Fundamentos de Organización de Datos

Árboles B

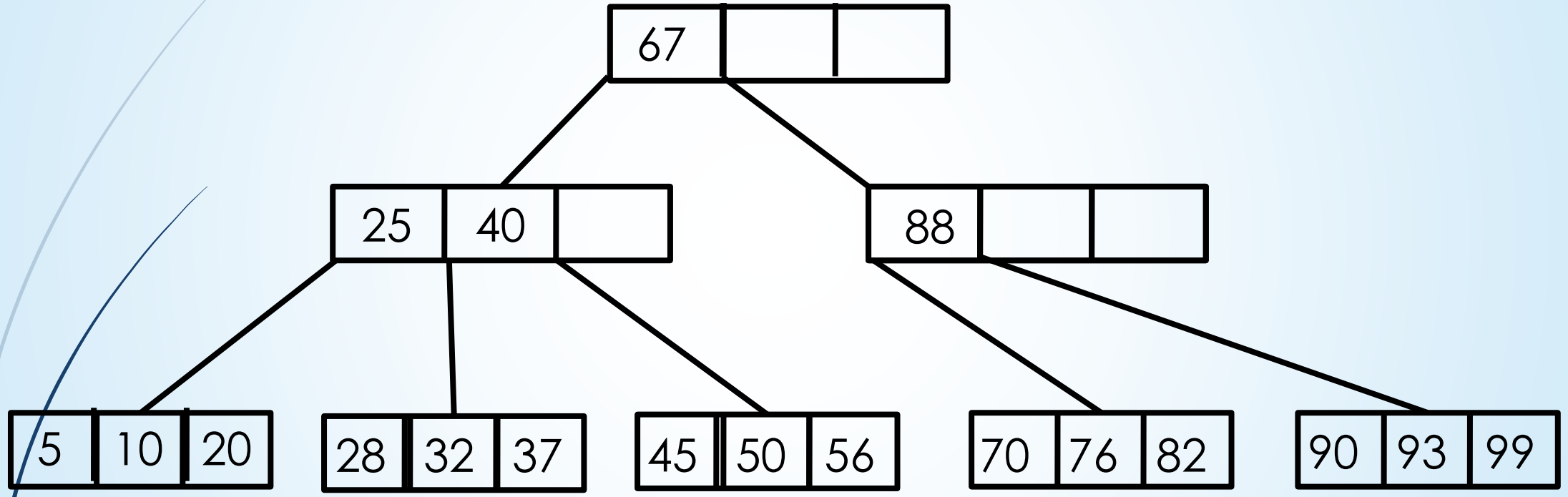
Árboles B y B+

Los árboles B son **árboles multicamino** con una **construcción especial en forma ascendente** que permite mantenerlos balanceados a bajo costo.

Propiedades de un Árbol B de orden M

- Cada nodo del árbol puede contener como **máximo M descendientes directos (hijos)**.
- La raíz no posee descendientes directos o tiene al menos dos.
- Un nodo interno con **X hijos** contiene **X-1 elementos**.
- Todos los nodos (salvo la raíz) tienen como **mínimo [M/2] – 1 elementos** y como **máximo M-1 elementos**.
- Todos los **nodos terminales** se encuentran al **mismo nivel**.
- Cada nodo tiene sus **elementos ordenados por clave**. Además, todos los elementos en el subárbol izquierdo de un elemento son menores o iguales que dicho elemento, mientras que todos los elementos en el subárbol derecho son mayores que ese elemento

Ejemplo de Árbol B de orden 4



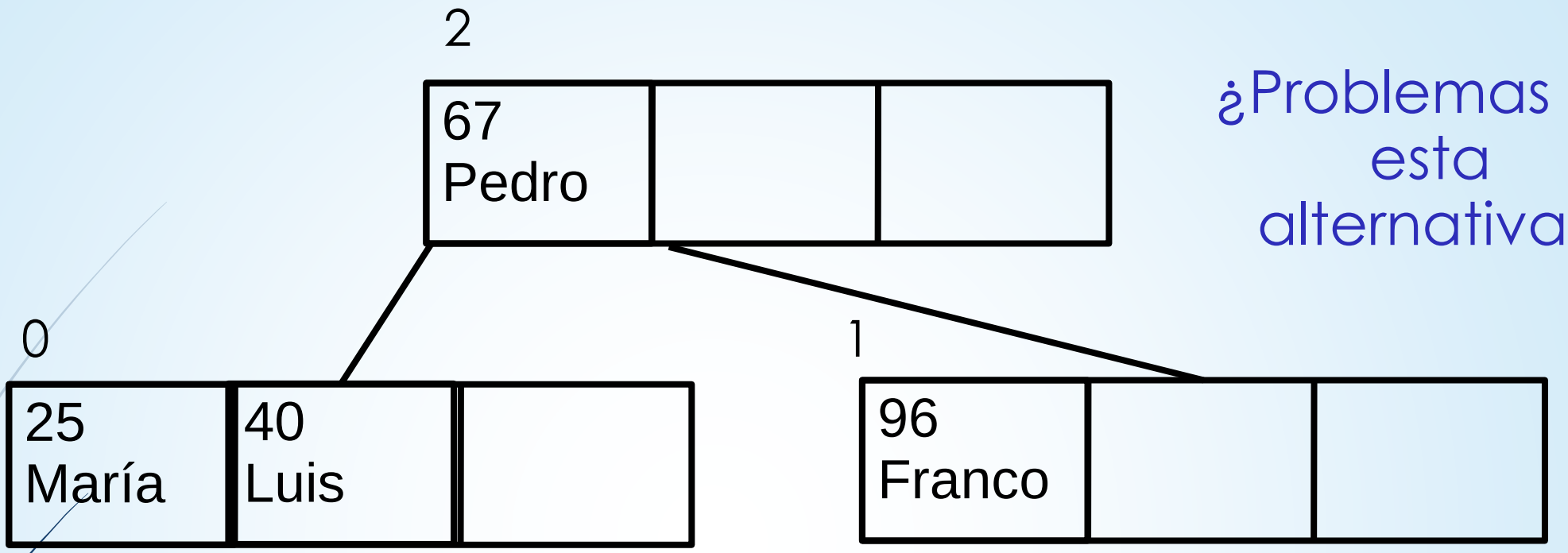
¿Para qué usamos los árboles B?

Alternativas:

- Organizar el archivo de datos como un árbol B
- Organizar el archivo índice un árbol B

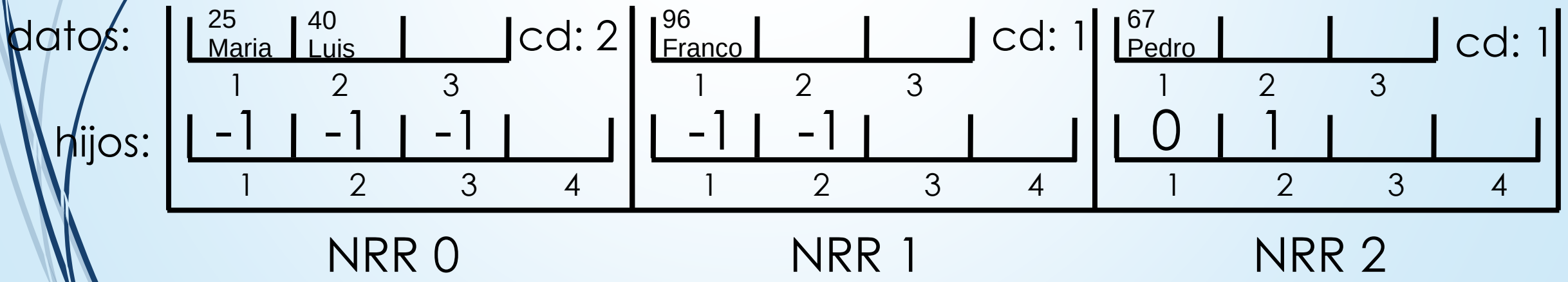
Archivo de datos como árbol B

```
const M = … ; {orden del árbol}
type
    TDato = record
      codigo: longint;
      nombre: string[50];
    end;
        TNodo = record
                cant_datos:  integer;
                datos: array[1..M-1] of TDato;
                hijos: array[1..M] of integer;
        end;
        arbolB = file of TNodo;
var
                archivoDatos: arbolB;
```



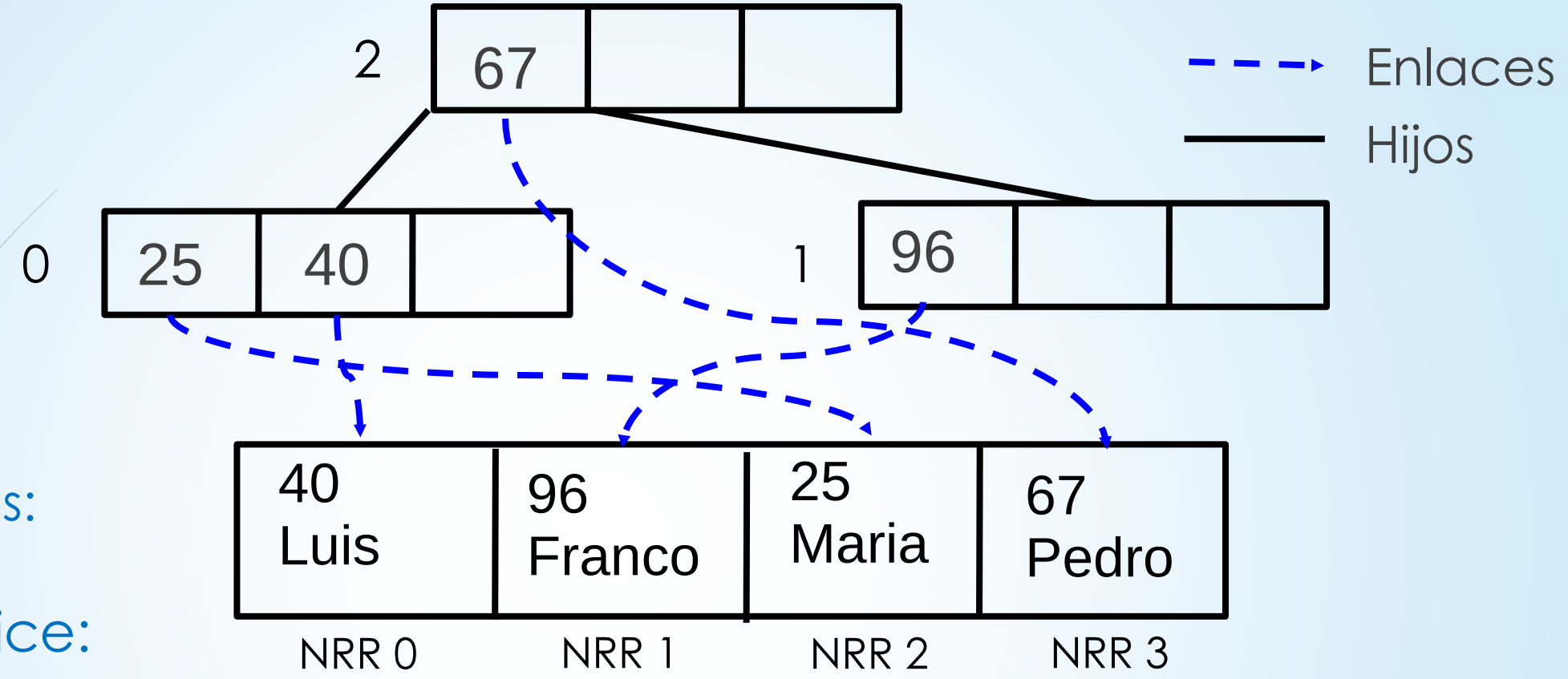
¿Problemas de esta alternativa?

Archivo:



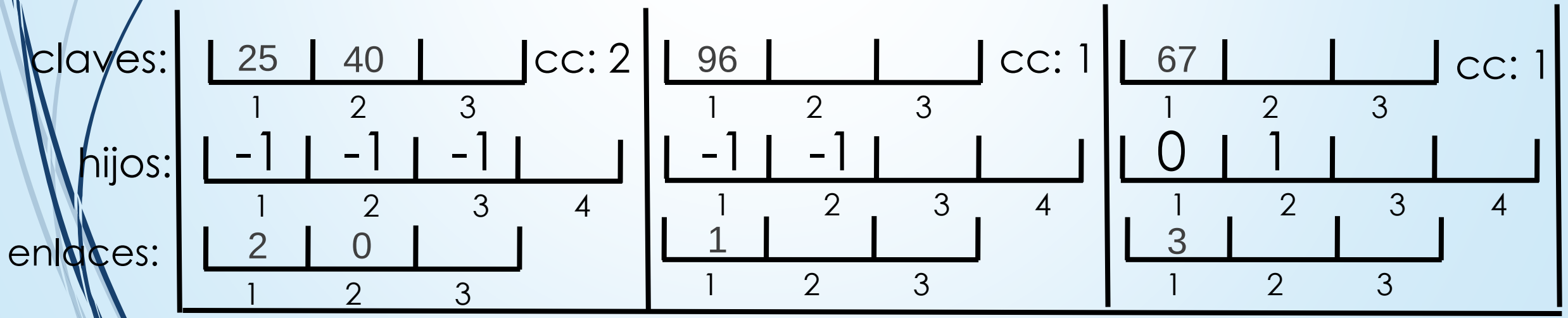
Archivo índice como árbol B

```
const M = … ; {orden del árbol}
type
    TDato = record
      codigo: longint;
      nombre: string[50];
    end;
        TNodo = record
                    cant_claves:    integer;
                    claves: array[1..M-1] of longint;
                    enlaces: array [1..M-1] of integer;
                    hijos: array[1..M] of integer;
            end;
            TArchivoDatos  = file of TDato;
            arbolB = file of TNodo;
var
                    archivoDatos: TArchivoDatos;
                    archivoIndice: arbolB;
```



Archivo Datos:

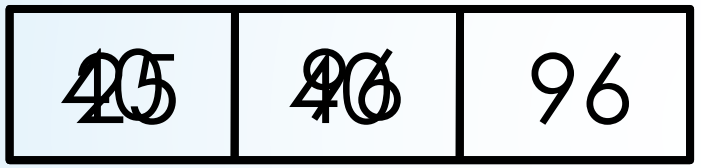
Archivo Indice:



Ejemplo – Árbol B de orden 4

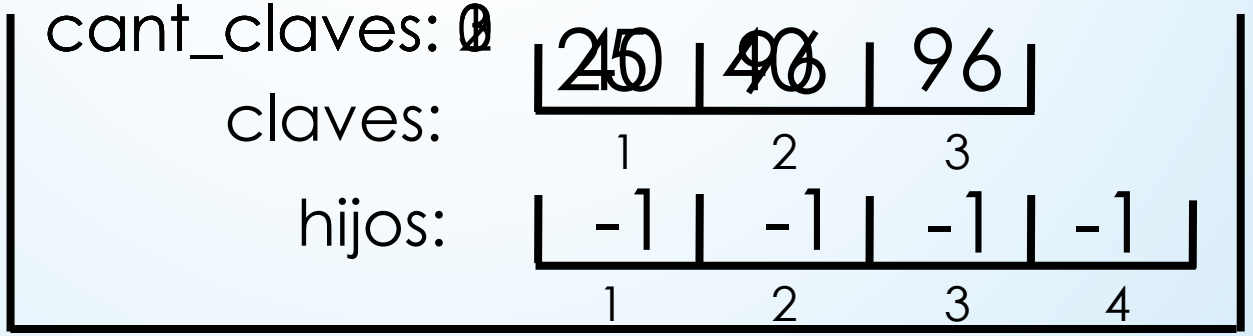
Árbol Inicial

Nodo 0



+40, +96, +25 , +67

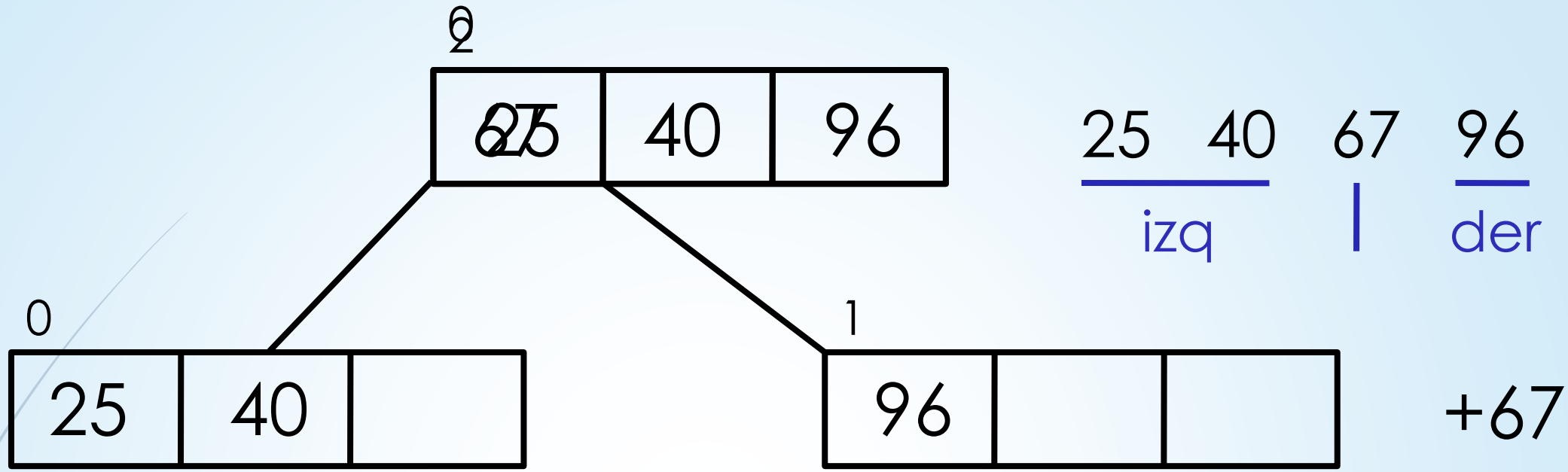
Archivo:



NRR 0

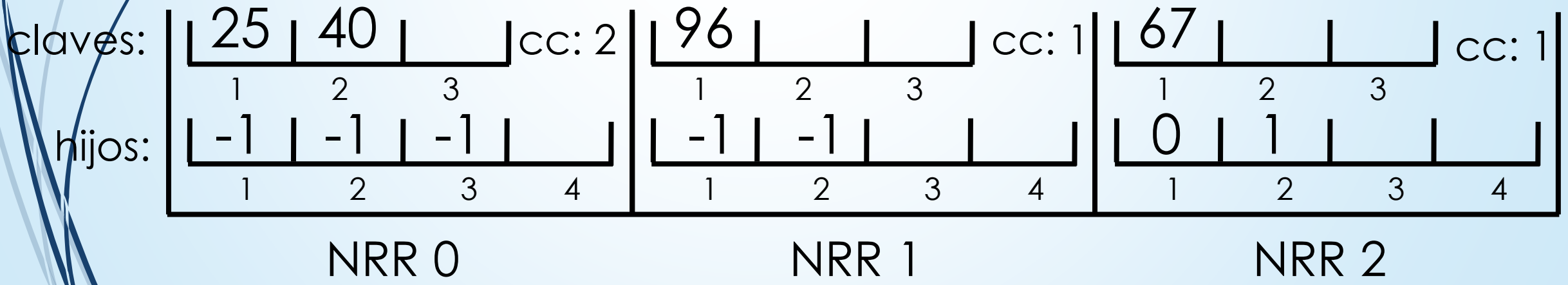
Overflow

- Se crea un nuevo nodo.
- La primera mitad de las claves se mantiene en el nodo con overflow.
- La segunda mitad de las claves se traslada al nuevo nodo.
- La menor de las claves de la segunda mitad se promociona al nodo padre.



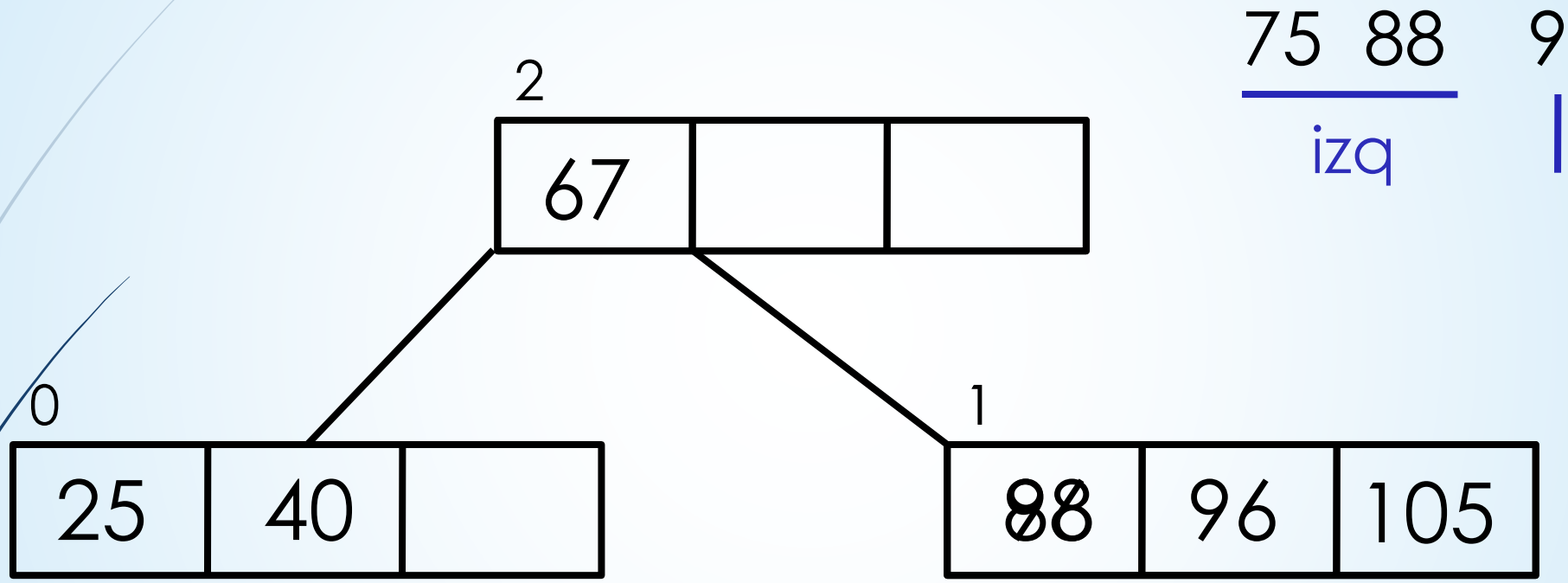
División de la raíz. Se incrementa la altura del árbol.

Archivo:



¡Notar la numeración de los nodos!

Ejemplo – Árbol B de orden 4



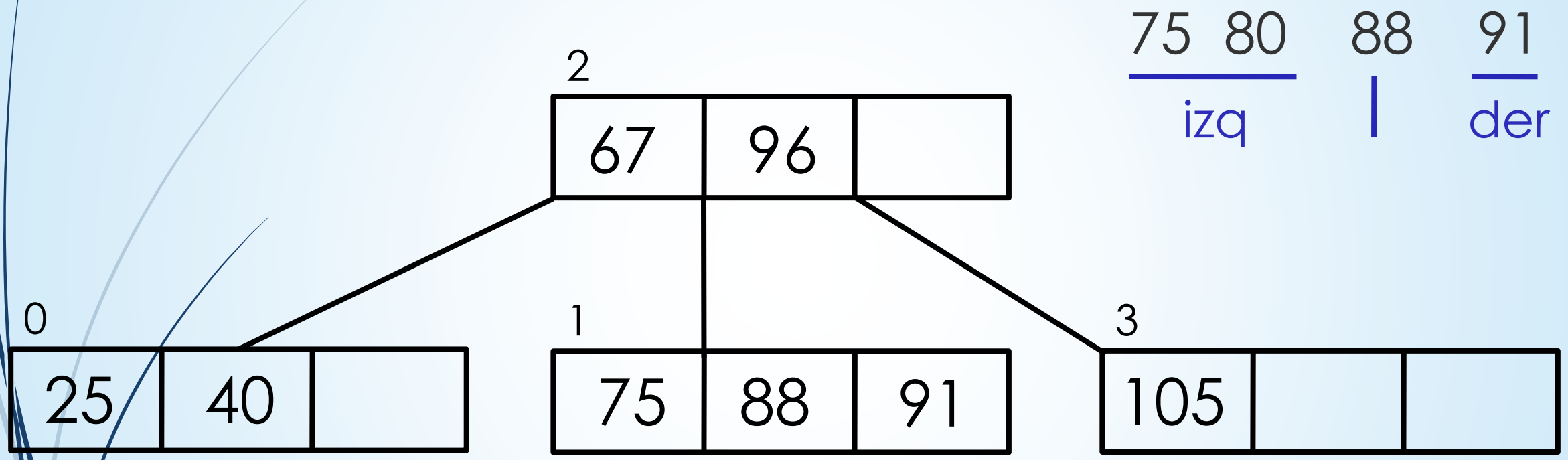
75 88 | 96 | 105

izq | der

+88 , +105 , +75

Overflow en el nodo 1. División del mismo y promoción de la clave 96.

Ejemplo – Árbol B de orden 4



75 80 | 88 | 91

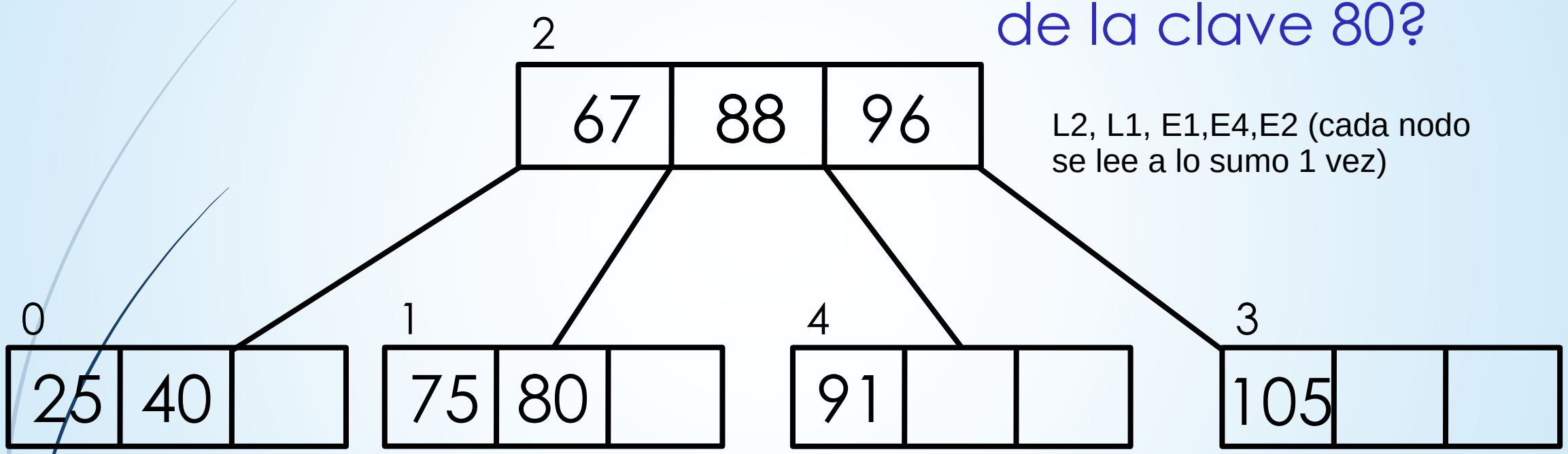
izq | der

+75 , +91 , +80

Overflow en el nodo 1. División del mismo y promoción de la clave 88.

Ejemplo – Árbol B de orden 4

¿L/E necesarias para el alta de la clave 80?



L2, L1, E1,E4,E2 (cada nodo se lee a lo sumo 1 vez)

+80 Completamos el árbol con las altas de: +86,+120,+230,+95,+55

Overflow en el nodo 1. División del mismo y promoción de la clave 80.

70 75 | 80 | 86

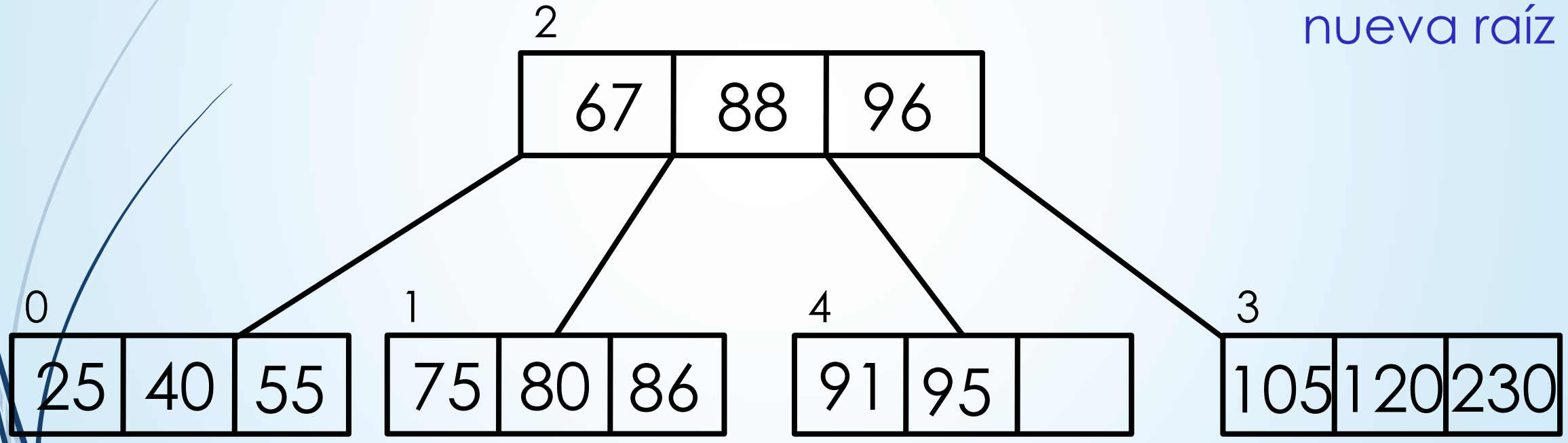
izq | der

Propagación del overflow a la raíz. División de la misma y aumento en la altura del árbol.

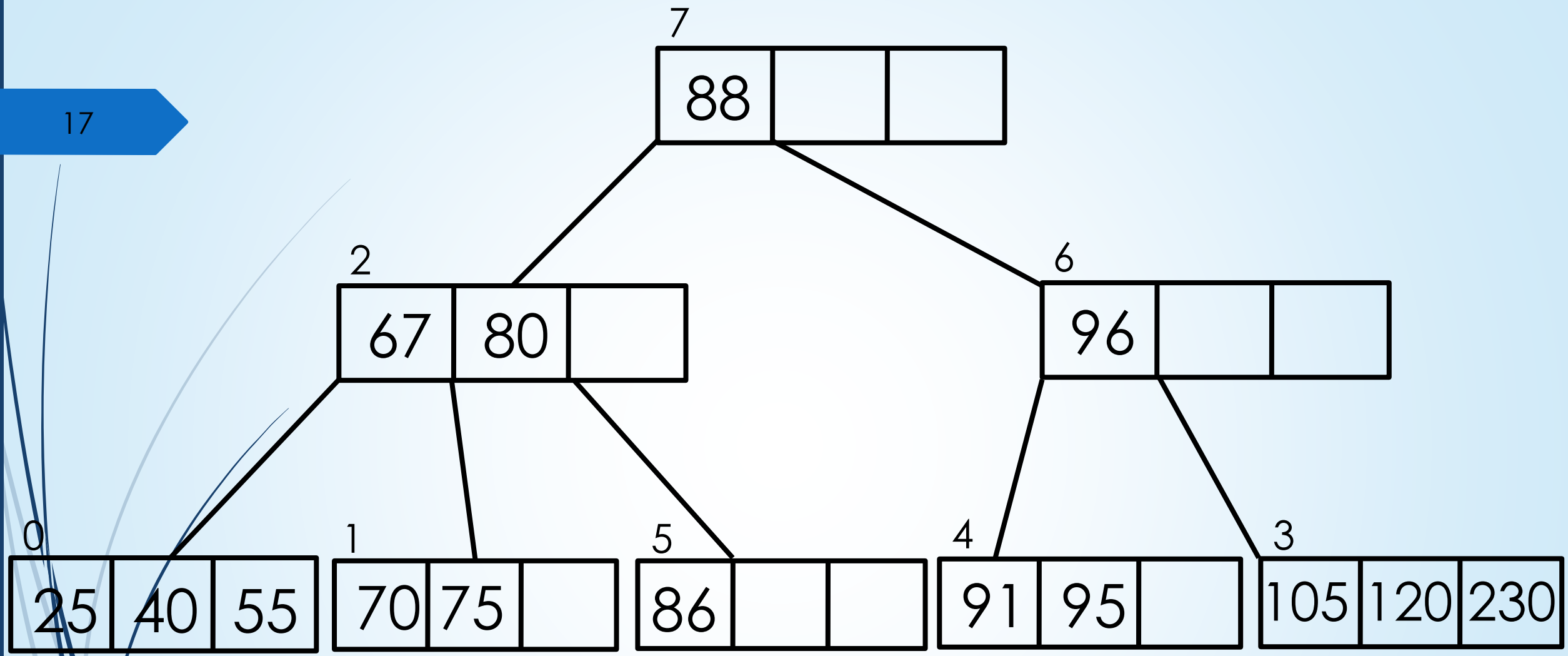
67 80 | 88 | 96

izq | der

nueva raíz

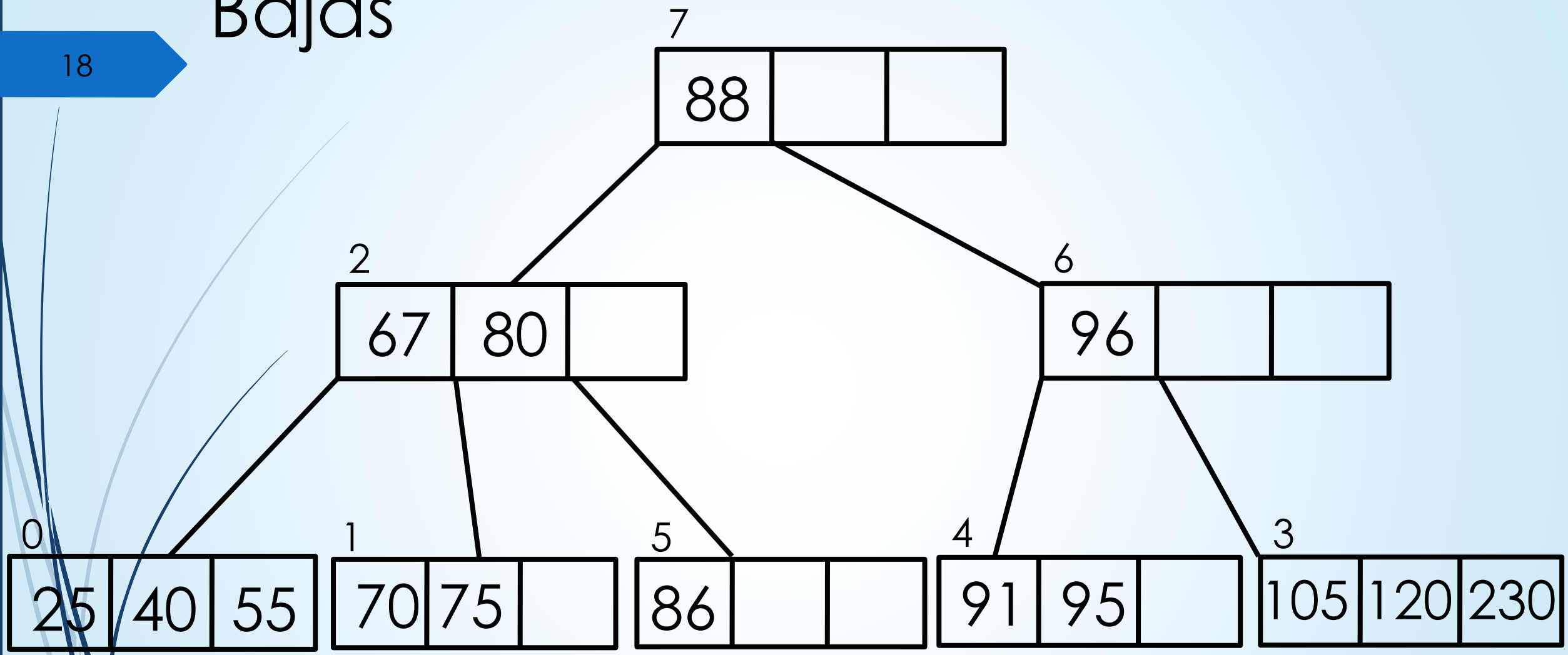


+86 , +120 , +230 , +95, +55 . Alta de +70



+70

Bajas



-75

Bajas

1. Si la clave a eliminar no está en una hoja, se debe reemplazar con la menor clave del subárbol derecho.
2. Si el nodo hoja contiene por lo menos el mínimo número de claves, luego de la eliminación, no se requiere ninguna acción adicional.
3. En caso contrario, se debe tratar el underflow

Bajas - Underflow

4. Primero *se intenta* **redistribuir** con un hermano adyacente. La redistribución es un proceso mediante el cual se trata de dejar cada nodo lo más equitativamente cargado posible.

5. Si la redistribución no es posible, entonces se debe **fusionar** con el hermano adyacente.

Políticas para la resolución de underflow:

1. **Política izquierda:** se intenta redistribuir con el hermano adyacente izquierdo, si no es posible, se fusiona con hermano adyacente izquierdo.
2. **Política derecha:** se intenta redistribuir con el hermano adyacente derecho, si no es posible, se fusiona con hermano adyacente derecho.
3. **Política izquierda o derecha:** se intenta redistribuir con el hermano adyacente izquierdo, si no es posible, se intenta con el hermano adyacente derecho, si tampoco es posible, se fusiona con hermano adyacente izquierdo.
4. **Política derecha o izquierda:** se intenta redistribuir con el hermano adyacente derecho, si no es posible, se intenta con el hermano adyacente izquierdo, si tampoco es posible, se fusiona con hermano adyacente derecho.

Políticas para la resolución de underflow:

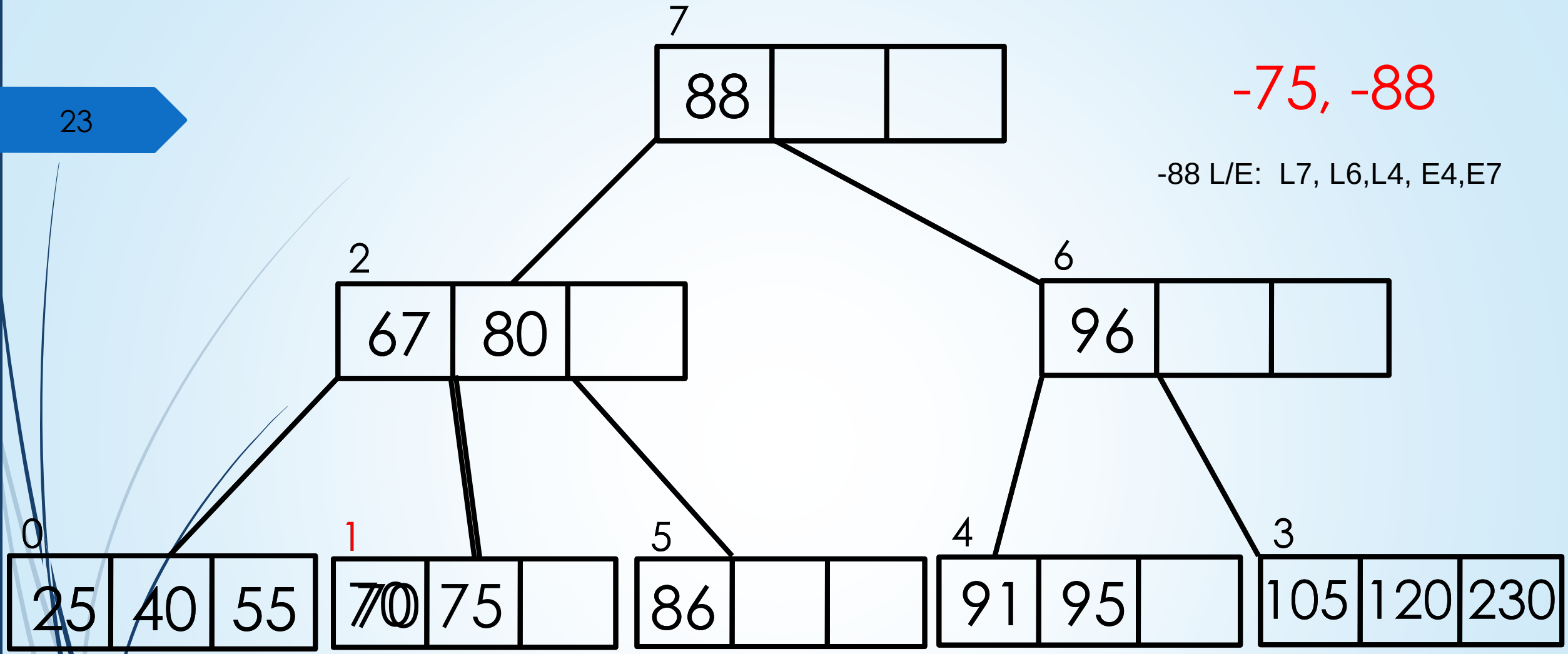
Casos especiales: en cualquier política si se tratase de un nodo hoja de un extremo del árbol debe intentarse redistribuir con el hermano adyacente que el mismo posea.

Aclaración:

- En caso de underflow lo primero que se intenta **SIEMPRE** es redistribuir si el hermano adyacente se encuentra en condiciones de hacer la redistribución y no se produce underflow en el.

-75, -88

-88 L/E: L7, L6,L4, E4,E7

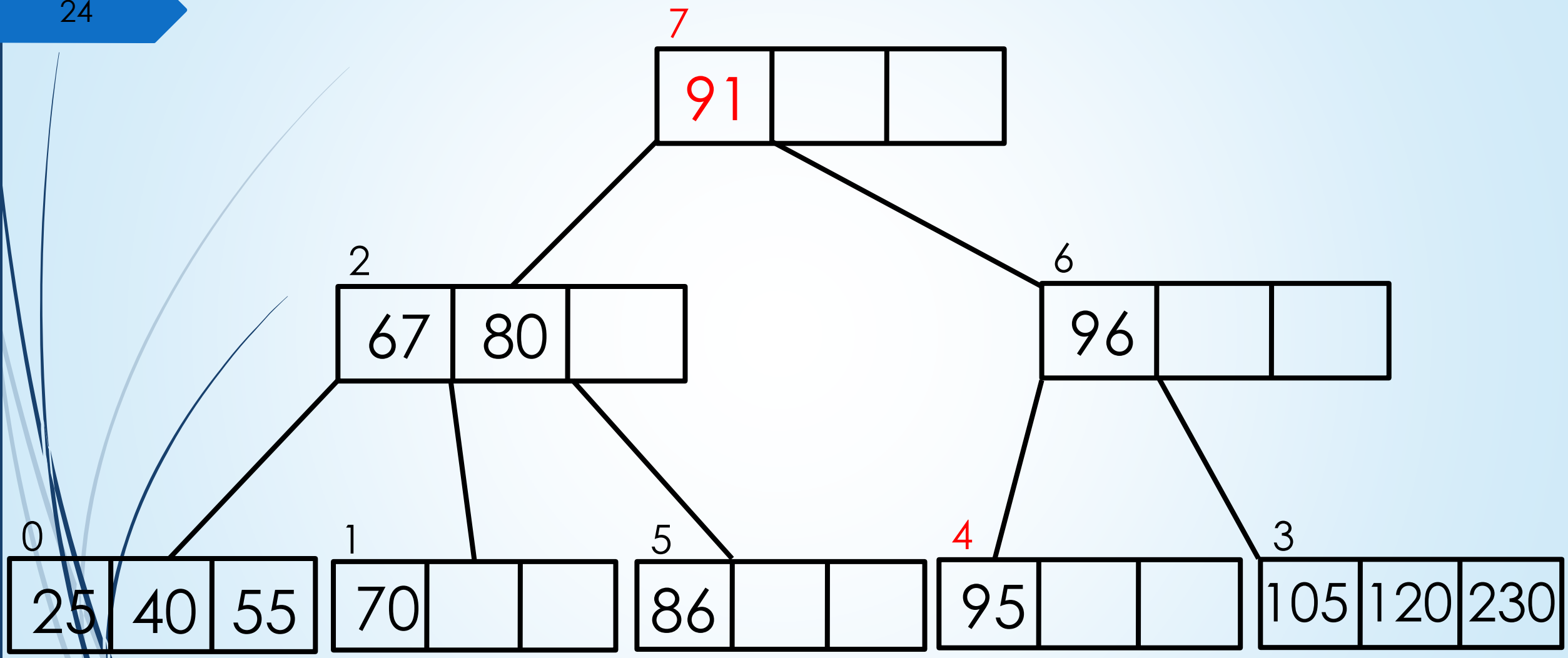


Eliminación de la clave 75 en el nodo 1.

Baja del 88, se reemplaza la clave por la menor clave del subárbol derecho.

No se genera underflow en la hoja

Ejemplo política derecha o izquierda



-88 , -70

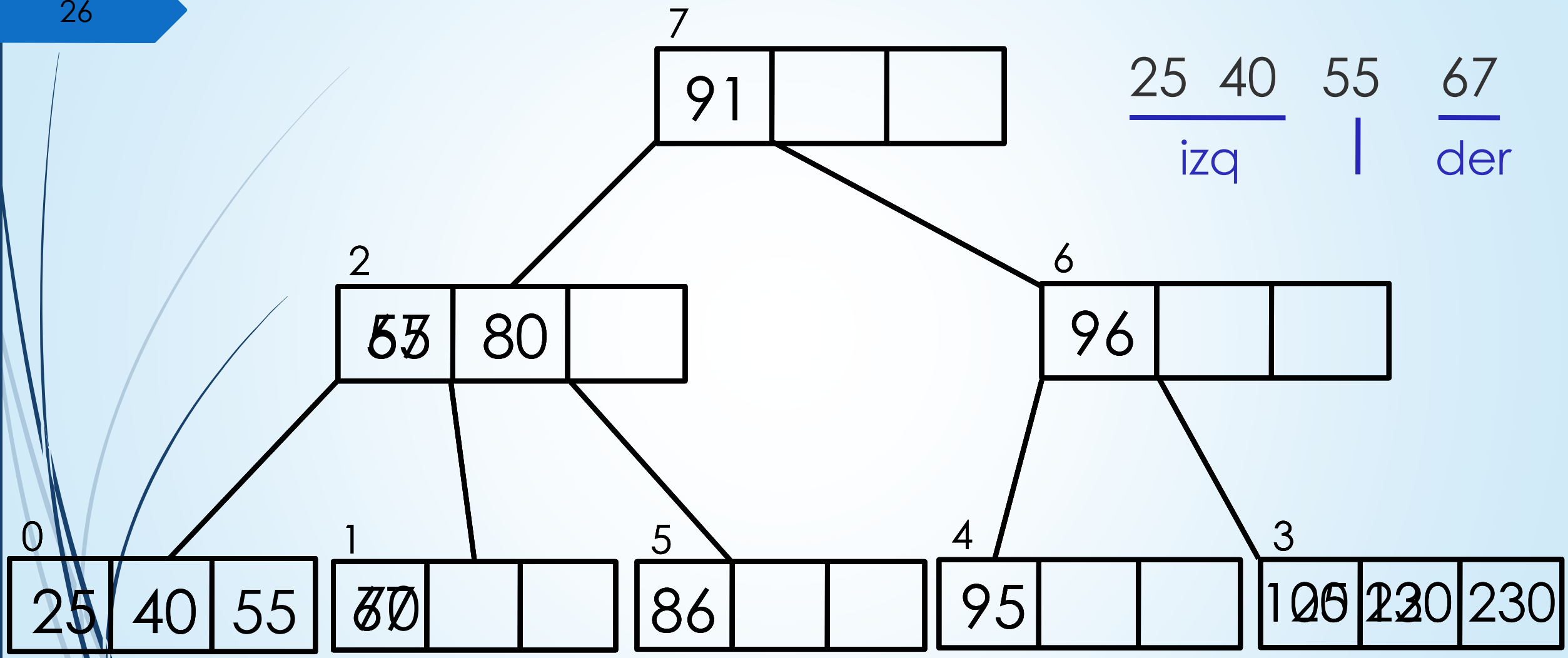
Baja de la clave 70 - política derecha o izquierda

La eliminación de la clave 70 en el nodo 1 produce underflow.

Se intenta redistribuir con el hermano derecho. No es posible ya que el nodo contiene la cantidad mínima de claves.

Se intenta redistribuir con el hermano izquierdo. La operación es posible y se rebalancea la carga entre los nodos 1 y 0.

Ejemplo - política derecha o izquierda

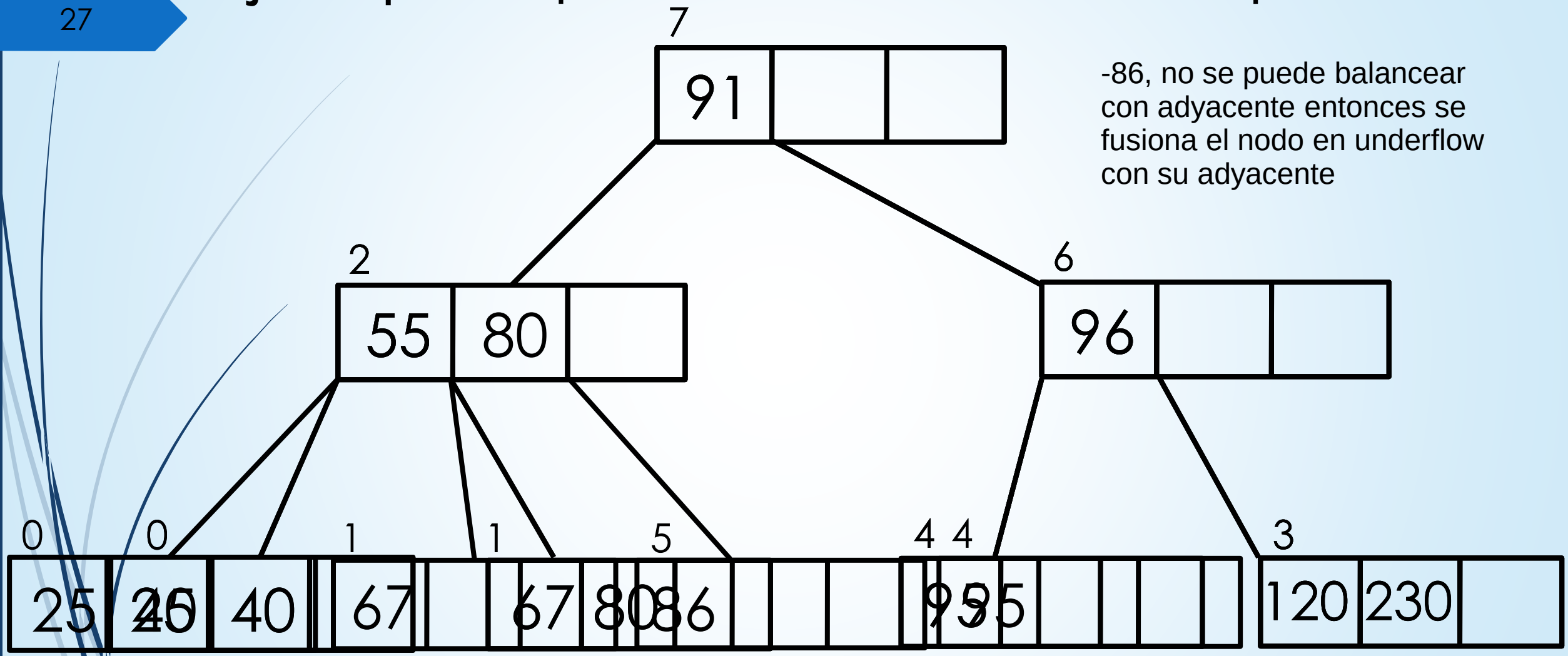


25 40 55 67

izq | der

-70 , -105, -86

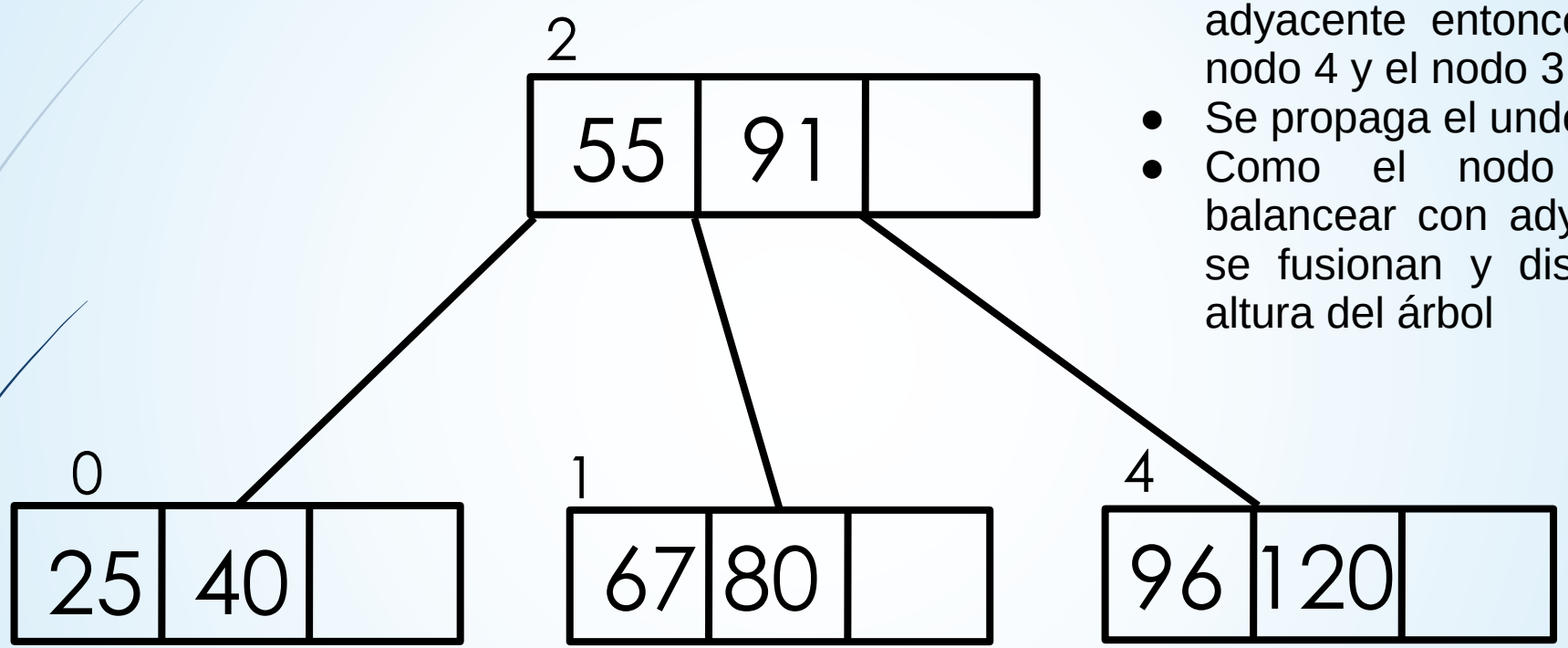
Ejemplo - política derecha o izquierda



-86, no se puede balancear con adyacente entonces se fusiona el nodo en underflow con su adyacente

-86, -230 , -95

Ejemplo - política derecha o izquierda



- -95, no se puede balancear con adyacente entonces se fusiona el nodo 4 y el nodo 3
- Se propaga el underflow al nodo 6
- Como el nodo 6 no puede balancear con adyacente (nodo 2) se fusionan y disminuye en 1 la altura del árbol

-95

Ej: Redistribución en nodo interno