Nota Cátedra: _____

Prom. Controles: _____

Universidad Católica del Norte
Facultad de Ingeniería y Ciencias Geológicas
Departamento de Ingeniería de Sistemas y Computación

ESTRUCTURA DE DATOS – CÁTEDRA 2 (I-2019)

Pauta

- Total: 240 pts (escala 60%) -

Nombre - RUT:
Fecha: 08 de junio del 2019.

Competencias a evaluar: Programación de estructuras de datos utilizando árboles.

60 pts 1. Cree un algoritmo recursivo para:
   30 a. En un ABB: Contar la cantidad de elementos mayores a X.
   30 b. En un AB: Indicar si existe algún nodo que sólo tenga un hijo izquierdo.

60 pts 2. Cree un algoritmo iterativo para recorrer un árbol binario por nivel.
   void recorridoPorNivel();

60 pts 3. En un árbol AVL mostrar gráficamente paso a paso al insertar:
   20, 30, 37, 10, 12, 29, 100, 90, 80, 85, 86, 5, 15, 13, 17, 1, 2

60 pts 4. En un árbol B+ de M=4 y M_{HOJA}=4, mostrar gráficamente paso a paso al:
   30 a. Insertar: 7, 10, 15, 40, 20, 8, 100, 74, 50, 18, 4, 61, 16, 13
   30 b. Eliminar: 10, 13, 40, 61, 100

1. a)
```
int contarMayor (Nodo p, int X) {
    if (!p) {
        return 0;
    }
    if (p->data > x) {
        return 1 + contarMayor(p->Izq, x) + contarMayor(p->Der, x);
    }else{
        return contarMayor (p->Der, x);
    }
}
```
5 pts · 5 pts · 10 pts · 5 pts · 5 pts · 30 pts

b)
```
bool buscarHIzq (Nodo p) {
    if (!p) {
        return FALSE;
    }
    if (p->Izq && !p->Der) {
        return TRUE;
    }
    return buscarHIzq (p->Izq) || buscarHIzq (p->Der);
}
```
5 pts · 5 pts · 5 pts · 15 pts · 30 pts

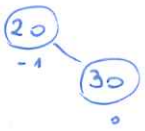
2.
```
void recorridoPorNivel () {
    Queue q;
    Nodo aux = raiz;

    while (aux) { //mientras aux no apunte a NULL
        print (aux->data);
        if (aux->Izq) {
            q.push (aux->Izq);
        }
        if (aux->Der) {
            q.push (aux->Der);
        }
        aux = q.pop();
    }
}
```
6 pts · 6 pts · 8 pts · 8 pts · 12 pts · 12 pts · 8 pts · 60 pts

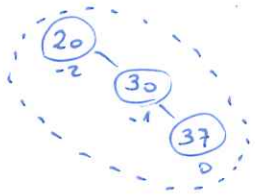
3.

+20

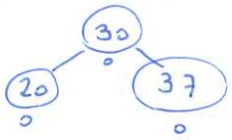
+30



+37

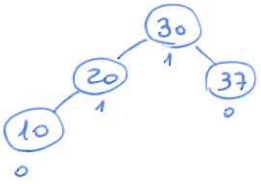


RR →

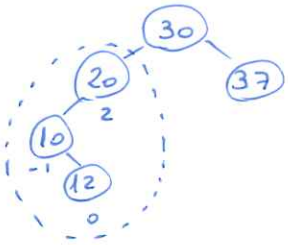


5 pts

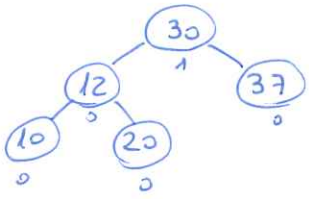
+10



+12

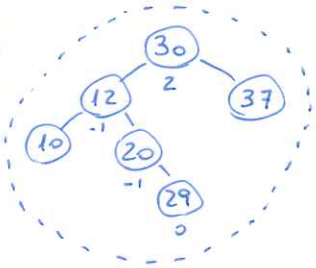


LR →

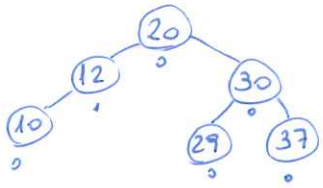


5 pts

+29

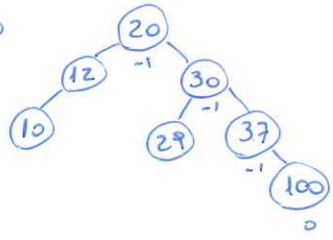


LR →

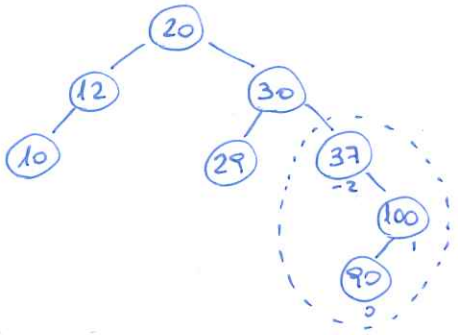


5 pts

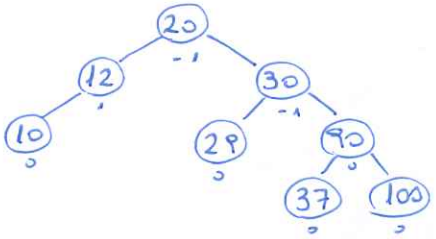
+100



+90

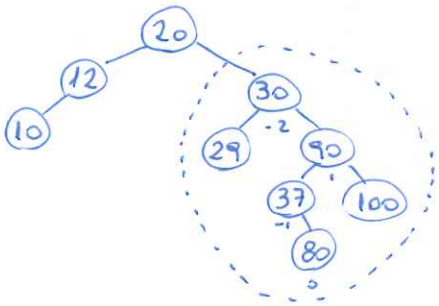


RL →

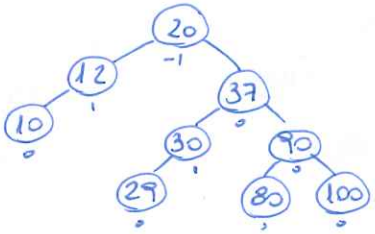


5 pts

+80

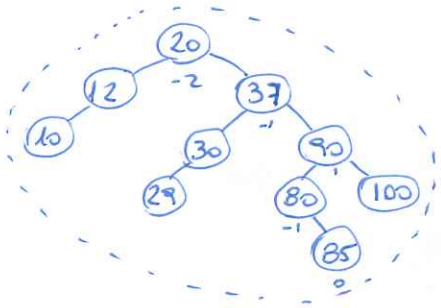


RL →

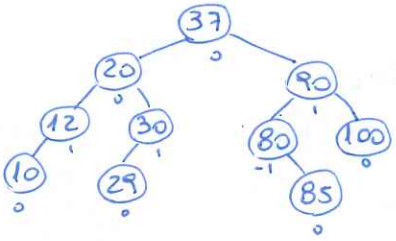


5 pts

+85

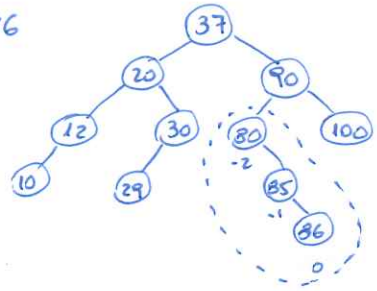


RR →

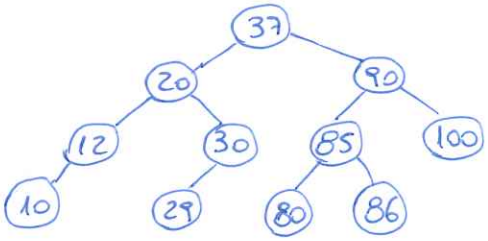


5 pts

+86

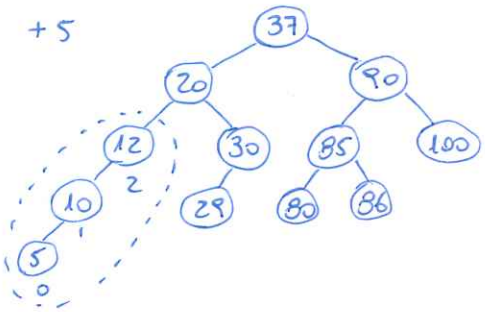


RR →

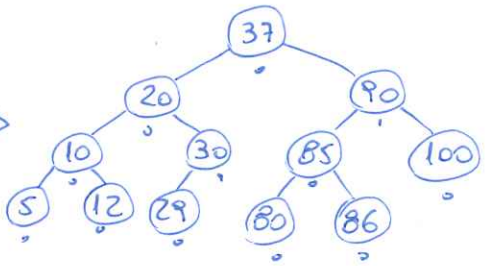


5 pts

+5

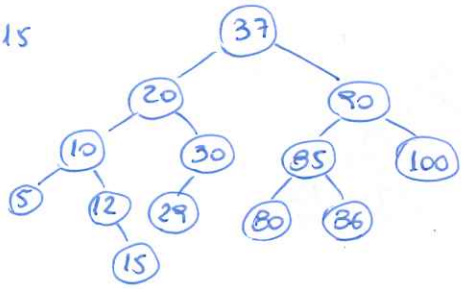


LL →

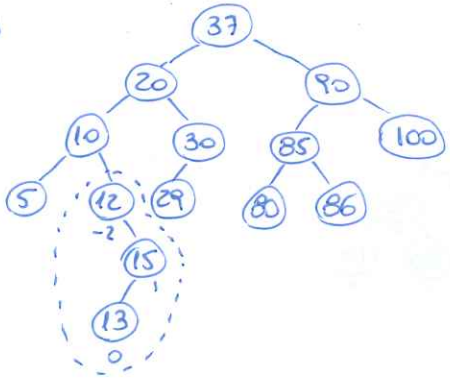


5 pts

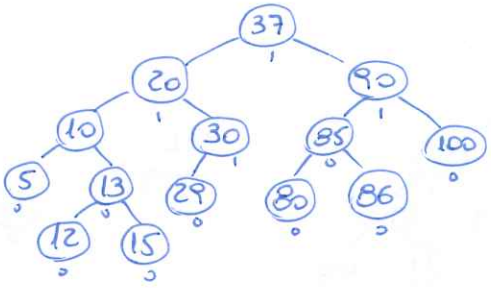
+15



+13

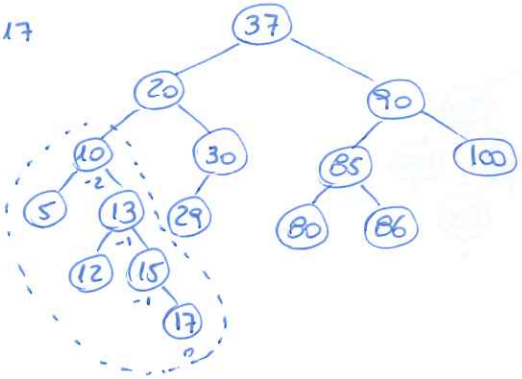


RL →

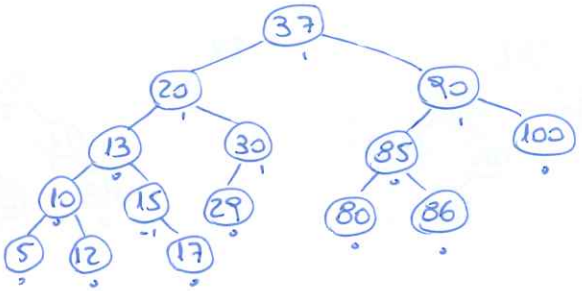


5 pts

+17

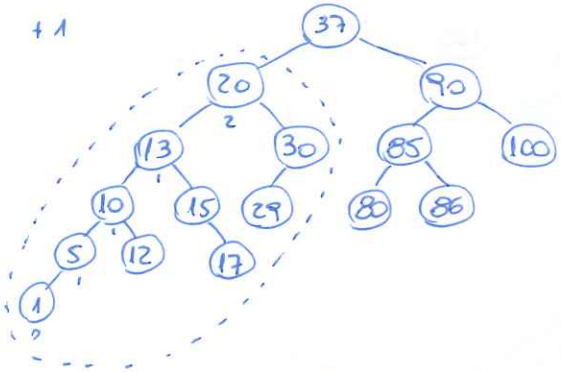


RR →

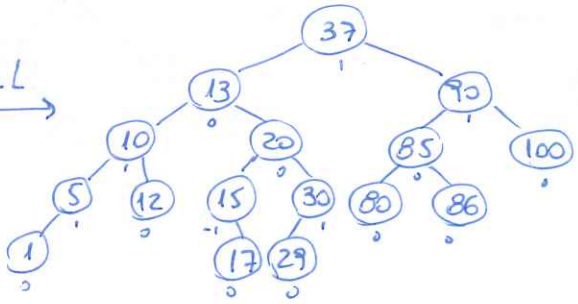


5 pts

+1

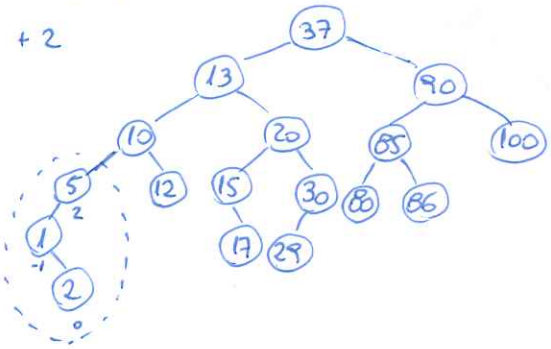


LL →

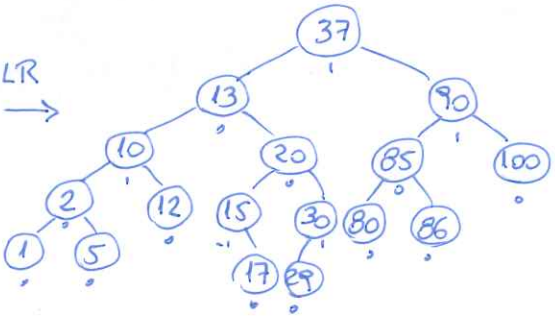


5 pts

+2



LR →



5 pts

60 pts

4. I) cant. máx. hijos = 4 => cant. máx. claves = 3

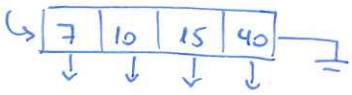
cant. min. hijos = 2 => cant. min. claves = 1

II) cant. máx. claves = 4

cant. min. claves = 2

$j = \left\lceil \frac{4+1}{2} \right\rceil = 3$

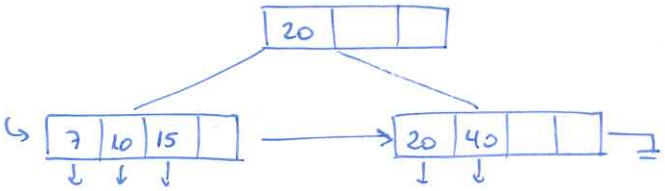
a) +7
+10
+15
+40



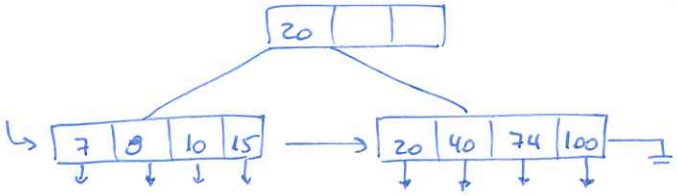
+20 overflow: "j" entradas en el nodo original

7 10 15 ‖ (20) 40 ($\uparrow j+1$)

6 pts

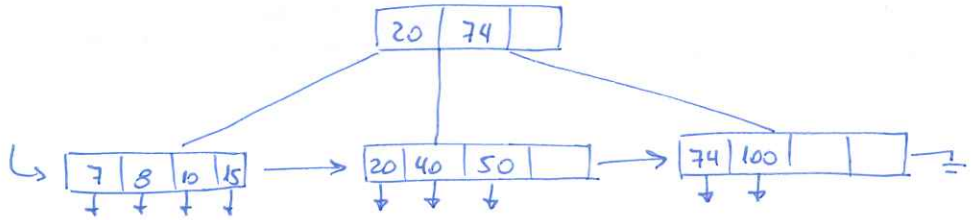


+8
+100
+74

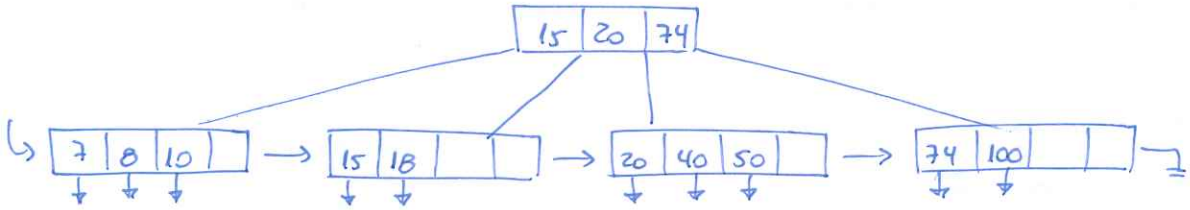


+50 overflow: 20 40 50 ‖ (74) 100

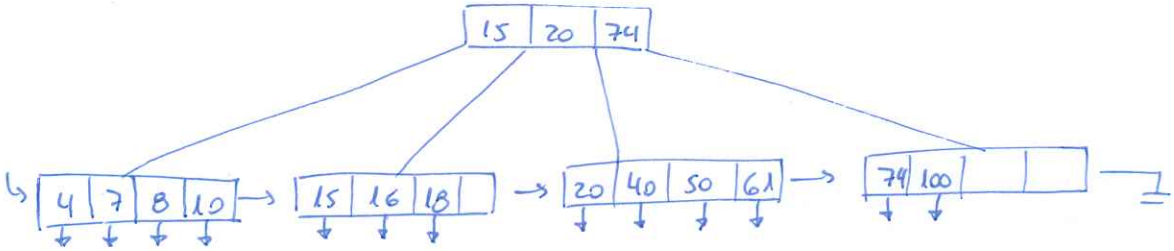
6 pts



+18 overflow: 7 8 10 ‖ (15) 18 6 pts



+4
+61
+16

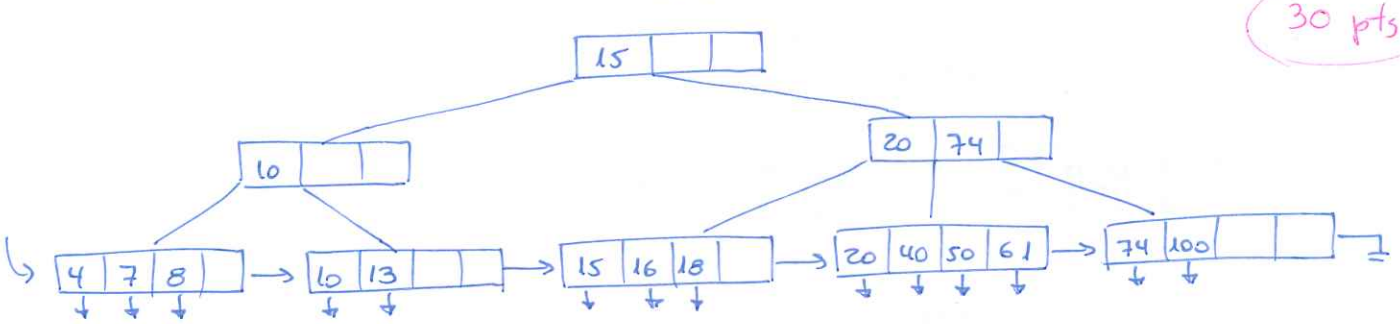


+13 overflow: 4 7 8 ‖ (10) 13 6 pts

overflow (B): 10 (15) 20 74 6 pts

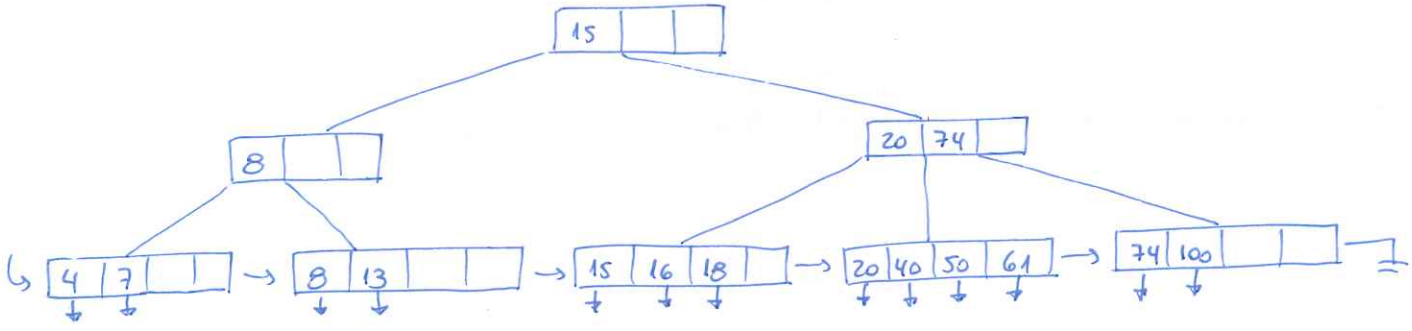
$n = 4 \Rightarrow \lceil \frac{n}{2} \rceil = 2$

30 pts



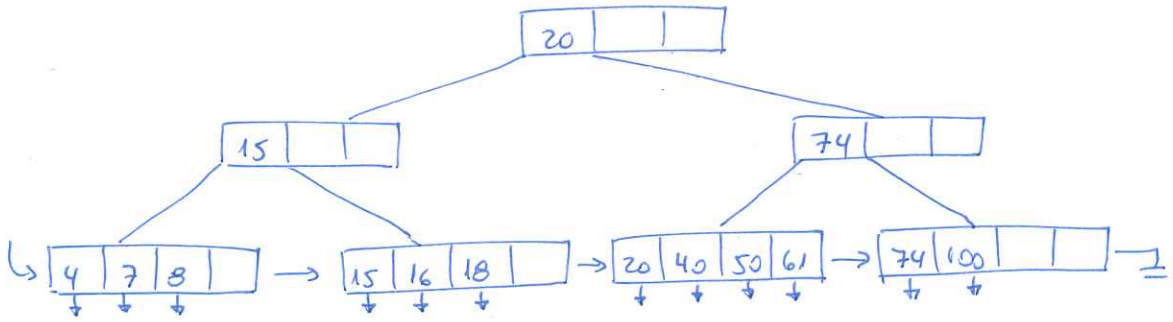
---

b) - 10 underflow: 1) Redist ✓ 6 pts

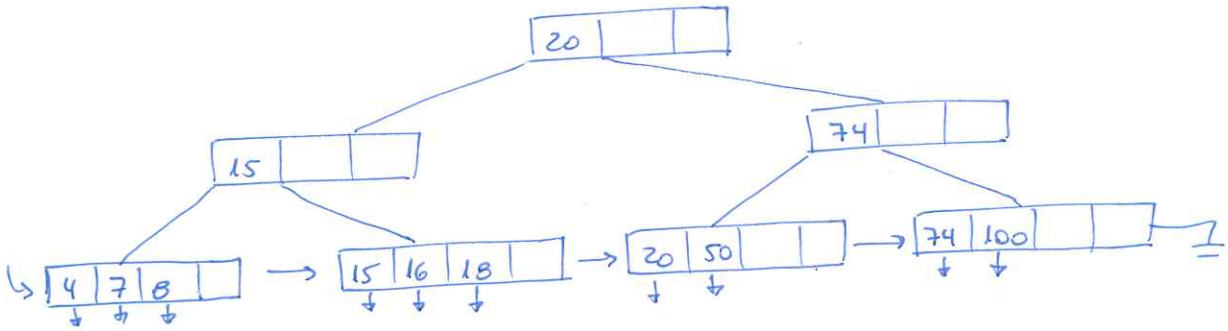


-13 underflow: 1) Redist ✗ 6 pts
2) Fusión: 4 7 8

underflow (B): 1) Redist ✓ 6 pts

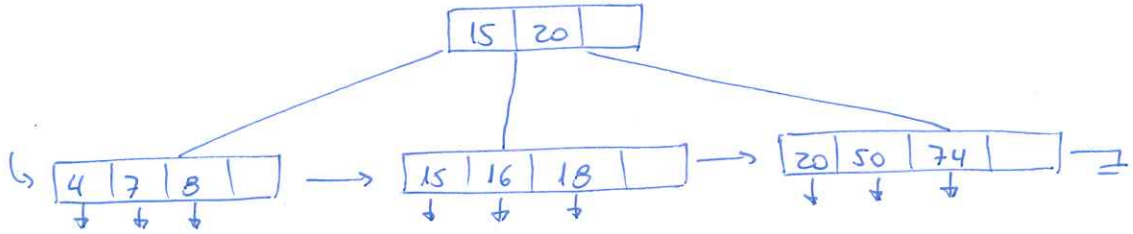


-40
-61



-100 underflow: 1) Redist ✗ 6 pts
2) Fusión: 20 50 74

underflow (B): 1) Redist ✗
2) Mezcla: 15 20 6 pts



30 pts