

Arboles

Arboles permiten acceso y actualización eficiente.

Arbol Binario:

- Se compone de un conjunto finito de elementos: nodos
- Contiene un nodo llamado raíz.
- La raíz se conecta con 2 árboles binarios: subárboles



- Los subárboles son disjuntos: que no tienen nodos en común
- Los subárboles son hijos de la raíz
- Hay una arista desde un nodo padre a un nodo hijo

Ruta: Si n_1, n_2, \dots, n_k es una secuencia de nodos en el árbol
Tal que n_i es padre de n_{i+1} para $1 \leq i < k$

↳ Esta secuencia es llamada ruta de n_1 a n_k

Ejm: $k=5 \Rightarrow n_1, n_2, n_3, n_4, n_5$

Padre	Hijo
n_1	n_2
n_2	n_3
n_3	n_4
n_4	n_5



Ruta: de n_1 a n_5

Longitud de la Ruta: $k-1$ Ejm: longitud = 4

Si hay una ruta de el nodo R al nodo M:



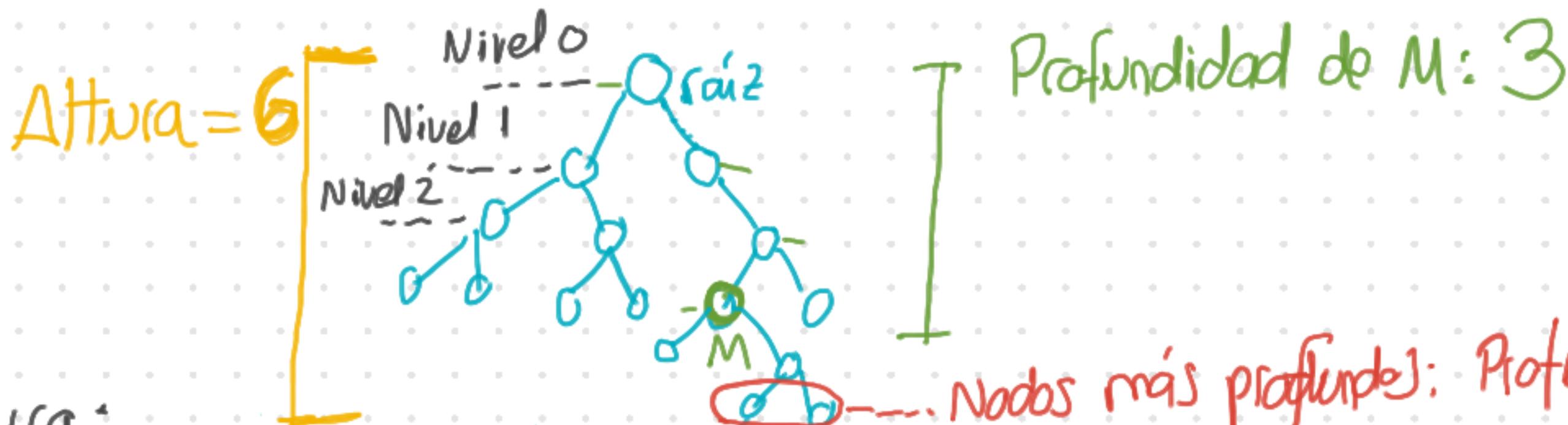
Ancestro: R es ancestro de M

Descendiente: M es descendiente de R

- La raíz es ancestro de todos los nodos del árbol

- Todos los nodos del árbol son descendientes de la raíz

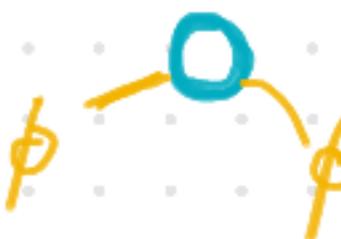
• Profundidad: de un nodo M en el árbol es la longitud de la ruta (del raíz del árbol) a M



Altura: es la profundidad del árbol con su nodo más profundo + 1

Todos los nodos de profundidad d están en nivel d en el árbol

Nodo hoja: aquel nodo que  Tiene 2 hijos vacío).



Nodo interno: cualesquier nodo que Tiene al menos un hijo vacío



Nivel 0...



Nivel 3

• Profundidad de I: 3

• La altura del árbol: 4

- B y C son hijos de A
- B y D forman un subárbol
- B tiene 2 hijos: 1 hijo vacío y a su derecha D
- A, C y E son ancestros de G,
- D, E y F hacen el nivel 2 del árbol
- Las aristas desde A - C - E - G forman una ruta de longitud 3
- D, G, H, I son nodos hojas
- A, B, C, E y F son nodos internos

Arboles diferentes:



Formas Especiales de Arboles:

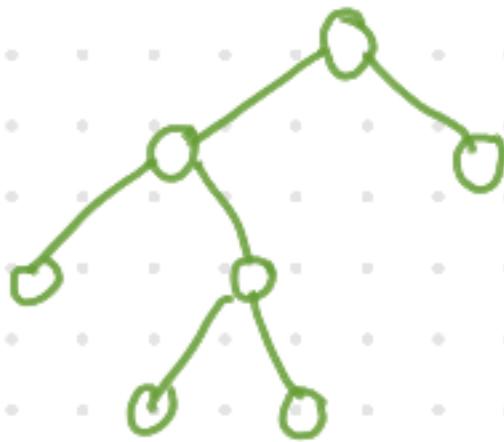
↳ Arbol lleno: es un árbol binario donde cada nodo es:

1. Un nodo interno con 2 hijos no vacíos
2. Un nodo hoja

↳ Arbol completo:

- Inicia en la raíz y se llena por niveles de izquierda a derecha
- Si tiene una altura d, todos los niveles excepto posiblemente el nivel d-1 están llenos
- El último nivel tiene sus nodos llenados de izquierda a derecha.

Arbol lleno



pero no completo

Arbol completo



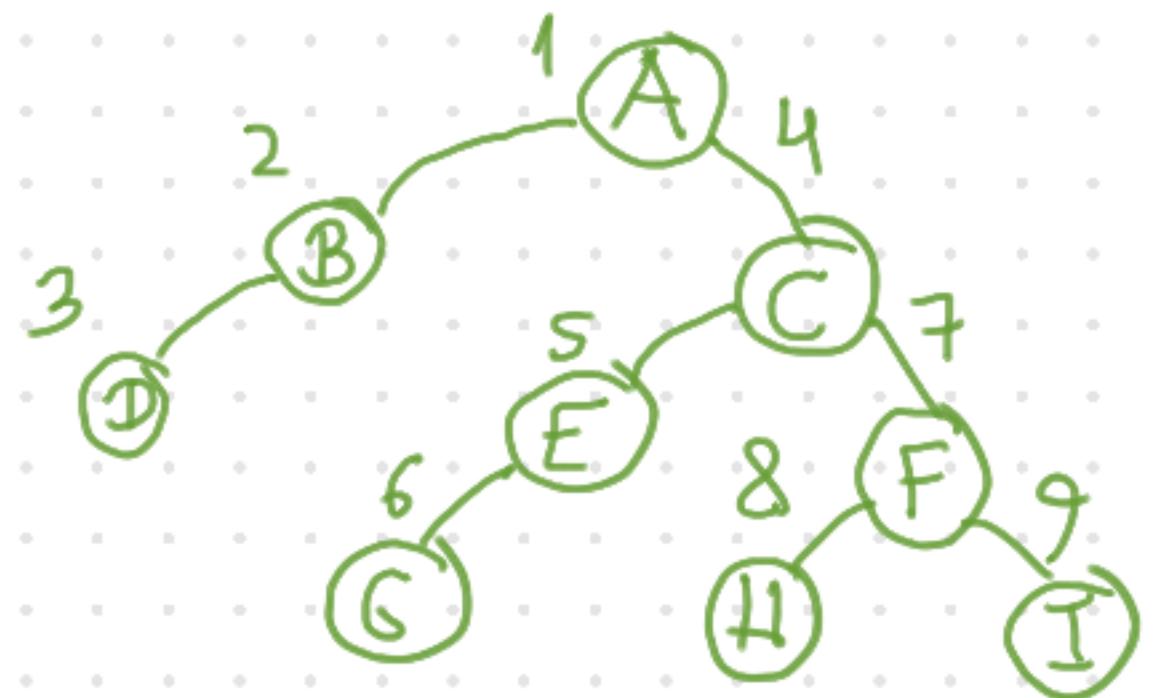
pero no lleno

Recorridos del Árbol Binario

Recorrido: proceso de visitar todos los nodos del árbol en algún orden

Enumeración: cualquier recorrido que liste cada nodo en el árbol exactamente una vez.

Recorrido Pre-orden: recorrer el árbol siguiéndolo de visita cualquier nodo antes de visitar sus hijos

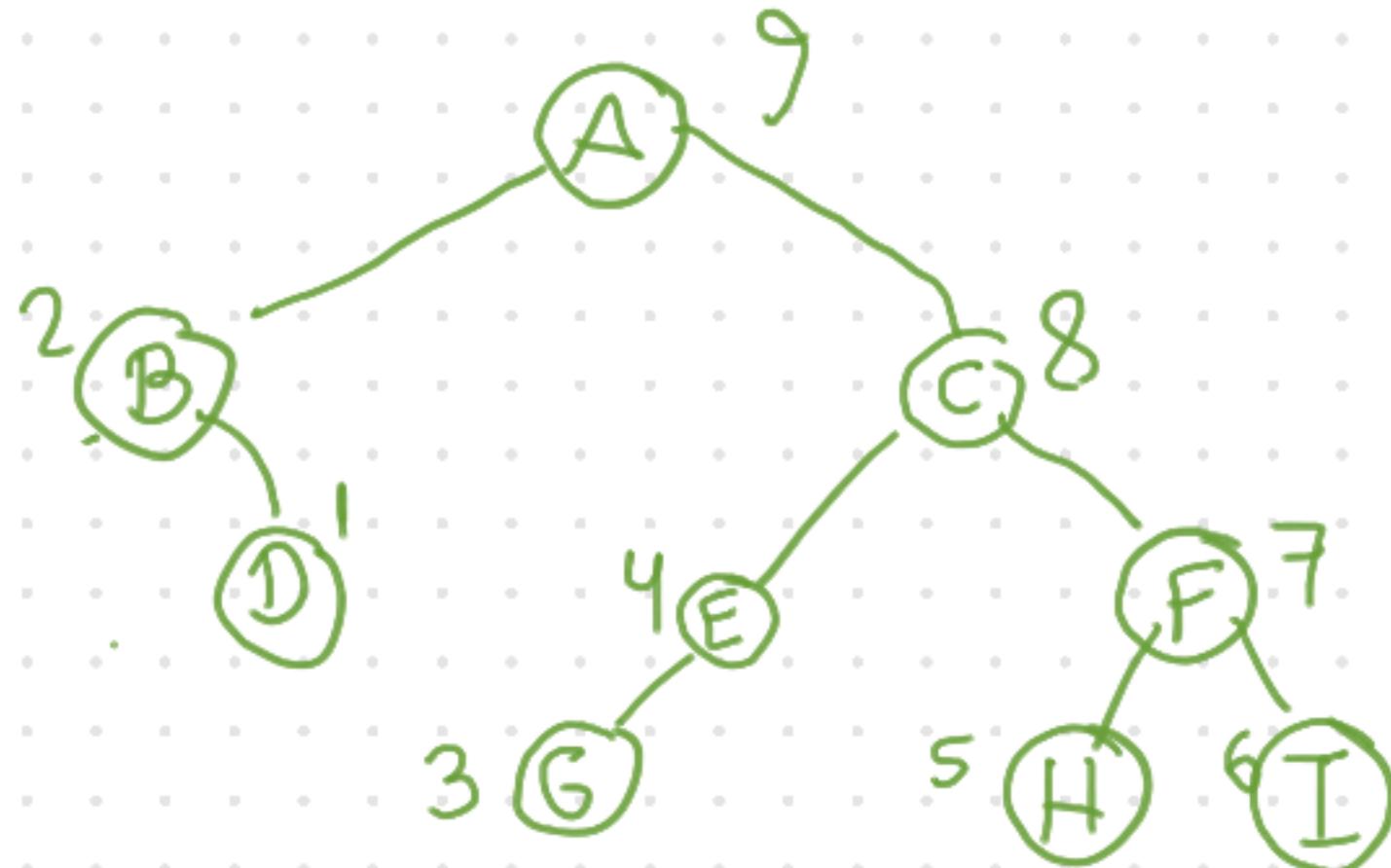


ABD CEGFHI

Recorrido pos-orden: visita cada nodo solo después de visitar a sus hijos y sus subárboles.

Útil para liberación de almacenamiento.

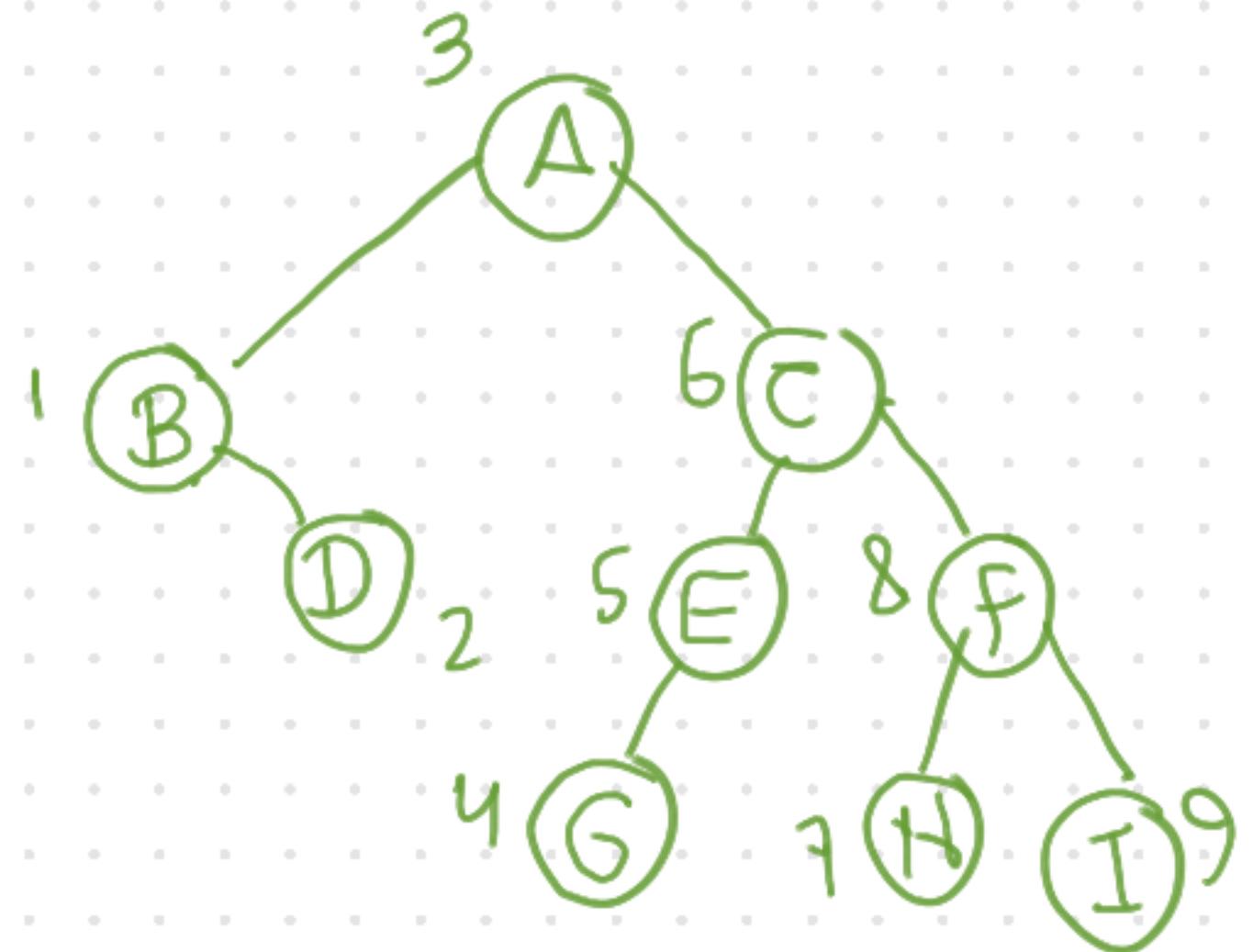
Nos gustaría eliminar el nodo hijo antes de eliminar el nodo padre.



DBGEHIFCA

Recorrido en-orden: primero visita el hijo izquierdo (incluyendo su sub-árbol entero), luego visita el nodo y finalmente visita el hijo derecho (incluyendo su sub-árbol entero).

Útil para imprimir todos los nodos en orden ascendente de valor

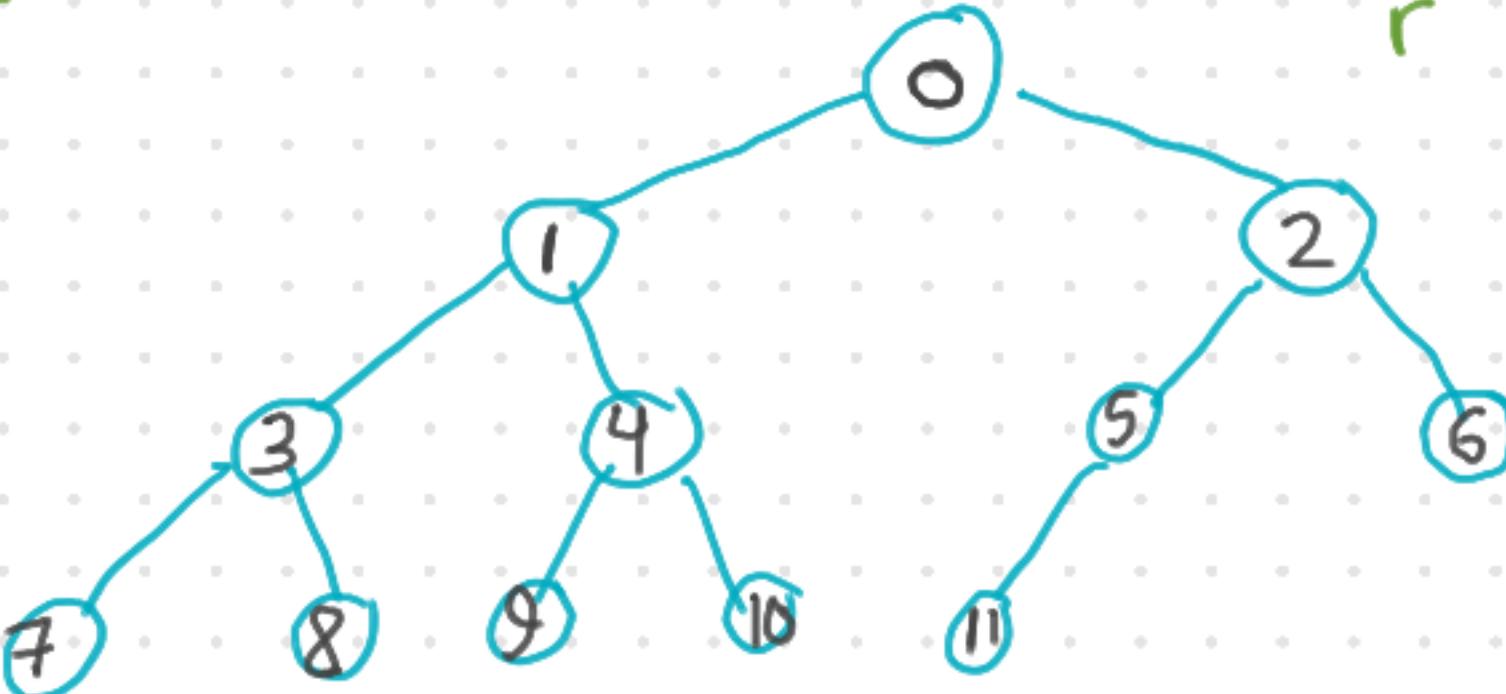


B D A G E C H F I

Implementación de Árboles basada en arreglos (Para árboles completos)

$$\lfloor 2.5 \rfloor = 2$$

floor



Indices:	Node	Hijo Izq.	Hijo Derecho	Padre
0	0			
1	1	3	5	0
2	2	4	6	1
3	3	7	8	2
4	4			2
5	5			3
6	6			3
7	7			
8	8			
9	9			
10	10			
11	11			

$$\text{Hijo Izquierdo}(r) = 2r + 1$$

$$\text{Hijo Derecho}(r) = 2r + 2$$

$$HI(0) = 2(0) + 1 = 1$$

$$HI(1) = 2(1) + 1 = 3$$

$$HI(2) = 2(2) + 1 = 5$$

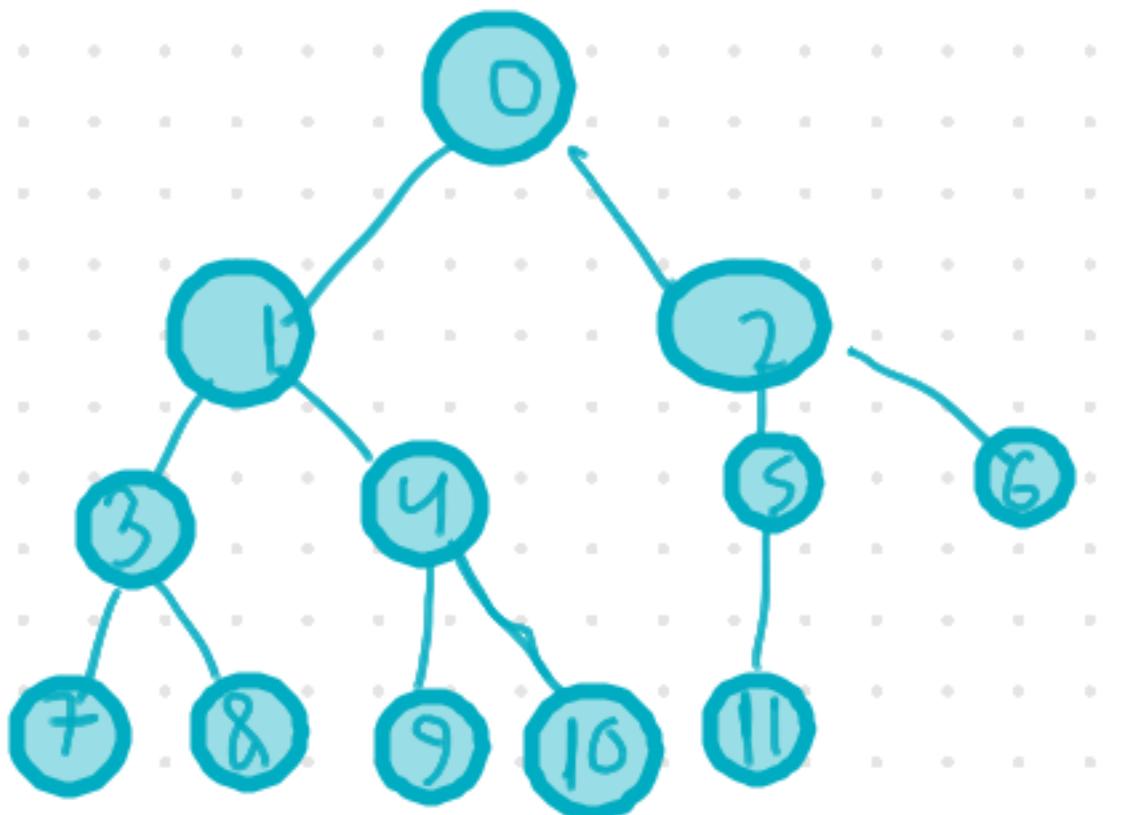
$$HD(0) = 2(0) + 2 = 2$$

$$HD(1) = 2(1) + 2 = 4$$

$$HD(2) = 2(2) + 2 = 6$$

$$\text{Padre}(r) = \lfloor (r-1)/2 \rfloor, \text{ si } r \neq 0$$

$$\text{Hermano Izquierdo}(r) =$$

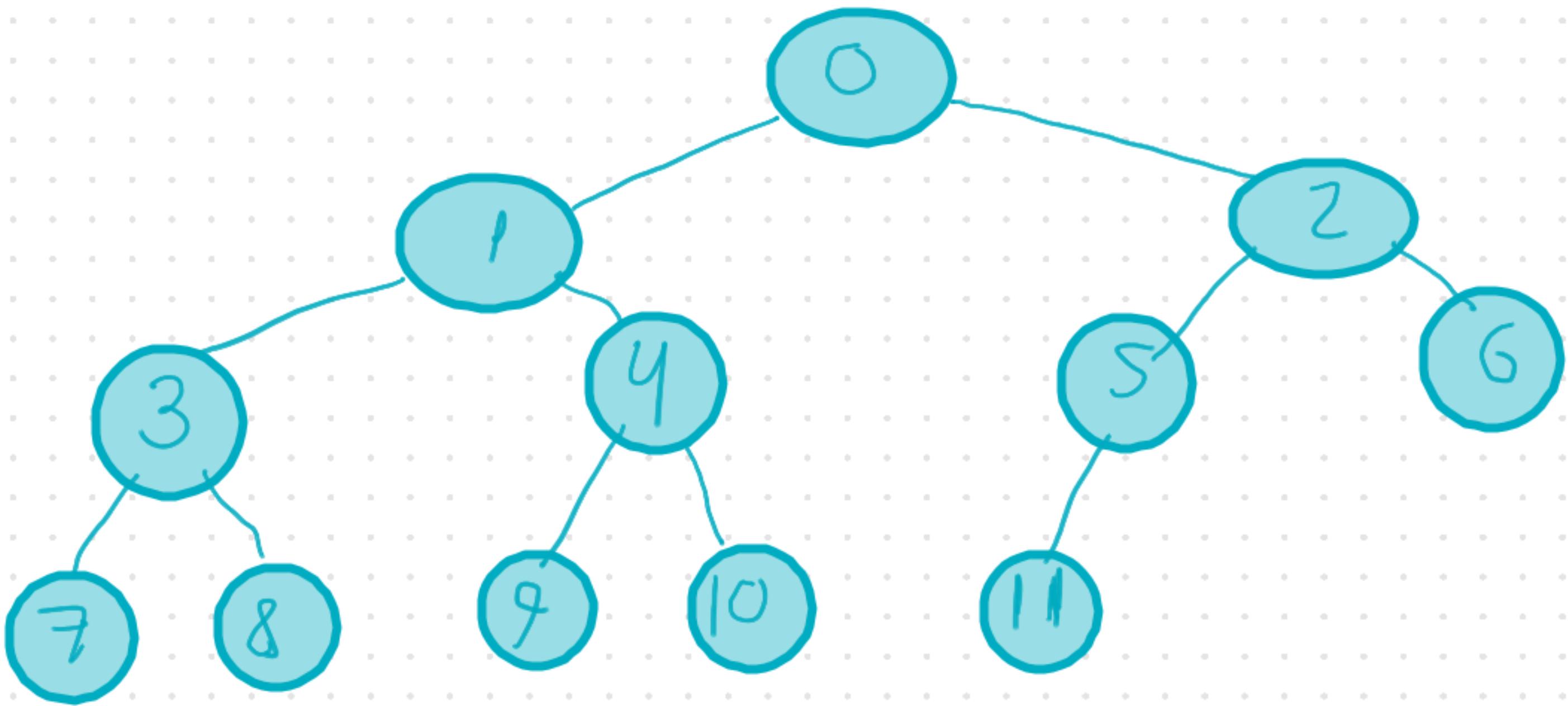


Indice Nodo	Hermano Izq.	Hermano Derecho
1	-	2
2	-	1
3	-	4

Hermano Izquierdo(r) = $r - 1$, si r es par y $r > 1$

Hermano Derecho(r) = $r + 1$, si r es impar y $r > 0$

Índice	0	1	2	3	4	5	6	7	8	9	10	11
Padre	-	0	0	1	1	2	2	3	3	4	4	5
Hijo Izq.	1	3	5	7	9	11	-	-	-	-	-	1
Hijo Der.	2	4	6	8	10	-	-	-	-	-	-	1
Hermano Izq	-	-	1	-	3	-	5	-	7	2	9	1
Hermano Der	-	2	~	X4	-	6	-	8	-	10	-	1
V.	Cáceres	E.	Prinque	D.	Gómez	G.-Díaz	Joel	Asturias	Alex	Saturno	Ricardo	Santos
									Lijan		Sakaiishi	Madrigal
											Diego	Velazquez
											D.	Guadalupe

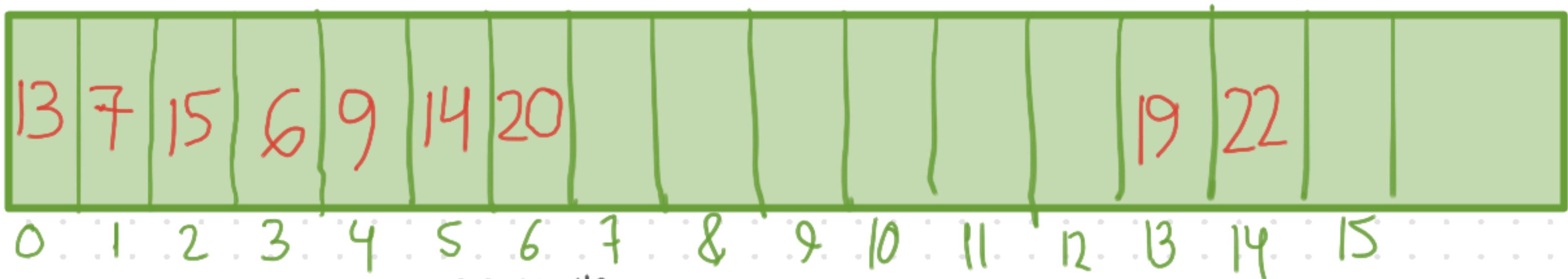


Orden de los índices en un árbol completo

Ejemplo Inserción:

Ingresar: 13, 7, 15, 20, 22, 9, 6, 14, 19 \Rightarrow 9 Valores a ingresar

Árbol \rightarrow ArrayList TamMax=15



menores que \leftarrow mayores que \rightarrow



Ingresar: 7, 15, 20, 13, 22, 9, 6, 14, 19
 \hookrightarrow Genera el árbol

Ejemplo Búsqueda:

Buscar: 19

$\hookrightarrow 19 = \textcircled{13} \Rightarrow F$

$\hookrightarrow 19 > 13 \Rightarrow V$

$\hookrightarrow 19 = \textcircled{15} \Rightarrow F$

$\hookrightarrow 19 > 15 \Rightarrow V$

$\hookrightarrow 19 = \textcircled{20} \Rightarrow F$

$\hookrightarrow 19 > 20 \Rightarrow F$

$\hookrightarrow 19 = \textcircled{19} \Rightarrow V$

7 comparaciones

Se comparó con 4 nodos

1 Buscar: 22

$\hookrightarrow 22 = \textcircled{13} F$

$\hookrightarrow 22 > 13 V$

$22 = \textcircled{15} F$

$22 > 15 V$

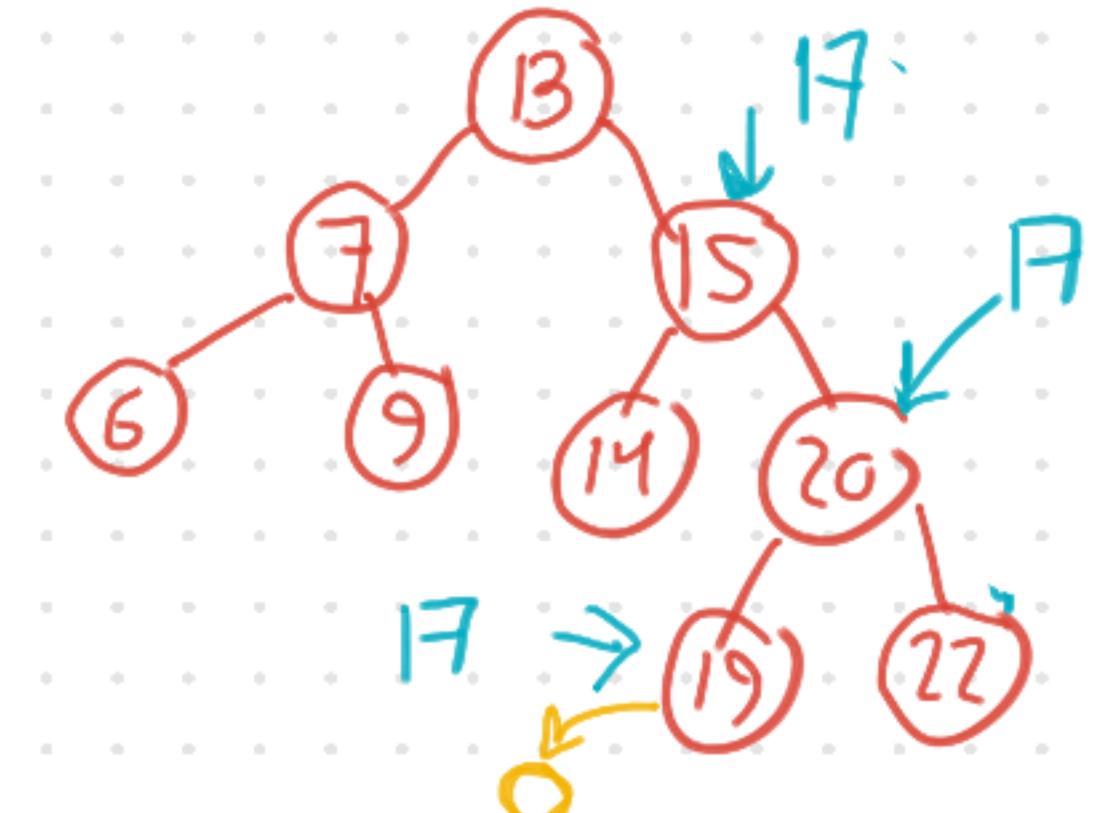
$22 = \textcircled{20} F$

$22 > 20 V$

$22 = \textcircled{22} V$

Se comparó con 4 nodos

7 comparaciones



Buscar: 17

$\hookrightarrow 17 = 13 F$

$17 > 13 V$

$17 = \textcircled{15} F$

$17 > 15 V$

$17 = \textcircled{20} F$

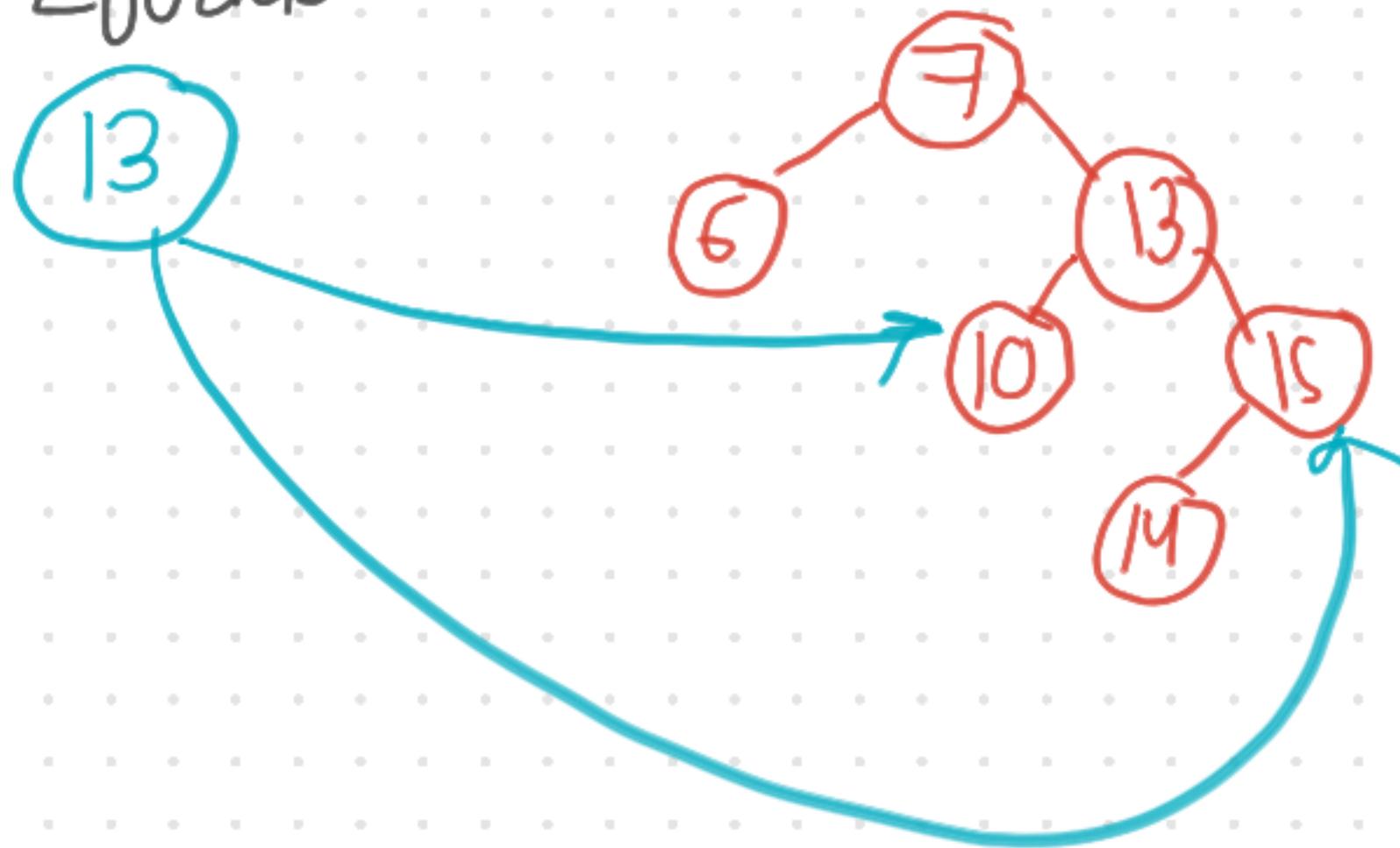
$17 > 20 f$

$17 = \textcircled{19} F$

$17 > 19 F$

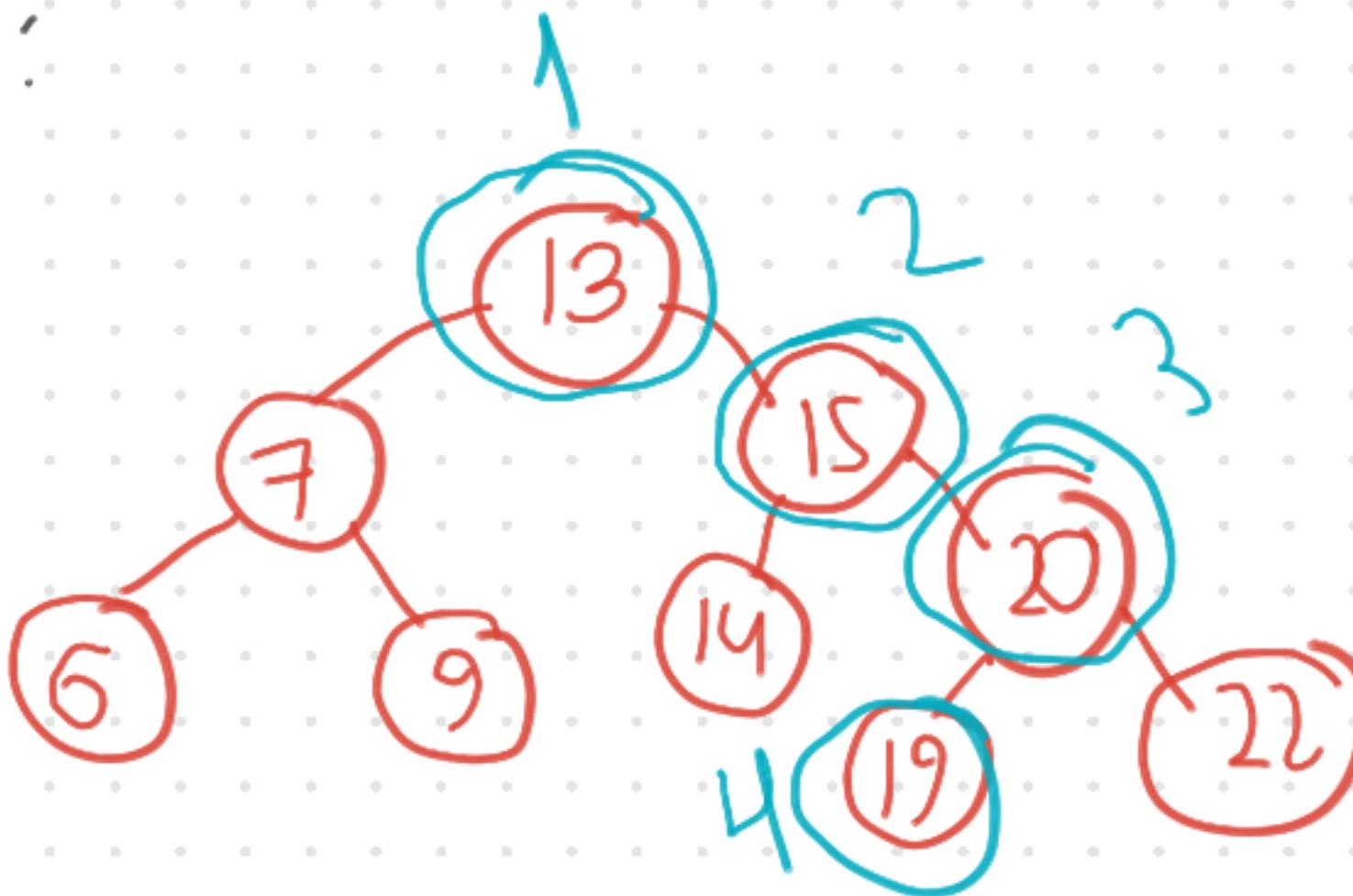
No se encontró 17

Ejercicio 1:



Ejercicio 2:

19

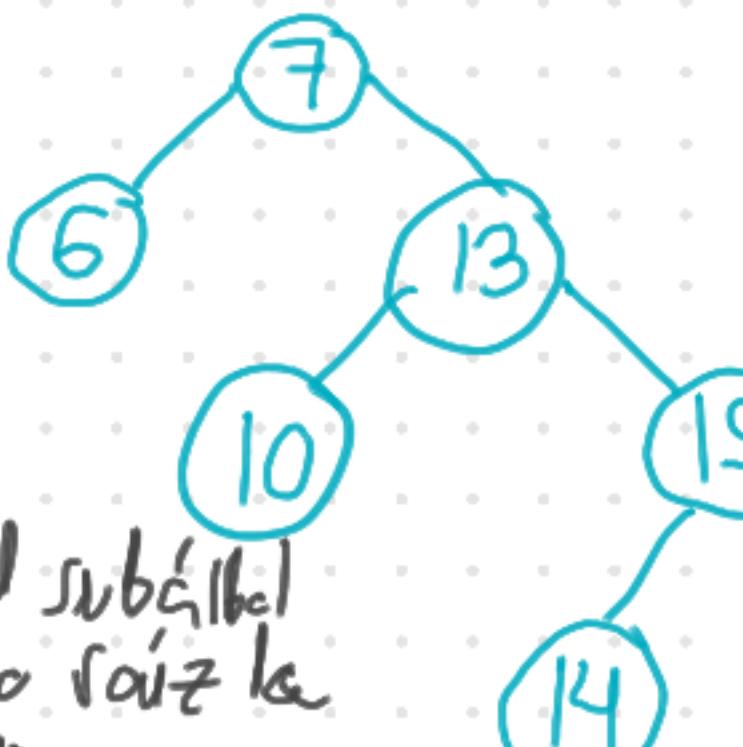


4 composaciones

Eliminación: 13

Eliminar nodo hoja: sencilla

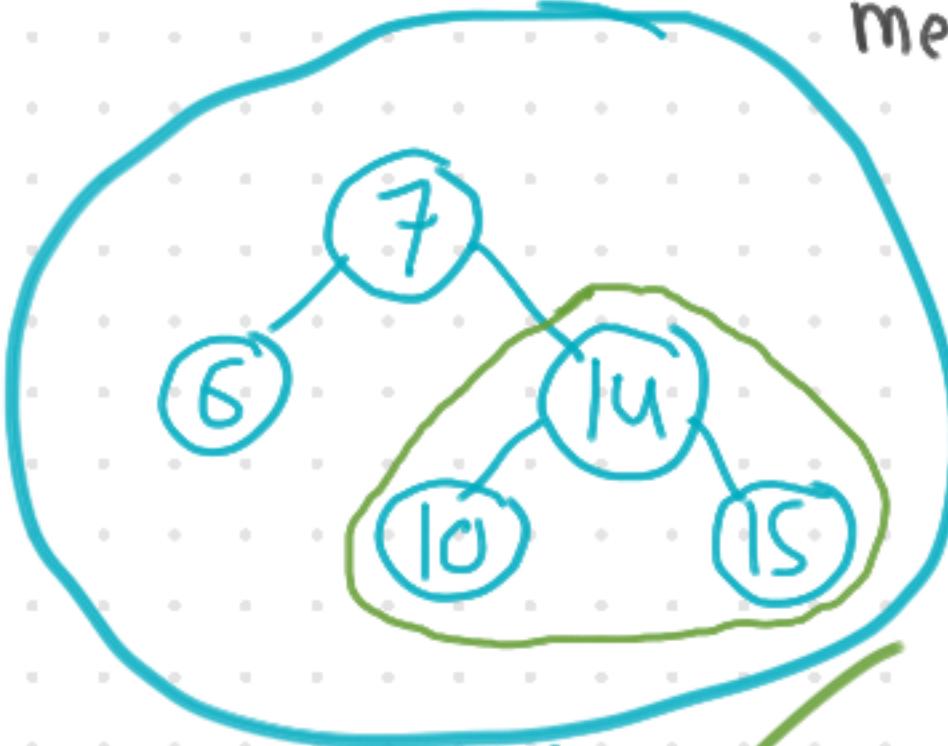
Eliminar nodo intermedio: complejo



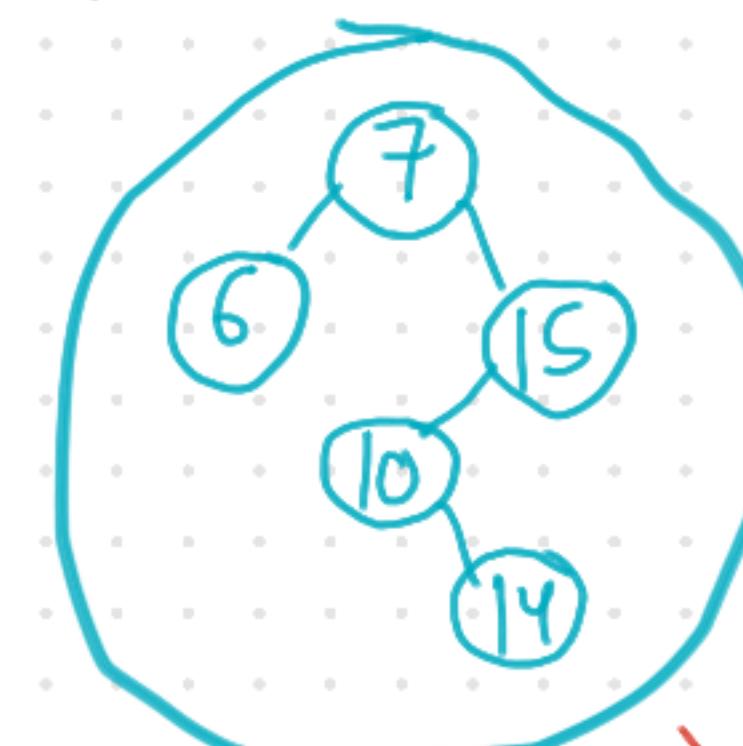
10, 12, 14, 15
↑ Hay 2 medianas: elegir la de la izquierda



(2) Reconstruir el subárbol
Tomando como raíz la mediana



Opc 3. ✓



Opc 2 X

Candidatos:
10, 14, 15
mediana (1)
Hallar la mediana
con todos los elementos
del subárbol

Tarea:

1. Implementación del algoritmo de eliminación
 2. Recorrido pre-orden
 3. Recorrido pos-orden
 4. Recorrido in-orden
-

Recorrido pre-orden

ABD

Si árbol está vacío \rightarrow retornar

Visitar raíz // cout << raiz \rightarrow void;

pre-orden (raiz \rightarrow izquierdo)

pre-orden (raiz \rightarrow derecha)

Ejm: pre-orden ('A')

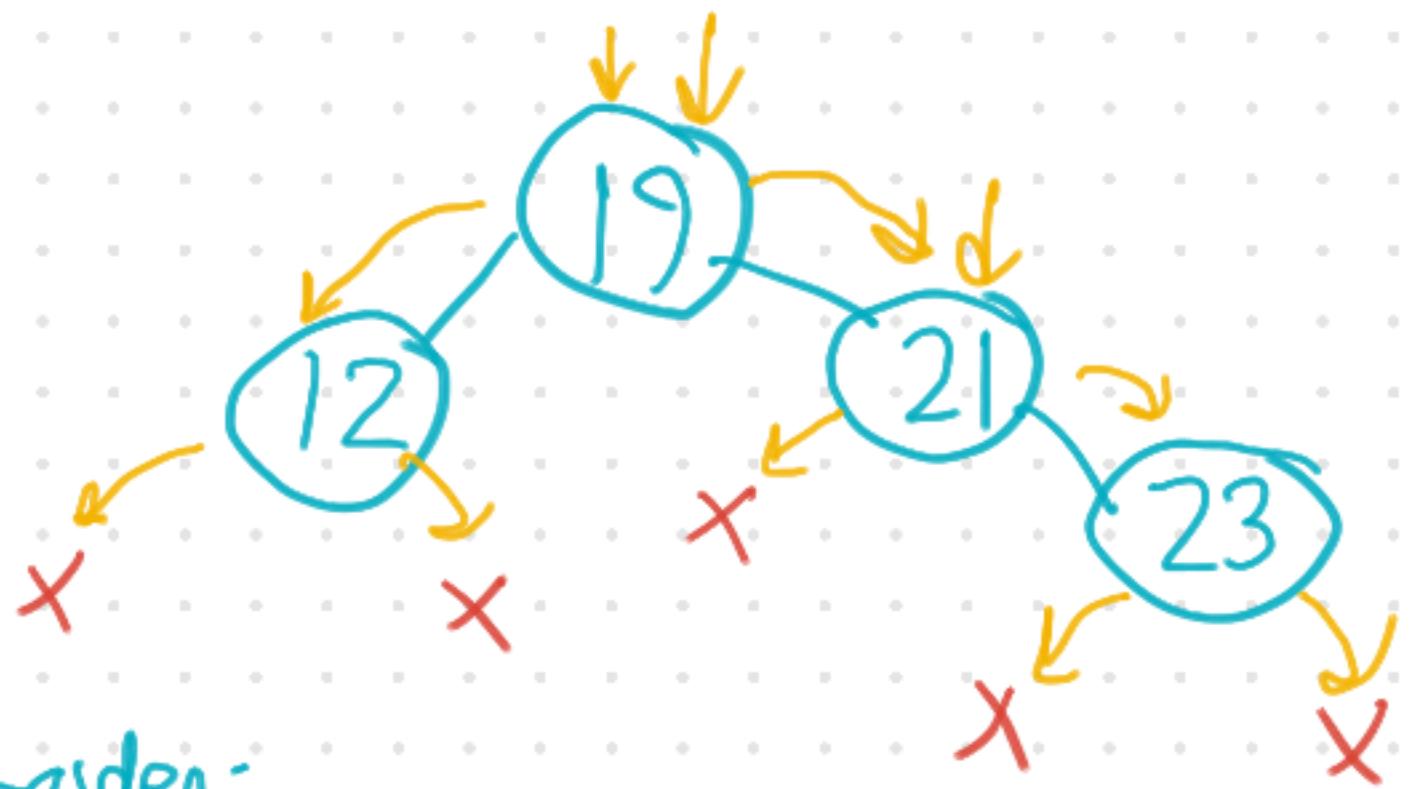
\hookrightarrow pre-orden ('B')

\hookrightarrow pre-orden ('D')

\hookrightarrow pre-orden ('C')



Pre-orden: ABD CEG FHI



In-order:

1º hijo izq.

2º nodo actual

3º hijo derecho

Elim: 21



