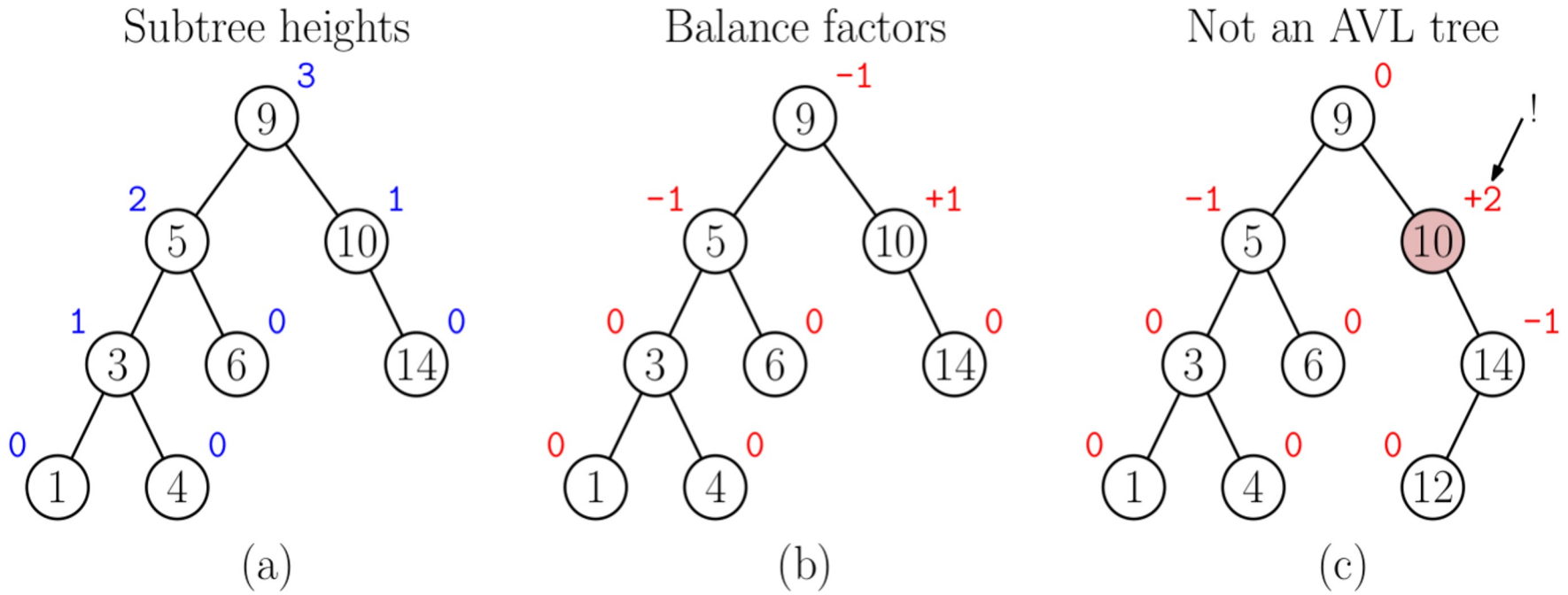
AVL Examples



Implementación de AVL

```
Módulo ArbolAVL implementa conjunto = {
   var raíz: NodoAVL
}

Struct NodoAVL x {
    x.izq,
    x.der,
    x.dato
    x.altura
}
```

`new NodoAvl(x)`: crea un nodo AVL con x como dato

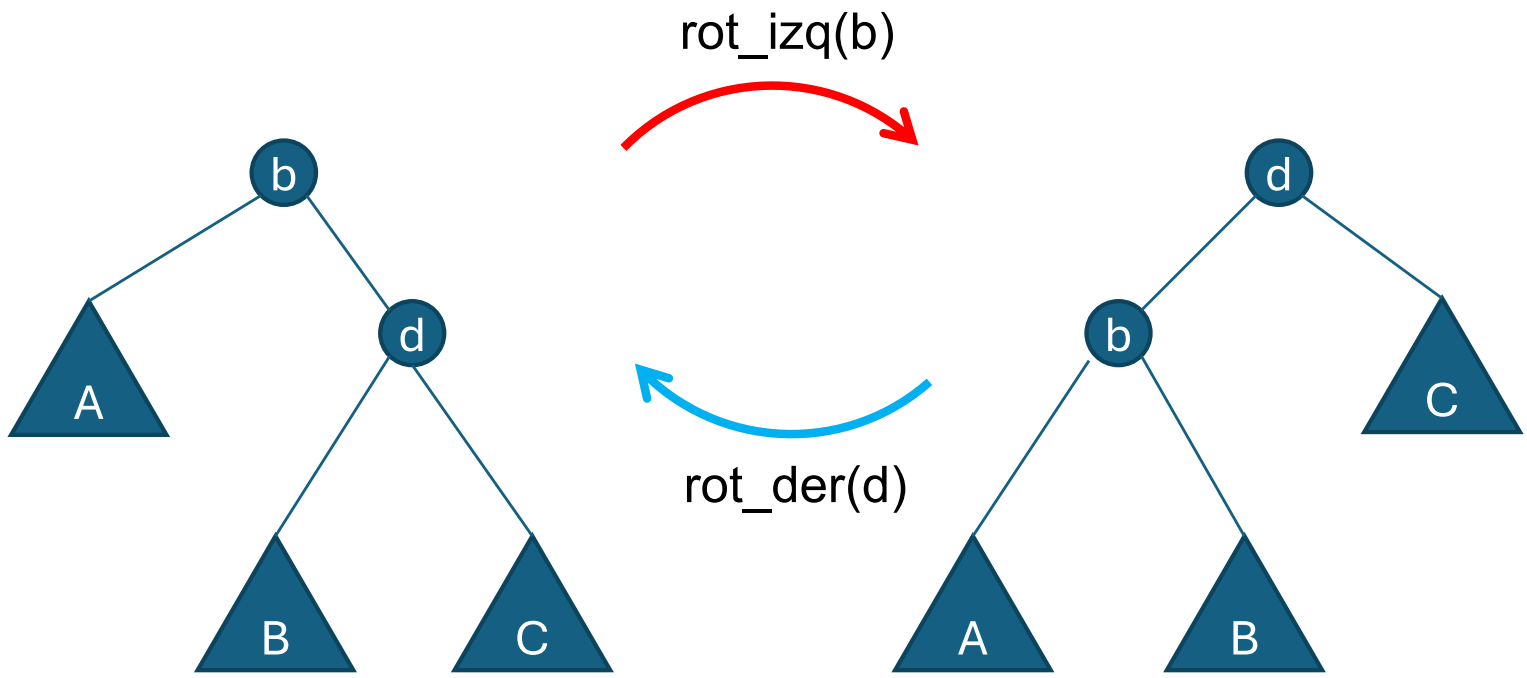
Funciones útiles

```
impl aux altura(p:NodoAVL): int
   if p == null
      return -1
   else
      return p.altura


impl aux actualizarAltura(p:NodoAVL)
    p. altura = 1 + max(altura(p.izq), altura(p.der))


impl aux factorBalanceo(p:NodoAVL): int
   return (altura(p.der) - altura(p.izq))
```

inserción en AVL

1. Insertar el nuevo nodo como en un ABB “clásico”
   - el nuevo nodo es una hoja
2. Recalcular los factores de balanceo que cambiaron por la inserción
   - sólo en la rama en la que ocurrió la inserción (los otros factores no pueden cambiar!), de abajo hacia arriba
3. Si en la rama aparece un factor de balanceo de ±2 hay que rebalancear
   - A través de “rotaciones”

Rotaciones

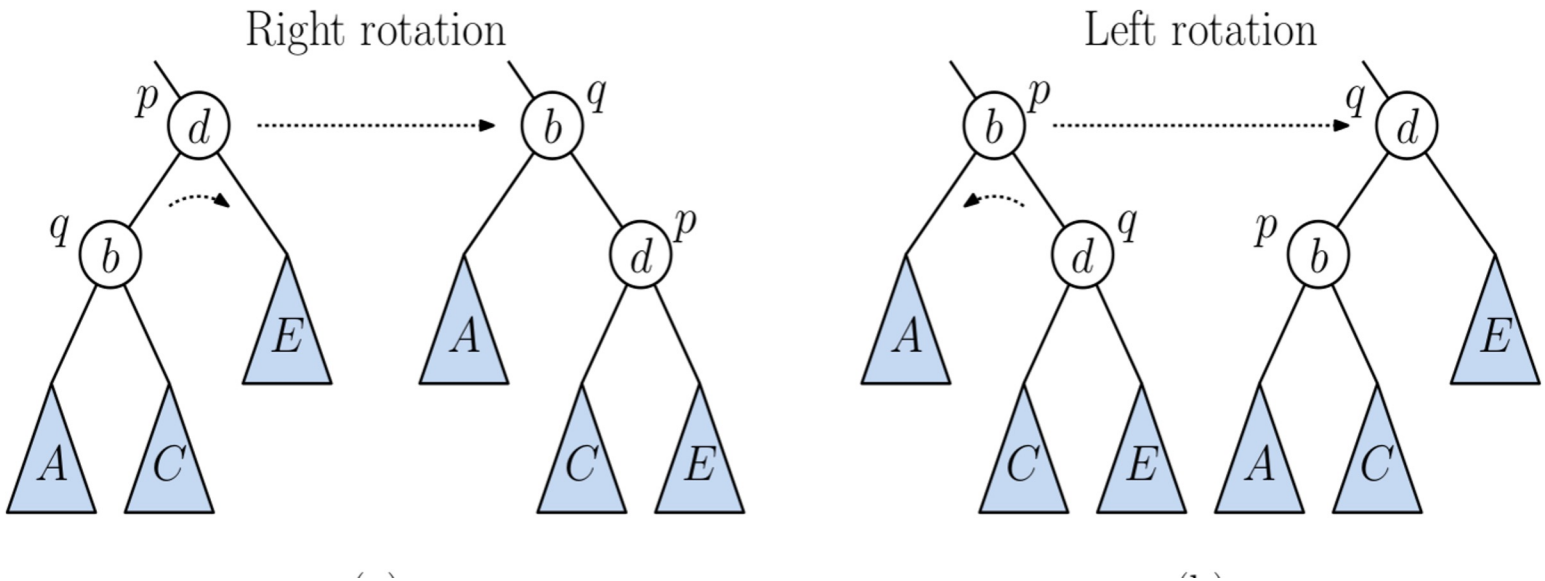


AbBdC

AbBdC

Rotaciones

Rotación: modificación local que modifica las alturas de los subárboles, pero conserva el orden

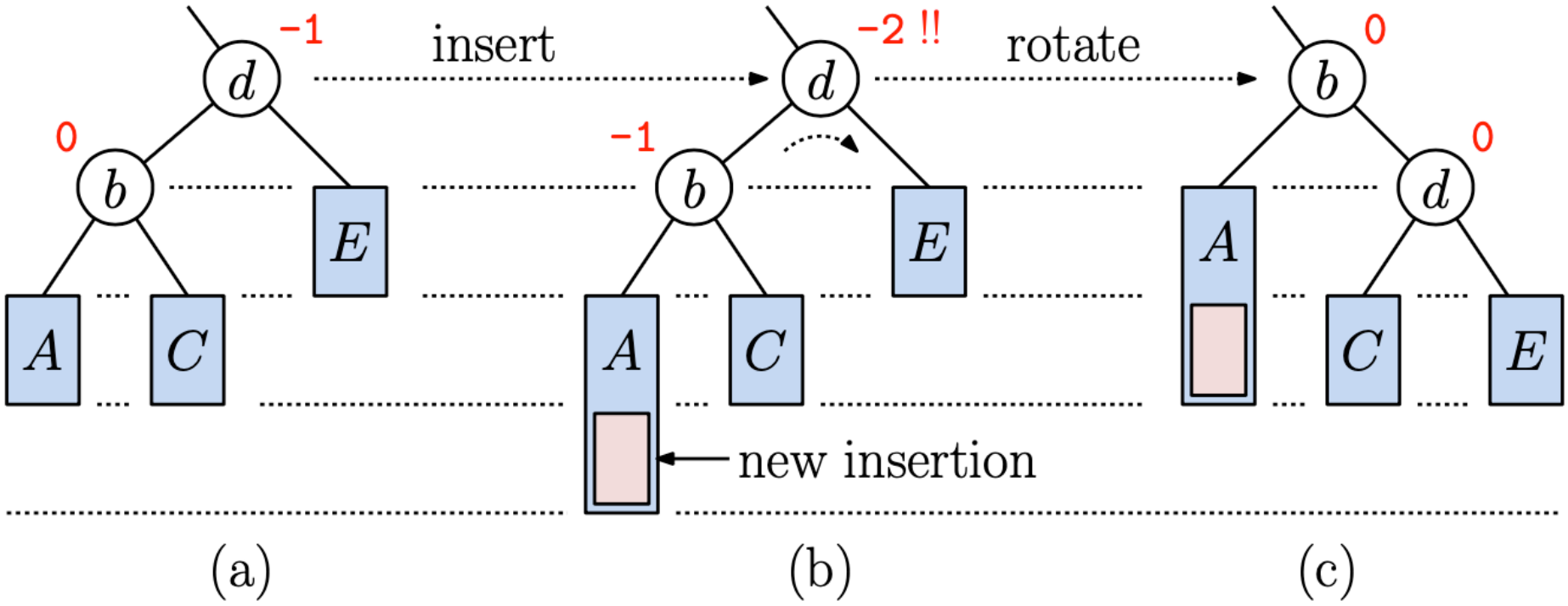


Una rotación no siempre es suficiente para rectificar un nodo que está desequilibrado (por ejemplo, el subárbol C)

Rotaciones en los AVL (Casos)

- II: inserción en el subárbol **izquierdo** de un hijo **izquierdo** (del nodo que se desbalancea)
  - Rotar Derecha
- DD: inserción en el subárbol **derecho** de un hijo **derecho** (del nodo que se desbalancea)
  - Rotar Izquierda

Insertar con rotación simple (II)



Notar que ademas de rotar temenos que llamar a actualizarAltura para b y d

Rotación doble

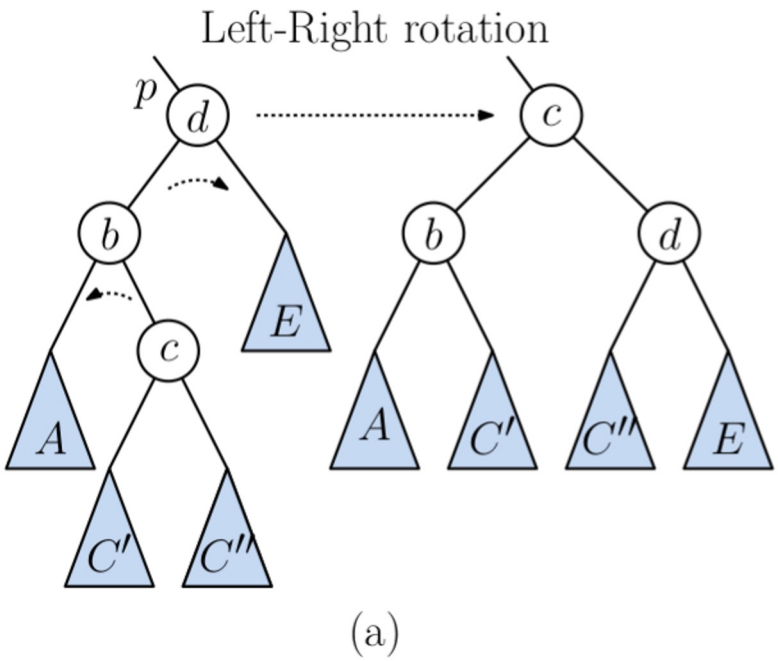
Una sola rotación no siempre es suficiente para rectificar un nodo que está desequilibrado. Por ejemplo: la rotación simple no altera la altura del subárbol C.

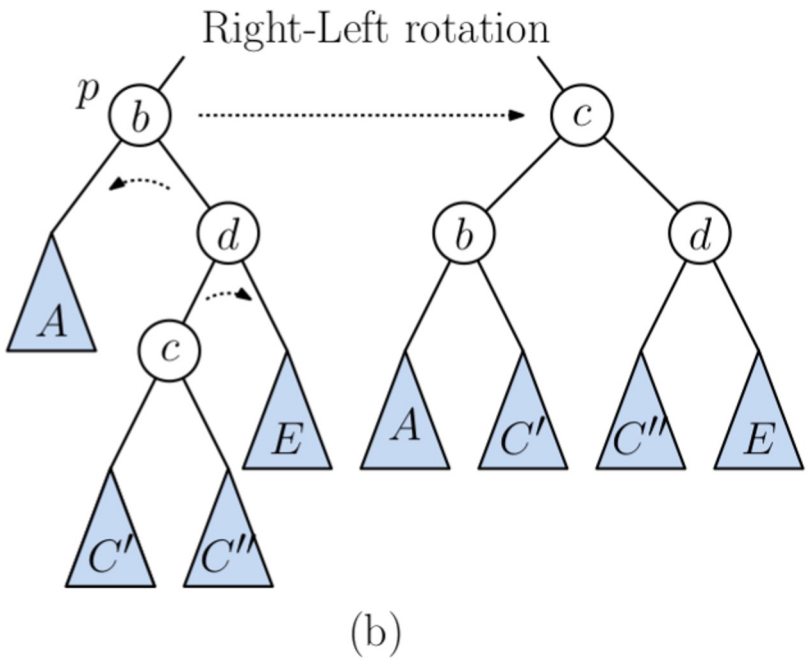
Doble rotación: combinación de dos rotaciones.

rotarIzquierdaDerecha(p): una rotación a la izquierda de p.izq seguida de una rotación a la derecha hacia p.

rotarDerechaIzquierda(p): una rotación a la derecha de p.der seguida de una rotación hacia la izquierda hacia p.

Rotación doble

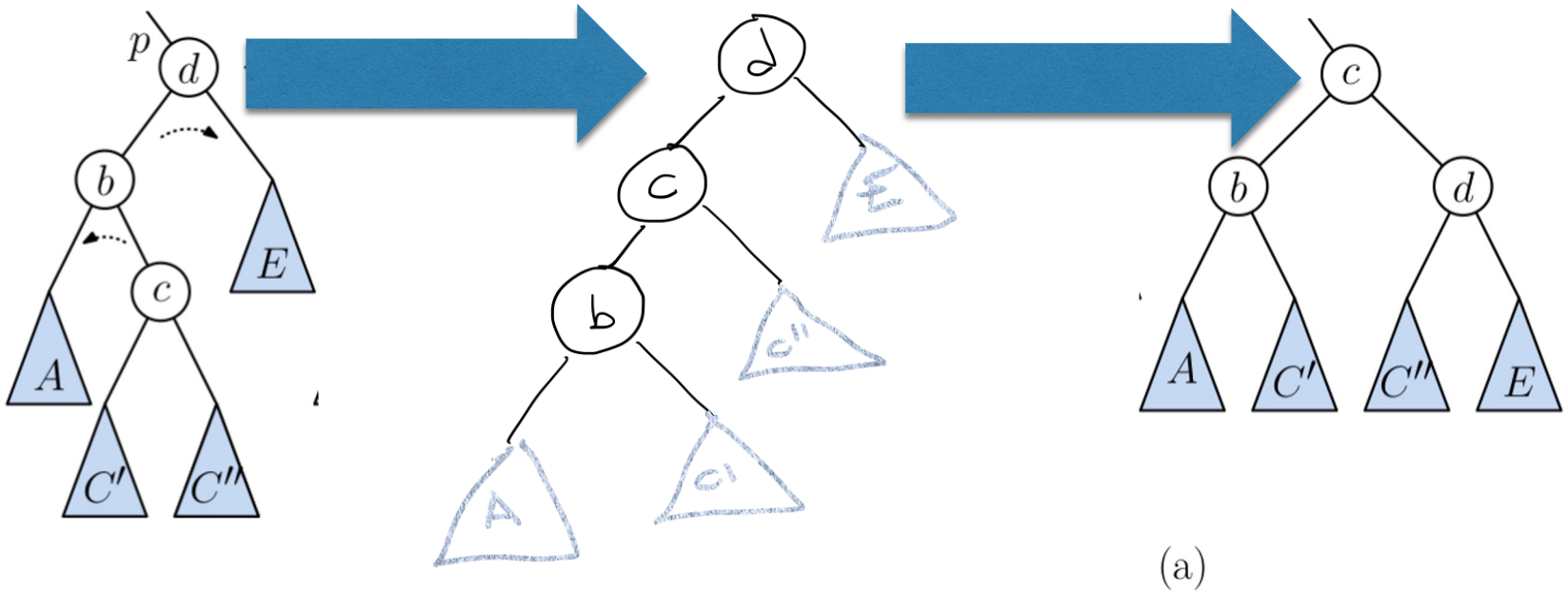




rotaciónIzquierdaDerecha(p):una rotación a la izquierda hacia la p.izq seguida de una rotación a la derecha hacia p.

rotaciónDerechaIzquierda(p): una rotación a la derecha hacia la p.der seguida de una rotación hacia la izquierda hacia p.

Rotación doble

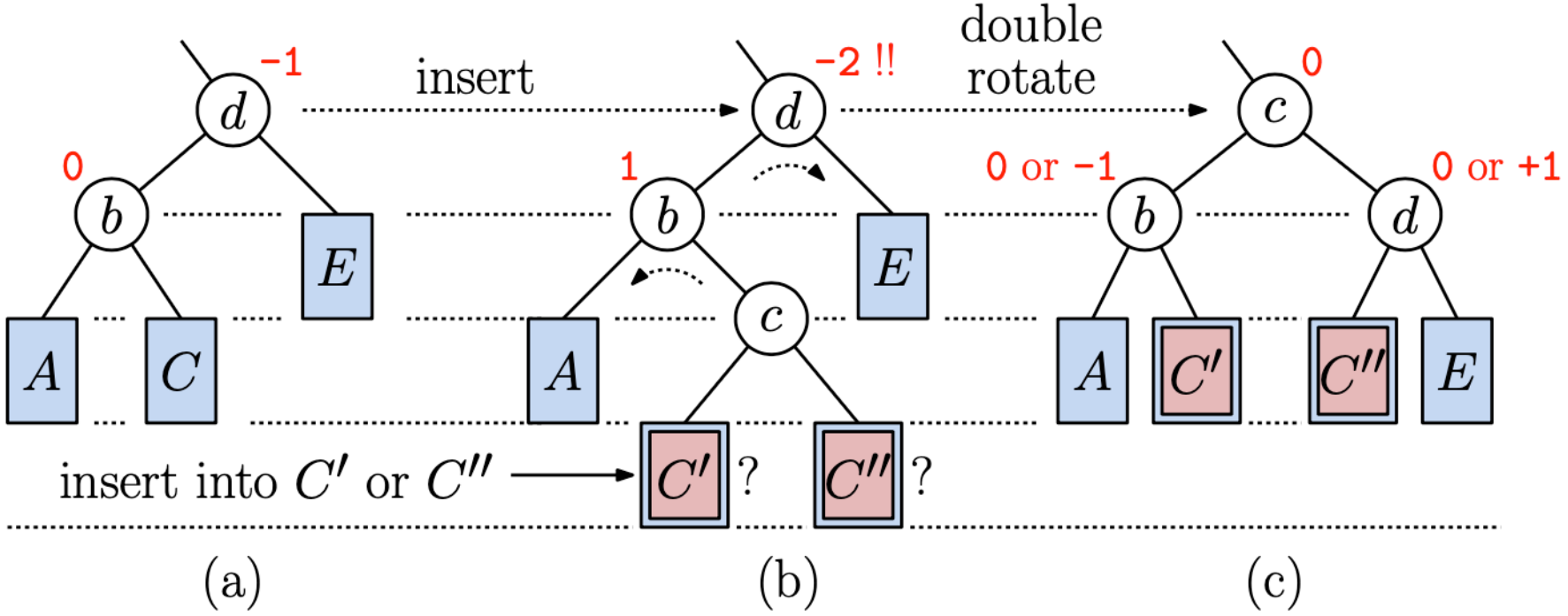


rotaciónIzquiedaDerecha(p): una rotación a la izquierda de p.izq seguida de una rotación a la derecha hacia p.

Rotaciones en los AVL (Casos)

- II: inserción en el subárbol **izquierdo** de un hijo **izquierdo** (del nodo que se desbalancea)
  - Rotar Derecha
- DD: inserción en el subárbol **derecho** de un hijo **derecho** (del nodo que se desbalancea)
  - Rotar Izquierda
- DI: inserción en el subárbol **derecho** de un hijo **izquierdo** (del nodo que se desbalancea)
  - Rotar Izquierda-Derecha
- ID: inserción en el subárbol **izquierdo** de un hijo **derecho** (del nodo que se desbalancea)
  - Rotar Derecha-Izquierda

Insertar con rotación doble (DI)



Notar que ademas de rotar temenos que llamar a actualizarAltura para b y d (y c)

Insertar doble

```
NodoAVL insertAVL(p:NodoAVL, x:int) {
  if (p == null)
     var p = new NodoAVL(x)
  else if (x < p.dato)
      p.izq = insertAVL(p.izq, x)
  else if (x > p.key)
      // x is larger - insert right
     p.der = insertAVL(p.der, x)
  else error 'Ya estaba el dato'
  return rebalance(p)
  }
```

Nota: el rebalanceo se invoca para todos los nodos de la rama desde p hasta la raíz

Rotaciones en los AVL (Como los detectamos?

- II: inserción en el subárbol **izquierdo** de un hijo **izquierdo** (del nodo que se desbalancea)
  - FB(n) <-1 y FB(n.izq)<=0
- DD: inserción en el subárbol **derecho** de un hijo **derecho** (del nodo que se desbalancea)
  - FB(n) >1 y FB(n.der)>=0
- DI: inserción en el subárbol **derecho** de un hijo **izquierdo** (del nodo que se desbalancea)
  - FB(n) <-1 y FB(n.izq)>0
- ID: inserción en el subárbol **izquierdo** de un hijo **derecho** (del nodo que se desbalancea)
  - FB(n) >1 y FB(n.der)<0

Rebalanceo

```
rebalancear(NodoAVL nodo): NodoAVL
    actualizarAltura(nodo);
    var fbd = balanceFactor(nodo);
     if (fbd < -1 && balanceFactor(nodo.izq) <= 0) // II
         nodo = rotacionDerecha(nodo);
     if (fbd > 1 && balanceFactor(nodo.der) >= 0)    // DD
           nodo = rotacionIzquierda(nodo);
     if (fbd < -1 && balanceFactor(nodo.izq) > 0)  // DI
          nodo = rotacionzquierdaDerecha(nodo);
     if (fbd > 1 && balanceFactor(nodo.der) < 0)   // ID
          nodo = rotacionDerechaIzquierda(nodo));
      actualizarAltura(nodo);
      return nodo;
}
```

inserción en AVL

1. Insertar el nuevo nodo como en un ABB “clásico”
   - el nuevo nodo es una hoja
2. Recalcular los factores de balanceo que cambiaron por la inserción
   - sólo en la rama en la que ocurrió la inserción (los otros factores no pueden cambiar!), de abajo hacia arriba
3. Si en la rama aparece un factor de balanceo de ±2 hay que rebalancear
   - A través de “rotaciones”

inserción en los AVL/costo

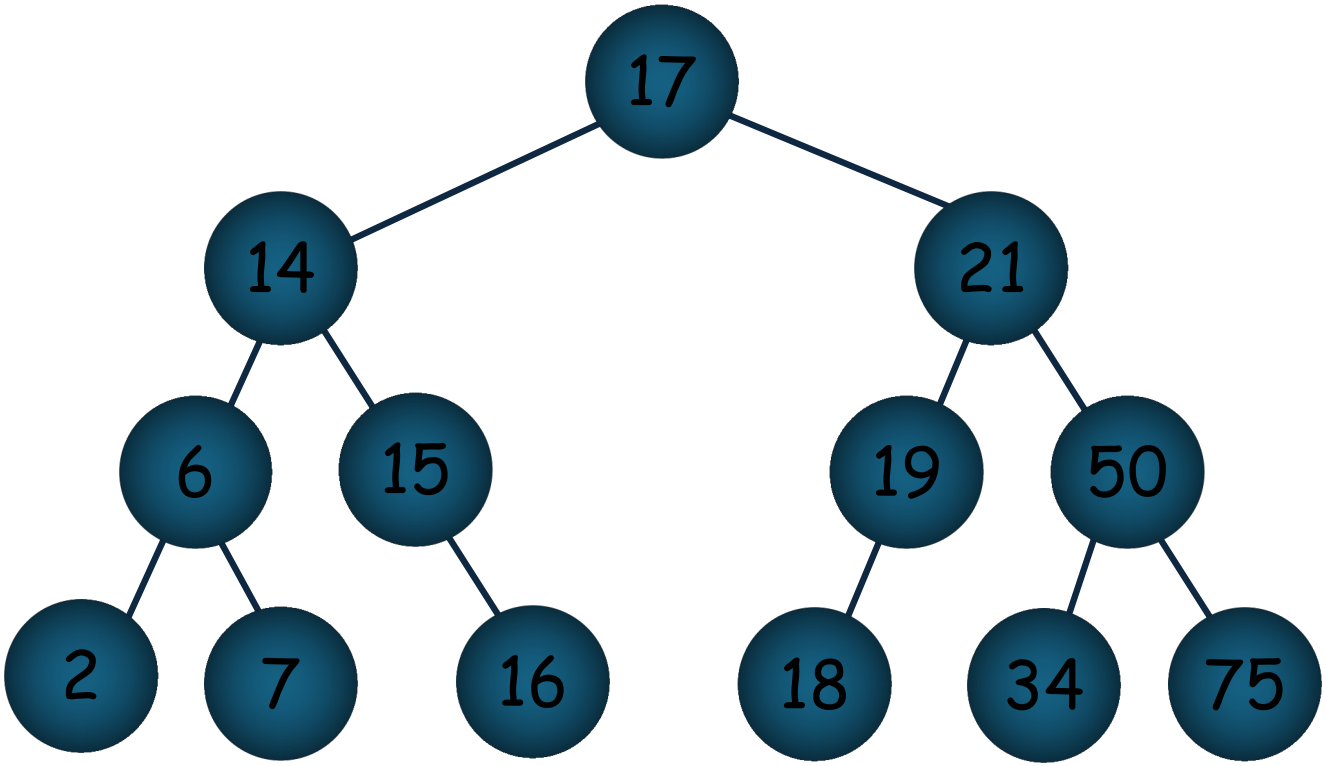
- paso 1: proporcional a la altura del árbol $\Theta(\lg n)$
- paso 2: proporcional a la altura del árbol $\Theta(\lg n)$
- paso 3: $O(1)$ (se hace una o dos rotaciones por inserción)

En total: $\Theta(\lg n)$

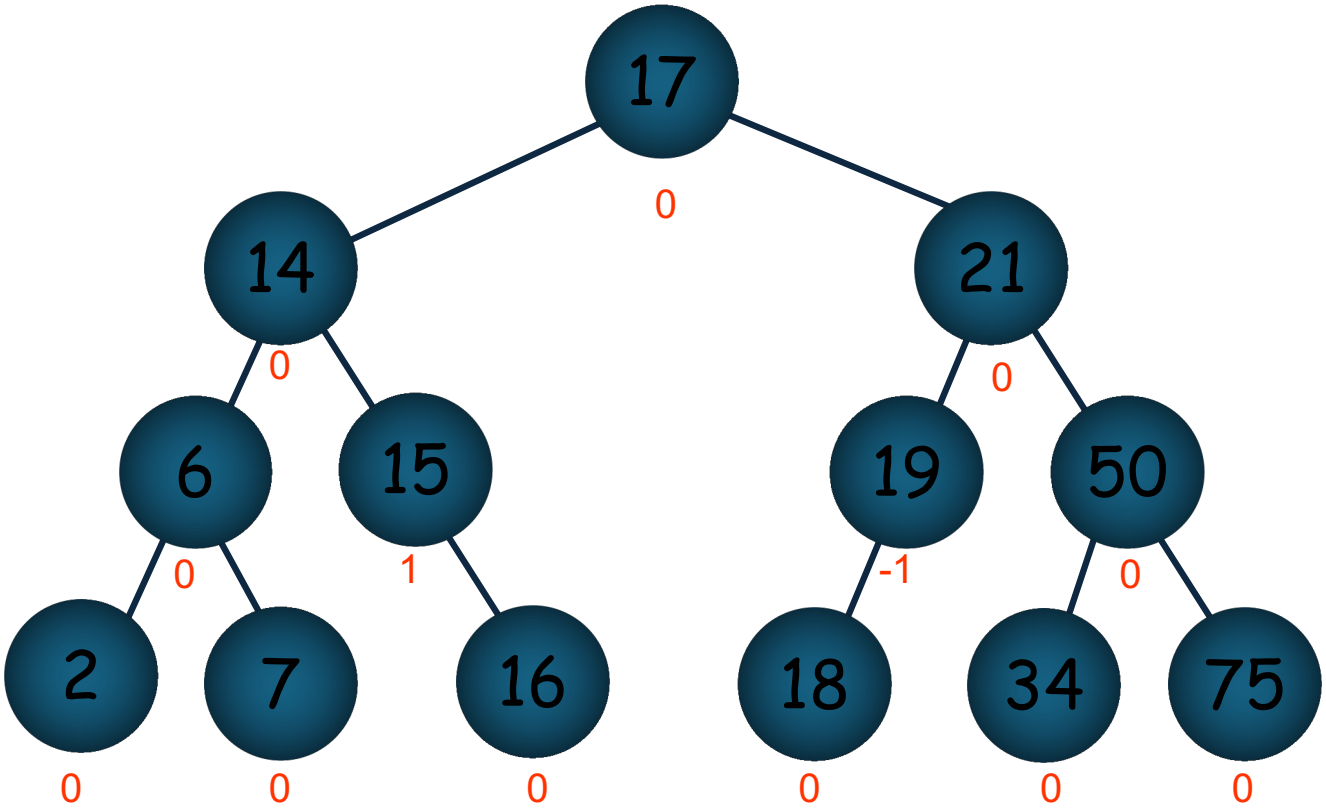
borrado en los AVL

1. Borrar el nodo como en un ABB “clásico”
2. recalcular los factores de balanceo que cambiaron por el borrado
   - sólo en la rama en que ocurrió el borrado, de abajo hacia arriba
3. <u>para cada nodo</u> con factor de balanceo ±2 hay que hacer una rotación simple o doble
   - O(lg *n)* rotaciones en el caso peor

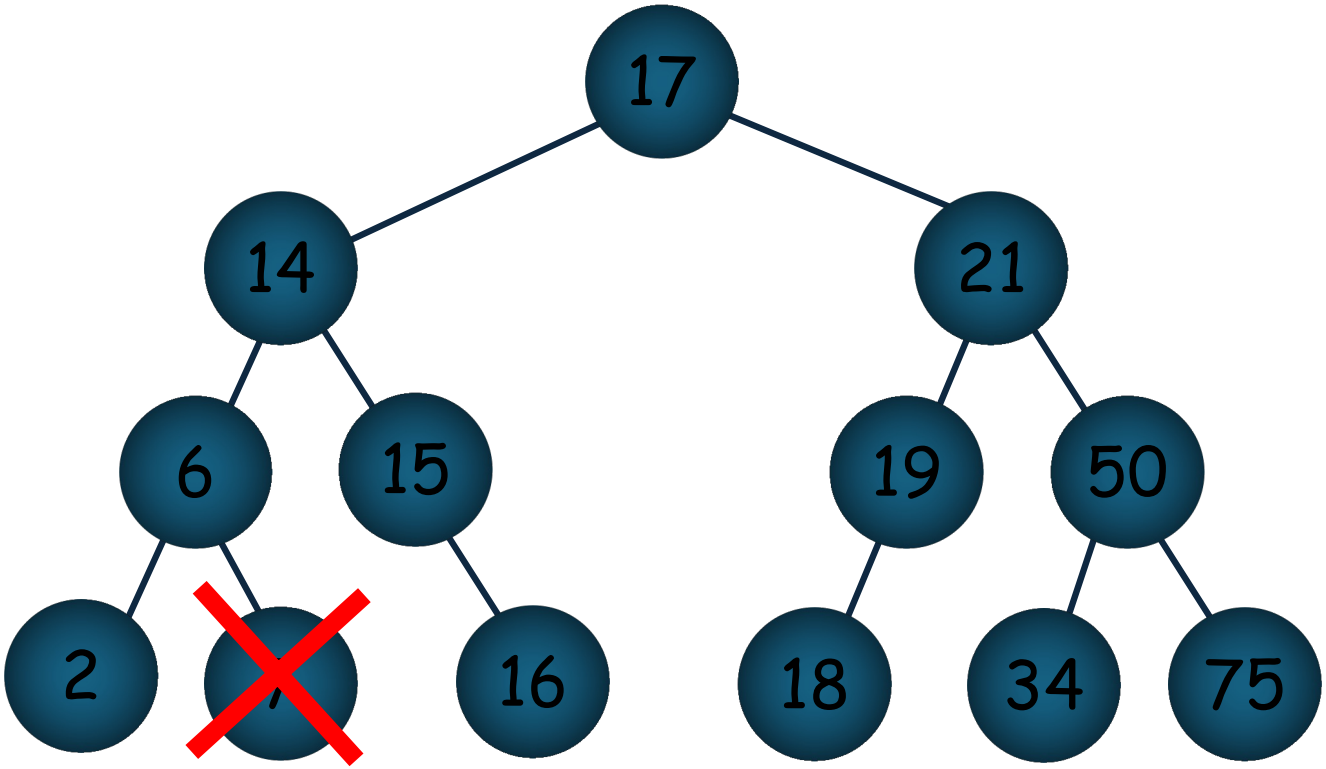
borrado en los AVL



borrado en los AVL

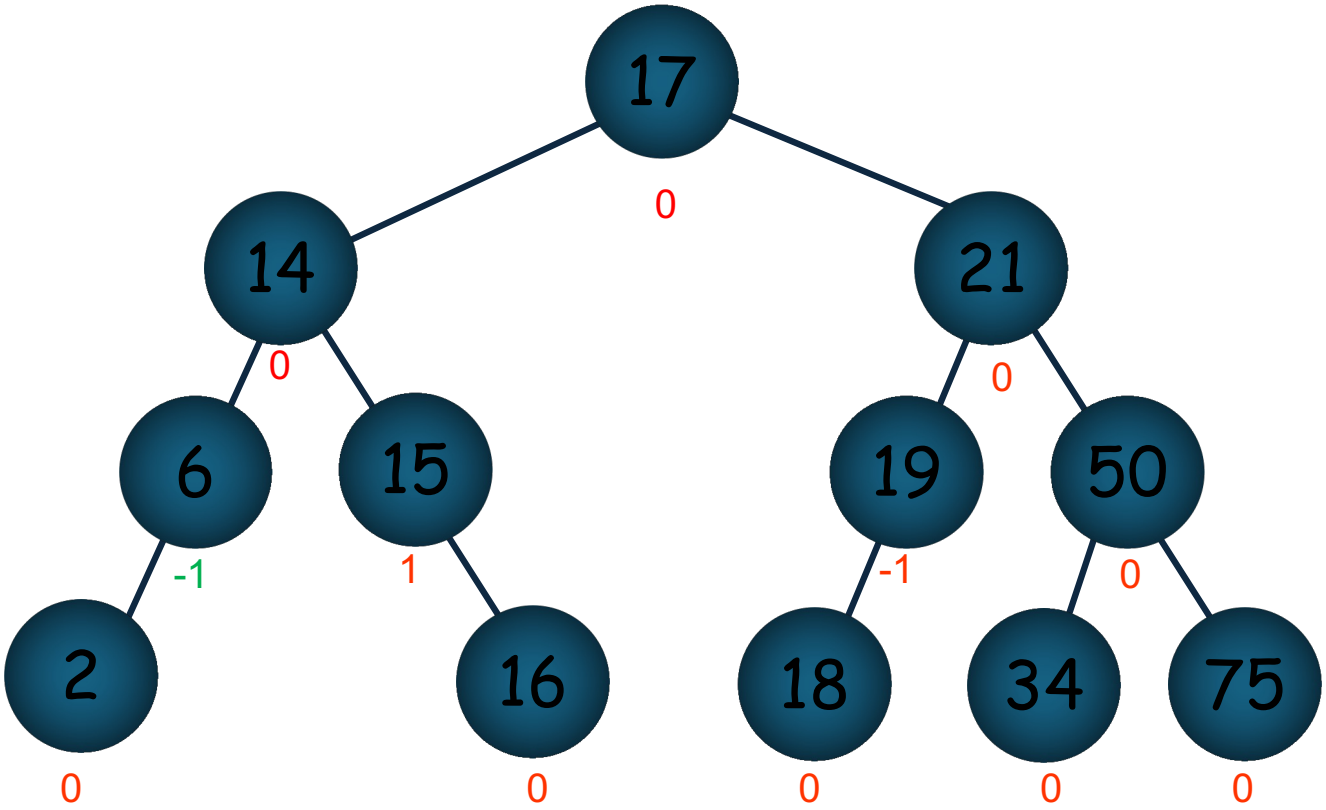


borrado en los AVL



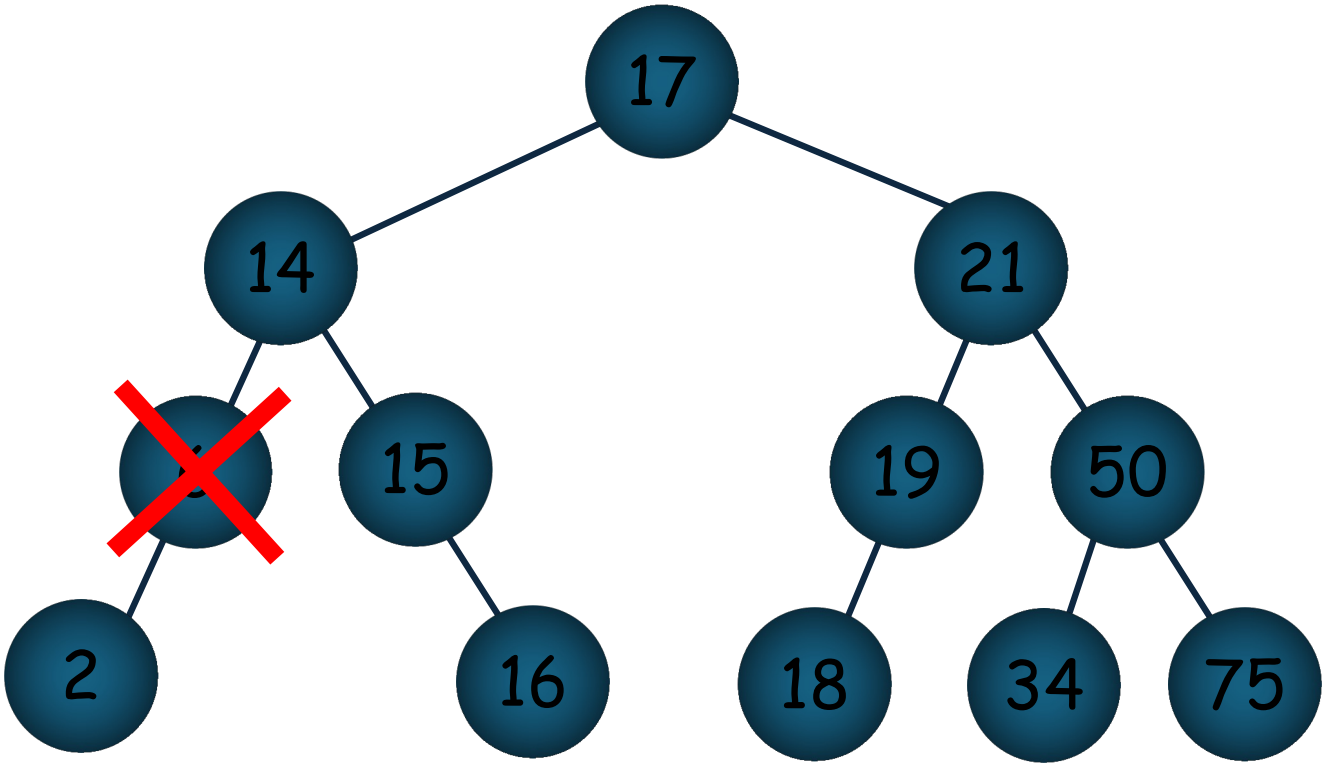
Borrar 7

borrado en los AVL

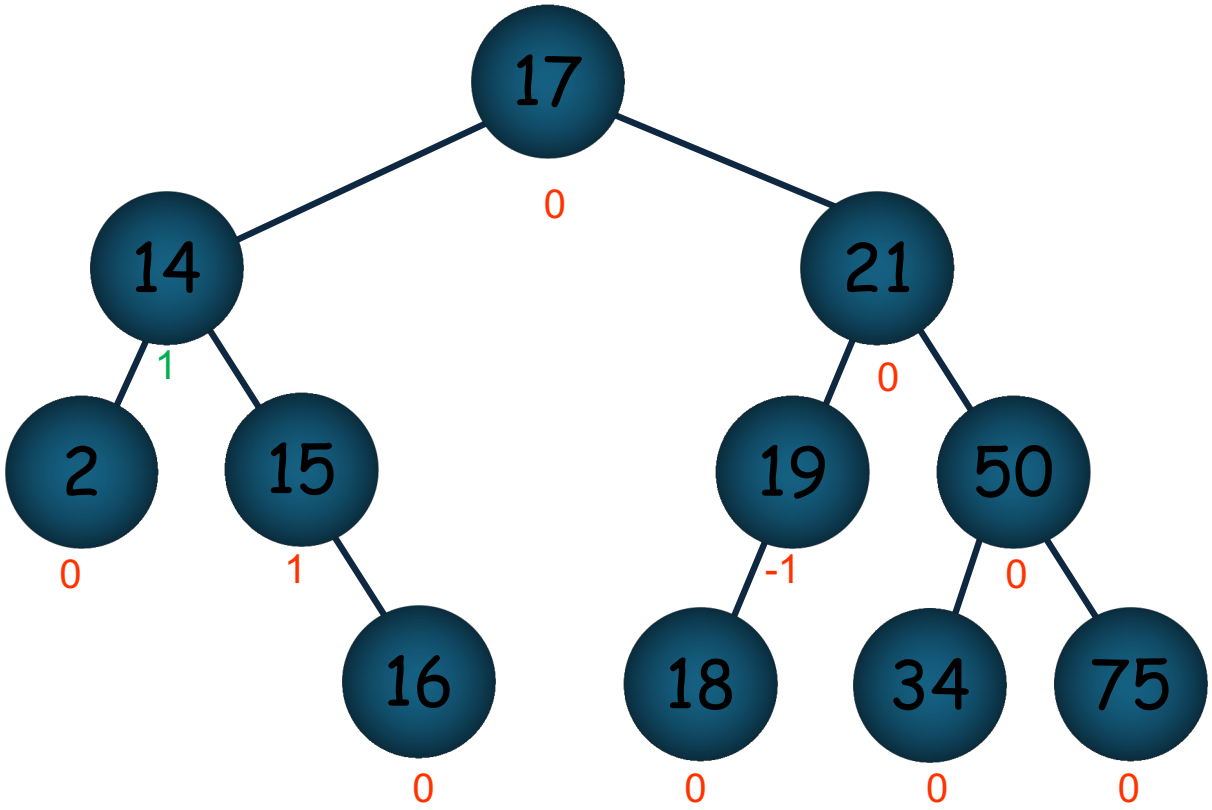


borrado en los AVL

Borrar 6

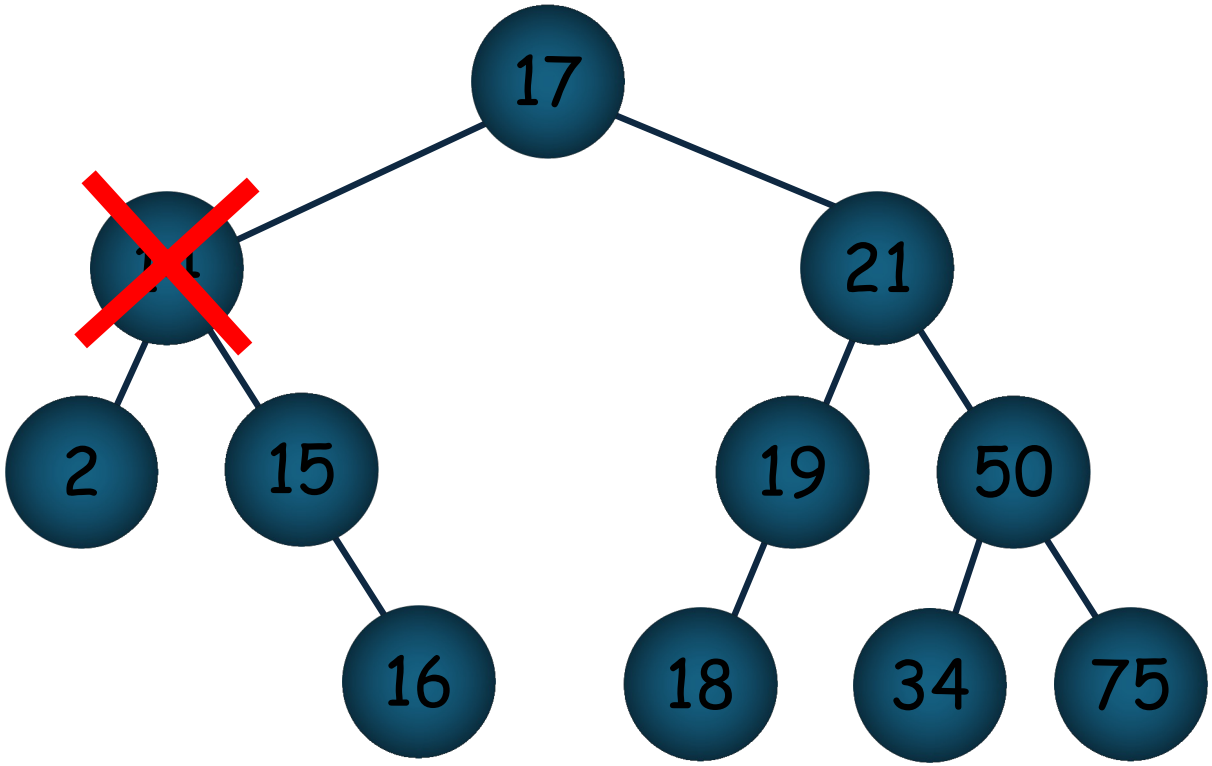


borrado en los AVL

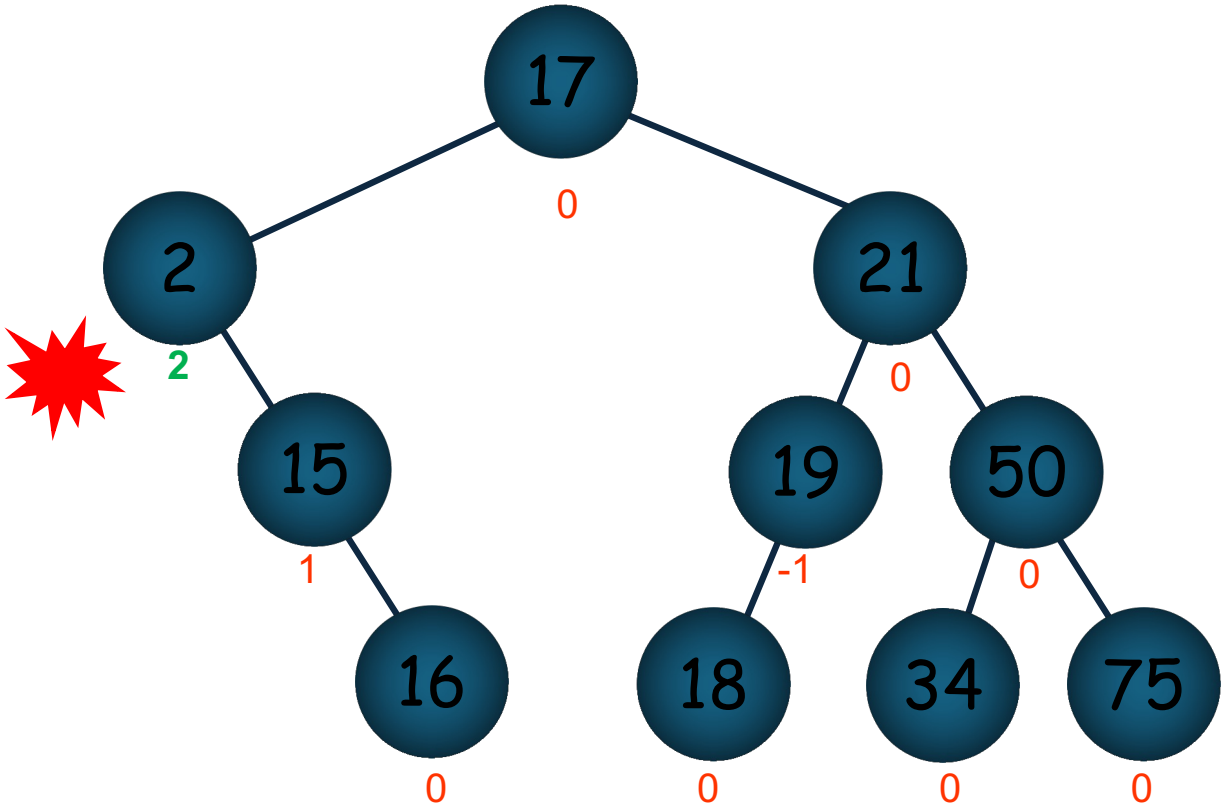


borrado en los AVL

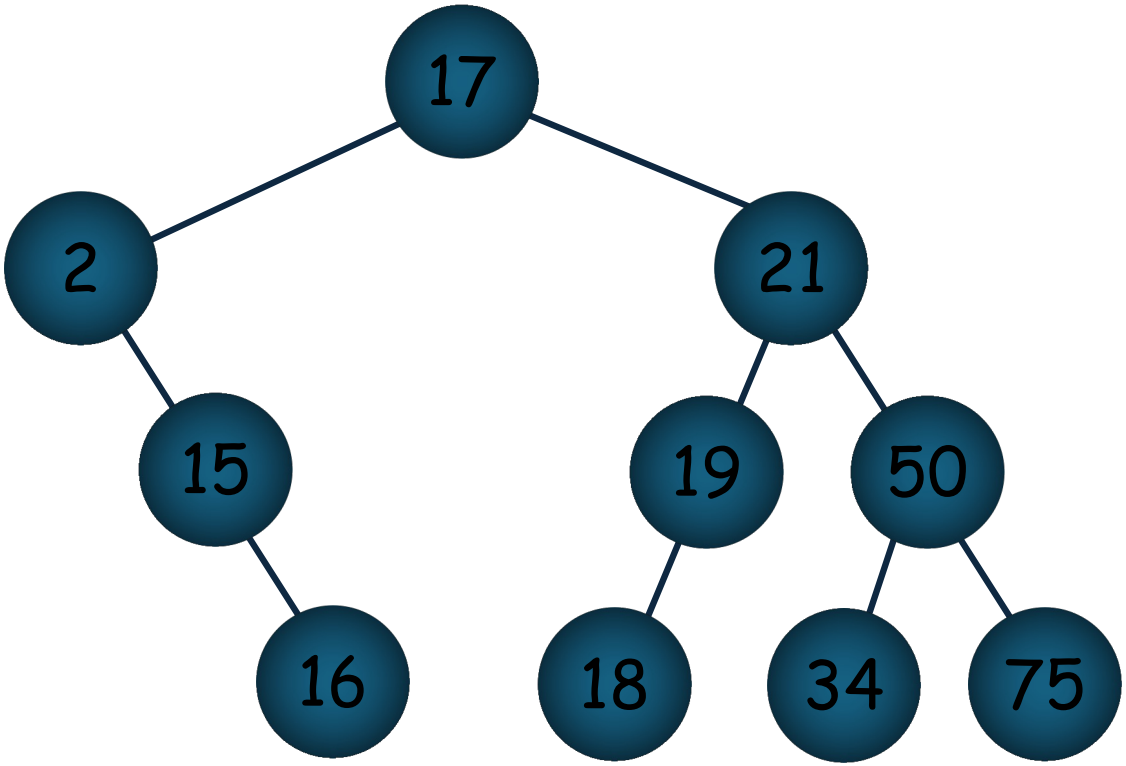
Borrar 14



borrado en los AVL

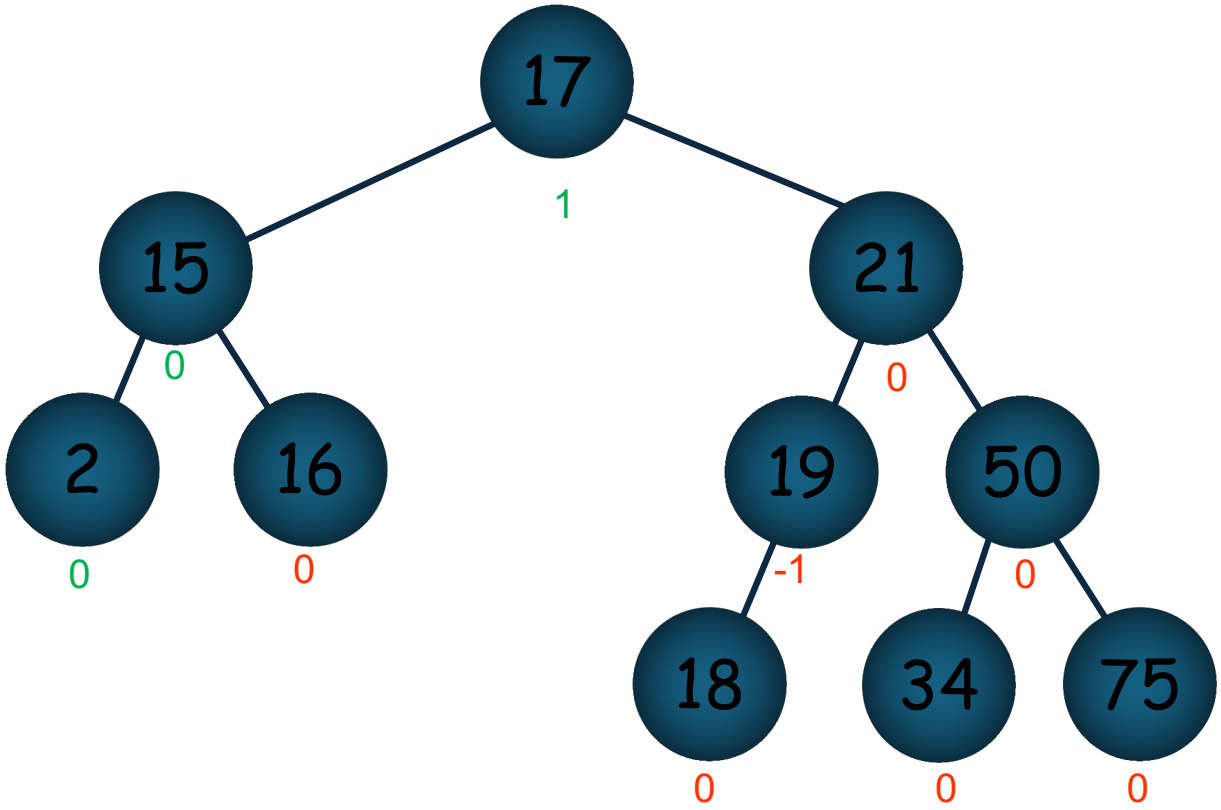


borrado en los AVL



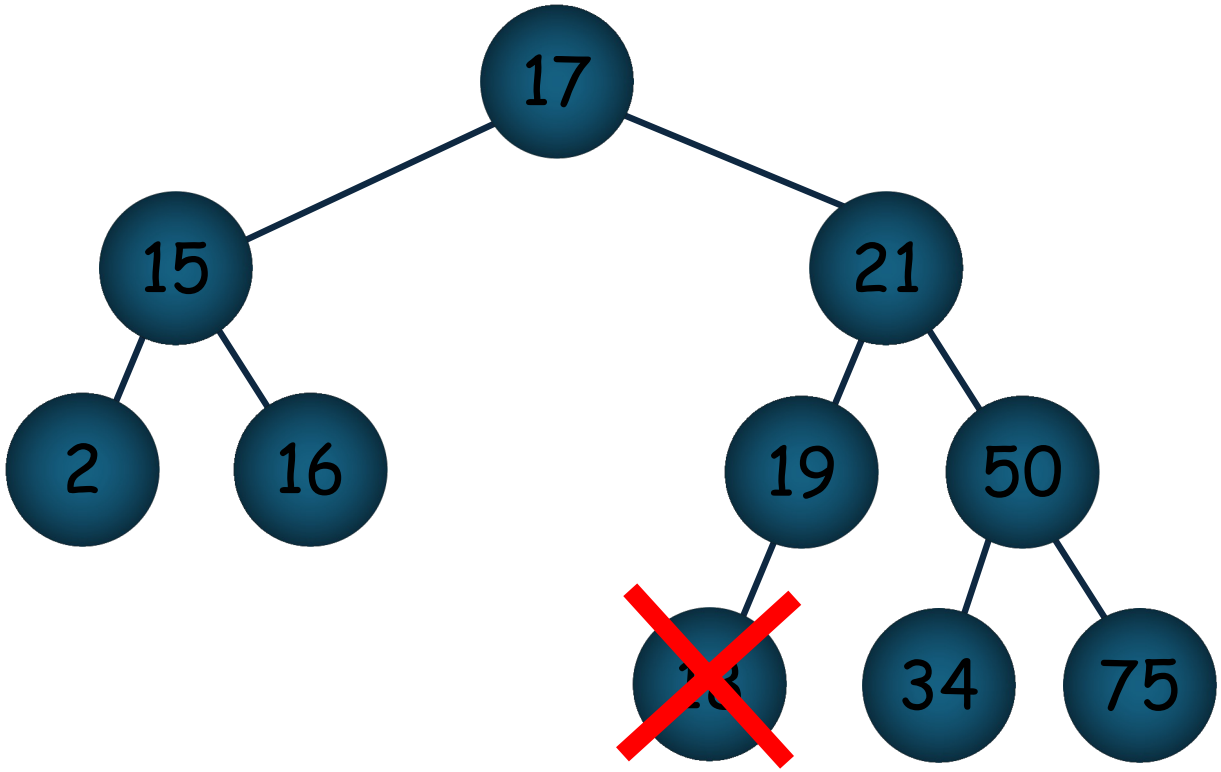
Balanceo DD:
rotación a izquierda(2)

borrado en los AVL

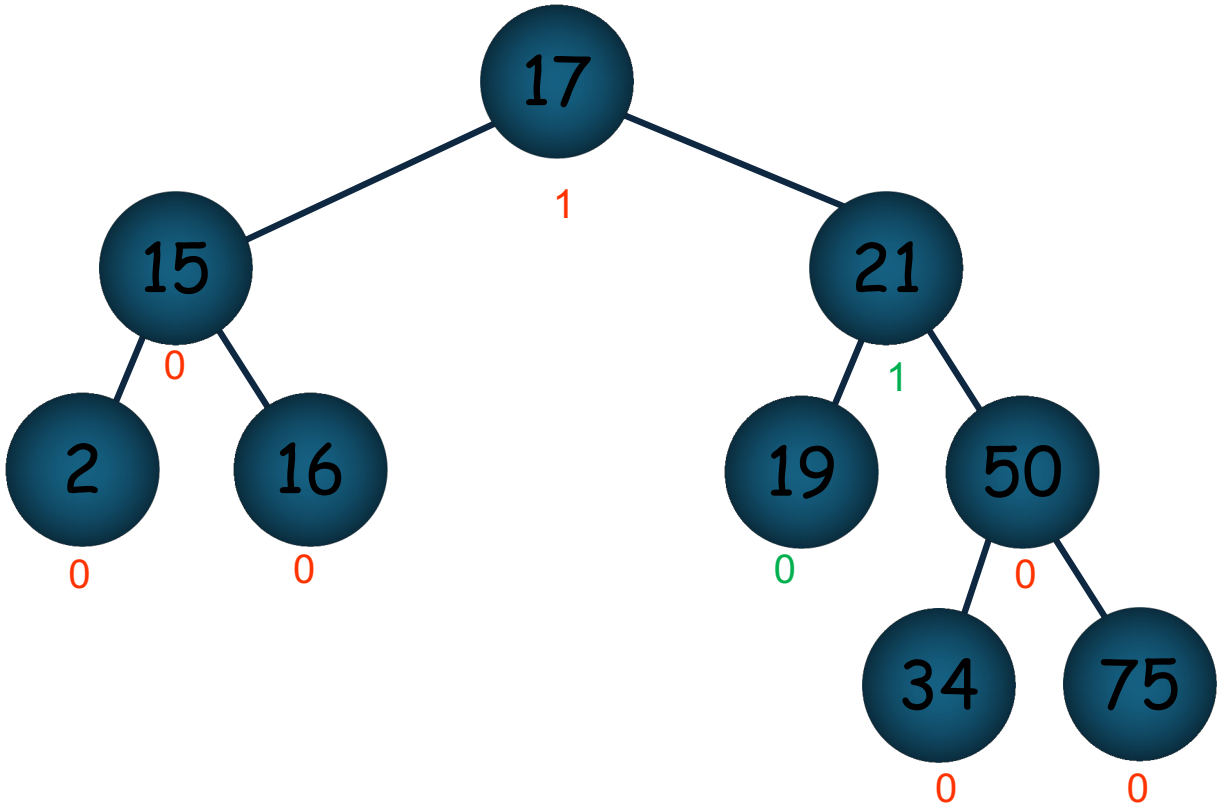


borrado en los AVL

Borrar 18

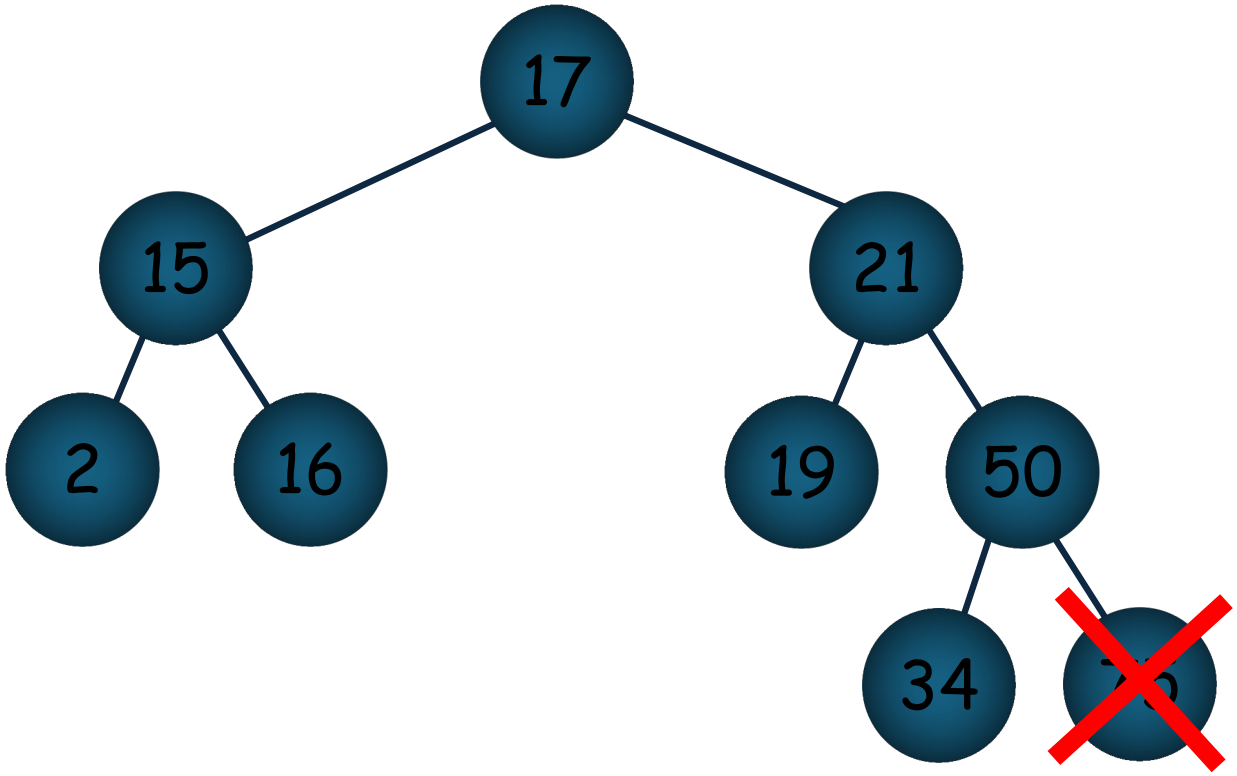


borrado en los AVL

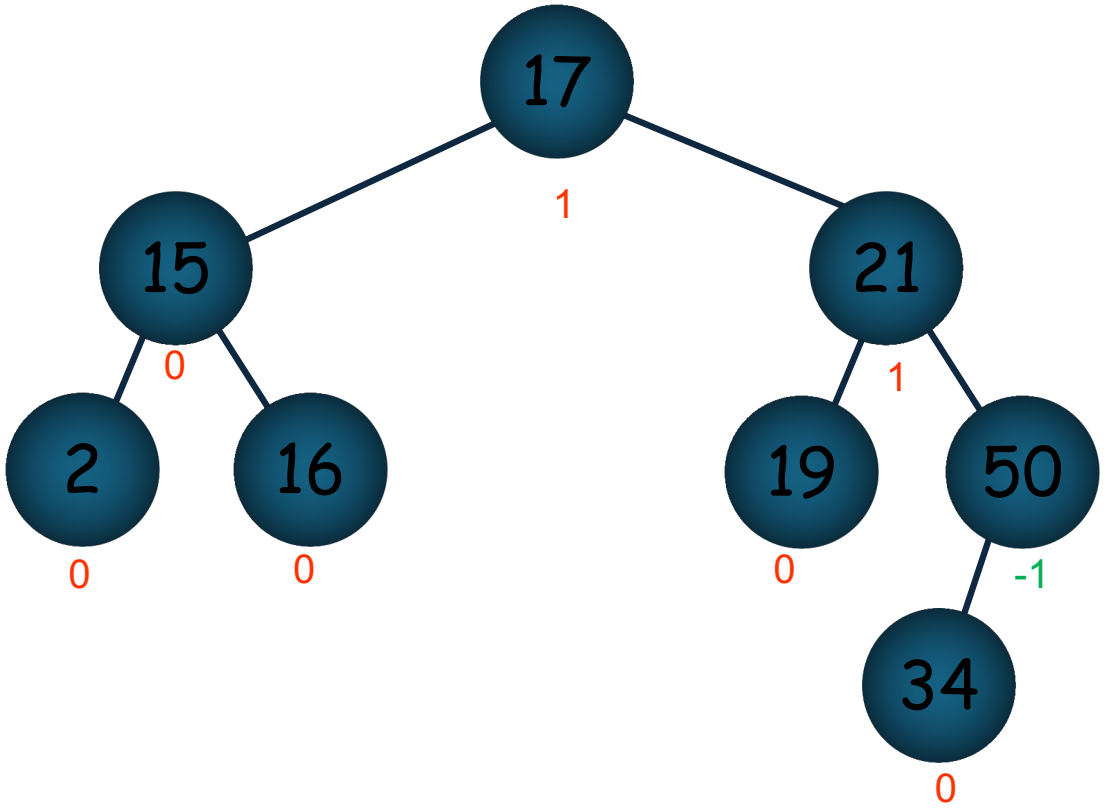


borrado en los AVL

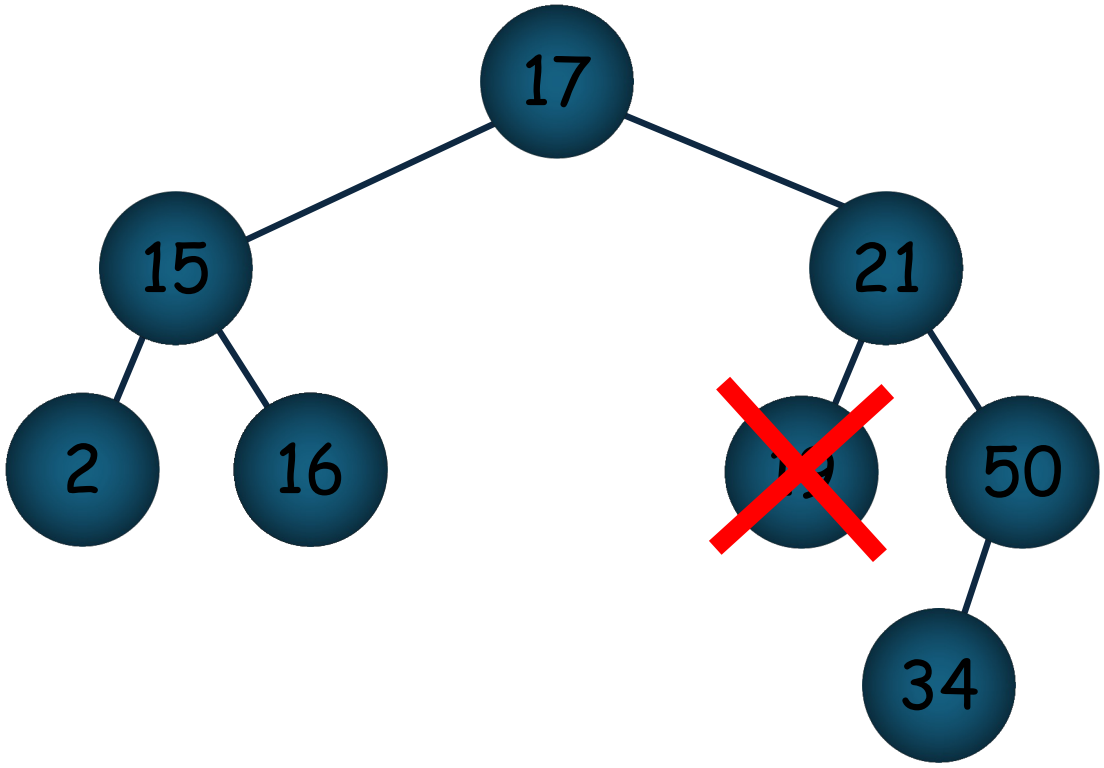
Borrar 75



borrado en los AVL

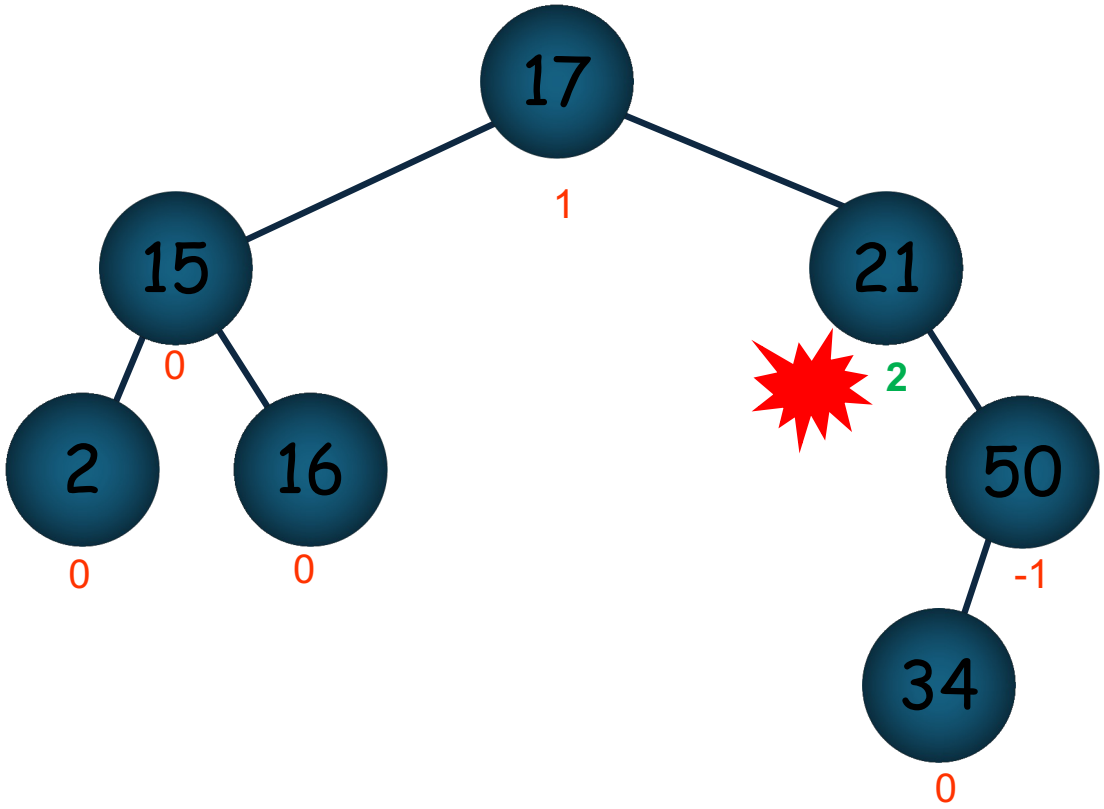


borrado en los AVL

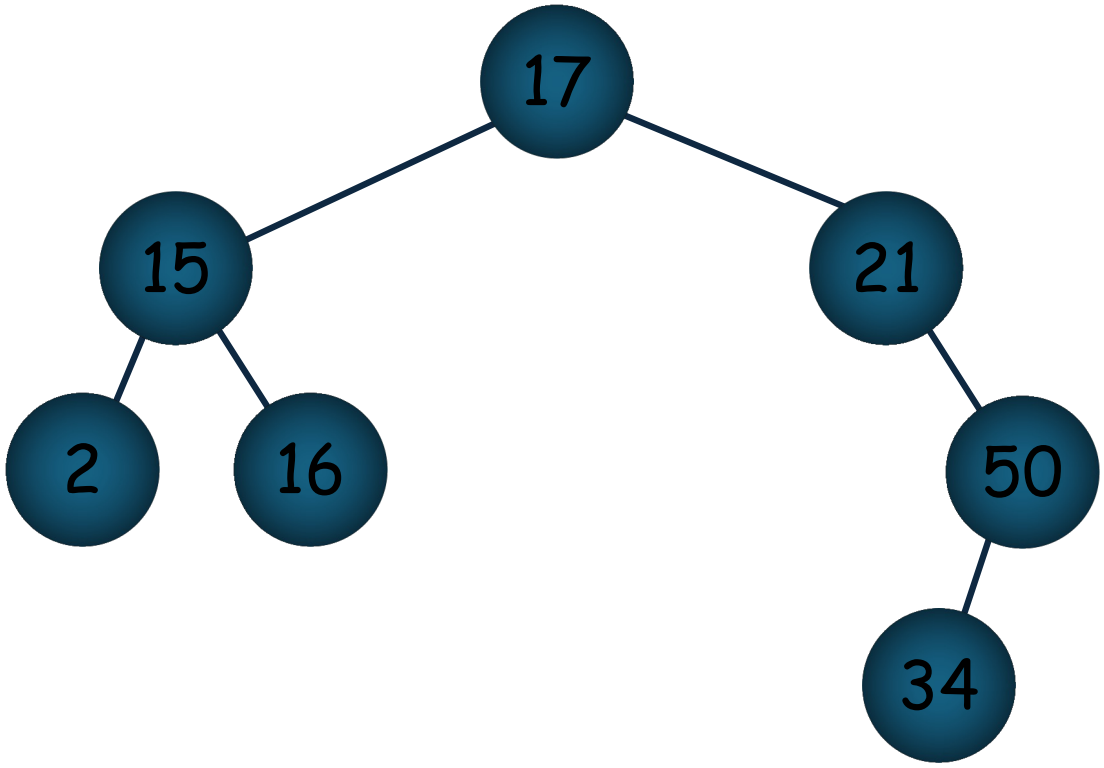


Borrar 19

borrado en los AVL

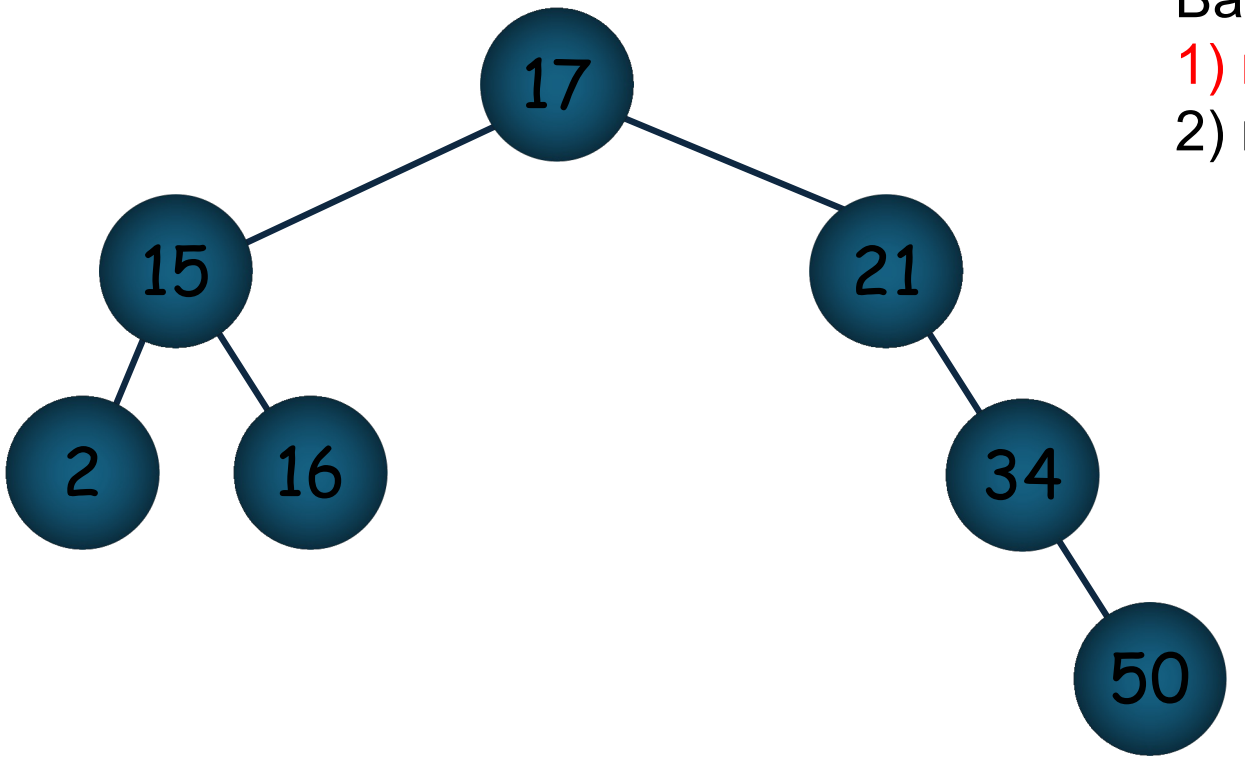


borrado en los AVL



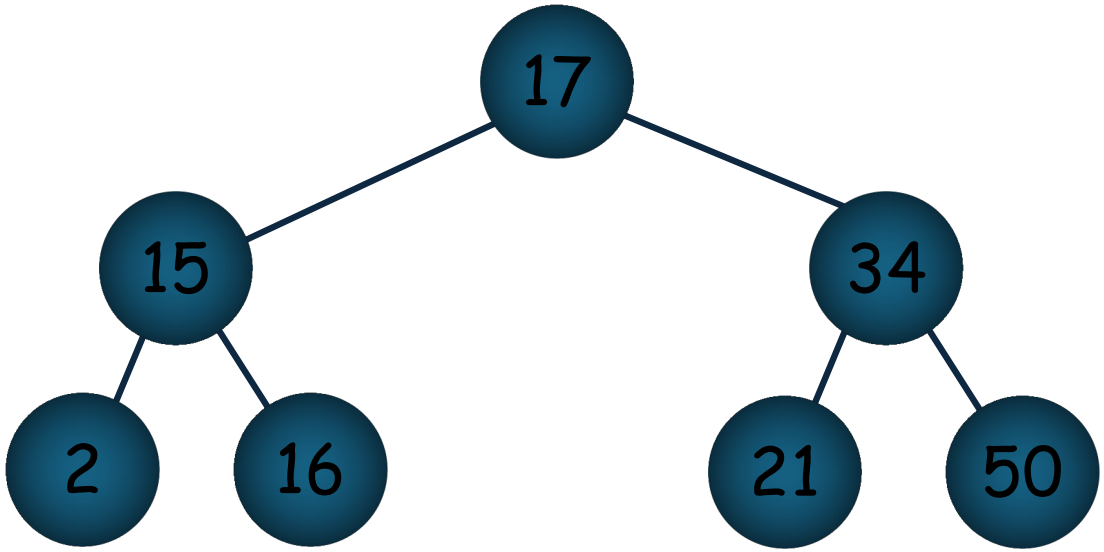
Balanceo ID: derecha-izq
1) rotación a derecha(50)
2) rotación a izquierda(21)

borrado en los AVL



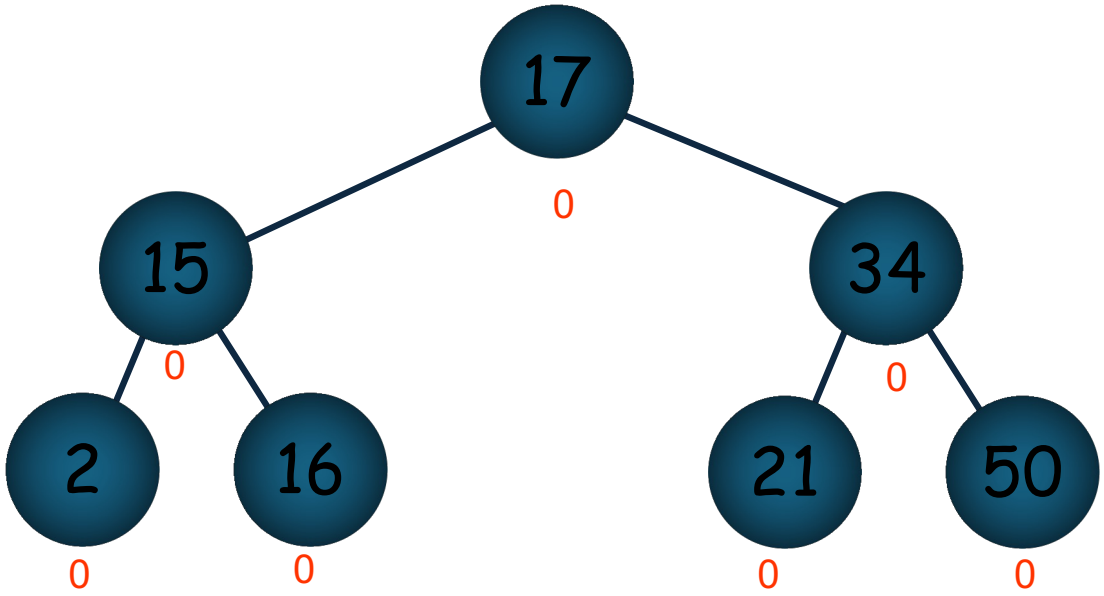
Balanceo ID: derecha-izq
1) rotación a derecha(50)
2) rotación a izquierda(21)

borrado en los AVL



Balanceo ID: derecha-izq
1) rotación a derecha(50)
2) rotación a izquierda(21)

borrado en los AVL



borrado en los AVL/costo

- en el caso peor hay que hacer rotaciones (simples o dobles) a lo largo de toda la rama
- paso 1: proporcional a la altura del árbol $\Theta(\lg n)$
- paso 2: proporcional a la altura del árbol $\Theta(\lg n)$
- paso 3: $\Theta(\lg n) \cdot \Theta(1)$

En total: $\Theta(\lg n)$