Teoría de Grafos y Algoritmia
Parte 1: Teoría de Arboles

Topicos

- Introducción.
- Definición
- Propiedades
- Árboles con raíz
  - Árbol binario
    - Recorridos

Teoría de Árboles – Introducción

Continuando con nuestro estudio de la Teoría de Grafos nos centraremos en un tipo especial de Grafo llamado Árbol.

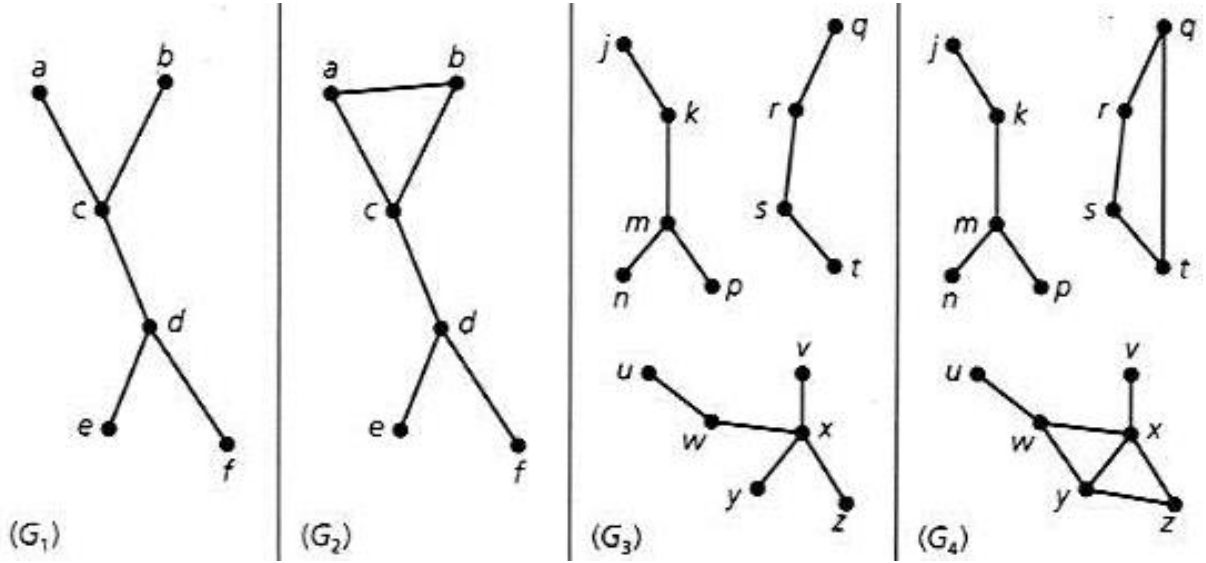
Los árboles fueron utilizados por primera vez en 1847 por Gustav Kirchhoff en su trabajo de redes eléctricas, aunque posteriormente fueron desarrollados y definidos de nuevo por Arthur Cayley.

Con la aparición de las computadora se encontraron nuevas aplicaciones para los árboles.

Teoría de Árboles – Definición

Sea $G = (V, E)$ un grafo no dirigido sin lazos. El grafo G es un Árbol si G es conexo y no tiene ciclos.

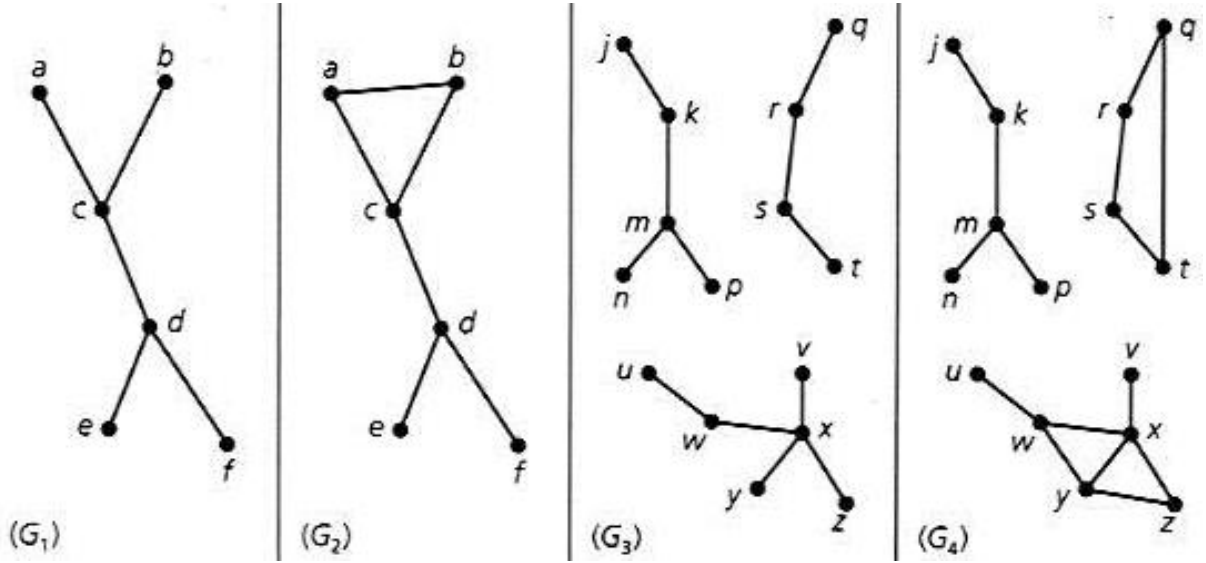
Aquí podemos ver algunos ejemplos de grafos. ¿Cuáles son árboles y cuáles no? ¿Por qué?



Teoría de Árboles – Definición

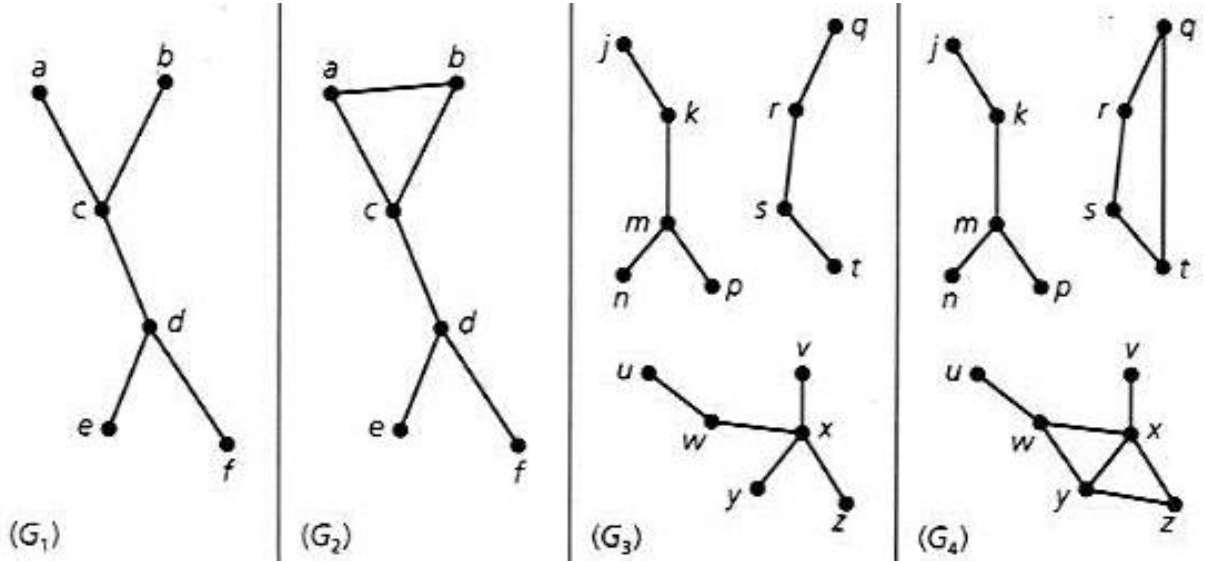
Aquí podemos ver que G_1 es un árbol y G_2 no por tener un ciclo. El grafo G_3 no es conexo, por lo que no puede ser un árbol. Sin embargo, cada componente de G_3 es un árbol; en ese caso es un Bosque.

Si un grafo es un árbol, escribimos T en lugar de G, para remarcar esta estructura.



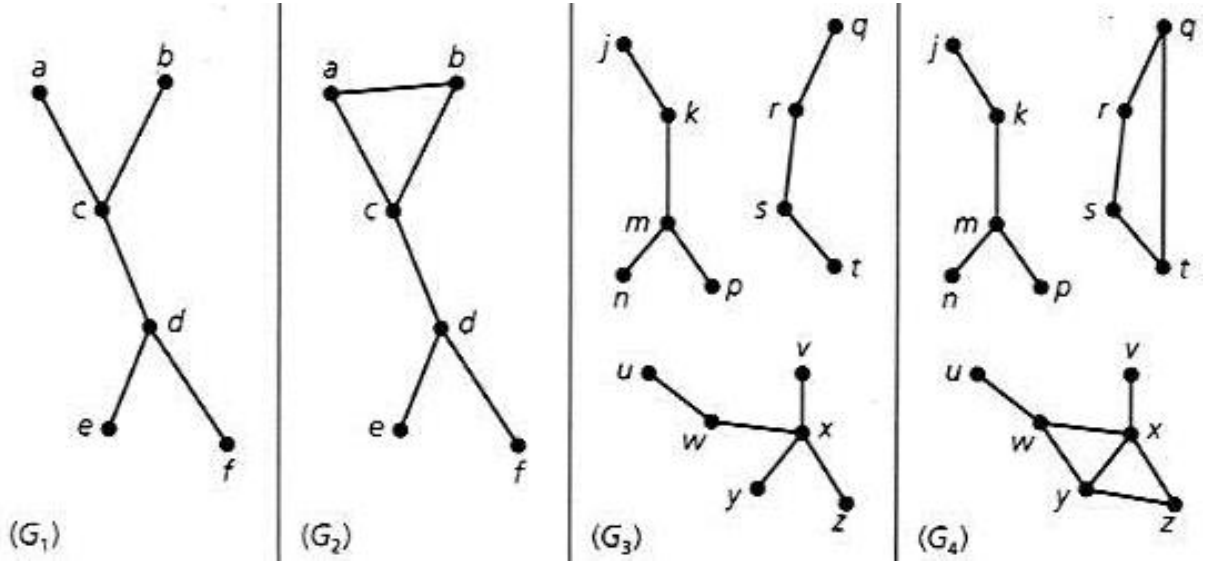
Teoría de Árboles – Definición

Aquí podemos ver que G_1 es un subgrafo de G_2, que G_1 contiene todos los vértices de G_2 y que G_1 es un árbol. En este caso, diremos que G_1 es un Árbol recubridor de G_2. Por lo tanto un árbol recubridor de un grafo conexo es un subgrafo recubridor que también es un árbol.



Teoría de Árboles – Definición

Podemos pensar en un árbol recubridor como aquel que proporciona una conectividad mínima para el grafo y, como el esqueleto mínimo que une los vértices. Análogamente podemos decir que G_3 es un Bosque recubridor para el grafo G_4.



Teoría de Árboles – Propiedades

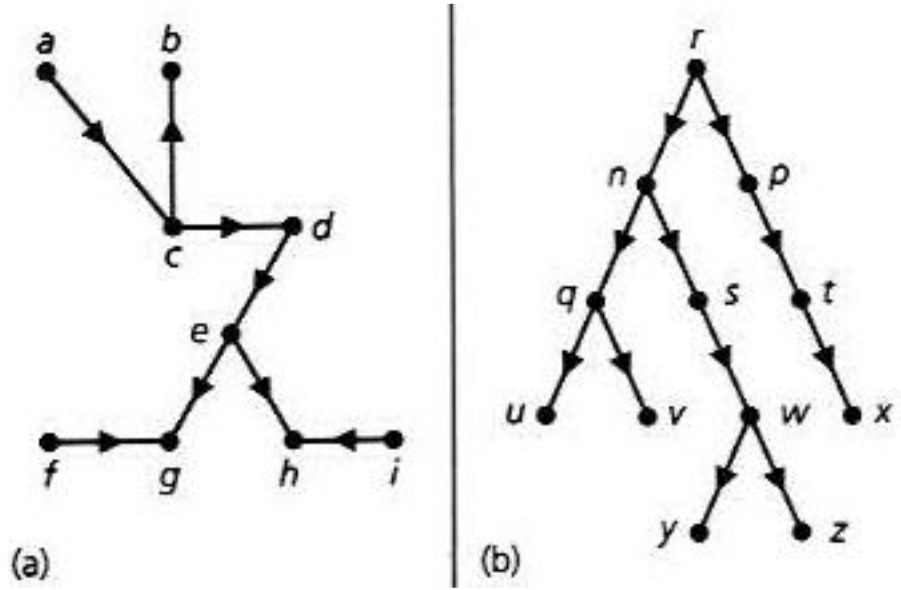
- Si a y b son vértices distintos en un árbol $T = (V, E)$, entonces hay un único camino que conecta estos vértices.
- Si $G = (V, E)$ es un grafo no dirigido, entonces G es conexo si y sólo si G tiene un árbol recubridor.
- Para todo árbol $T = (V, E)$, $|V| = |E| + 1$.
- Para todo árbol $T = (V, E)$, $|V| \geq 2$, entonces T tiene al menos dos vértices colgantes (vértices de grado 1).

Teoría de Árboles – Árboles con raíz

Si G es un grafo dirigido, entonces G es un Árbol dirigido si el grafo no dirigido asociado con G es un árbol. Si G es un árbol dirigido, G es un Árbol con raíz si existe un único vértice r en G, llamado la raíz tal que: el grado de entrada de $r = \delta^+(r) = 0$ y, para todos los demás vértices v, el grado de entrada de $v = \delta^+(v) = 1$.

Teoría de Árboles – Árboles con raíz

El árbol de la parte (a) es dirigido pero sin raíz; el árbol de la parte (b) tiene a r como raíz.

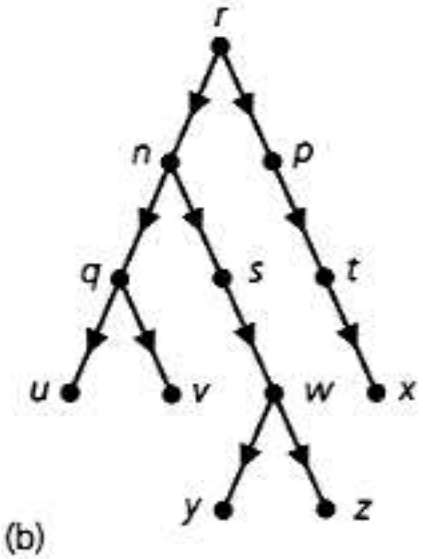


Teoría de Árboles – Árboles con raíz

Graficaremos los árboles con raíz como el de la figura de la parte (b), pero con la convención de que las direcciones van del nivel superior al inferior.

En un árbol con raíz, un vértice v con grado de salida $\delta^{-}(v) = 0$ es una Hoja o Vértice terminal. En nuestro ejemplo, los vértices u, v, y, z, x son hojas.

Todos los demás vértices es decir, los que no son Hojas, incluida la raíz, se llaman Vértices internos.



Teoría de Árboles – Árboles con raíz

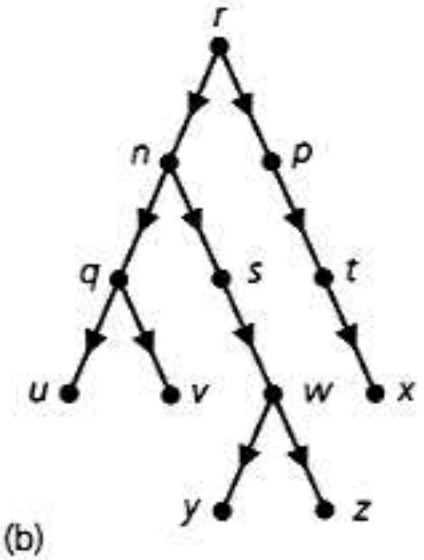
El camino desde la raíz r hasta el vértice s en nuestro ejemplo es de longitud 2, por lo que decimos que s está en el Nivel 2. Análogamente x está en el Nivel 3 mientras que y está en el Nivel 4 .

Decimos que s es un Hijo de n y, n es Padre de s.

Los vértices w, y y z son Descendientes de s, n y r; mientras que s, n y r son Ascendientes de w, y y z.

Dos vértices con un padre en común diremos que son Hermanos. Por ejemplo, q y s lo son ya que su padre es n.

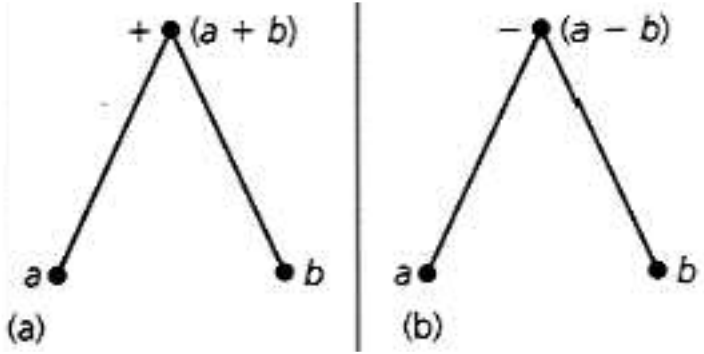
Si es v_1 cualquier vértice del árbol, el Subárbol en v_1 es el subgrafo inducido por la raíz v_1 y todos sus descendientes.



Teoría de Árboles – Árboles con raíz – Árbol Binario

Un árbol con raíz decimos que es un Árbol Binario si para cada vértice v, su grado de salida es $\delta^-(v) = 0$, 1 o 2; es decir, si v tiene como mucho 2 hijos.

Si $\delta^-(v) = 0$ o 2 para todo vértice, entonces decimos que el árbol con raíz es un Árbol binario completo. Este árbol puede representar una operación binaria, como se puede ver en la figura:



Teoría de Árboles – Árboles con raíz – Árbol Binario

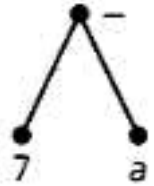
Usando esta idea podemos ver que para construir la expresión aritmética:

$$((7 - a)/5) \times ((a + b) \uparrow 3)$$

donde ↑ representa la potencia.

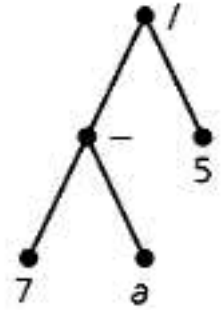
En este caso vamos a construir el árbol que representa la expresión de arriba hacia abajo.

En primer lugar construimos un subárbol para la expresión $7 - a$:



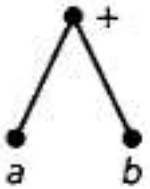
Teoría de Árboles – Árboles con raíz – Árbol Binario

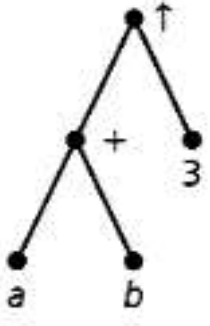
Esto se incorpora (como el subárbol izquierdo de $/$) en el árbol binario con raíz correspondiente a la expresión $(7 - a)/5$ de la figura:



Teoría de Árboles – Árboles con raíz – Árbol Binario

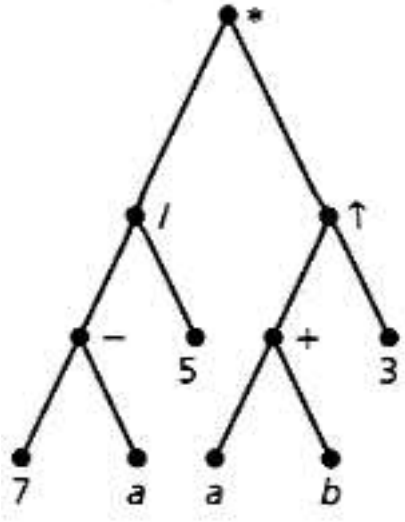
Luego, de modo similar, construimos los árboles binarios correspondientes a la expresión: $(a + b) \uparrow 3$.





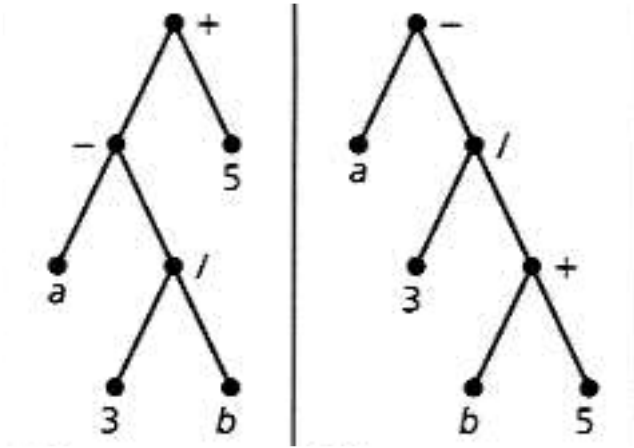
Teoría de Árboles – Árboles con raíz – Árbol Binario

Obteniendo, finalmente, el árbol de la expresión aritmética buscado:



Teoría de Árboles – Árboles con raíz – Árbol Binario

Los siguientes árboles binarios completos representan las expresiones: $(a - (3/b)) + 5$ y $a - (3/(b + 5))$



Vamos a ver a continuación 3 formas de recorrer los vértices que forman un árbol binario. Las 3 formas lo recorren de la misma manera, lo que cambia en cada una de ellas es el momento en que se muestra la información del vértice en que estamos.

Cada uno de estos recorridos están escritos con un algoritmo que tiene la característica de ser recursivo, es decir, se llama a sí mismo. Esto es porque el algoritmo está escrito como una función que toma como dato la raíz del árbol que está procesando.

Debido a la expresión resultante obtenida es que estos recorridos se denominan: preorder, inorder y postorder.

Algoritmo $preorder(s)$

Si s es vacío $Entonces$

$Retornar$

$Mostrar$ el valor de s

$L \coloneqq hijo\ izquierdo\ de\ s$

$preorder(L)$

$R \coloneqq hijo\ derecho\ de\ s$

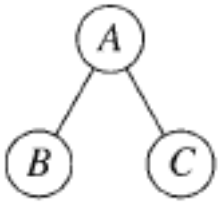
$preorder(R)$

Ahora, analizaremos el algoritmo para algunos casos sencillos:

Si el árbol binario es vacío, nada se procesa pues, en este caso, el algoritmo termina simplemente en la línea 2.

Supongamos que la entrada consta de un árbol con un único vértice. Hacemos s igual a la raíz y llamamos a $preorder(s)$. Como s no es vacío pasamos a la línea 3 donde mostramos el valor de la raíz. En la línea 5, llamamos a $preorder$ con s siendo el hijo izquierdo (vacío) de la raíz. Sin embargo hemos dicho que en la entrada de un árbol vacío $preorder$ no muestra información alguna. De manera análoga, en la línea 7 utilizamos un árbol vacío como entrada de $preorder$ y nada se procesa. Así cuando la entrada consta de un árbol con un único vértice, mostramos la raíz y terminamos.

Ahora, supongamos que la entrada es el siguiente árbol:

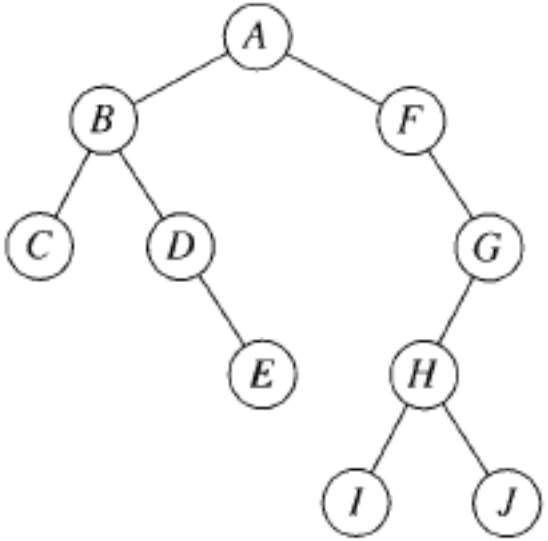


Hacemos s igual a la raíz y llamamos a $preorder(s)$. Como s no es vacío pasamos a la línea 3 donde mostramos el valor de la raíz (A). En la línea 5, llamamos a $preorder$ con s siendo el hijo izquierdo de la raíz:

Acabamos de ver que si la entrada de $preorder$ consta de un único vértice, $preorder$ muestra ese vértice. Así, a continuación, mostramos el vértice B. De manera análoga en la línea 7, mostramos el vértice C.

Cuando el algoritmo termina, la salida obtenida es : ABC.

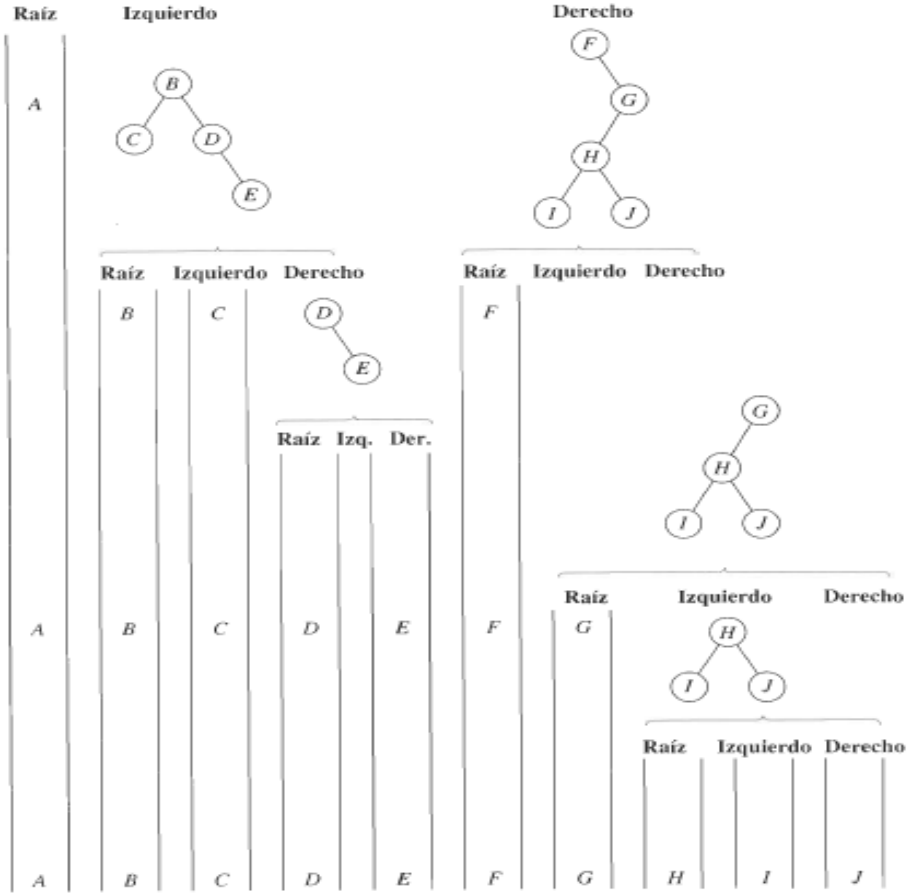
¿Cuál sería la salida del algoritmo $preorder$ con el siguiente árbol?



Teoría de Árboles – Árboles con raíz – Árbol Binario - Recorridos

Si vamos siguiendo las líneas $3-7$ $(raíz/izquierda/derecha)$, del algoritmo $preorder$ el recorrido se muestra como indica la figura.

Así, la salida del algoritmo es : $ABCDEFGHIJ$.



Los recorridos $inorder$ y $postorder$ se obtienen solamente cambiando la posición de la línea 3 (raíz). Los prefijos pre, in y $post$ se refieren a la posición de la raíz en el recorrido; es decir, $preorder$ significa que primero va la raíz, $inorder$ quiere decir que la raíz va luego de mostrar el subárbol de la izquierda y, $postorder$ significa que la raíz va al final luego de mostrar el subárbol de la izquierda y, el de la derecha.

Algoritmo $inorder(s)$

Si s es vacío $Entonces$

$Retornar$

$L \coloneqq hijo\ izquierdo\ de\ s$

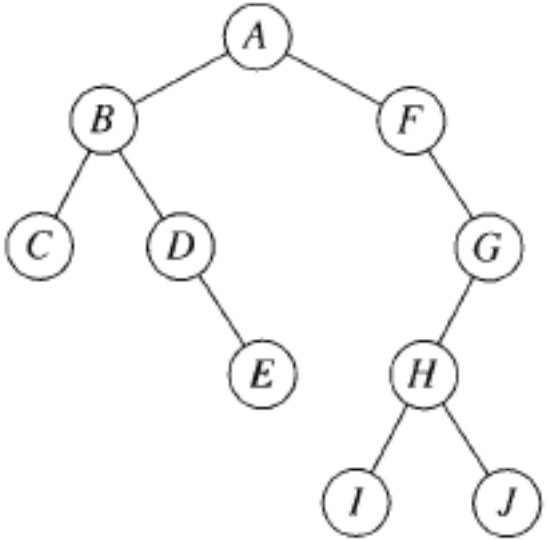
$inorder(L)$

$Mostrar$ el valor de s

$R \coloneqq hijo\ derecho\ de\ s$

$inorder(R)$

¿Cuál sería la salida del algoritmo $inorder$ con el siguiente árbol?



Si vamos siguiendo las líneas $3 - 7$ $(raíz/izquierda/derecha)$, del algoritmo $inorder$ obtenemos como salida: $CBDEAFIHJG$.

Algoritmo $postorder(s)$

Si s es vacío $Entonces$

$Retornar$

$L \coloneqq hijo\ izquierdo\ de\ s$

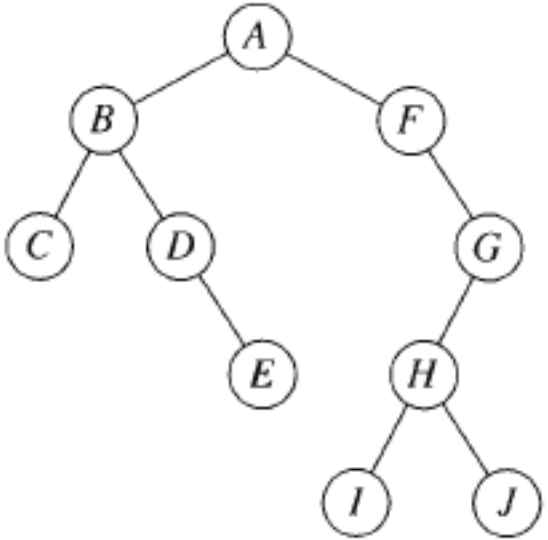
$postorder(L)$

$R \coloneqq hijo\ derecho\ de\ s$

$postorder(R)$

$Mostrar$ el valor de s

¿Cuál sería la salida del algoritmo $postorder$ con el siguiente árbol?



Si vamos siguiendo las líneas $3-7$ $(raíz/izquierda/derecha)$, del algoritmo $postorder$ obtenemos como salida: $CEDBIJHGFA$.

Si volvemos al árbol del slide 16 cuando vimos expresiones aritméticas y aplicamos los recorridos dados, la salida sería:

- del algoritmo $preorder$ es: $\times \div - 7\ \mathrm{a}\ 5 \uparrow + \mathrm{a}\ b\ 3$.
- del algoritmo $inorder$ es $7 - \mathrm{a} \div 5 \times \mathrm{a} + b \uparrow 3$
- del algoritmo $postorder$ es $7\ \mathrm{a} - 5 \div \mathrm{a}\ b + 3 \uparrow \times$.