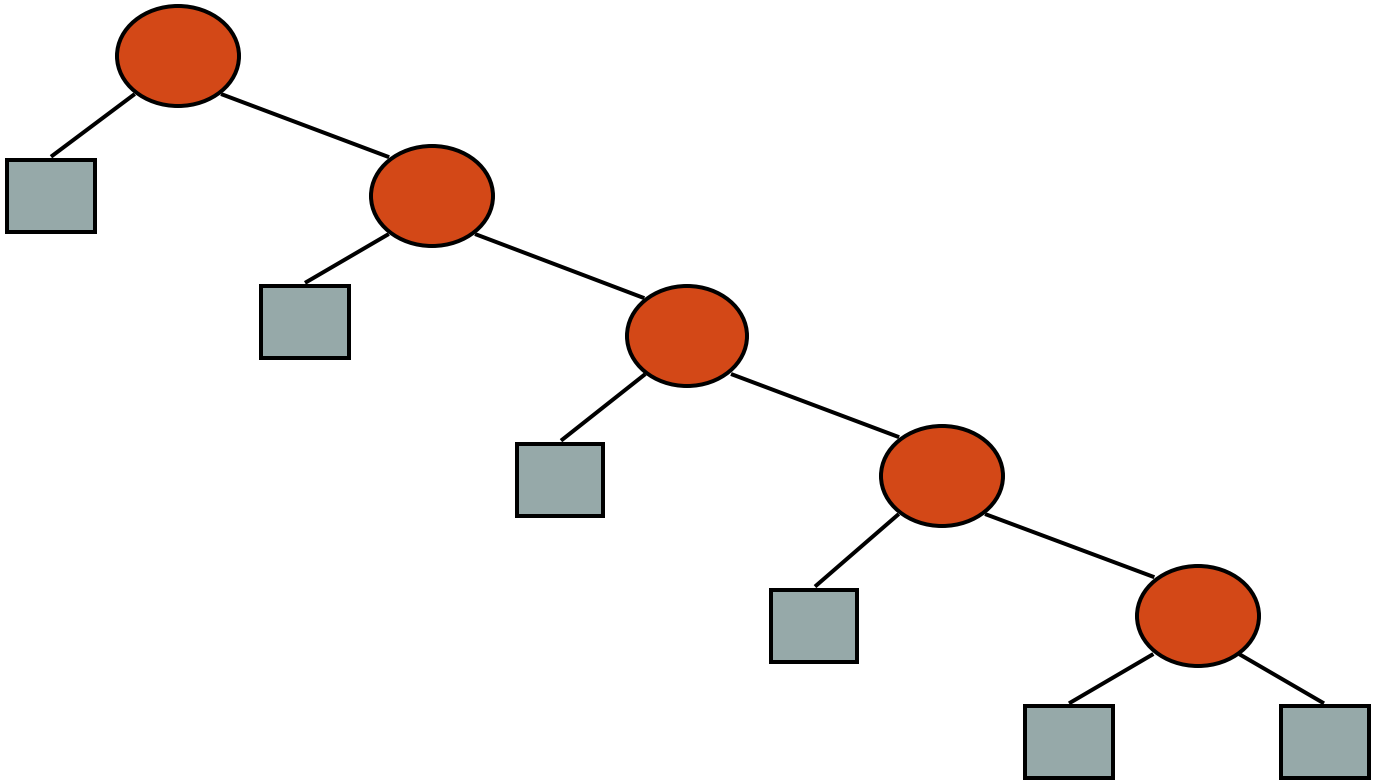
- El peor caso en un ABB es cuando se insertan elementos que se encuentran ordenados (ya sea de menor a mayor o viceversa)

- Por esta razón su uso no es recomendable cuando se maneja información extraída por ejemplo de un diccionario

Desempeño

- Si se insertan n elementos en un ABB de altura h:
  - El espacio utilizado en el peor de los casos es $\mathbf{O}(n)$
  - Los métodos de inserción, búsqueda y eliminación son de tiempo $\mathbf{O}(h)$
  - La altura en el peor de los casos es $\mathbf{O}(n)$ y en el mejor de los casos $\mathbf{O}(\log n)$

Peor de los casos



Montículo Binario

Definición

- Un Montículo Binario es un Árbol Binario Completo
- Todos los niveles están llenos a excepción del nivel de hojas
- Una característica es que los elementos izquierdo y derecho deben ser mayores que su padre
- Debido a la representación de árbol completo, un Montículo Binario se representa como un arreglo

Representación

- Aquí la distribución de padres e hijos está dada por la posición de cada elemento en el arreglo, de esta forma:
- Un nodo padre en (i):
  - Tiene a un hijo izquierdo en (2*i + 1) si 2*i+1 < N
  - Tiene a un hijo derecho en (2*i + 2) si 2*i+2 < N
  - En donde:
    - N es el número de elementos
- Un nodo hijo tiene a su padre en (i-1)/2 si i es distinto a 0

Llenado

- La representación más común de un Montículo Binario es a través de un arreglo
- Aquí, cada que se inserta un elemento, es necesario compararlo con aquel, que de acuerdo a su posición sería su padre
- Si el padre es menor, es necesario intercambiar los elementos y volver a comparar con quien ahora sería su nuevo padre
- Así sucesivamente hasta que el padre ya no sea menor o el elemento quede en la raíz

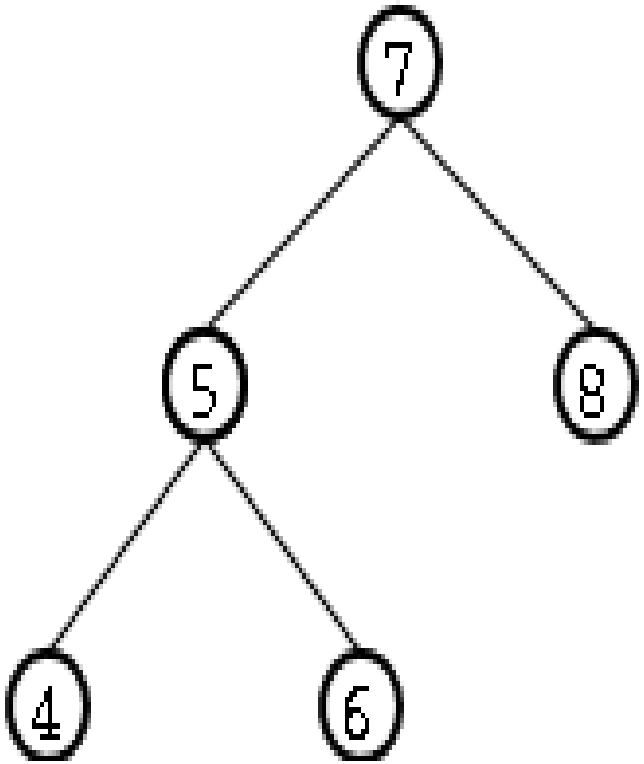
Árboles Adelson-Velskii y Landis (AVL)

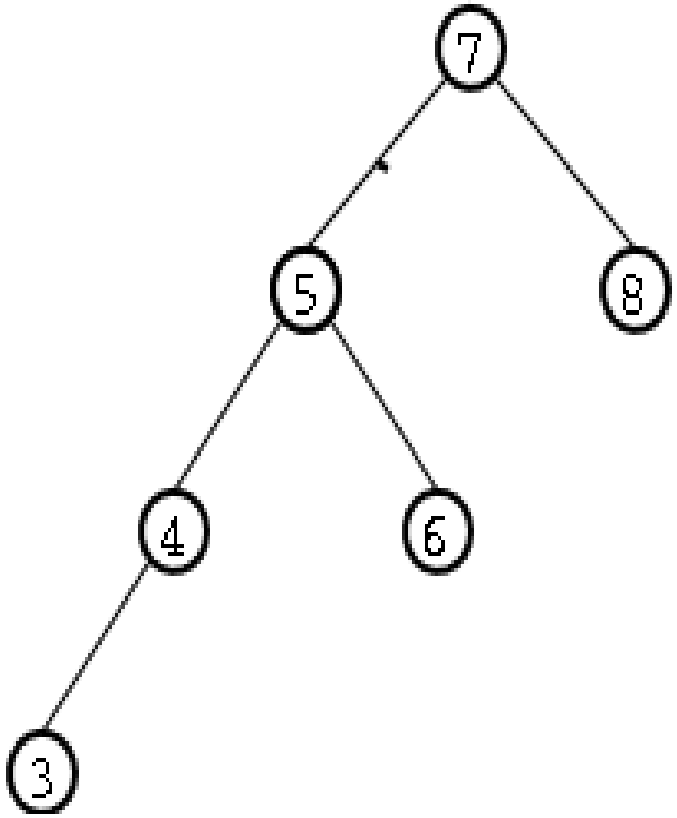
Definiciones

- Es un árbol binario de búsqueda equilibrado

- Un árbol AVL es un ABB en donde las alturas de los hijos izquierdo y derecho solo pueden diferir a lo máximo en uno

Definiciones





Operaciones con árboles AVL

- Inserción
- Búsqueda
- Eliminación
- Recorrido
- Balanceo

Operaciones

- Las operaciones de búsqueda y recorridos son las mismas que para un ABB

- Las operaciones de inserción y borrado de un elemento son las mismas, con la excepción de que después de ejecutarlas requieren una operación de balanceo

Balanceo

- Tras una inserción, los nodos pueden ver alterado su equilibrio

- Existen cuatro casos que se deben considerar:
  1. Una inserción en el subárbol izquierdo del hijo izquierdo de X.
  2. Una inserción en el subárbol derecho del hijo izquierdo de X.
  3. Una inserción en el subárbol izquierdo del hijo derecho de X.
  4. Una inserción en el subárbol derecho del hijo derecho de X.

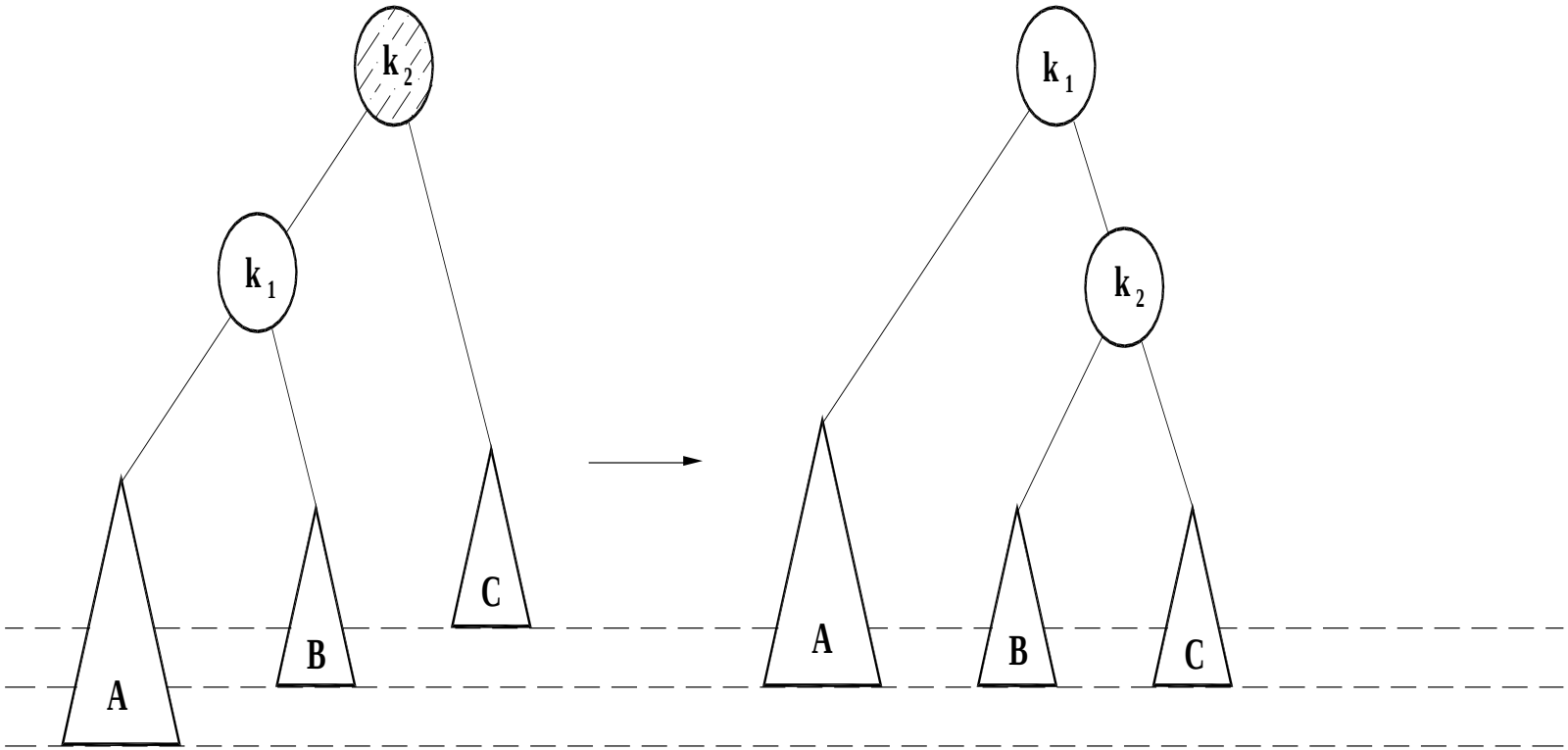
Balanceo

- En el primer y cuarto caso, en donde la inserción se produce en los márgenes, el balance se recupera con una rotación simple.

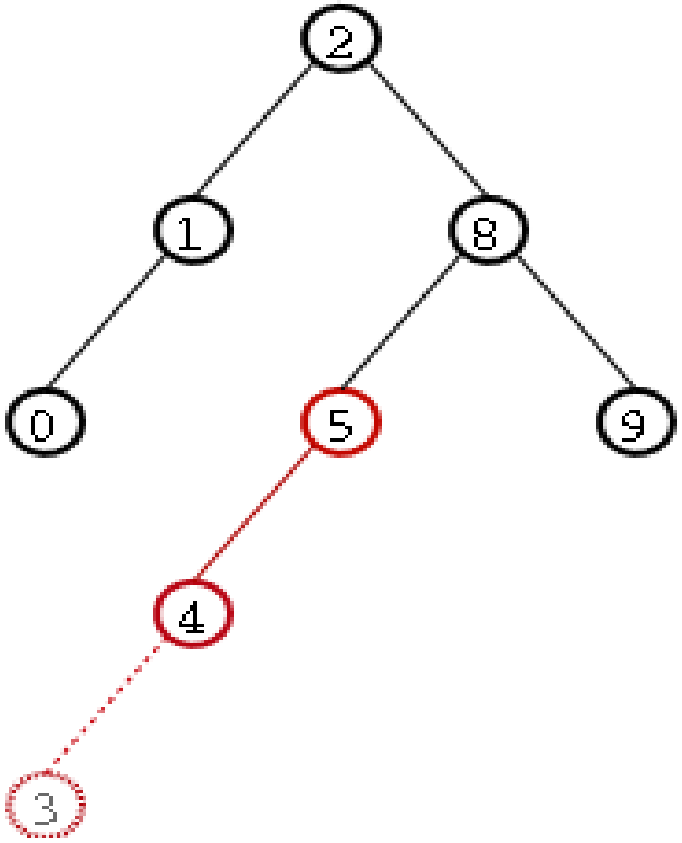
Balanceo simple

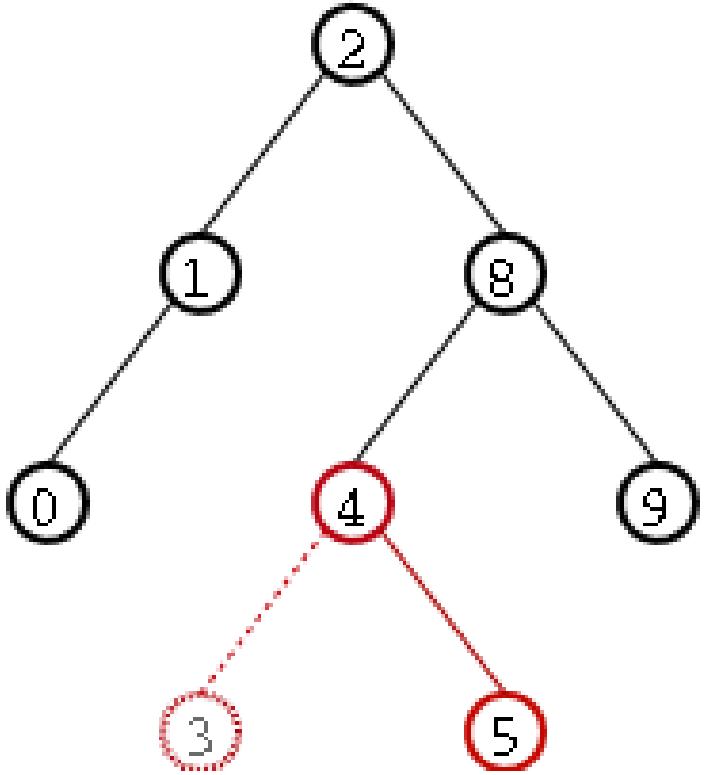
- El subárbol *A* ha crecido en un nivel, provocando que sea dos niveles más profundo que *C*
- Para re equilibrar de manera correcta el árbol, hay que subir *A* un nivel y bajar *C*
- El resultado es que k_1 será la nueva raíz
- La propiedad de los árboles binarios de búsqueda indica que en el árbol inicial, $k_2 > k_1$, así que en el nuevo árbol k_2 se convierte en el hijo derecho de k_1

Rotación simple a la derecha

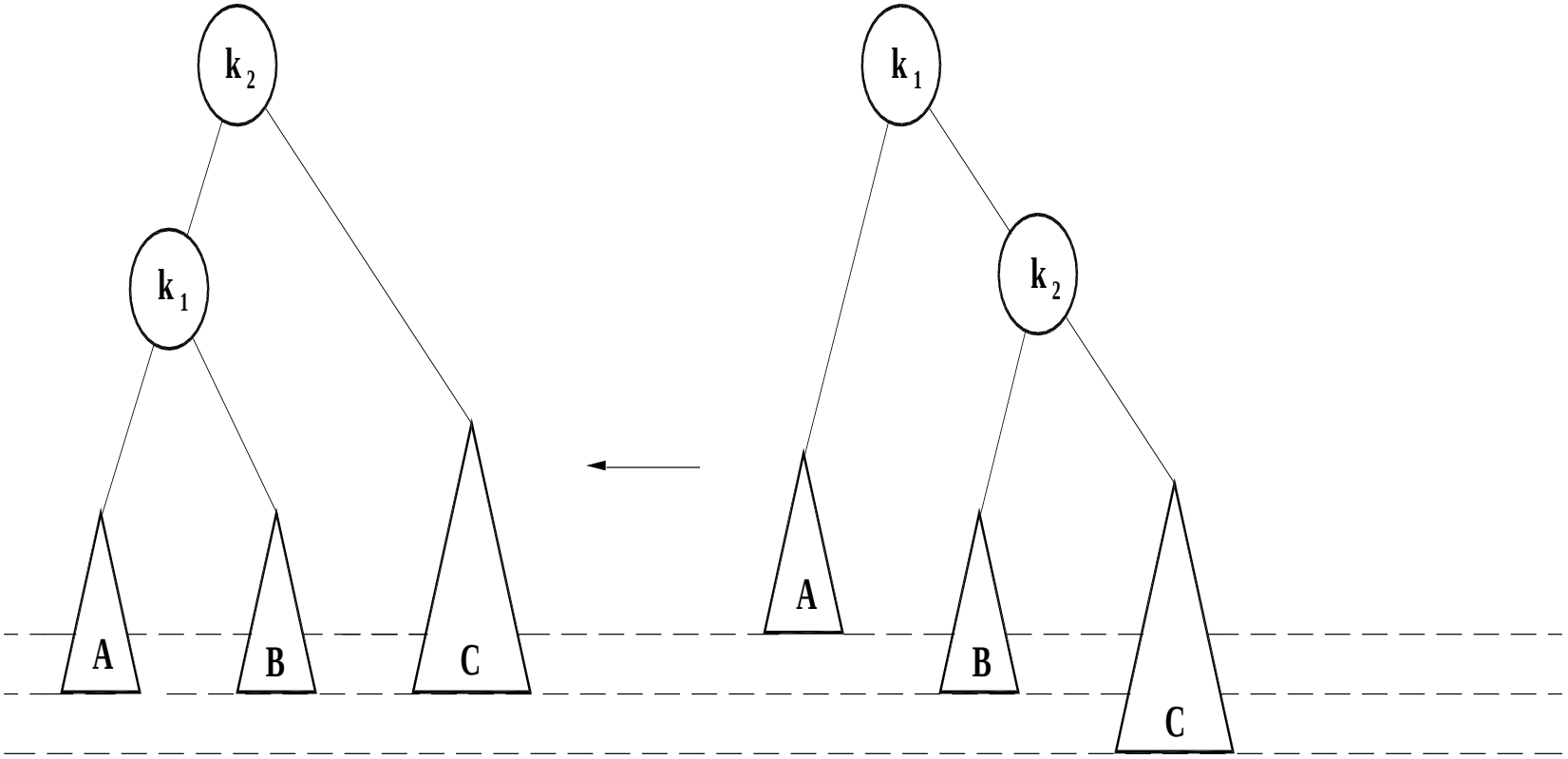


Ejemplo

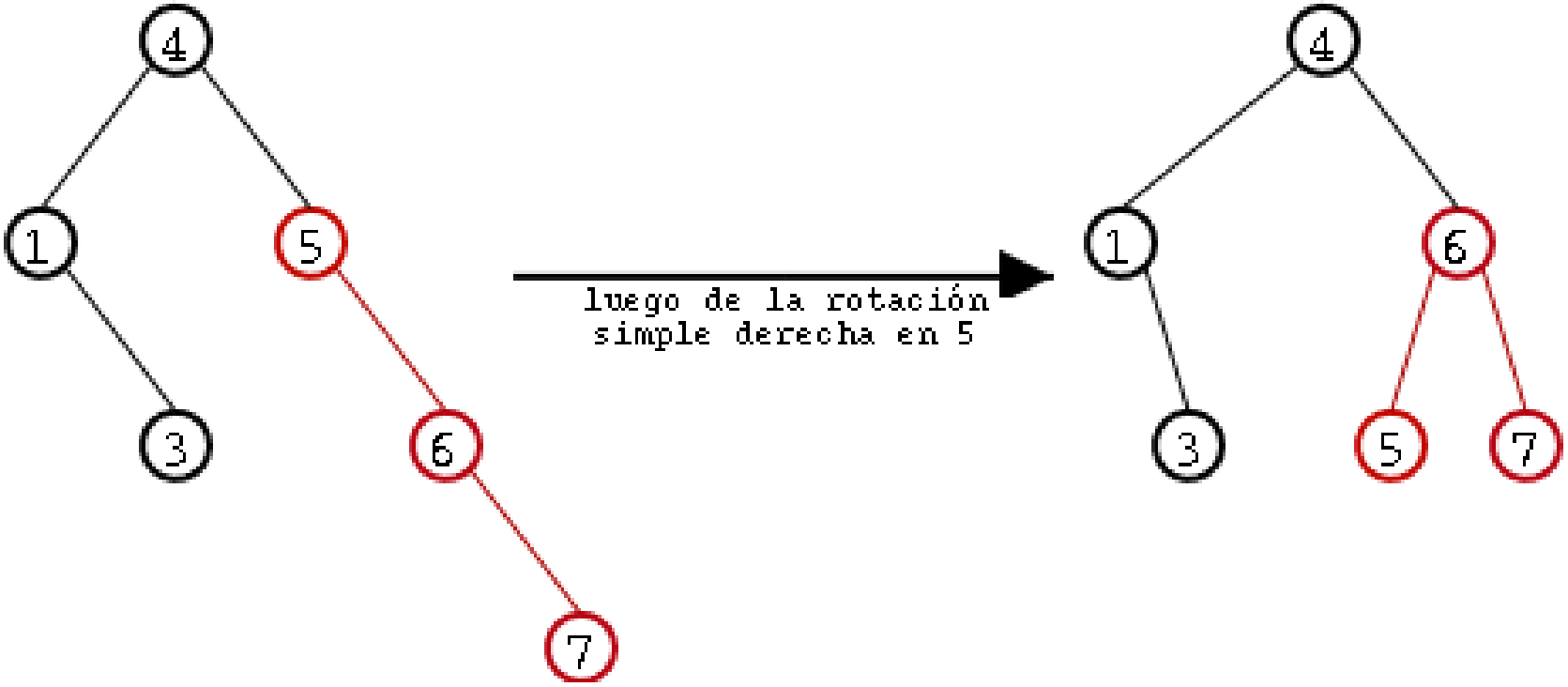




Rotación simple a la izquierda



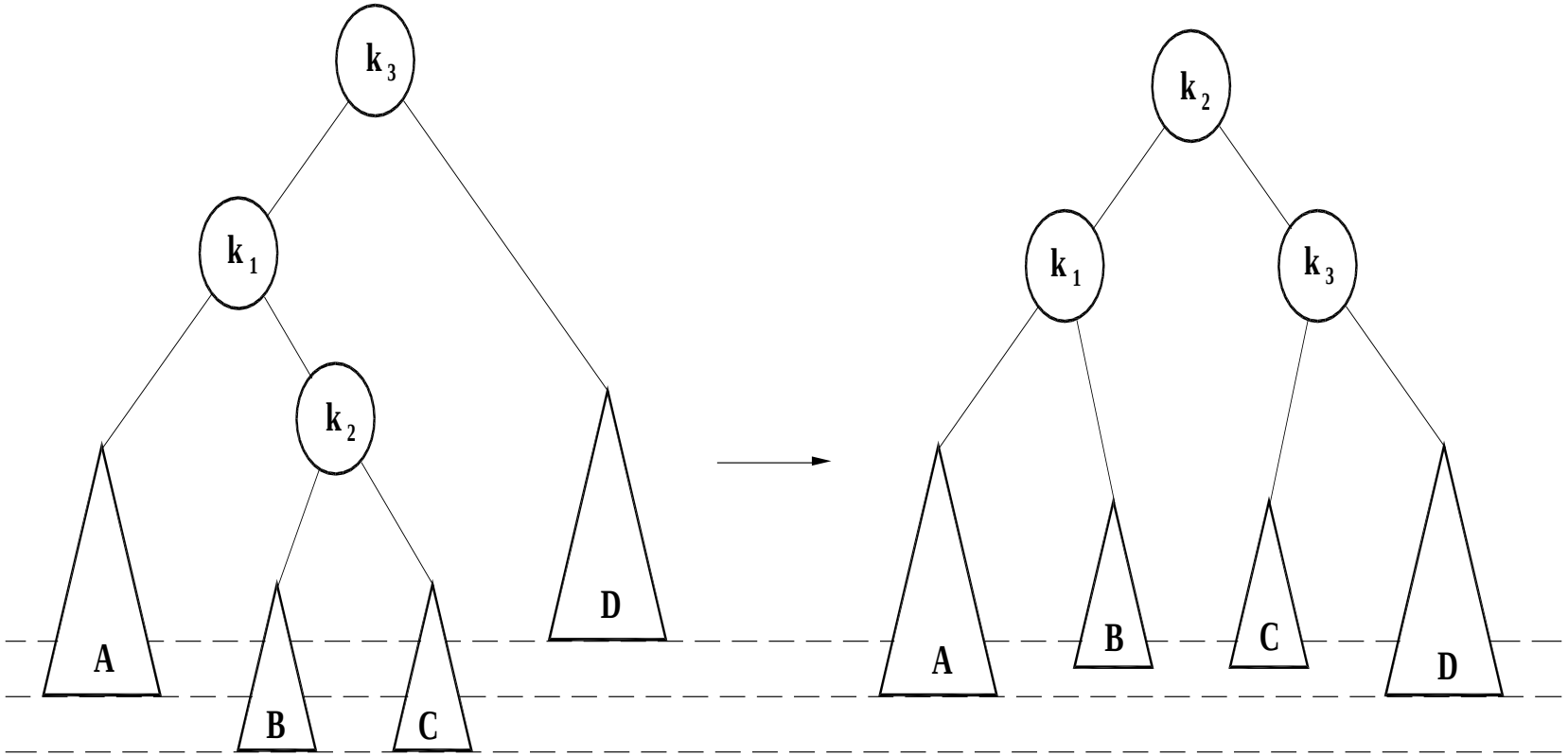
Ejemplo



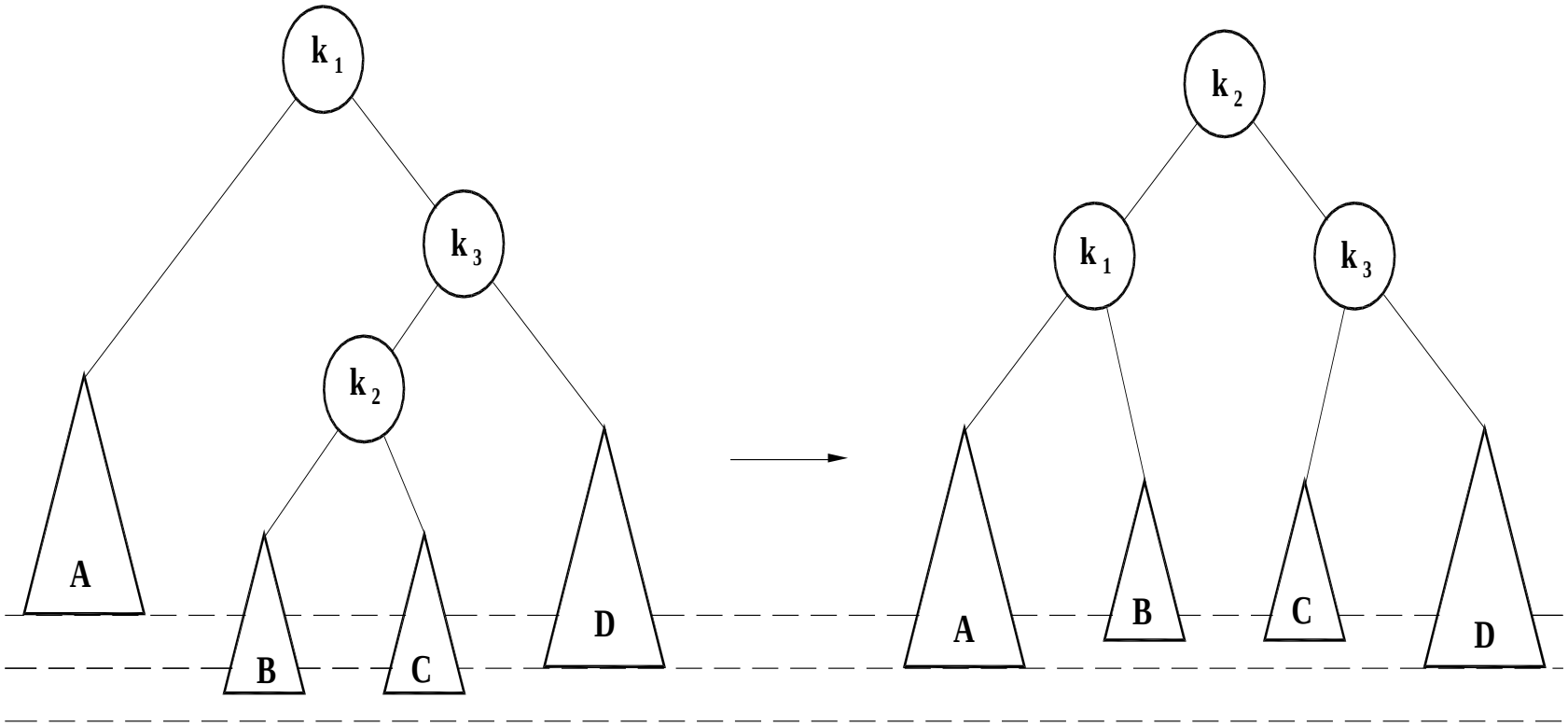
Rotaciones dobles

- La rotación simple tiene el problema de que no resuelve el problema en los casos 2 y 3
- Para resolver el problema se deben efectuar dos rotaciones sencillas:
  - Una rotación simple entre el hijo de *X* y su nieto, seguida de una rotación simple entre *X* y su nuevo hijo

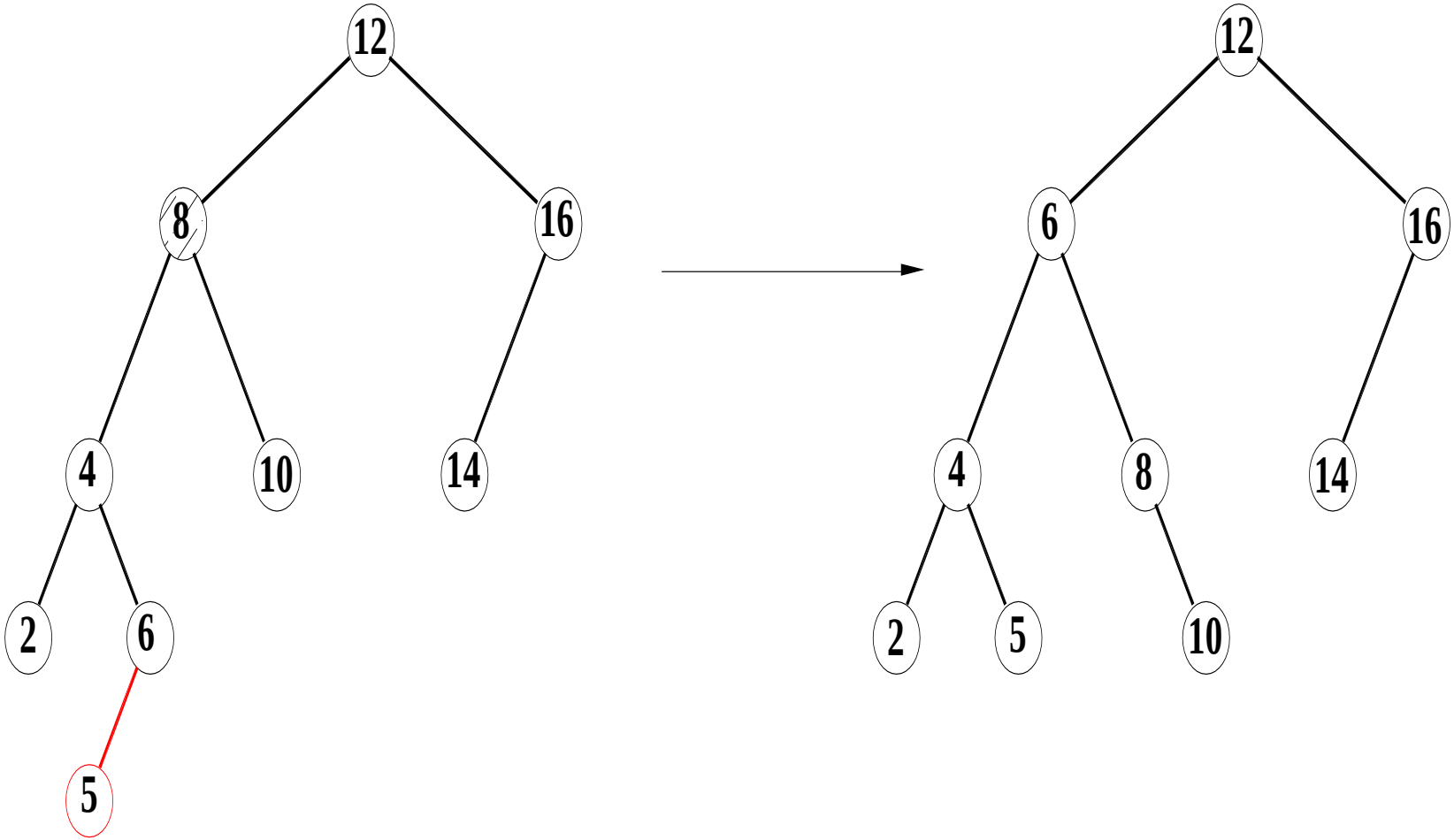
Rotación doble



Rotación doble



Ejemplo



Desempeño

- Al ser los árboles AVL balanceados, no importa en que orden se ingresen las llaves, la altura siempre será de O(log *n*):
  - La altura mínima de un árbol AVL de *n* llaves es: piso $|\log n|$
  - La altura máxima : techo $|\log n| + 1$