

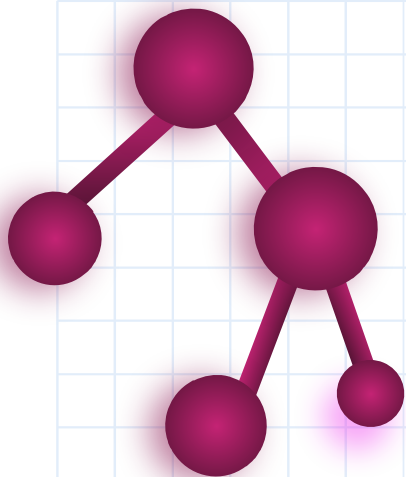
04/11/21

# Introducción a los árboles binarios



*(árboles  
orden = 2)*

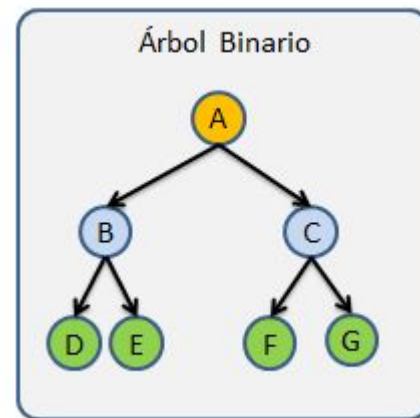
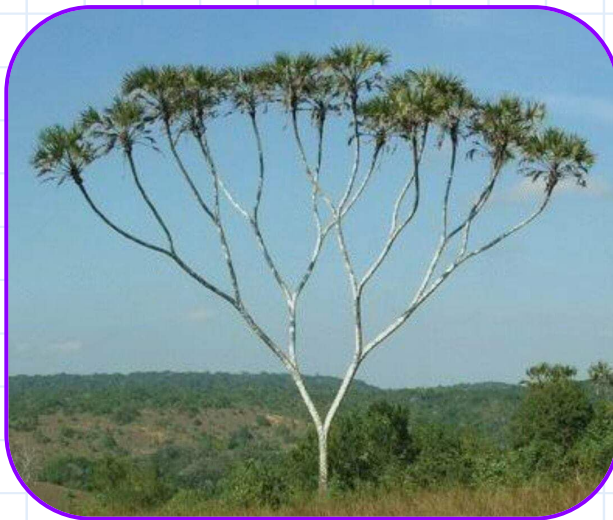




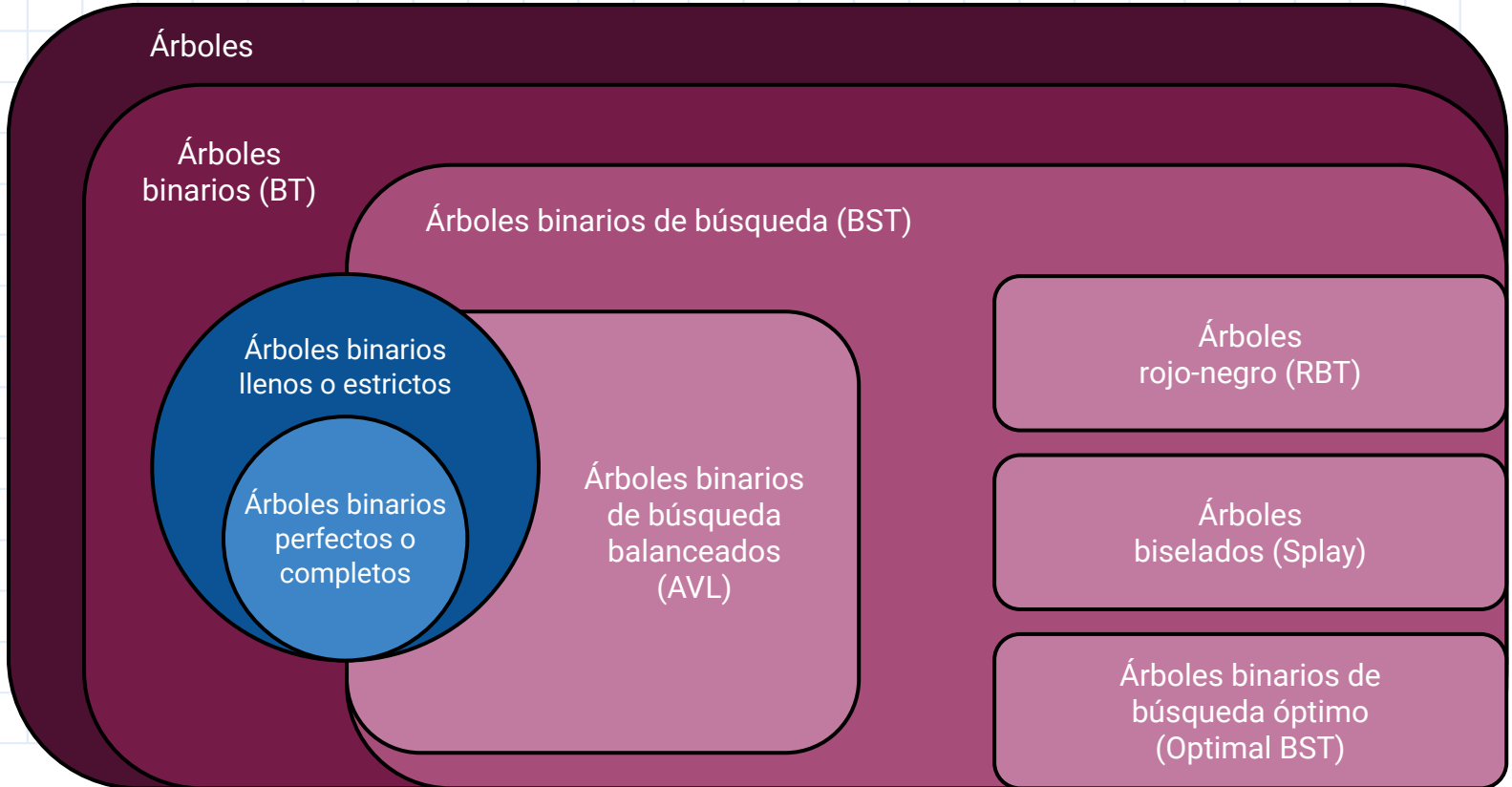
# Árboles binarios

Un árbol binario es un conjunto finito de elementos que está ya sea vacío o conformado por tres subconjuntos: un elemento llamado la raíz del árbol y otros dos que son, a su vez, árboles binarios.

Ellos se conocen con el nombre de sub-árbol izquierdo y sub-árbol derecho del árbol original. Un sub-árbol puede estar vacío.



# Clasificación de los Árboles Binarios

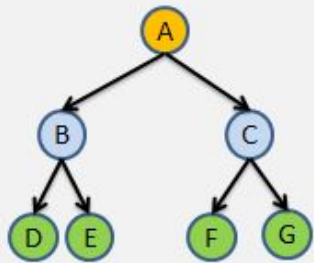


# Árbol binario lleno o estricto

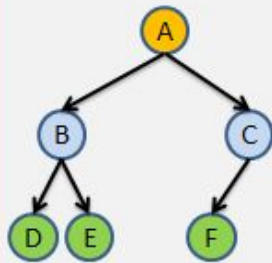


Árbol binario en el que todos sus nodos tienen cero o 2 hijos.

Árbol Binario Llento



Árbol Binario **NO** Llento



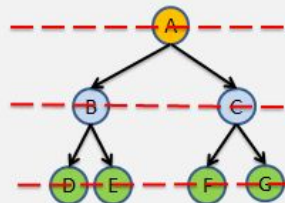
# Árbol binario perfecto o completo



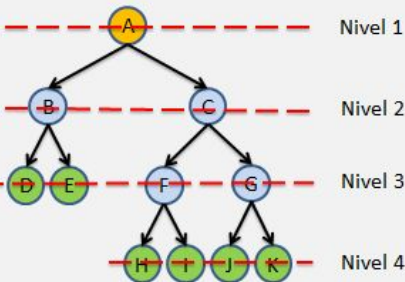
Árbol binario estricto de profundidad  $d$ , donde todas las hojas están en el nivel  $d$ .

Otra forma de verlo: Árbol binario estricto en donde todas las hojas están en el mismo nivel.

Árbol Binario Perfecto



Árbol Binario **NO** Perfecto



Nivel 1

Nivel 2

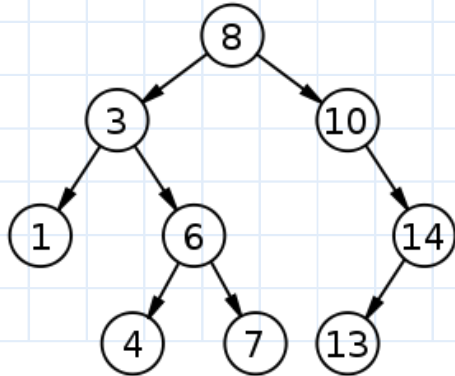
Nivel 3

Nivel 4

# Árbol binario de búsqueda



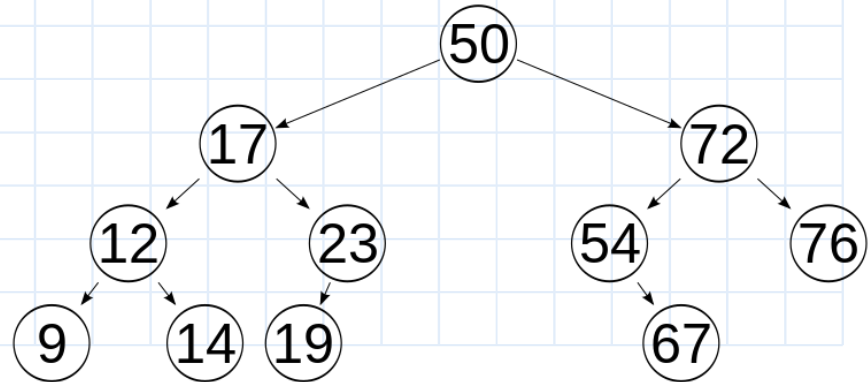
Es un **árbol binario** que cumple que el subárbol **izquierdo** de cualquier nodo contiene valores **menores** que el que contiene dicho nodo, y el subárbol **derecho** contiene valores **mayores**.


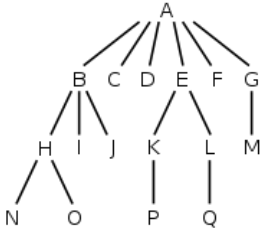

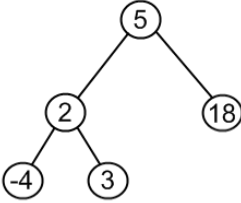





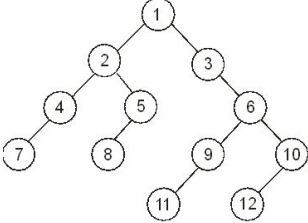






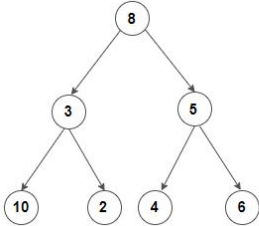




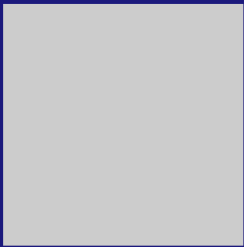
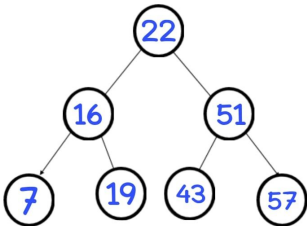





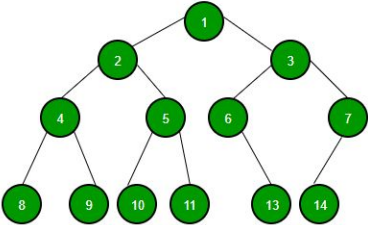




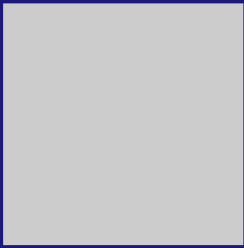
# Árbol binario de búsqueda balanceado


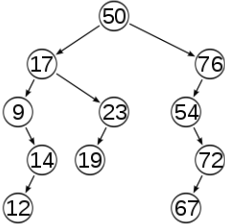


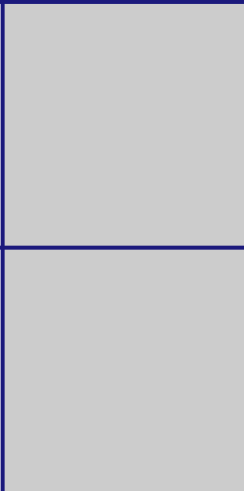

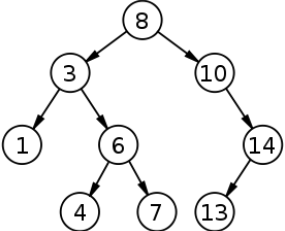



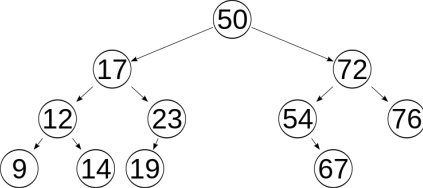





**ABB** que están siempre **equilibrados** de tal modo que para todos los nodos, la altura de la rama izquierda no difiere en más de una unidad de la altura de la rama derecha o viceversa. Para conseguir esta propiedad de equilibrio, **la inserción y el borrado** de los nodos se ha de realizar de una forma especial.


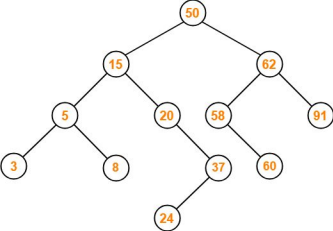
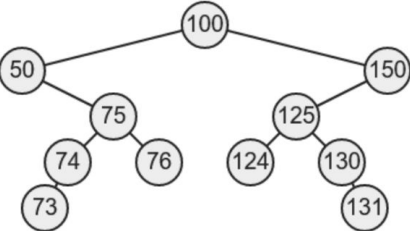
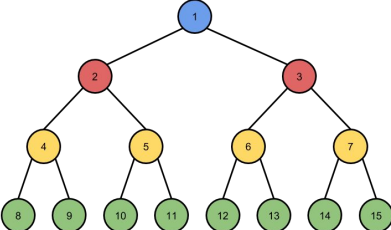


 <b>Árbol</b>	<b>Árbol binario</b>	<b>AB lleno o estricto</b>	<b>AB perfecto o completo</b>	<b>AB de búsqueda</b>	<b>ABB balanceado</b>
					
					
					

 <b>Árbol</b>	<b>Árbol binario</b>	<b>AB lleno o estricto</b>	<b>AB perfecto o completo</b>	<b>AB de búsqueda</b>	<b>ABB balanceado</b>
					
					
					

 <b>Árbol</b>	<b>Árbol binario</b>	<b>AB lleno o estricto</b>	<b>AB perfecto o completo</b>	<b>AB de búsqueda</b>	<b>ABB balanceado</b>
					
					
					



 <b>Árbol</b>	Árbol binario	AB lleno o estricto	AB perfecto o completo	AB de búsqueda	ABB balanceado
	✓	✗		✓	?
	✓	✗		✓	?
	✓	✓	✓	✗	

# Recorridos

## In-orden:

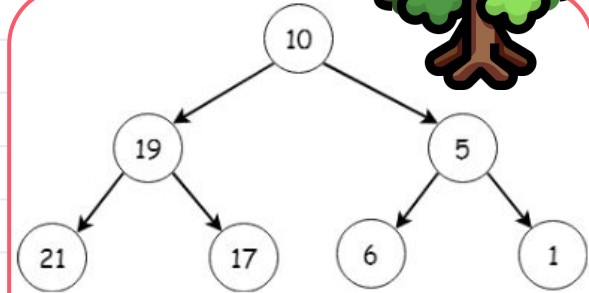
- Se recorre en **in-orden** el sub-árbol izquierdo.
- Se recorre la raíz.
- Se recorre en **in-orden** el sub-árbol derecho.

## Pre-orden:

- Se recorre la raíz.
- Se recorre en **pre-orden** el sub-árbol izquierdo.
- Se recorre en **pre-orden** el sub-árbol derecho.

## Post-orden:

- Se recorre en **post-orden** el sub-árbol izquierdo.
- Se recorre en **post-orden** el sub-árbol derecho.
- Se recorre la raíz.



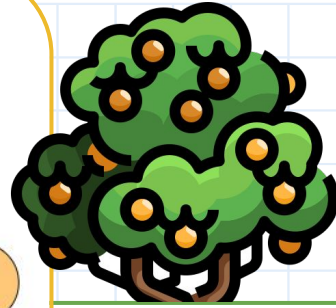
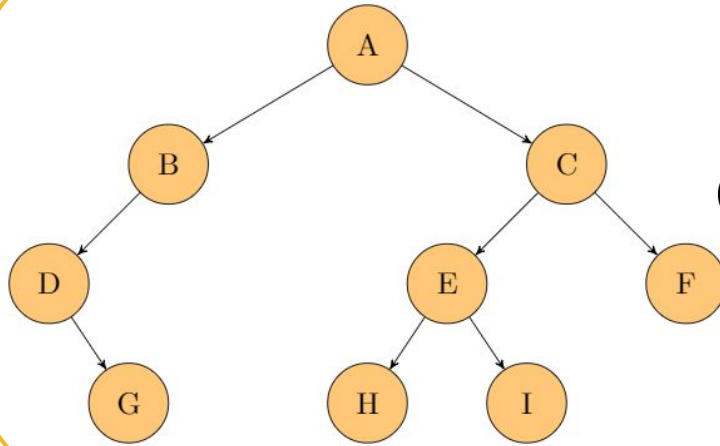
## Recorridos

In-orden: 21, 19, 17, 10, 6, 5, 1.

Pre-orden: 10, 19, 21, 17, 5, 6, 1.

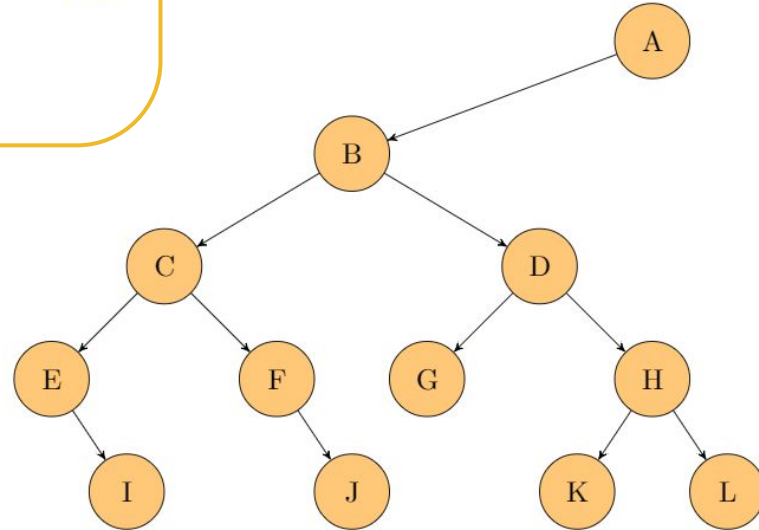
Post-orden: 21, 17, 19, 6, 1, 5, 10.



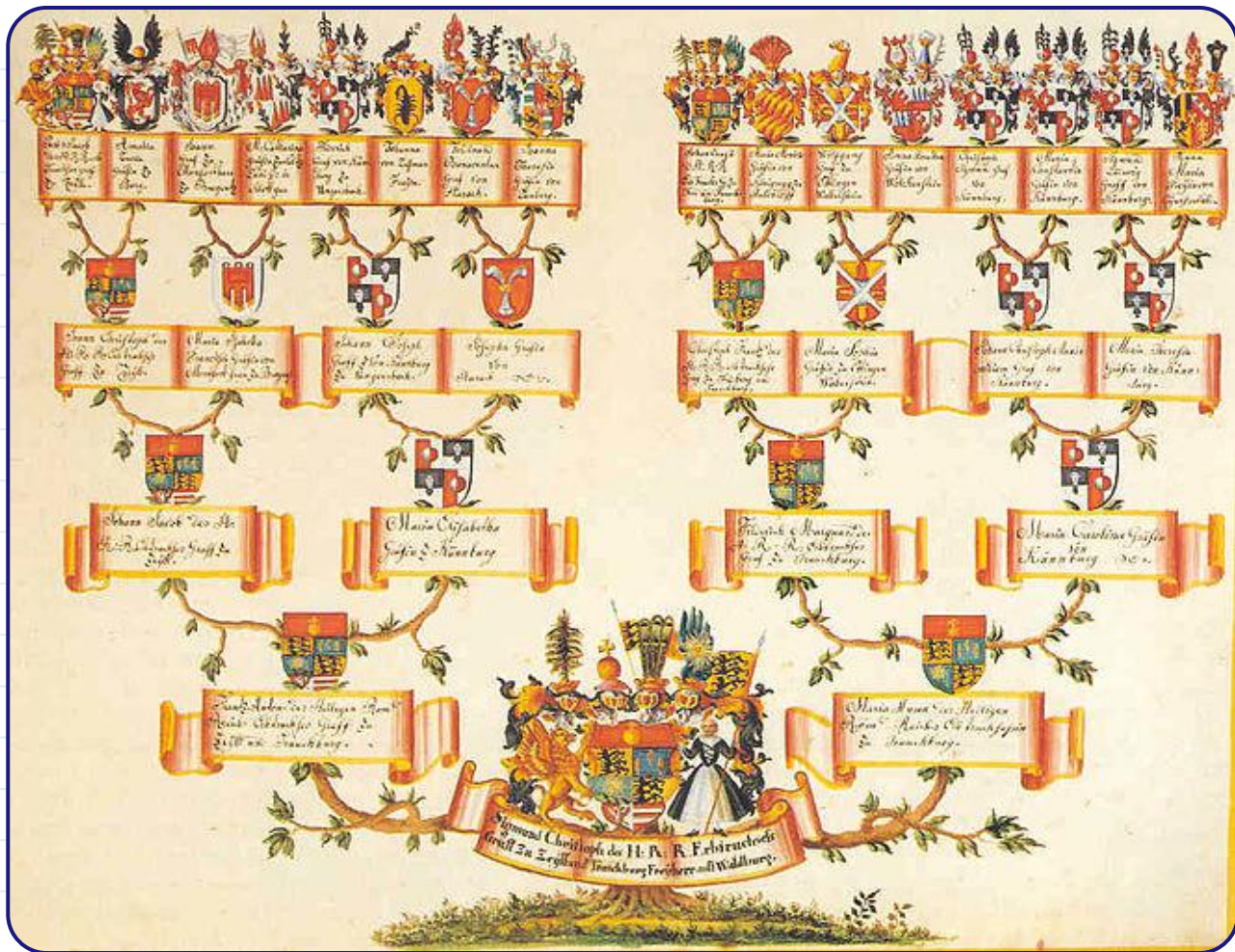
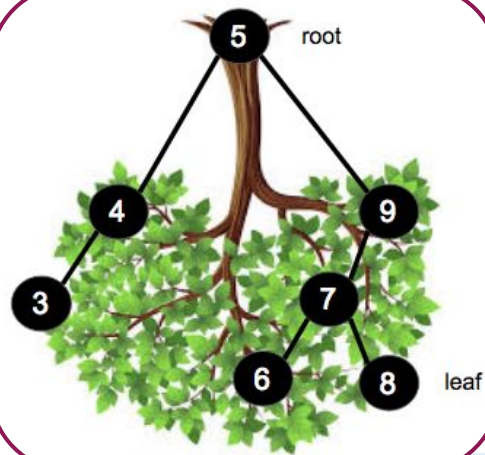


## Ejercicio AB's

1. ¿Es un árbol binario lleno?
2. ¿Es un árbol binario perfecto?
3. Recorrido in-orden.
4. Recorrido pre-orden.
5. Recorrido post-orden.



# Árboles binarios de búsqueda

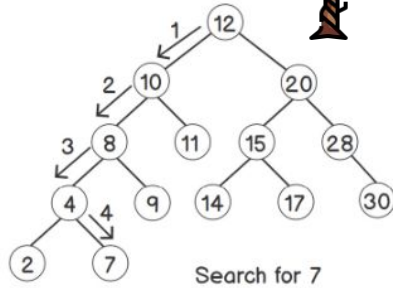




## 1. Búsqueda en un árbol binario de búsqueda

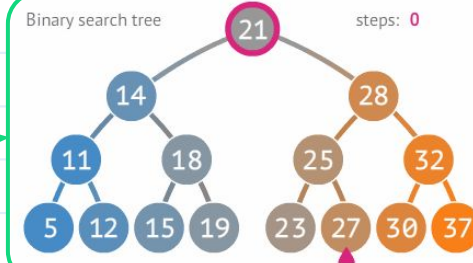
### Algoritmo:

- Comparar el dato buscado con la raíz del árbol. Si es mayor, se sigue con el sub-árbol derecho. Si es menor, se continúa con el sub-árbol izquierdo.
- Repetir sucesivamente hasta encontrar el dato o llegar a NULL.



Binary search tree

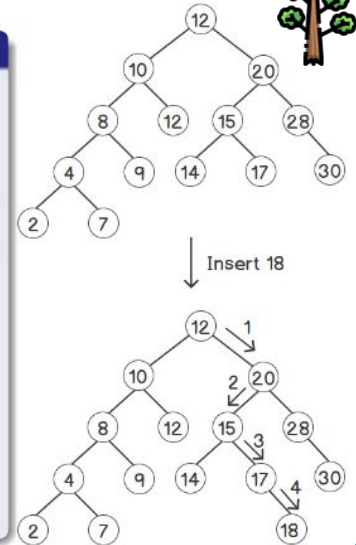
steps: 0



## 2. Inserción en un árbol binario de búsqueda

### Algoritmo:

- Comparar el dato a insertar con la raíz del árbol. Si es mayor, se sigue con el sub-árbol derecho. Si es menor, se continúa con el sub-árbol izquierdo.
- Repetir sucesivamente el paso 1 hasta que se cumpla alguna de las siguientes condiciones:
  - El sub-árbol derecho, o el sub-árbol izquierdo, es igual a vacío, en cuyo caso se procederá a insertar el elemento en el lugar que le corresponde.
  - El dato que se quiere insertar está en el nodo analizado, por lo tanto no se lleva a cabo la inserción.



120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56

120



120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56

120

87

43

120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56

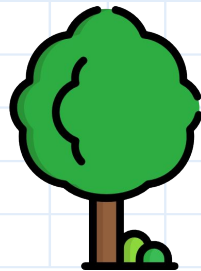
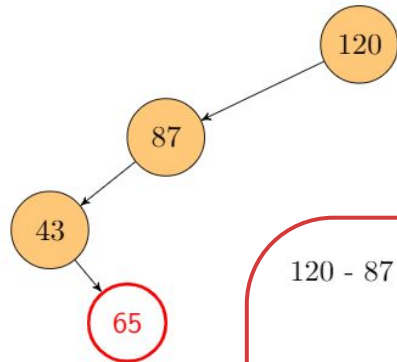
120

87

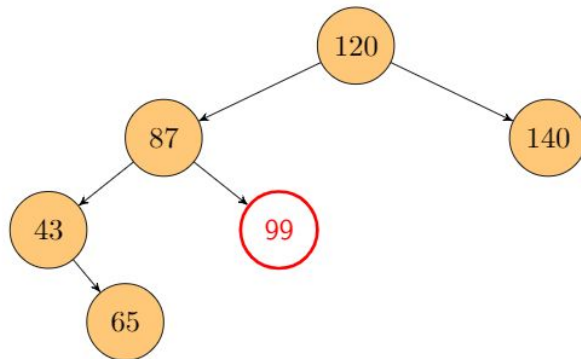


120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56

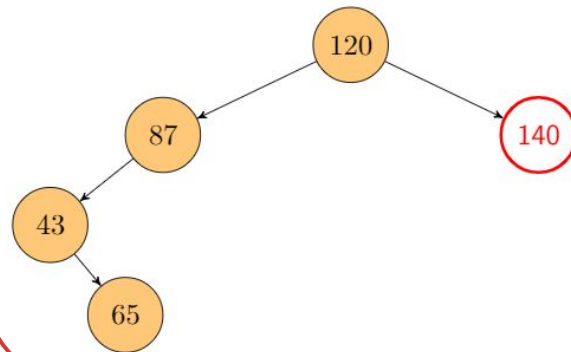
120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56



120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56



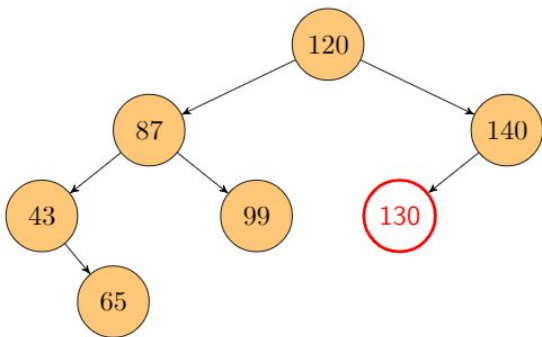
120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56



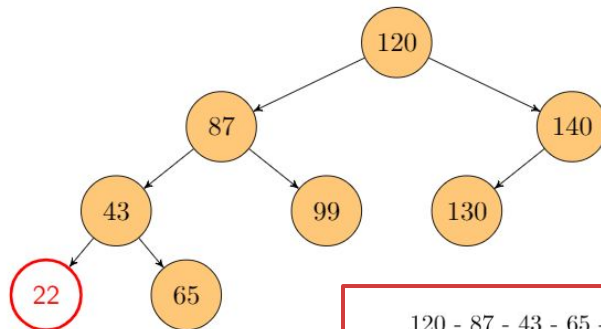




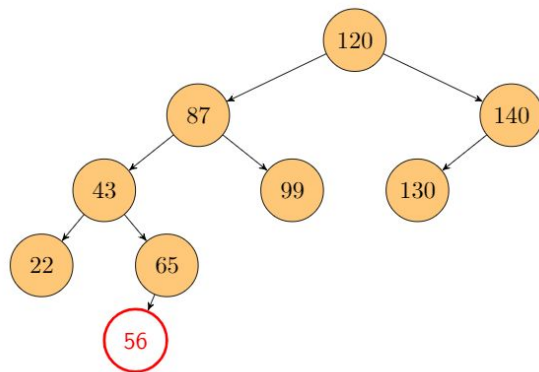
120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56



120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56



120 - 87 - 43 - 65 - 140 - 99 - 130 - 22 - 56

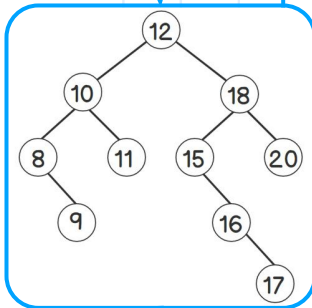


### 3. Eliminación en un árbol binario de búsqueda

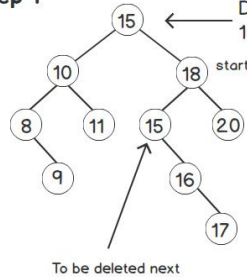


#### Algoritmo:

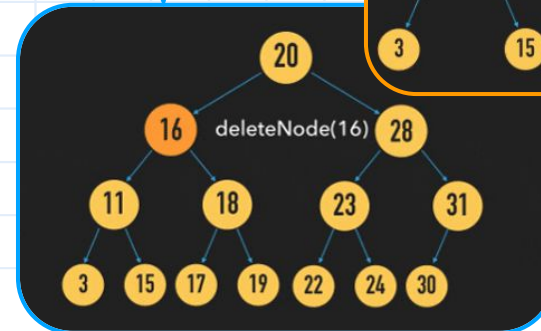
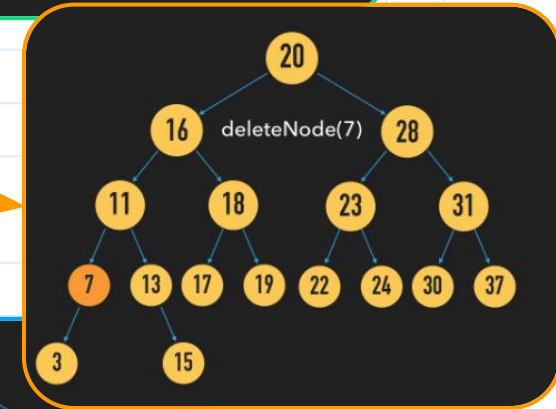
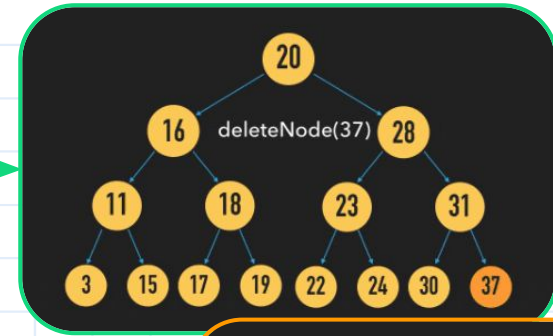
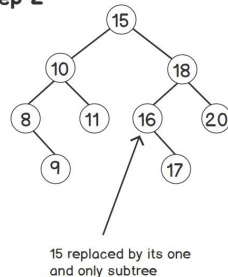
- Si el elemento a eliminar es **terminal u hoja**, simplemente se suprime redefiniendo el puntero de su predecesor.
- Si el elemento a eliminar **tiene un solo descendiente**, entonces tiene que sustituirse por ese descendiente.
- Si el elemento a eliminar **tiene los dos descendientes**, entonces se tiene que sustituir por ~~el nodo que se encuentra más a la izquierda en el sub-árbol derecho o por el nodo que se encuentra más a la derecha en el sub-árbol izquierdo~~.



#### Step 1

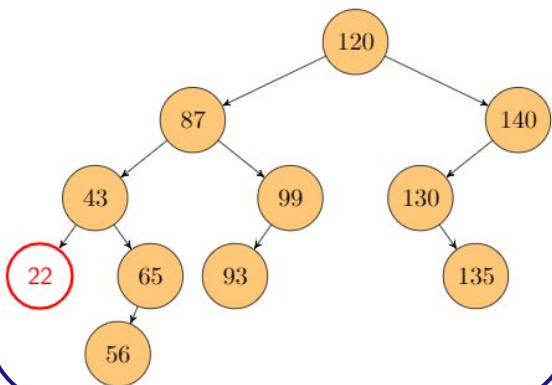


#### Step 2

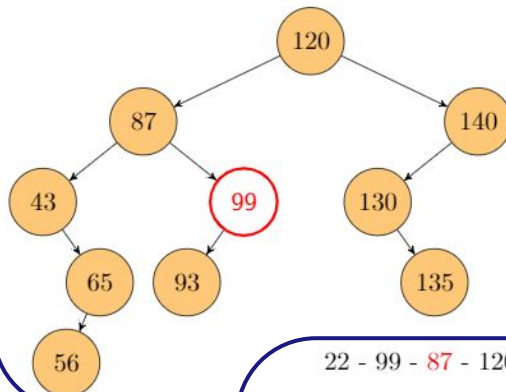




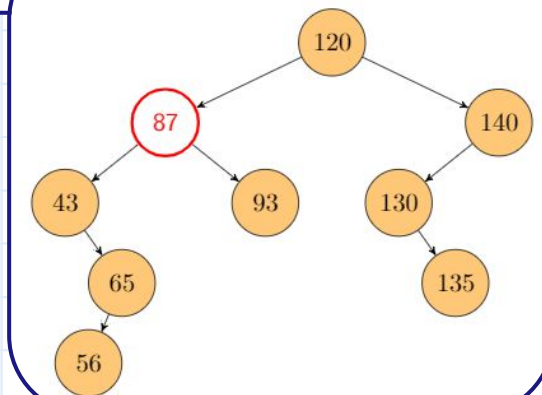
22 - 99 - 87 - 120 - 140 - 135 - 56



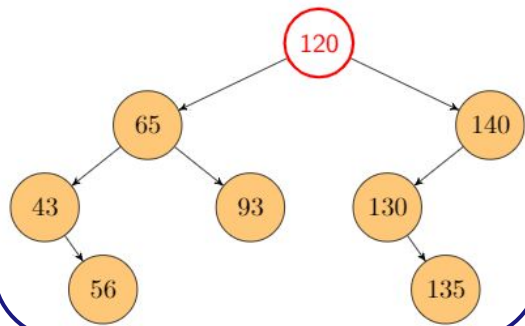
22 - 99 - 87 - 120 - 140 - 135 - 56



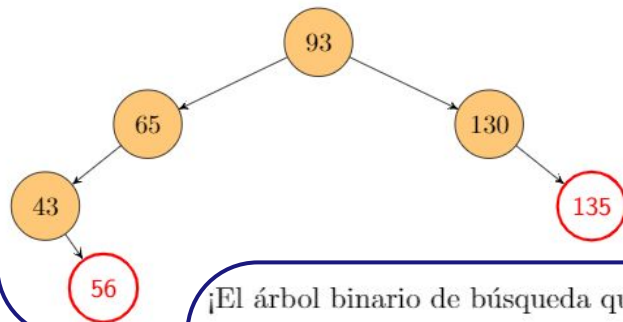
22 - 99 - 87 - 120 - 140 - 135 - 56



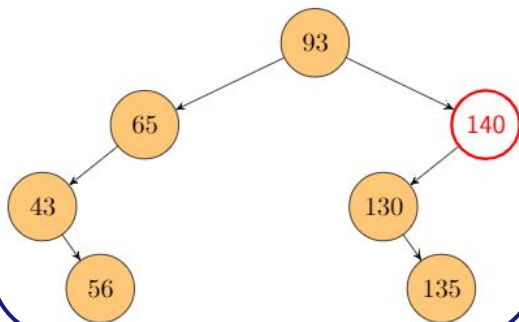
22 - 99 - 87 - 120 - 140 - 135 - 56



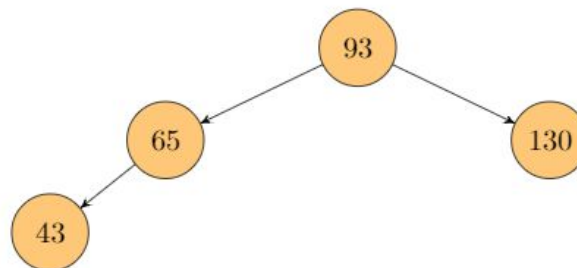
22 - 99 - 87 - 120 - 140 - 135 - 56



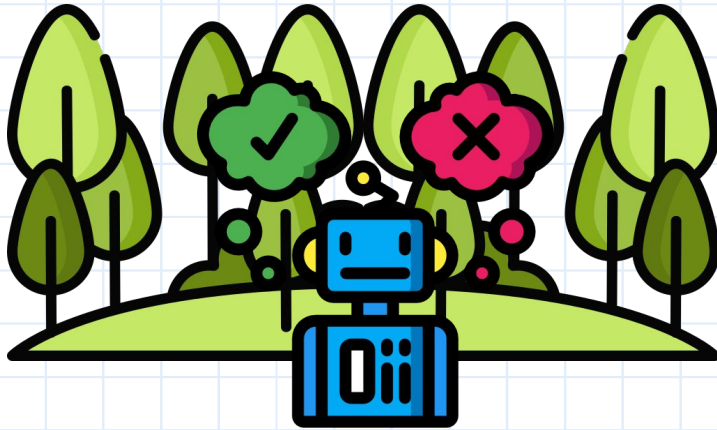
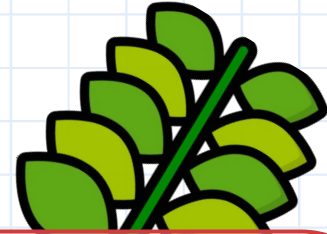
22 - 99 - 87 - 120 - 140 - 135 - 56



¡El árbol binario de búsqueda quedó ordenado!



# ¿Dudas, consultas o comentarios?



¿Cómo quedaría si se insertan los datos 93 y 135?

