Árboles

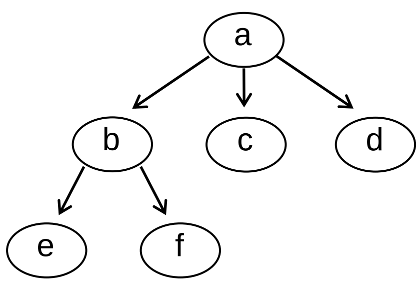
Un árbol es una estructura jerárquica, organizada y dinámica aplicada sobre una colección de objetos llamados nodos.

- *Jerárquica* porque los componentes están a distinto nivel.
- *Organizada* porque importa la forma en que este dispuesto el contenido.
- *Dinámica* porque su forma, tamaño y contenido pueden variar durante la ejecución.

Los árboles genealógicos y los organigramas son ejemplos comunes de árboles. Entre otras cosas, los árboles son útiles para analizar circuitos eléctricos, para representar la estructura de fórmulas matemáticas, para organizar información en una base de datos, para representar el sistema de archivos y para analizar la estructura sintáctica de un programa fuente en los compiladores.

Existen diferentes formas de representación de un árbol, entre las más comunes se tienen las siguientes:

Mediante círculos y flechas:



Mediante paréntesis anidados:

(a (b (e, f), c, d))

Mediante notación decimal de Dewey:

1a, 1.1b, 1.1.1e, 1.1.2f, 1.2c, 1.3d

Identado, mediante nodos. Un buen ejemplo de esto, es la forma de representar gráficamente las carpetas (directorios) de un sistema de archivos. En este caso, una carpeta es un nodo padre de los archivos y subcarpetas contenidas en él.

1. a
    a. b
        i. e
        ii. f
    b. c
    c. d

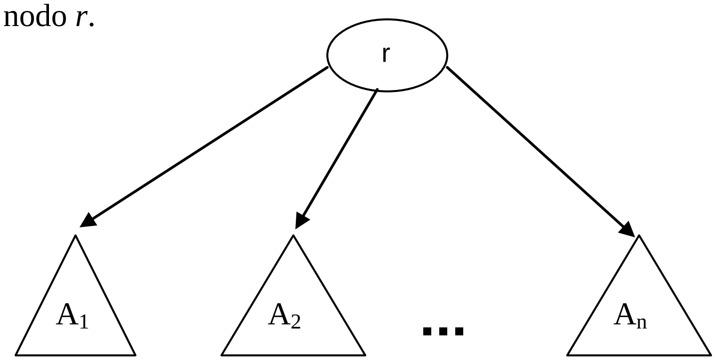
La forma de representación más fácil, común es la representación mediante círculos y flechas.

<u>Conceptos básicos</u>

Definición: Un árbol se puede definir recursívamente como sigue:

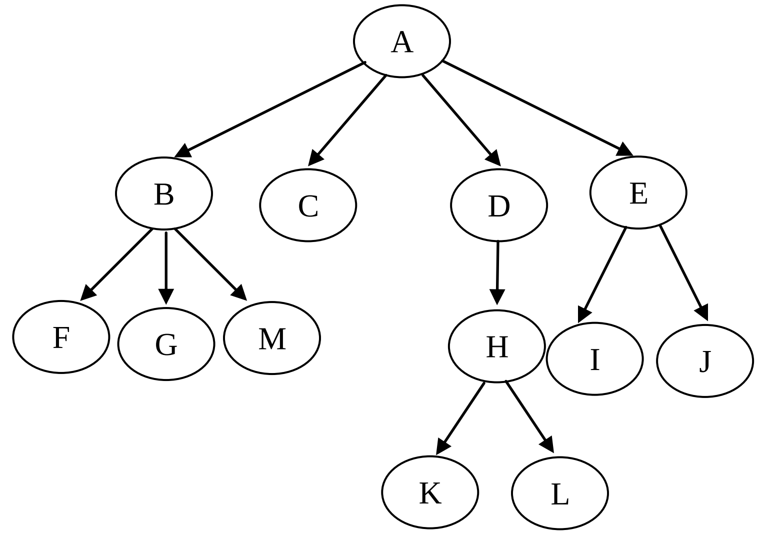
> Un solo nodo es, por sí mismo, un árbol. Ese nodo es también la raíz de dicho árbol.

Supóngase que *r* es un nodo y que A_1, $A_{2,}$ n..., A_n son árboles con raíces r_1, $r_{2, \ldots} r_n$, respectivamente. Se puede construir un nuevo árbol diciendo que *r* se constituya en el padre de los nodos r_1, $r_{2, \ldots} r_{n.}$ Por lo que, en dicho árbol, *r* será ahora la raíz y A_1, $A_{2, \ldots} A_n$ serán los *subárboles* de *r*. Los nodos r_1, $r_{2, \ldots} r_n$ serán ahora también *hijos* del nodo *r*.



Algunas veces se incluye entre los árboles el árbol *nulo* *vacío*, el cual, es un árbol sin nodos que se representa mediante la letra Λ.

Generalmente, se crea una relación o parentesco entre los nodos de un árbol que impone una estructura jerárquica y que da lugar a términos como padre, hijo, hermano, antecesor, sucesor, etc. Se dice que la raíz de cada subárbol A_k es un *hijo* de r y que *r* es el *padre* de cada raíz de los subárboles. En principio cualquier nodo del árbol podría tener un número arbitrario de nodos hijos, a esto se le conoce como un *árbol general*, como se muestra en la siguiente figura. Si se limita el número de nodos hijos para cada nodo del árbol, digamos a un número *n > 2* (llamado la aridad del árbol), entonces el árbol de aridad n es llamado *n-ario*.



- ✓ El nodo A es la raíz (padre).
- ✓ Los hijos de A son B, C, D, E
- ✓ Los nodos F, G, M son hermanos e hijos de B
- ✓ A es abuelo de H
- ✓ K y L son hijos de H y nietos de A

Con estas consideraciones se pueden definir las siguientes características y propiedades de los árboles. Algunos de los siguientes conceptos; sin embargo, no son uniformes en toda la literatura referente a la teoría de árboles.

- ✓ *Si hay un camino de A hasta B, se dice que A es* antecesor *de B, y que B es* sucesor *de A.*

- ✓ *Padre* es el antecesor inmediato de un nodo
- ✓ *Hijo*, cualquiera de sus descendientes inmediatos.
- ✓ *Antepasado* de un nodo, es cualquier antecesor de dicho nodo.
- ✓ *Descendiente* de un nodo, es cualquier sucesor de dicho nodo.
- ✓ *Hermano* de un nodo, es otro nodo con el mismo padre.
- ✓ *Raíz* es el nodo que no tiene ningún predecesor.
- ✓ *Hoja* (o nodo terminal) es el nodo que no tiene sucesores.
- ✓ Los nodos que tienen predecesor y sucesor se llaman nodos *interiores*.
- ✓ *Rama* es cualquier camino del árbol.
- ✓ *Bosque* es un conjunto de árboles desconectados.
- ✓ *Grado de un nodo*, es el número de flechas que salen de ese nodo. El número de flechas que entran siempre es uno.
- ✓ *Grado de un árbol*, es el mayor grado que puede hallarse en sus nodos.
- ✓ *Nivel* o profundidad de un nodo, es la longitud del camino desde la raíz hasta ese nodo. El nivel puede definirse como 1 para la raíz y nivel(predecesor)+1 para los demás nodos.
- ✓ *Generación*, es un conjunto de nodos con la misma profundidad.
- ✓ *Altura de un nodo*, es la longitud del camino desde ese nodo hasta la hoja más alejada (la altura de una hoja es 0 y la de un árbol vacío se considera -1).
- ✓ *Altura de un árbol*, es la altura desde la raíz. Esto es, es el máximo de los niveles de todos los nodos del árbol.
- ✓ Un *camino* de un nodo n_1 a otro n_k, se define como la secuencia de nodos n_1, n_2, ... n_k tal que n_i es padre de n_{i+1} para $1 \leq i < k$.
- ✓ *Longitud del camino entre 2 nodos*: Es el número de arcos que hay entre ellos.

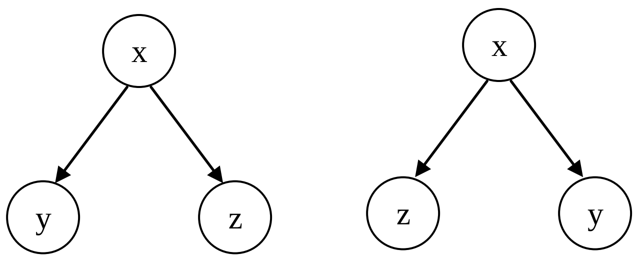
Ejemplo: Utilizando el árbol de la figura anterior, se tiene:

- A es antecesor de F y F es sucesor de A
- B es el padre de G y H es el padre de K
- I y J son hijos de E y K y L son hijos de H.
- A, D y H son antepasados de K y L
- Los descendientes de D son H, K y L
- I y J son hermanos. B, C, D, y E son también hermanos.
- El nodo A es la raíz
- C, F, G, K, M, L, I y J son hojas del árbol
- B, D, H, E son nodos interiores
- El grado de A es 4
- El grado de B es 3
- El grado de C es 0
- El grado del árbol es 4
- El nivel de A es 1
- El nivel de B es 2
- El nivel de H es 3
- El nivel de K es 4
- F, G, H, I, y J son de la generación 3
- La altura del nodo D es 2
- La altura del nodo H es 1
- La altura del nodo G es 0
- La altura del árbol es 3

- El camino de A a K es único y lo forman los nodos A-D-H-K
- El nodo B tiene longitud de camino 1 desde A
- El nodo I tiene longitud de camino 2 desde A
- El nodo K tiene longitud de camino 3 desde A

Orden de los nodos

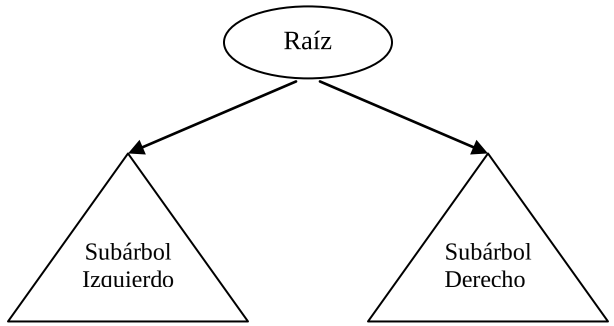
Generalmente los árboles de un nodo se ordenan de izquierda a derecha. Por ejemplo, los árboles de en la figura son distintos porque los dos hijos del nodo *x* aparecen en diferente orden en los dos árboles. Si no se toma en cuenta el orden de los nodos hijos, entonces se habla de un árbol *no ordenado*.



El orden de izquierda a derecha de los *hermanos* se puede extender para comparar dos nodos cualesquiera entre los cuales no exista la relación antecesor-descendiente. La regla que se aplica es que si *y* y *z* son hermanos y *y* está a la izquierda de *z*, entonces todos los descendientes de *y* estarán a la izquierda de todos los descendientes de *z*. Esto es, *y* es menor que *z*.

Árboles Binarios

Un **árbol binario** es un árbol de grado 2, en el que todo nodo del árbol tiene un subárbol binario izquierdo y derecho asociados.



Árbol Binario Completo o Lleno: Es un árbol binario en el que todos sus nodos, excepto las hojas, tienen siempre dos hijos (el subárbol izquierdo y el derecho) no nulos. El número de nodos de un árbol completo se calcula por la fórmula:

$$\textit{Número de nodos} = 2^h - 1 \quad \textit{(donde h es la altura)}$$

Además, siendo 1 el nivel de la raíz, el número máximo de nodos en un nivel k será 2^{k-1}.

Árbol Binario Completo de Altura o Profundidad H: Es un árbol Binario Completo en donde todas las hojas están en el nivel H. Esta es una de las pocas estructuras de árbol que se pueden representar eficientemente usando arreglos.

Árboles de Expresión

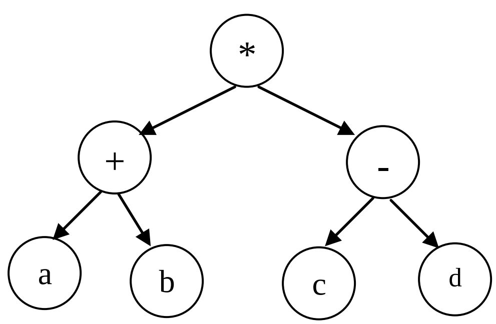
Una de las aplicaciones de árboles binarios son los llamados *árboles de expresión*.

Una expresión es una secuencia de componentes léxicos (tokens), que siguen reglas preescritas. Un token puede ser un operador o un operando.

Las propiedades de un árbol de expresión son las siguientes:

- Cada hoja es un operando
- El nodo raíz y los nodos internos son operadores
- Los subárboles son sub-expresiones en las que el nodo raíz es un operador

La siguiente figura muestra un ejemplo de un árbol de expresión de la expresión (a+b) * (c-d)

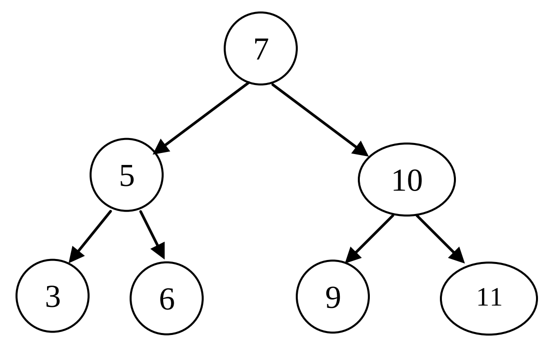


Árboles Binarios de búsqueda

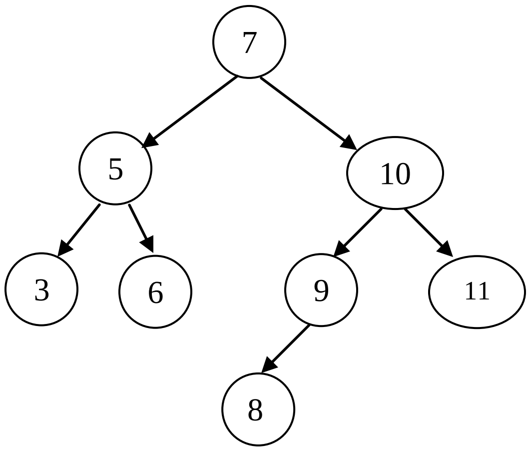
Un árbol binario de búsqueda es un árbol en el que todo nodo existente tiene un **sólo** elemento y cumple lo siguiente:

- ✓ todas las claves del subárbol izquierdo son menores que la raíz,
- ✓ todas las claves del subárbol derecho son mayores que la raíz,
- ✓ los subárboles izquierdo y derecho son también árboles de búsqueda.

Los nodos insertados en árboles de búsqueda binarios se insertan como hojas. Realizarlo de otro modo no solo no mejoraría la eficiencia buscada, sino que además habría que reajustar el árbol tras cada inserción. La figura muestra un ejemplo de un árbol de búsqueda de número ordenados.



Por ejemplo, al insertar la clave 8, el árbol de la figura anterior, quedaría de la siguiente forma:



Recorridos

Muchas de las operaciones del TDA – Árbol Binario implican recorrer o visitar cada uno de los nodos del árbol, ya sea para insertar, eliminar, visitar o buscar un elemento de una forma eficiente.

Existen en general cuatro formas de hacerlo, tres de naturaleza recursiva y uno más de naturaleza iterativa.

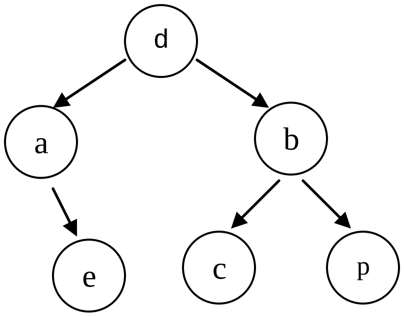
- *Recorrido en PreOrden (u orden previo):* Iniciando en la raíz, primero se visita ésta, luego se hace un recorrido en PreOrden del subárbol Izquierdo y luego en el subárbol derecho, también en PreOrden.

- *Recorrido en InOrden (orden simétrico):* Iniciando en la raíz, primero se efectúa un recorrido en InOrden en el subárbol izquierdo, luego se visita la raíz, y luego se visita el subárbol derecho también en InOrden.

- *Recorrido en PostOrden (u orden posterior)*: Iniciando en la raíz, primero se visita en PostOrden el subárbol izquierdo, luego el subárbol derecho, también en PostOrden, y por último se visita la raíz.

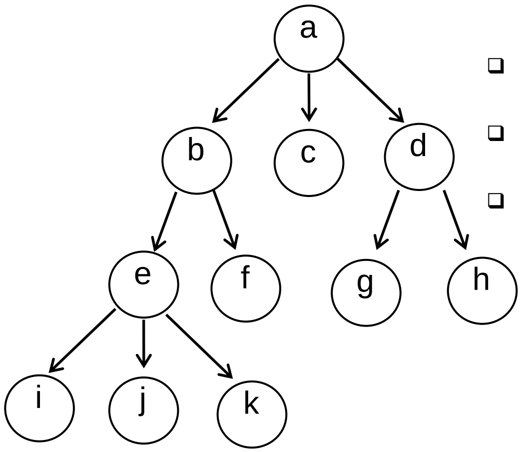
- *Recorrido por niveles:* Iniciando en la raíz, primero se visita la raíz, y luego se visitan los elementos del segundo nivel de izquierda a derecha, seguidos por los del nivel 3 en el mismo orden, y así sucesivamente hasta terminar de visitar todos los elementos.

Como ejemplo consideremos el siguiente árbol.



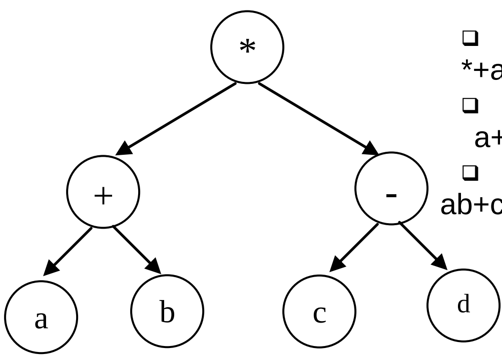
- PreOrden: d a e b c p
- InOrden: a e d c b p
- PostOrden: e a c p b d
- Niveles: d a b e c p

Otro ejemplo:



- PreOrden:
  a b e i j k f c d g h
- InOrden:
  i e j k b f a c g d h
- Postorden:
  i j k e f b c g h d a

El siguiente ejemplo, muestra el recorrido en un árbol de expresión.



- PreOrden:
  *+ab-cd (expresión Prefija)
- InOrden:
  a+b*c-d (expresión Infija)
- Postorden:
  ab+cd-* (expresión Postfija)