

Árboles Binarios

Agenda

- Definición
- Descripción y terminología
- Representaciones
- Recorridos
- Aplicación: Árboles de expresión

Árbol Binario: Definición

➤ *Un árbol binario es una colección de nodos, tal que:*

- *puede estar vacía*
- *puede estar formada por un nodo distinguido R , llamado **raíz** y dos sub-árboles T_1 y T_2 , donde la raíz de cada subárbol T_i está conectado a R por medio de una arista*

Descripción y terminología

- Cada nodo puede tener a lo sumo dos nodos hijos.
- Cuando un nodo no tiene ningún hijo se denomina *hoja*.
- Los nodos que tienen el mismo nodo padre se denominan *hermanos*.

Descripción y terminología

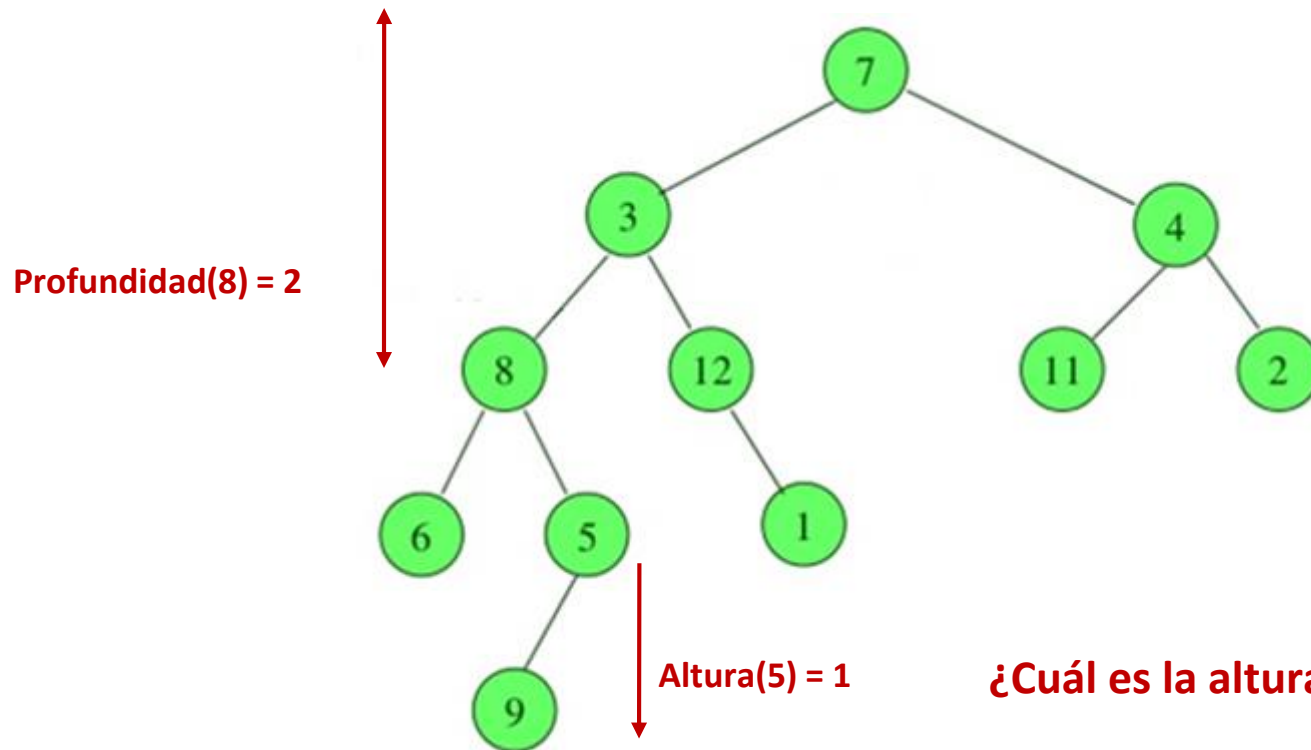
➤ Conceptos a usar:

- ***Camino***: desde n_1 hasta n_k , es una secuencia de nodos n_1, n_2, \dots, n_k tal que n_i es el padre de n_{i+1} , para $1 \leq i < k$.
 - La longitud del camino es el número de aristas, es decir $k-1$.
 - Existe un camino de longitud cero desde cada nodo a sí mismo.
 - Existe un único camino desde la raíz a cada nodo.
- ***Profundidad***: de n_i es la longitud del único camino desde la raíz hasta n_i .
 - La raíz tiene profundidad cero.

Descripción y terminología

- *Grado* de n_i es el número de hijos del nodo n_i .
- *Altura* de n_i es la longitud del camino más largo desde n_i hasta una hoja.
 - Las hojas tienen altura cero.
 - La altura de un árbol es la altura del nodo raíz.
- *Ancestro/Descendiente*: si existe un camino desde n_1 a n_2 , se dice que n_1 es ancestro de n_2 y n_2 es descendiente de n_1 .

Descripción y terminología

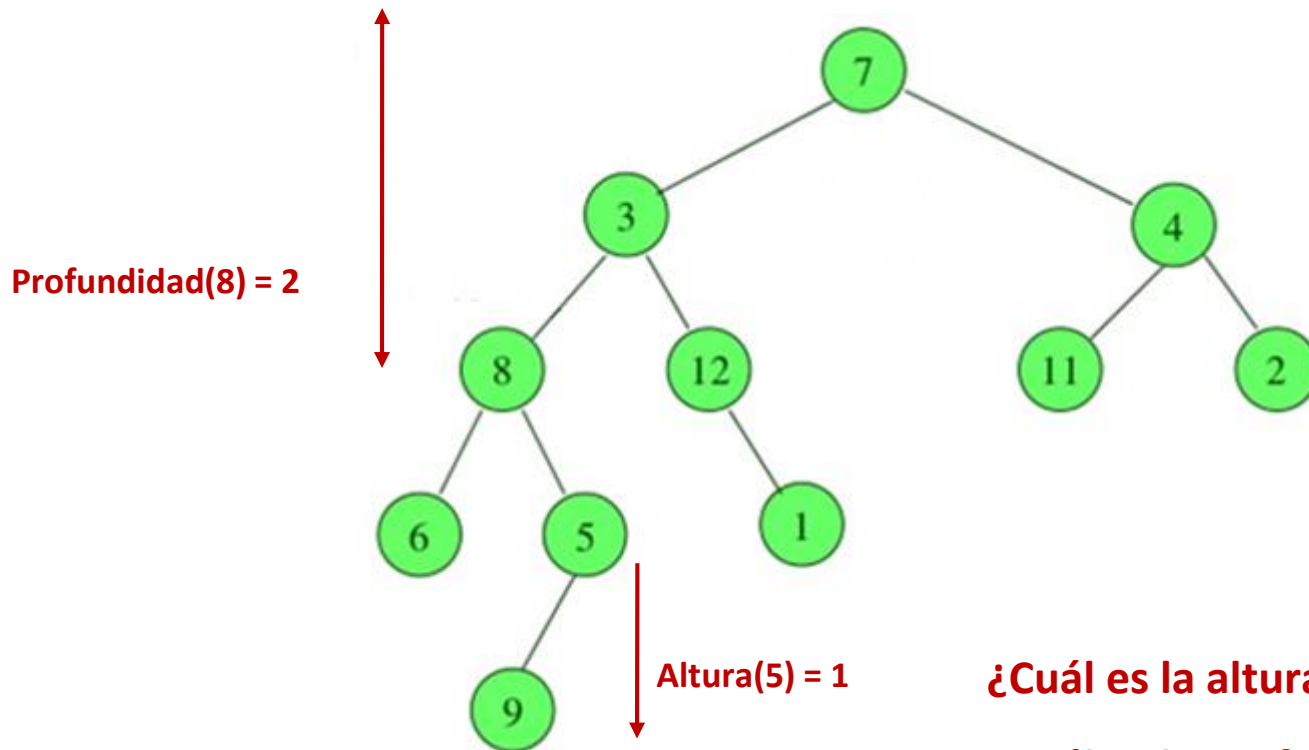


¿Cuál es la altura del nodo 8?

¿Cuál es la profundidad del nodo 12?

¿Cuáles son los ancestros del nodo 11?

Descripción y terminología



¿Cuál es la altura del nodo 8? 2

¿Cuál es la profundidad del nodo 12? 2

¿Cuáles son los ancestros del nodo 11? 7 y 4

Descripción y terminología

- *Árbol binario lleno*: Dado un árbol binario T de altura h , diremos que T es *lleno* si cada nodo interno tiene grado 2 y todas las hojas están en el mismo nivel (h).

Es decir, recursivamente, T es *lleno* si :

- 1.- T es un nodo simple (árbol binario lleno de altura 0), o
- 2.- T es de altura h y sus sub-árboles son llenos de altura $h-1$.

Descripción y terminología

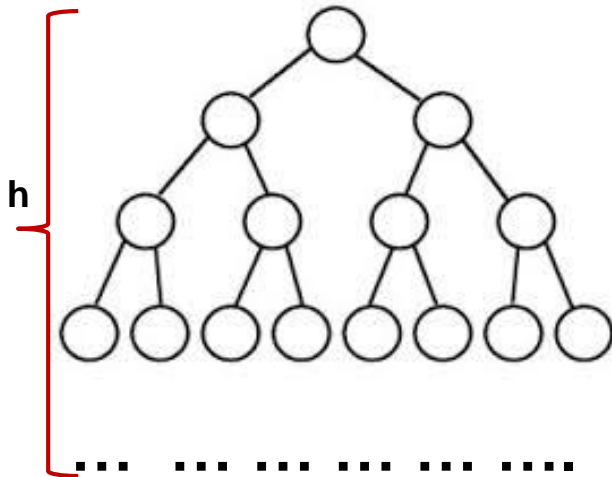
- *Cantidad de nodos en un árbol binario lleno:*

Sea T un árbol binario lleno de altura h , la cantidad de nodos N es $(2^{h+1} - 1)$

Descripción y terminología

- *Cantidad de nodos en un árbol binario lleno:*

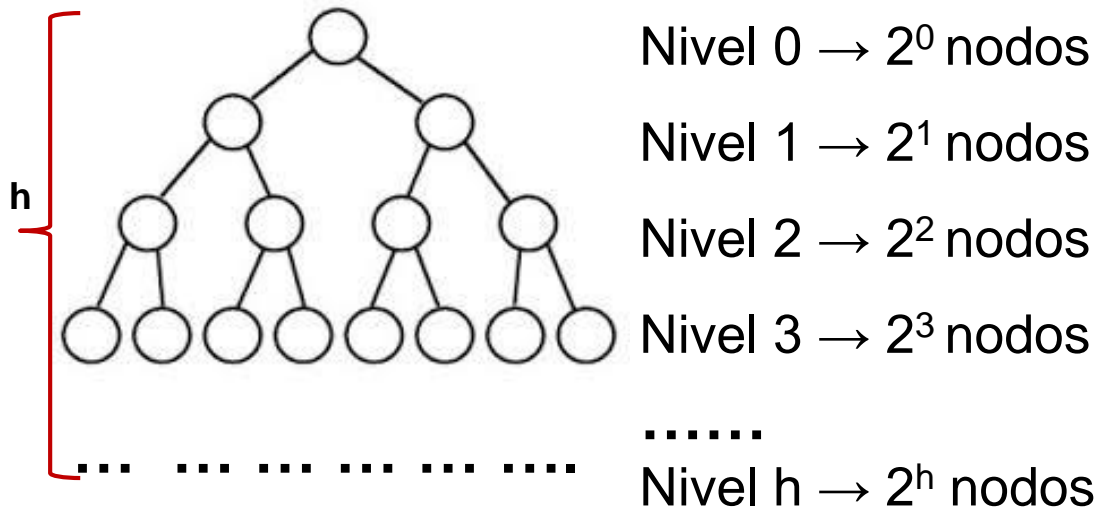
Sea T un árbol binario lleno de altura h , la cantidad de nodos N es $(2^{h+1} - 1)$



Descripción y terminología

- Cantidad de nodos en un árbol binario lleno:*

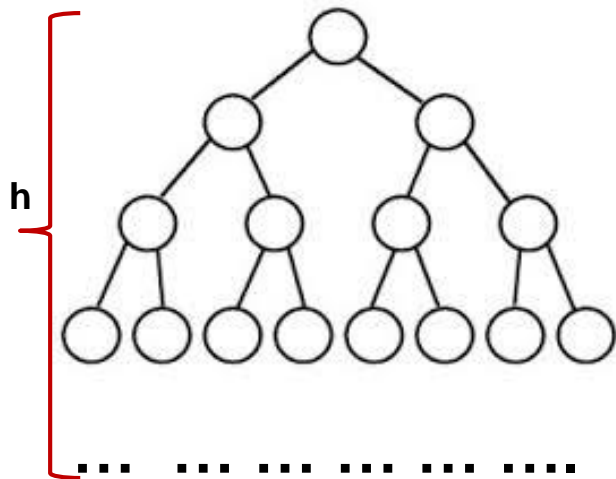
Sea T un árbol binario lleno de altura h , la cantidad de nodos N es $(2^{h+1} - 1)$



Descripción y terminología

- Cantidad de nodos en un árbol binario lleno:*

Sea T un árbol binario lleno de altura h , la cantidad de nodos N es $(2^{h+1} - 1)$



Nivel 0 $\rightarrow 2^0$ nodos

Nivel 1 $\rightarrow 2^1$ nodos

Nivel 2 $\rightarrow 2^2$ nodos

Nivel 3 $\rightarrow 2^3$ nodos

.....

Nivel $h \rightarrow 2^h$ nodos

$$N = 2^0 + 2^1 + 2^2 + 2^3 + \dots + 2^h$$

La suma de los términos de una serie geométrica de razón 2 es:

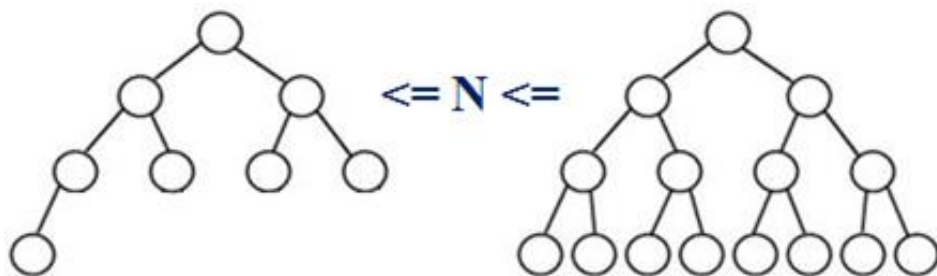
$$(2^{h+1} - 1)$$

Descripción y terminología

- *Árbol binario completo*: Dado un árbol binario T de altura h , diremos que T es completo si es lleno de altura $h-1$ y el nivel h se completa de izquierda a derecha.
- *Cantidad de nodos en un árbol binario completo*:
Sea T un árbol binario completo de altura h , la cantidad de nodos N varía entre (2^h) y $(2^{h+1} - 1)$

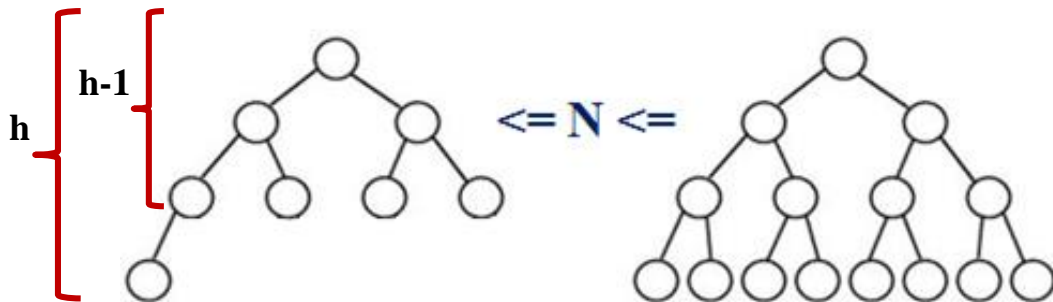
Descripción y terminología

- *Árbol binario completo*: Dado un árbol binario T de altura h , diremos que T es completo si es lleno de altura $h-1$ y el nivel h se completa de izquierda a derecha.
- *Cantidad de nodos en un árbol binario completo*:
Sea T un árbol binario completo de altura h , la cantidad de nodos N varía entre (2^h) y $(2^{h+1} - 1)$



Descripción y terminología

- *Árbol binario completo*: Dado un árbol binario T de altura h , diremos que T es completo si es lleno de altura $h-1$ y el nivel h se completa de izquierda a derecha.
- *Cantidad de nodos en un árbol binario completo*:
Sea T un árbol binario completo de altura h , la cantidad de nodos N varía entre (2^h) y $(2^{h+1} - 1)$

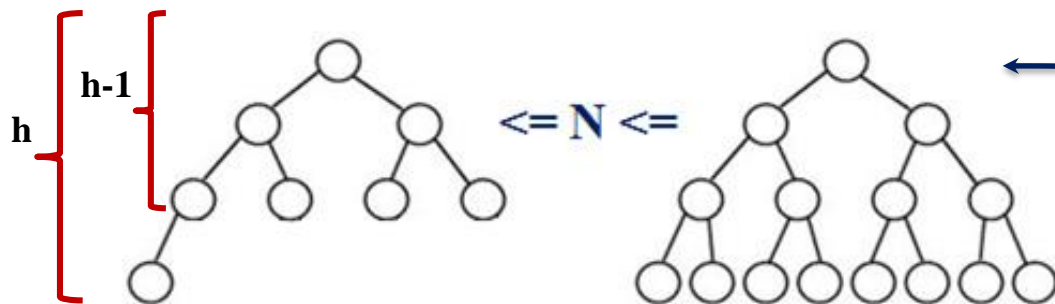


Descripción y terminología

- *Árbol binario completo*: Dado un árbol binario T de altura h , diremos que T es completo si es lleno de altura $h-1$ y el nivel h se completa de izquierda a derecha.

- *Cantidad de nodos en un árbol binario completo*:

Sea T un árbol binario completo de altura h , la cantidad de nodos N varía entre (2^h) y $(2^{h+1} - 1)$



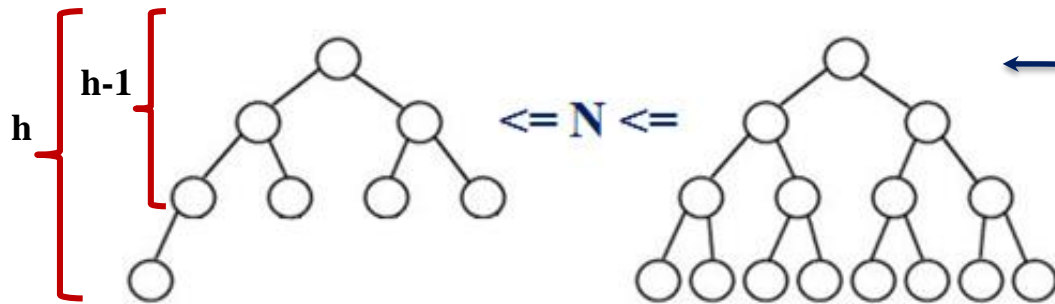
- Si el árbol es lleno
 $N = (2^{h+1} - 1)$

Descripción y terminología

- **Árbol binario completo:** Dado un árbol binario T de altura h , diremos que T es completo si es lleno de altura $h-1$ y el nivel h se completa de izquierda a derecha.

- **Cantidad de nodos en un árbol binario completo:**

Sea T un árbol binario completo de altura h , la cantidad de nodos N varía entre (2^h) y $(2^{h+1} - 1)$



- Si el árbol es lleno
 $N = (2^{h+1} - 1)$

- Si no, el árbol es lleno en la altura $h-1$ y tiene por lo menos un nodo en el nivel h :
 $N = (2^{h-1+1} - 1) + 1 = (2^h - 1 + 1)$

Juguemos a adivinar

- *Una persona piensa un animal y otra persona hace preguntas para adivinarlo*



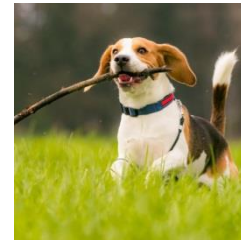
Limitamos las opciones a:

- *Dragón*
- *Dinosaurio*
- *Cóndor*
- *Caballo*
- *Perro*



Algunas preguntas

- *¿Es real?*
- *¿Está extinto?*
- *¿Vuela?*
- *¿Puede llevar personas?*
- *¿Es cuadrúpedo?*



Clasificamos los animales

irreal



real

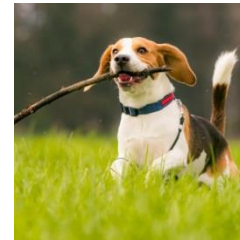


Clasificamos los animales

Vuela



No vuela

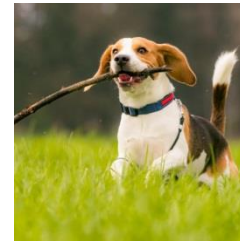


Clasificamos los animales

Cuadrúpedo



Bípedo



Sintetizamos las características

	¿Real?	¿Extinto?	¿Vuela?	¿Lleva personas?	¿Cuadrúpedo?
Dragón			X	X	X
Dinosaurio	X	X			
Cóndor	X		X		
Perro	X				X
Caballo	X			X	X

¿Cómo podemos organizar las preguntas?

De forma tal de ir descartando animales

Para identificar un sólo animal

¿Vuela?

Si



No



Vuela -> ¿Real?

No



Si



No vuela -> ¿Extinto?

Si



No



No vuela -> No extinto -> ¿Lleva personas?

Si



No



Estrategia completa

➤ *Vuela?*

➤ *Vuela-> Es real?*

➤ *Real-> Cóndor*

➤ *No es real-> Dragón*

➤ *No vuela-> Está extinto?*

➤ *Está extinto-> Dinosaurio*

➤ *No está extinto-> lleva personas?*

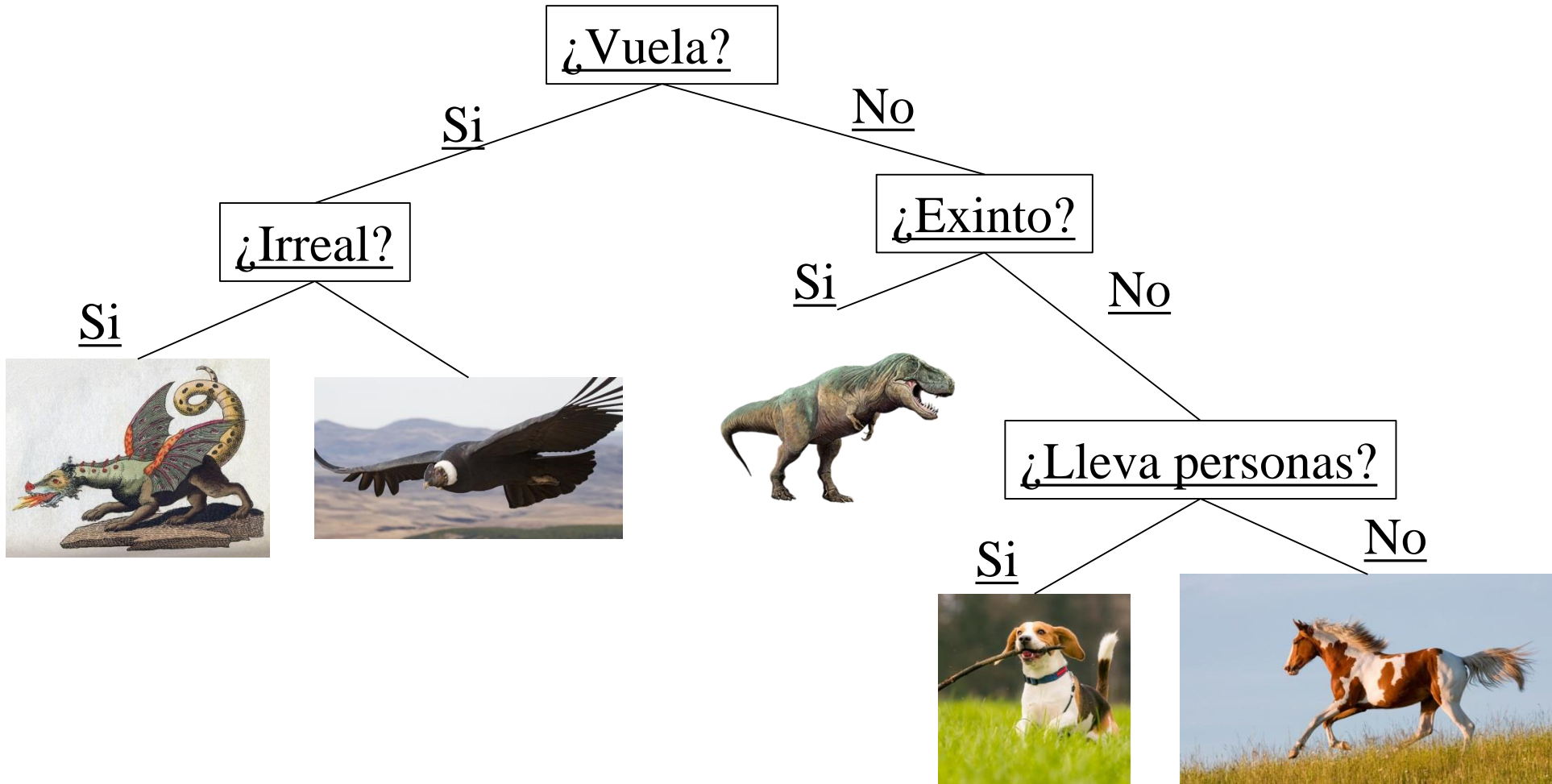
➤ *Lleva personas-> Caballo*

➤ *No lleva personas-> Perro*

Árbol de decisión

- *Herramienta de soporte a la toma de decisión que usa un modelo similar a un árbol donde se registran decisiones y sus posibles consecuencias*

Árbol de decisión



Árbol de decisión: usos

- *Son utilizados en investigación operativa para identificar la mejor estrategia para lograr un objetivo*
 - *Análisis financiero, considerando recursos y probabilidades*
 - *Ocurrencia de eventos, considerando probabilidades y resultados*
- *También son populares en Machine learning*

Representación

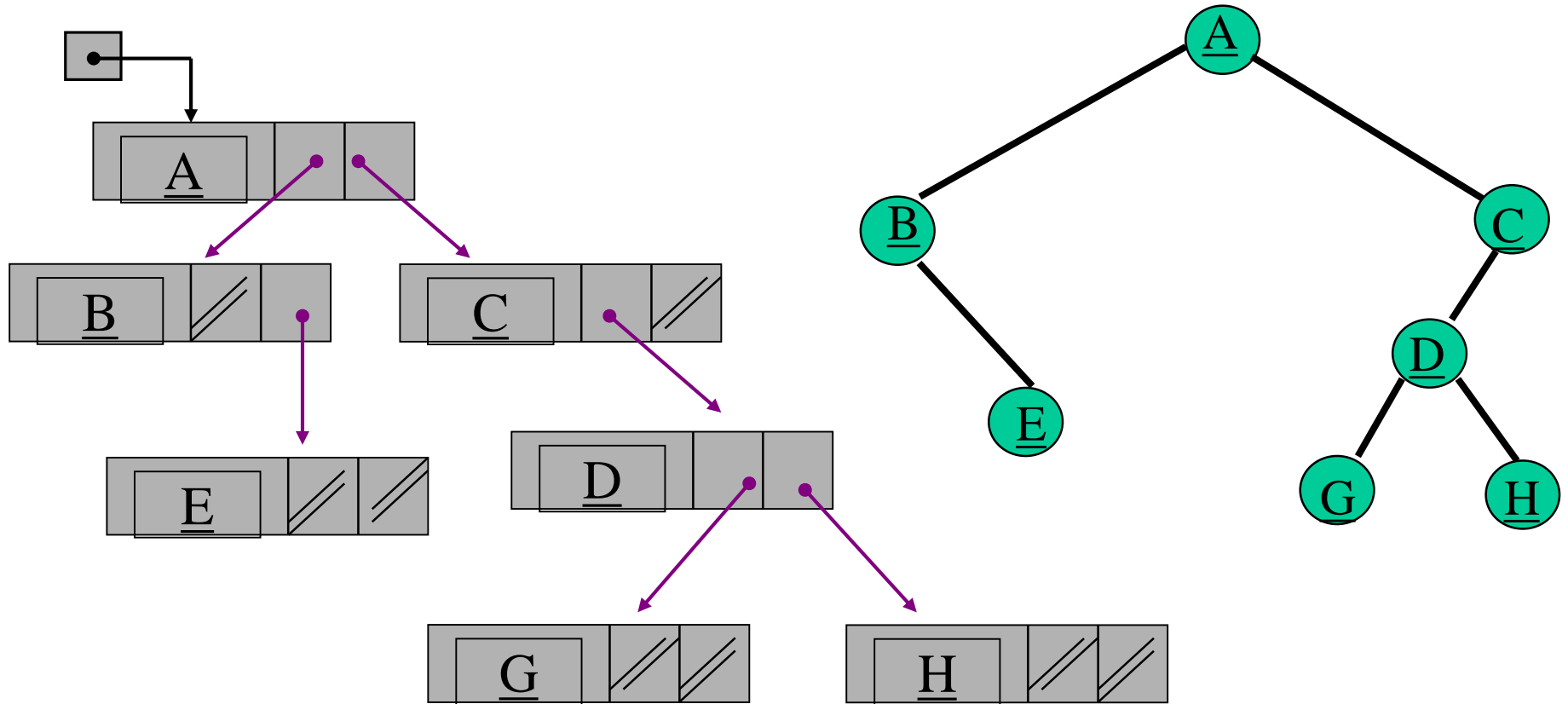
Hijo Izquierdo - Hijo Derecho

✓ Cada nodo tiene:

- Información propia del nodo
- Referencia a su hijo izquierdo
- Referencia a su hijo derecho

Representación

Hijo Izquierdo - Hijo Derecho



Recorridos

- **Preorden**
Se procesa primero la raíz y luego sus hijos, izquierdo y derecho.
- **Inorden**
Se procesa el hijo izquierdo, luego la raíz y último el hijo derecho
- **Postorden**
Se procesan primero los hijos, izquierdo y derecho, y luego la raíz
- **Por niveles**
Se procesan los nodos teniendo en cuenta sus niveles, primero la raíz, luego los hijos, los hijos de éstos, etc.

Recorrido: Preorden

```
public void preorden() {  
    imprimir (dato);  
    si (tiene hijo_izquierdo)  
        hijoIzquierdo.preorden();  
    si (tiene hijo_derecho)  
        hijoDerecho.preorden();  
}
```

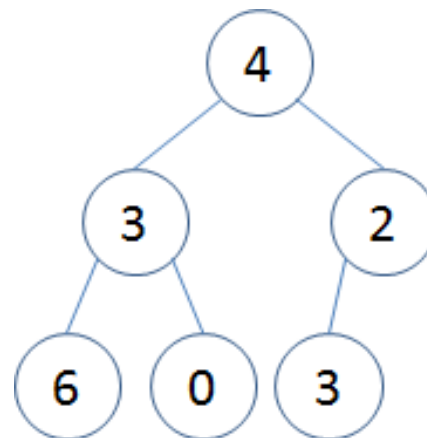
Recorrido: Por niveles

```
public void porNiveles() {  
    encolar(raíz);  
    mientras (cola no se vacíe) {  
        desencolar(v);  
        imprimir (dato de v);  
        si (tiene hijo_izquierdo)  
            encolar(hijo_izquierdo);  
        si (tiene hijo_derecho)  
            encolar(hijo_derecho);  
    }  
}
```

Ejercicio: Valencias

[SPOJ.com - Problem UCV2013J](https://SPOJ.com/Problem/UCV2013J)

El Sr. White ha encontrado una manera de maximizar la pureza de los cristales basados en ciertos compuestos químicos. Ha observado que cada compuesto está hecho de **moléculas** que están unidas entre sí siguiendo la estructura de un **árbol binario completo** donde cada nivel, excepto posiblemente el último, está completamente lleno, y todos los nodos están lo más a la izquierda posible. Cada nodo del árbol almacena la **valencia** de una molécula y se representa como un **número entero**. El Sr. White utiliza un microscopio electrónico que descarga la estructura de la molécula como un stream de números enteros y le gustaría tener su ayuda para obtener automáticamente la valencia total de sólo las **hojas del árbol dado**. Por ejemplo, la secuencia 4-3-2-6-0-3 representa el árbol que se muestra en la figura y la valencia total de las hojas es 9.



Ejercicio: Valencias

[SPOJ.com - Problem UCV2013J](https://www.spoj.com/problems/UCV2013J/)

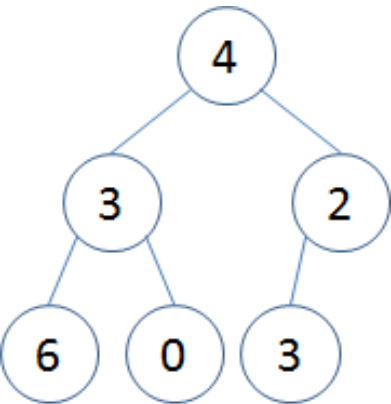
Input

La entrada contiene varios casos de prueba, cada uno correspondiente a un compuesto en particular. Cada caso de prueba consiste en una sola línea que comienza con un entero N ($1 \leq N \leq 1000000$), seguido de N números enteros V_i que representan las valencias de cada molécula separadas por espacios en blanco ($0 \leq V_i \leq 100$).

El final de la entrada se indica mediante un caso de prueba con $N = 0$.

Output

Para cada compuesto se produce una sola línea con la suma de las valencias de las hojas del árbol.



Ejemplo

Input:

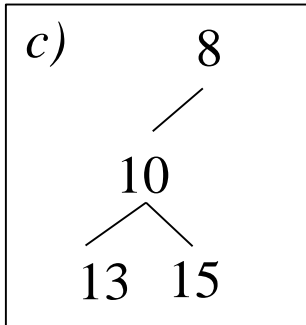
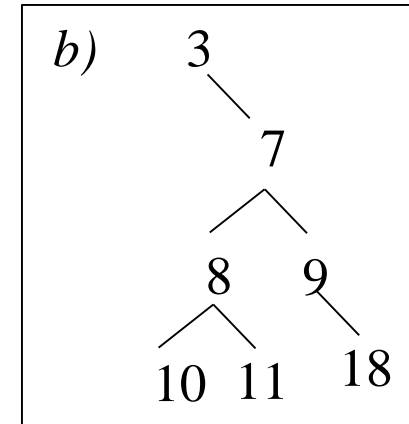
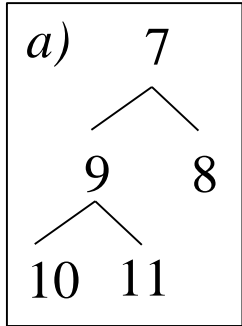
```
6 4 3 2 6 0 3
7 1 1 1 2 1 2 1
0
```

Output:

```
9
6
```

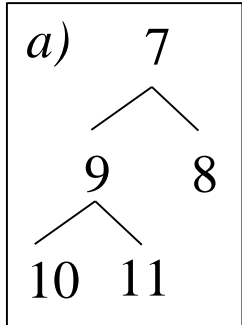
Ejercitación Árbol binario: Recorridos

Ejercicio 1



Ejercitación Árbol binario: Recorridos

Ejercicio 1

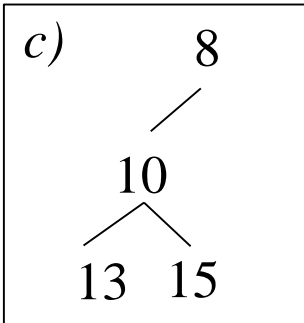
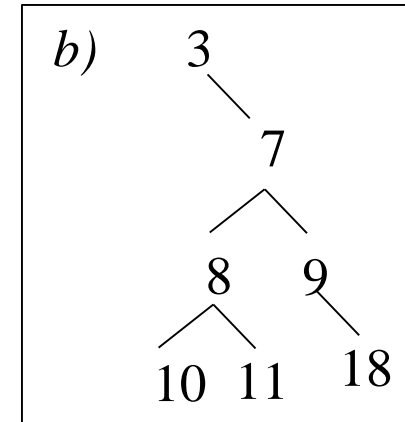


a)

✓ *inorden* : 10 9 11 7 8

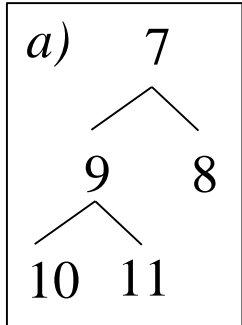
✓ *postorden* : 10 11 9 8 7

✓ *preorden* : 7 9 10 11 8



Ejercitación Árbol binario: Recorridos

Ejercicio 1

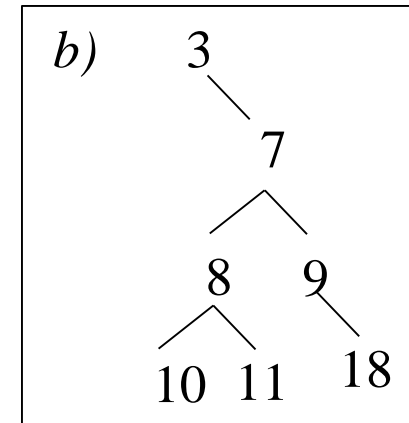
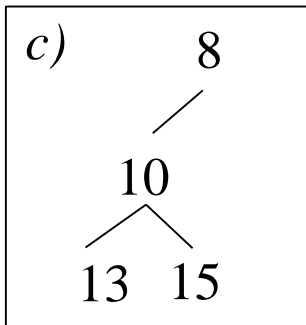


a)

✓ *inorden* : 10 9 11 7 8

✓ *postorden* : 10 11 9 8 7

✓ *preorden* : 7 9 10 11 8



b)

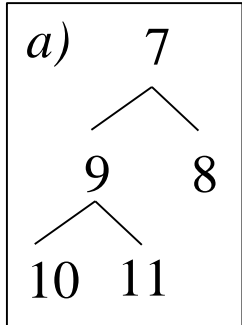
✓ *inorden* : 3 10 8 11 7 9 18

✓ *postorden* : 10 11 8 18 9 7 3

✓ *preorden* : 3 7 8 10 11 9 18

Ejercitación Árbol binario: Recorridos

Ejercicio 1

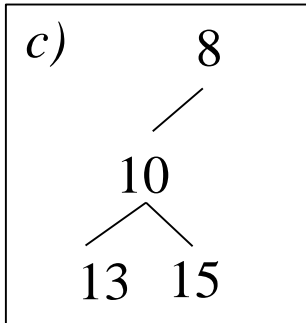


a)

✓ *inorden* : 10 9 11 7 8

✓ *postorden* : 10 11 9 8 7

✓ *preorden* : 7 9 10 11 8

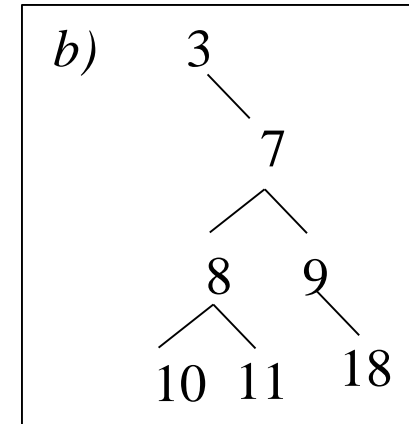


c)

✓ *inorden* : 13 10 15 8

✓ *postorden* : 13 15 10 8

✓ *preorden* : 8 10 13 15



b)

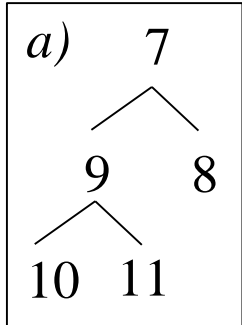
✓ *inorden* : 3 10 8 11 7 9 18

✓ *postorden* : 10 11 8 18 9 7 3

✓ *preorden* : 3 7 8 10 11 9 18

Ejercitación Árbol binario: Recorridos

Ejercicio 1

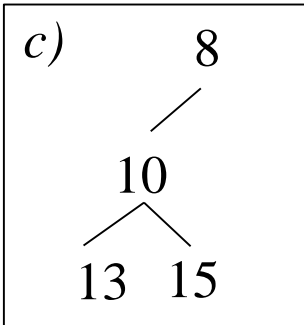


a)

✓inorden : 10 9 11 7 8

✓postorden : 10 11 9 8 7

✓preorden : 7 9 10 11 8

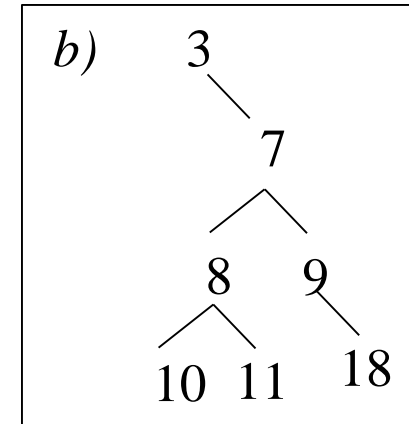


c)

✓inorden : 13 10 15 8

✓postorden : 13 15 10 8

✓preorden : 8 10 13 15



b)

✓inorden : 3 10 8 11 7 9 18

✓postorden : 10 11 8 18 9 7 3

✓preorden : 3 7 8 10 11 9 18

Ejercicio 2

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos:

inorden: **C B F E G A D I H** y postorden: **C F G E B I H D A**

Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:

inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

¿Por dónde empezamos?

¿Qué información podemos obtener de los recorridos dados?

¿De qué estamos seguros?

Ejercitación


Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:

inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**



Raíz

¿ Cómo seguimos ?

Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

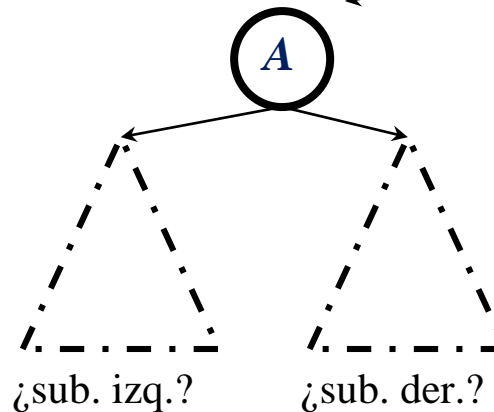
Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:

inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Raíz

¿Cómo armamos los subárboles?
¿Qué información podemos
obtener de los recorridos dados?



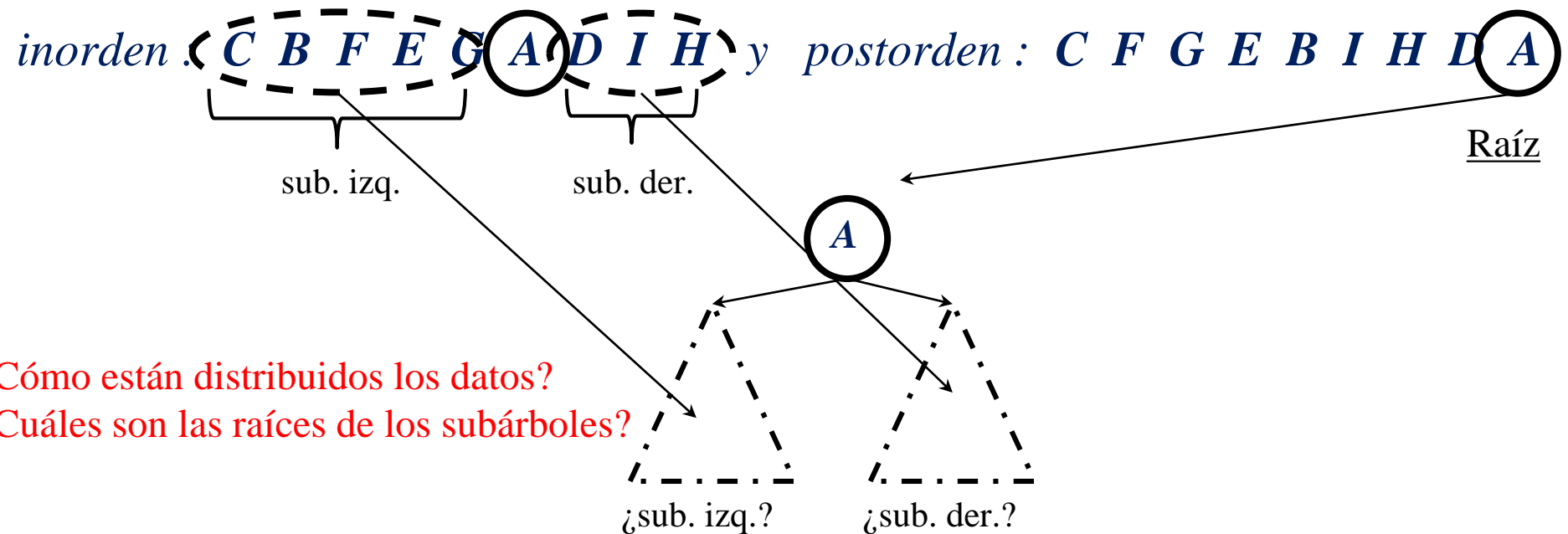
Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:



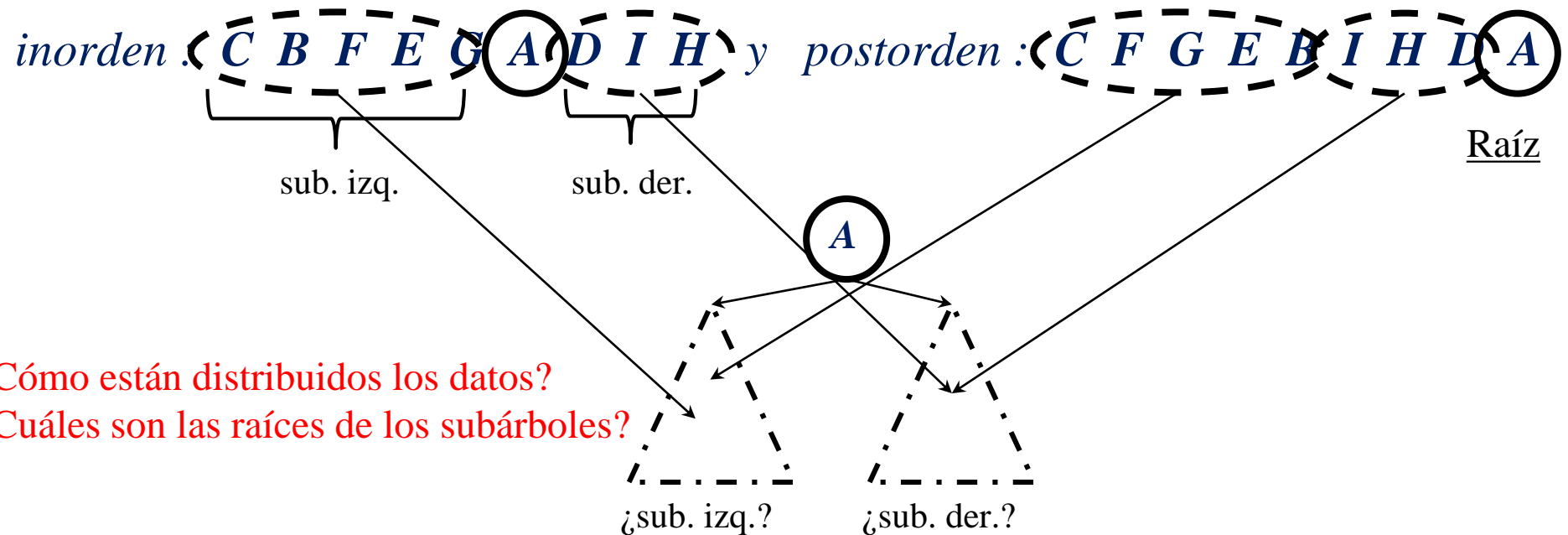
Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:



¿Cómo están distribuidos los datos?

¿Cuáles son las raíces de los subárboles?

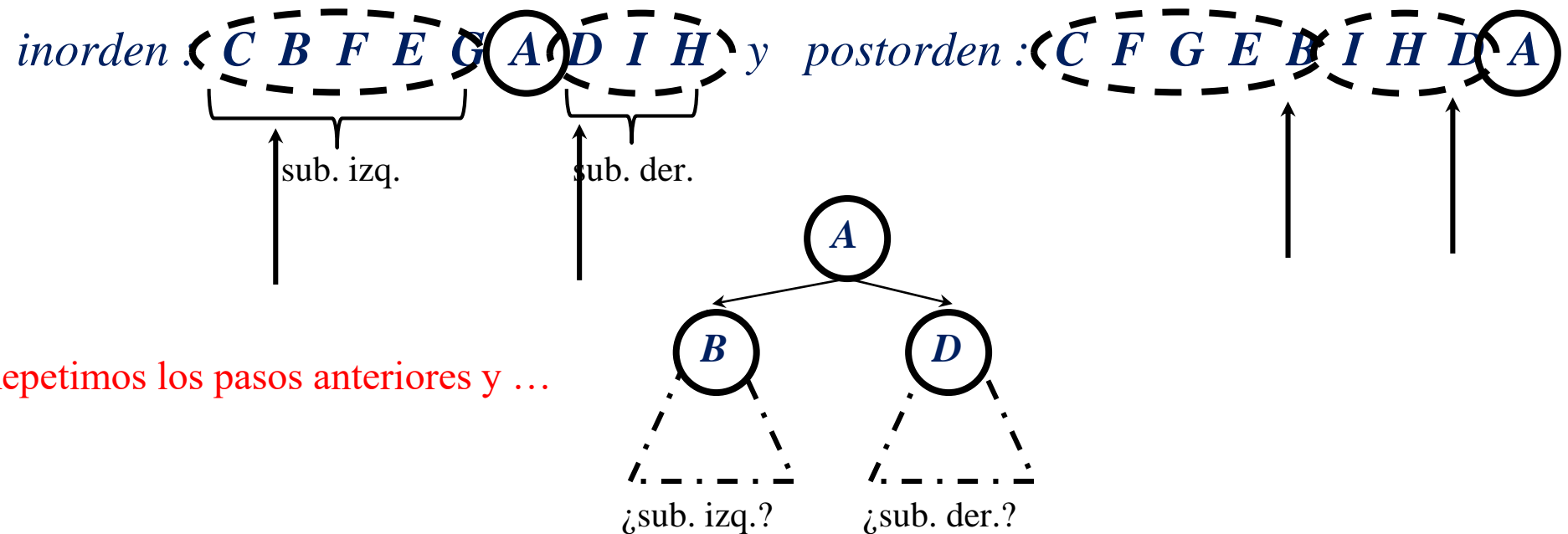
Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:



Repetimos los pasos anteriores y ...

Ejercitación

Árbol binario: Recorridos

Ejercicio 2.

Construya el árbol binario a partir del cual se obtuvieron los siguientes recorridos: inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

Resolución:

inorden : **C B F E G A D I H** y postorden : **C F G E B I H D A**

