2.1.4. Un ejemplo de uso: árboles de expresiones

Los árboles binarios se utilizan para almacenar expresiones aritméticas en memoria, esencialmente en compiladores de lenguajes de programación. Una expresión es una secuencia de tokens (componentes de léxicos que siguen unas reglas establecidas). Un token puede ser un operando o bien un operador.

Los paréntesis no se almacenan en el árbol, pero están implicados en la forma del árbol, como puede apreciarse en la Figura 14:

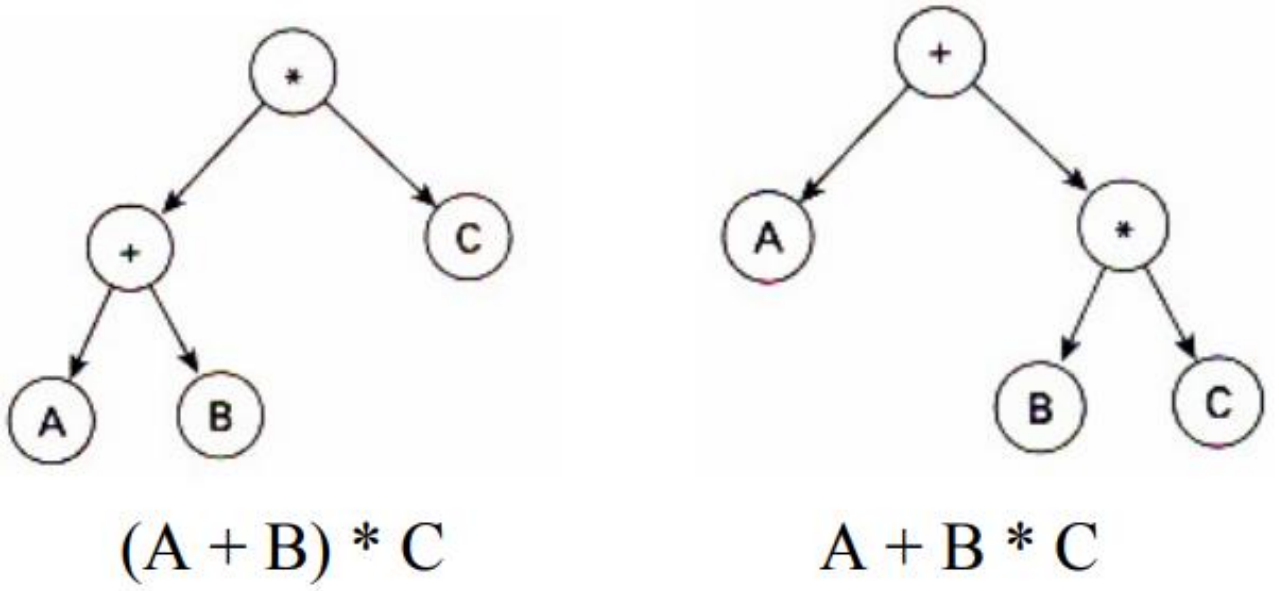


Figura 14

La Figura 15 representa la expresión infija **`a * (b + c) + d`** junto a su árbol de expresión. El nombre de infija es debido a que los operadores se sitúan entre los operandos.

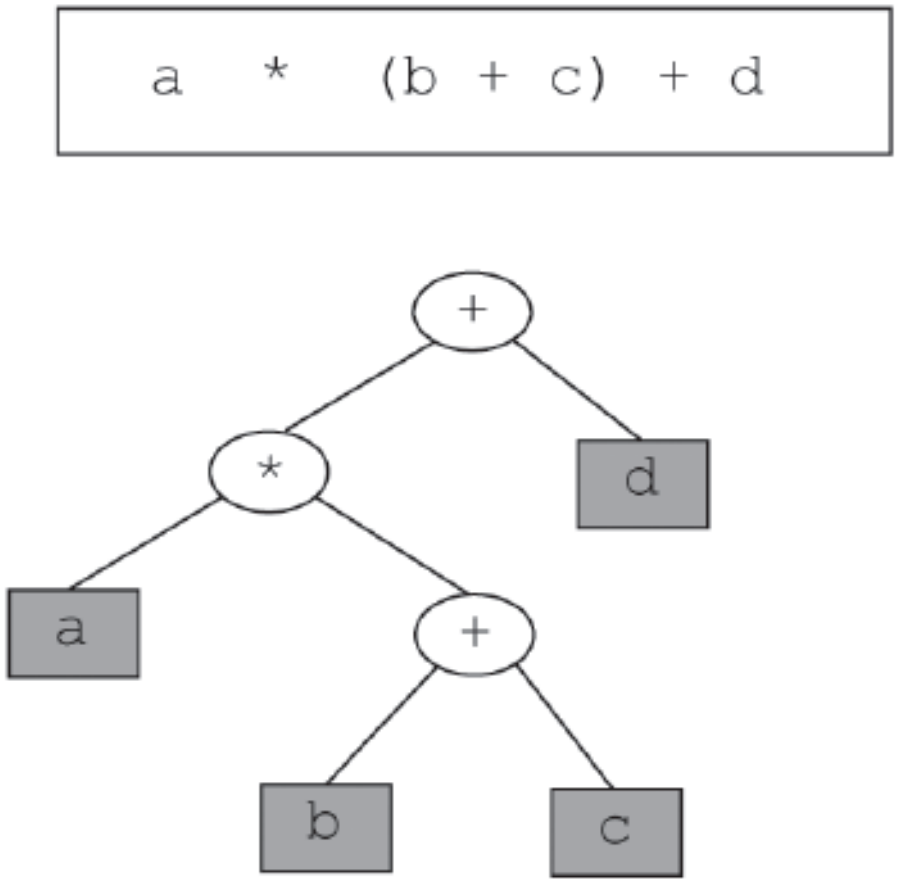


Figura 15

Un árbol de expresión es un árbol binario con las siguientes propiedades:

1. Cada hoja es un operando.
2. Los nodos raíz y los nodos internos son operadores.
3. Los subárboles son sub-expresiones cuyo nodo raíz es un operador.

Los árboles binarios se utilizan para representar expresiones en memoria, esencialmente en compiladores de lenguajes de programación. Se observa que los paréntesis de la expresión no aparecen en el árbol, pero están implicados en su forma, y esto resulta muy interesante para la evaluación de la expresión.

Si se supone que todos los operadores tienen dos operandos, se puede representar una expresión mediante un árbol binario cuya raíz contiene un operador y cuyos subárboles izquierdo y derecho son los operandos izquierdo y derecho, respectivamente. Cada operando puede ser una letra (x, y, a, b, etc.) o una subexpresión representada como un subárbol.

La Figura 16 muestra un árbol cuya raíz es el operador *,
su subárbol izquierdo representa la subexpresión (x + y)
y su subárbol derecho representa la subexpresión (a - b).

El nodo raíz del subárbol izquierdo contiene el operador (+) de la subexpresión izquierda y el nodo raíz del subárbol derecho contiene el operador (-) de la subexpresión derecha. Todos los operandos letras se almacenan en nodos hojas.

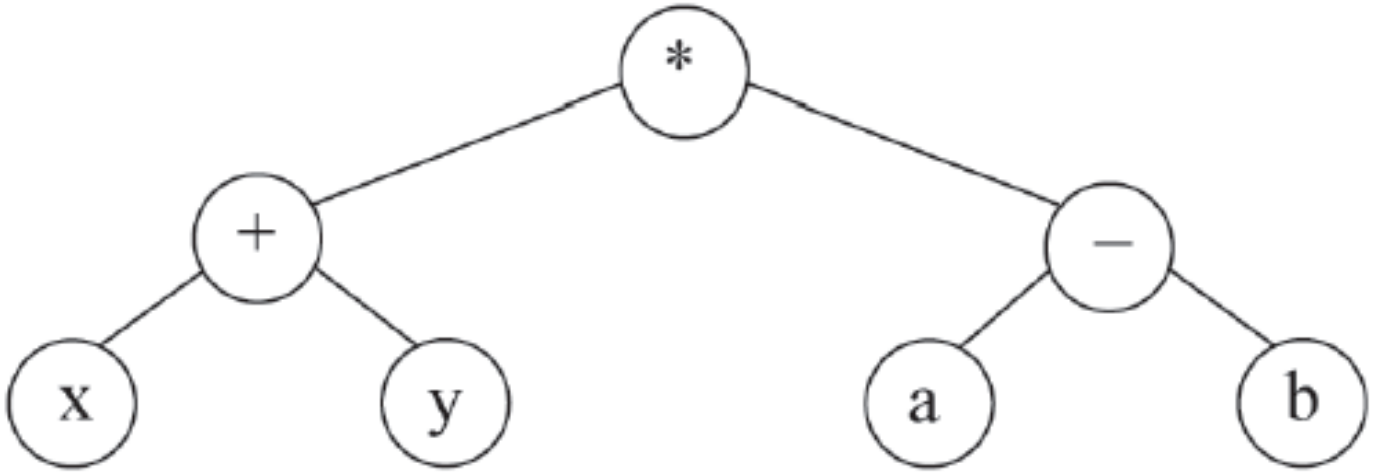


Figura 16

Utilizando el razonamiento anterior, la expresión **`(x * (y - z)) + (a - b)`** con paréntesis alrededor de las subexpresiones, forma el árbol binario de la Figura 17.

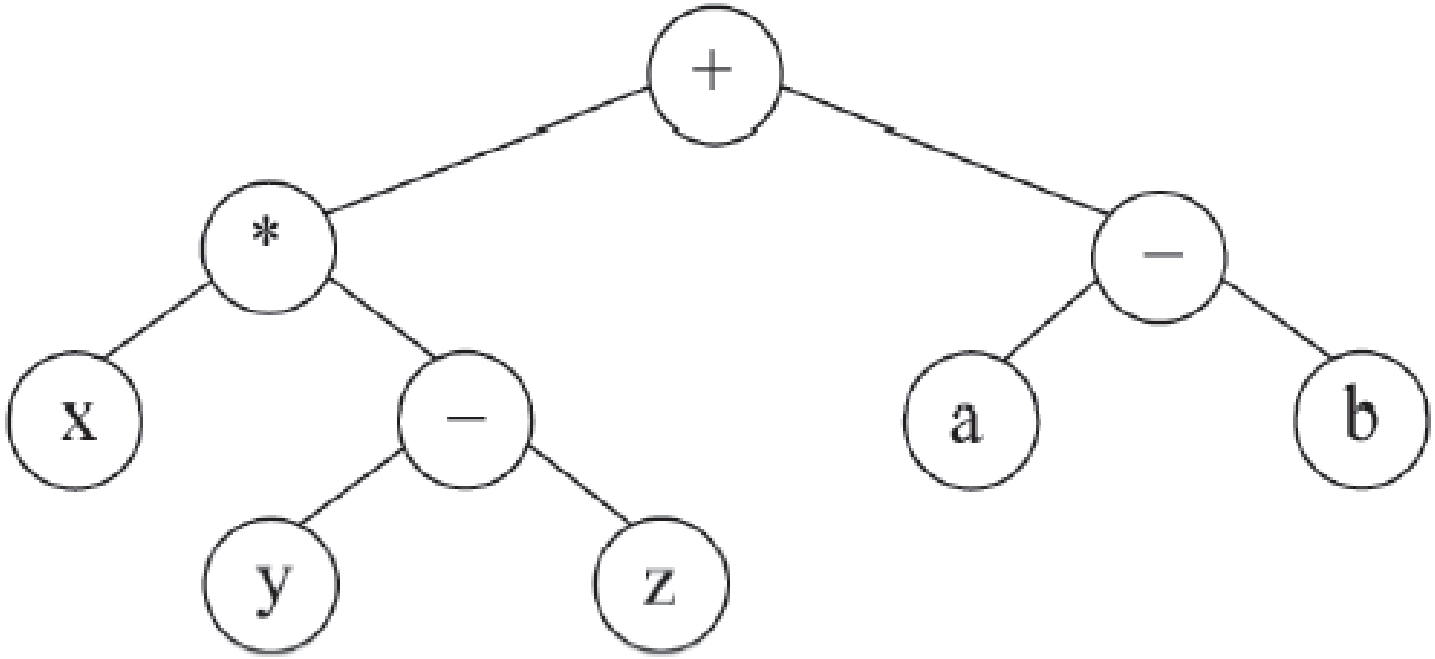


Figura 17

Ejemplos:

Deducir las expresiones que representan los árboles binarios de la Figura 18

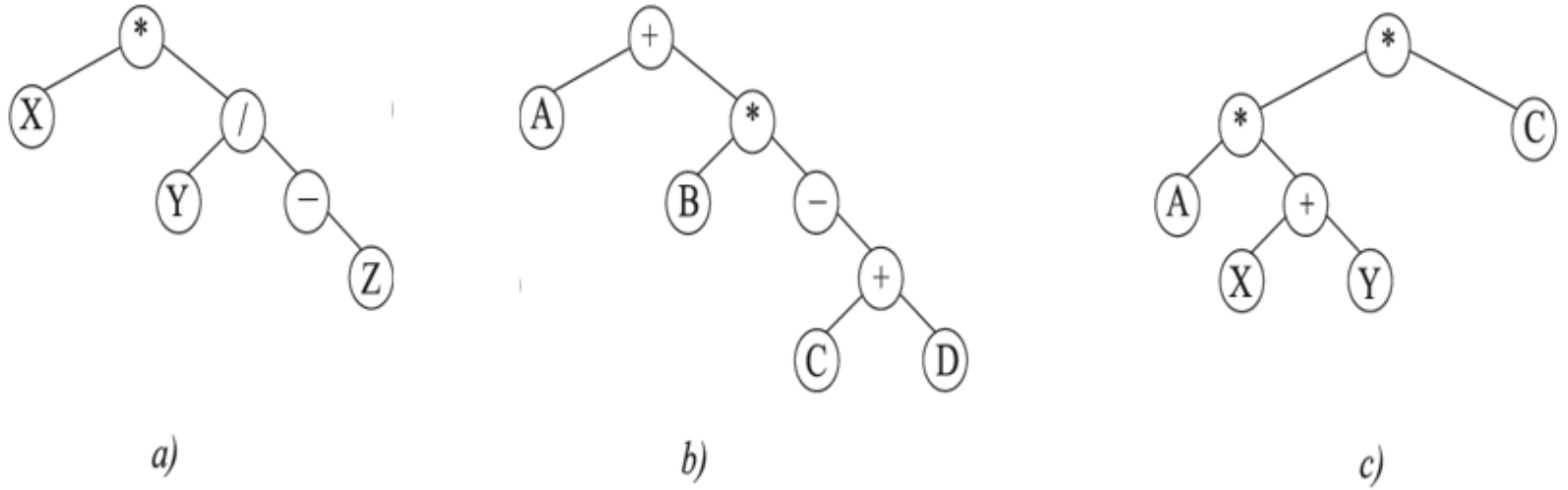


Figura 18

a) X * (Y / (-Z)) **b) A + B * (-(C + D))** **c) (A * (X + Y)) * C**

Dibujar la representación en árbol binario de cada una de las siguientes expresiones:

a) Y * X /(A + B)* C **b) X * Y / A + B * C**

Las representaciones aparecen en la Figura 19 a) y 19 b):

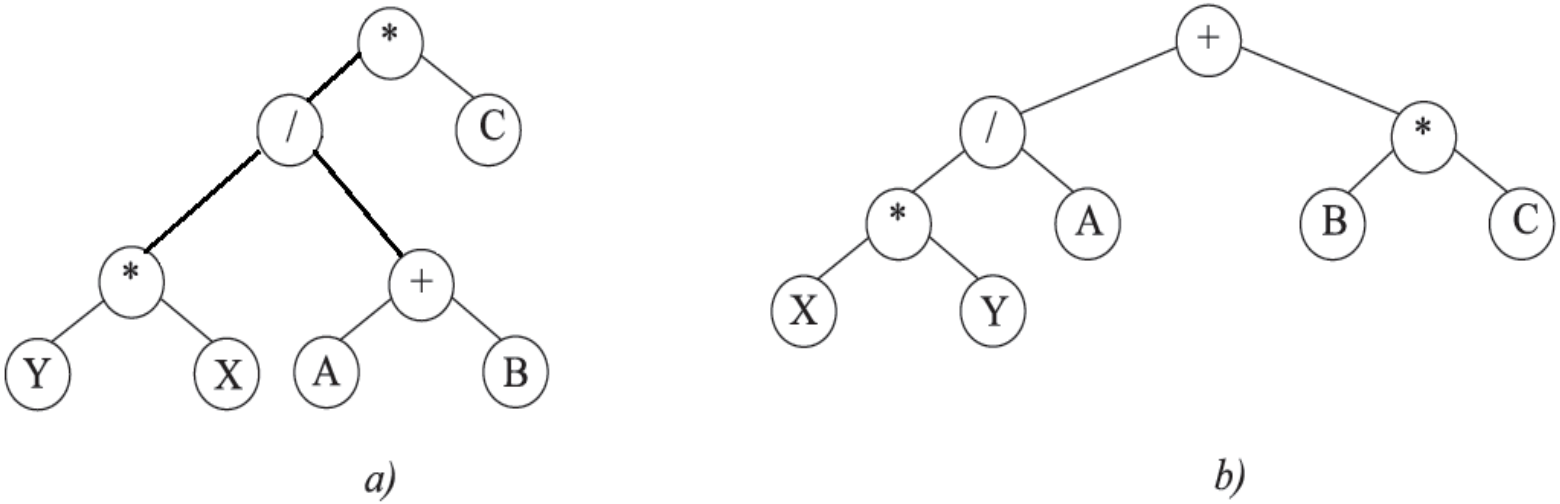


Figura 19

Representar la siguiente expresión con operadores lógicos mediante un árbol binario:

(¬ (p ^ q)) **v** (¬p **v** ¬q)

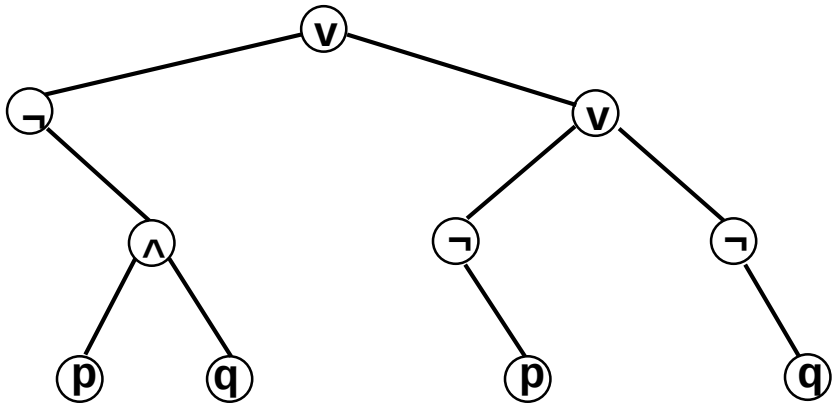


Figura 19 c)

Representar la siguiente expresión mediante un árbol binario:

$$(x + y)^2 + (x - 4) / 3$$

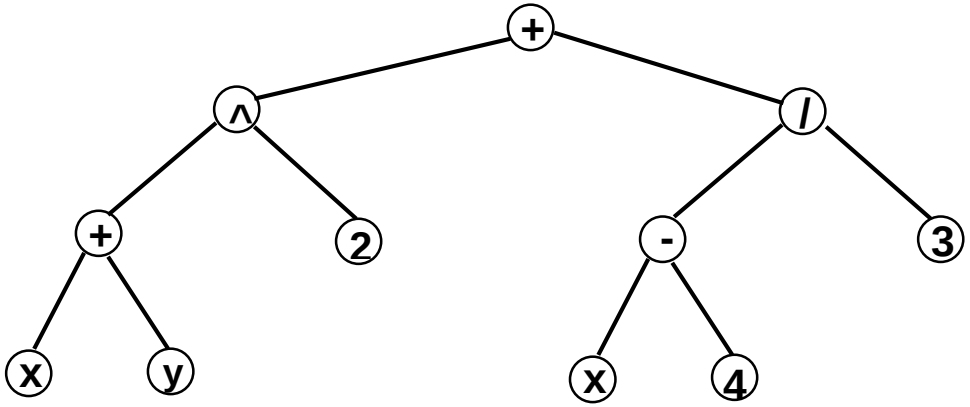


Figura 19 d)

2.1.5. El TAD Arbol Binario (ArBin). La estructura de árbol binario constituye un tipo abstracto de datos; las operaciones básicas que definen el TAD árbol binario son las siguientes:

Tipo de dato: Dato que se almacena en los nodos del árbol.

Operaciones:

CrearArbol	Inicia el árbol como vacío.
Construir	Crea un árbol con un elemento raíz y dos ramas, izquierda y derecha que son a su vez árboles.
EsVacio	Comprueba si el árbol no tiene nodos.
Raiz	Devuelve el nodo raíz.
Izquierdo	Obtiene la rama o subárbol izquierdo de un árbol dado.
Derecho	Obtiene la rama o subárbol derecho de un árbol dado.
Borrar	Elimina del árbol el nodo con un elemento determinado.
Pertenece	Determina si un elemento se encuentra en el árbol.

2.1.6. Estructuras y Representaciones. Un árbol binario se construye con nodos. Cada nodo debe contener el campo dato (datos a almacenar) y dos campos de enlace (apuntador), uno al subárbol izquierdo (izquierdo, izdo) y otro al subárbol derecho (derecho, dcho). El valor null indica un árbol o un subárbol vacío. Se puede observar (Figura 20) que los nodos de un árbol binario que son hojas se caracterizan por tener sus dos campos de enlace a null.

Estructura dinámica con enlaces a los hijos.

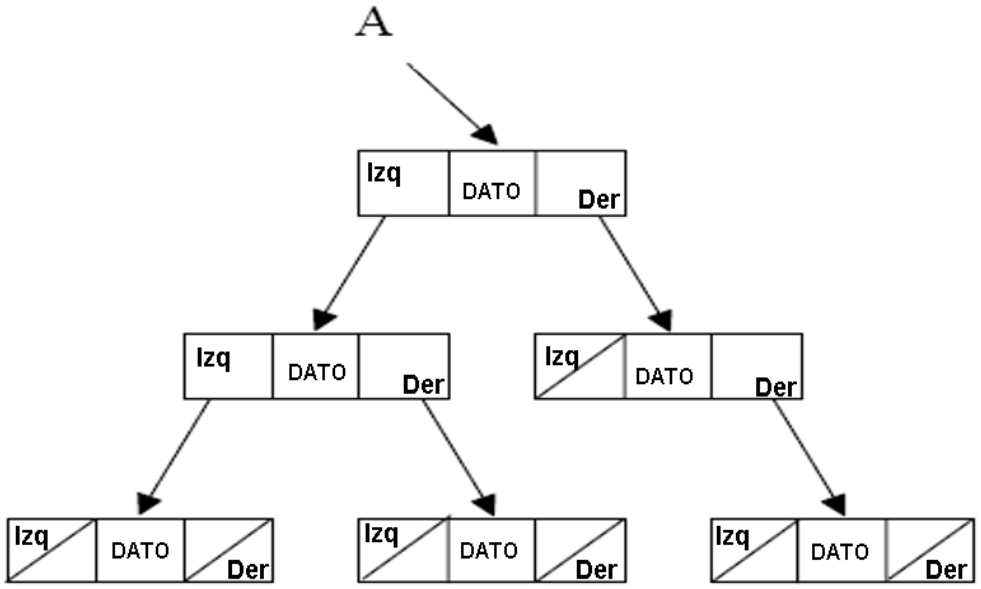


Figura 20

Pros:

- Permite obtener una implementación fácil y rápida para todas las operaciones.
- Permite aplicar las operaciones de forma recursiva.

Contra:

- No permite obtener caminos ascendentes.

Estructura dinámica con enlaces a los hijos y al padre.

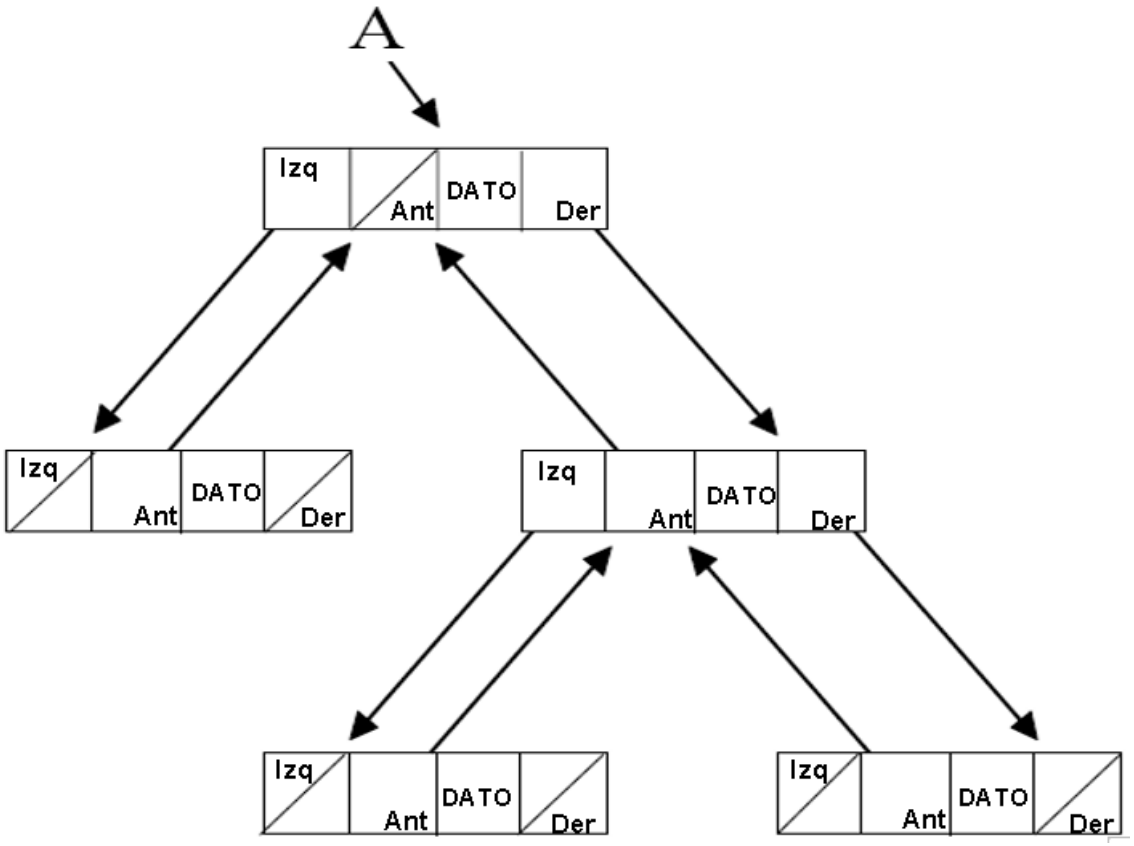


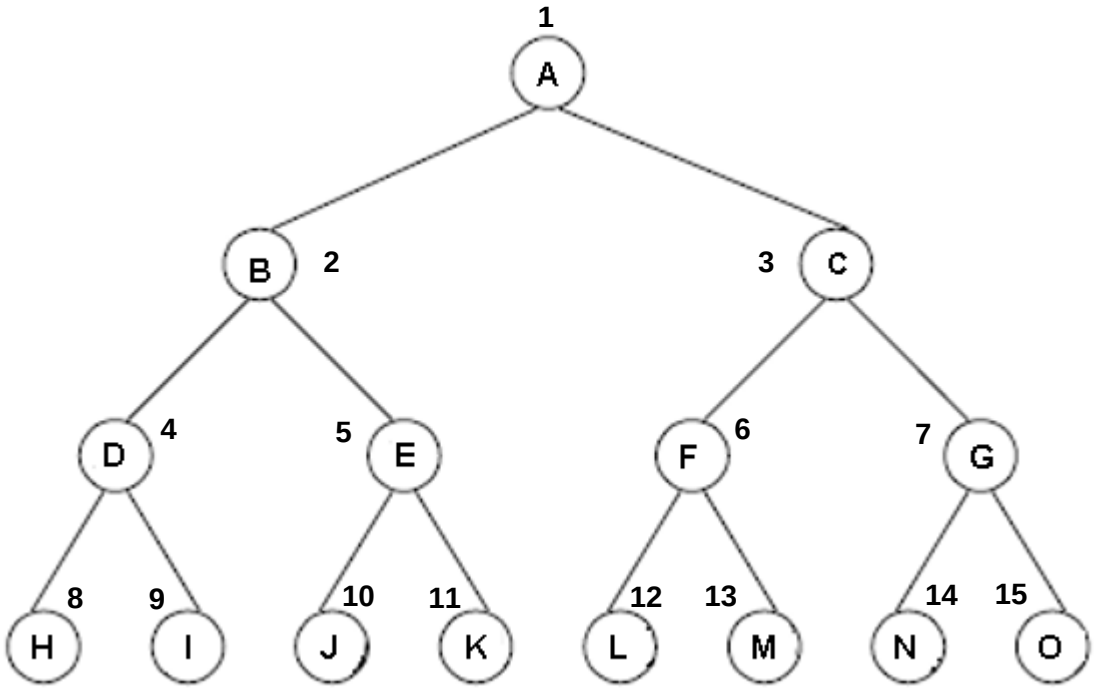
Figura 21

Pros:

- Permite efectuar retrocesos en el árbol sin necesidad de salir de puntos profundos en recursividad.
- Permitir recorridos fáciles de forma no recursiva.
- Permite obtener caminos ascendentes a partir de cualquier nodo.

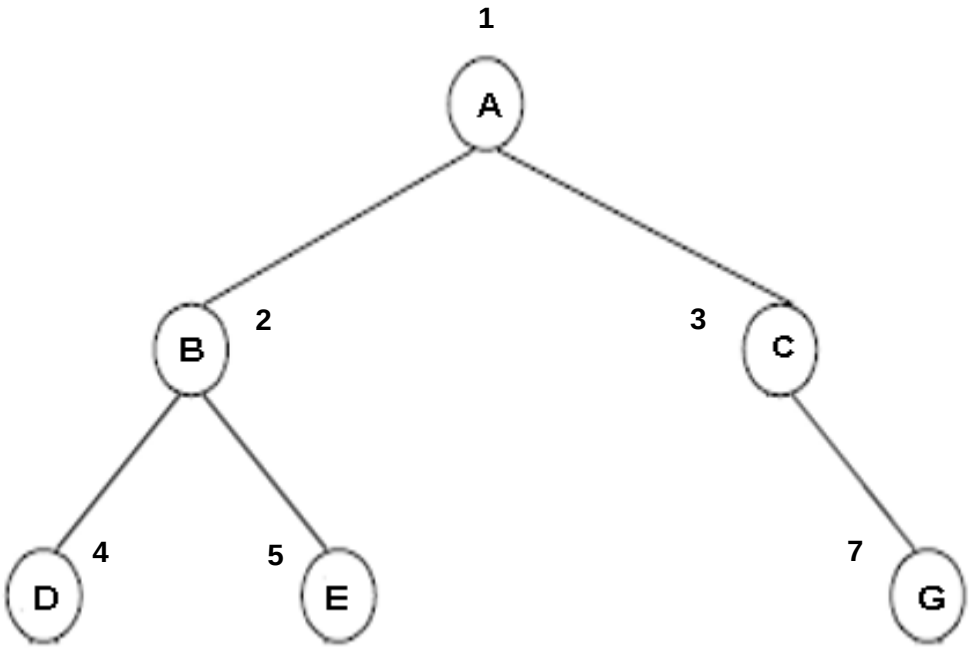
Representación estática dispersa en amplitud.

Se almacenan los subárboles siguiendo el recorrido en amplitud, de manera que cada nodo ocupa una posición en la tabla, dependiendo de la posición que ocupa en el árbol, suponiendo que está lleno (se respetan las posiciones no ocupadas). La posición que ocupará cada nodo se aprecia en la Figura 22.



POS	VAL	VACIO
1	A	F
2	B	F
3	C	F
4	D	F
5	E	F
6	F	F
7	G	F
8	H	F
9	I	F
10	J	F
11	K	F
12	L	F
13	M	F
14	N	F
15	O	F

Figura 22



POS	VAL	VACIO
1	A	F
2	B	F
3	C	F
4	D	F
5	E	F
6	-	V
7	G	F

Figura 23

Características:

- Acceso directo a cualquier subárbol de un determinado nivel utilizando la correspondencia entre niveles del árbol y segmentos de la tabla (nivel **i** : segmento[2^{i} , $2^{(i+1)} - 1$])

  Del ejemplo en la página anterior:

POS	VAL	VACIO	NIVEL QUE OCUPA	
1	A	F	0	Segmento [2^{i} , $2^{(i+1)} - 1$]: [2^{0} , $2^{(0+1)} - 1$] = [1 , 1]
2	B	F	1	Segmento [2^{i} , $2^{(i+1)} - 1$]: [2^{1} , $2^{(1+1)} - 1$] = [2 , 3]
3	C	F	1	
4	D	F	2	Segmento [2^{i} , $2^{(i+1)} - 1$]: [2^{2} , $2^{(2+1)} - 1$] = [4 , 7]
5	E	F	2	
6	-	V	2	
7	G	F	2	

- Nos ahorramos la existencia de punteros; sin embargo, los nodos vacíos también ocupan espacio. Se empleará pues para árboles casi llenos, o completos.

- Será necesario un valor especial o campo adicional para indicar los nodos vacíos.

- Es muy cómoda para representar árboles completos que no se modifican, y sólo se consultan.

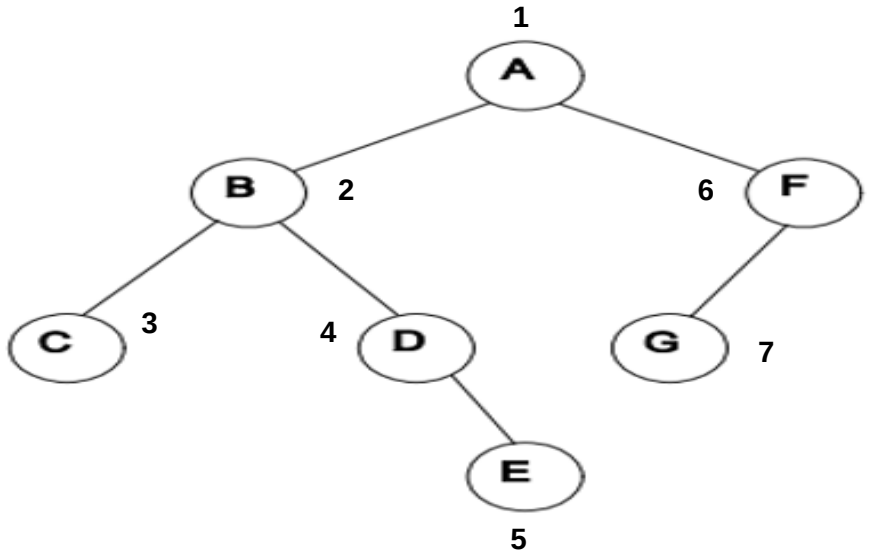
- Aparte de los problemas típicos de las estructuras estáticas, aquí se tiene el inconveniente de que el desperdicio de espacio es enorme en los árboles poco balanceados.

Representación estática mediante Cursores. La única diferencia entre el almacenamiento estático con cursores, y el dinámico con punteros, es que en el primero el número de nodos ya está predeterminado, es constante, y se hallan todos uno detrás de otro en secuencia, por lo que pueden ser denotados por su posición, también llamada cursor, en lugar de por un puntero. Asimismo, es necesario mantener una lista de nodos libres.

En definitiva, esta nueva alternativa consiste en emplear una tabla de registros de tres campos:

- Valor del nodo.
- Posición de la raíz del subárbol izquierdo.
- Posición de la raíz del subárbol derecho.

A continuación, aparece un ejemplo de almacenamiento de un árbol:



POS	VAL	HIJO IZQ	HIJO DER
1	A	2	6
2	B	3	4
3	C	0	0
4	D	0	5
5	E	0	0
6	F	7	0
7	G	0	0

Figura 24

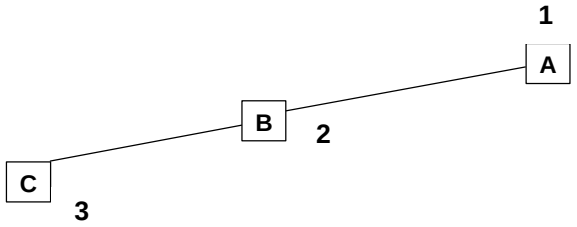
Como se podrá apreciar, los valores de la segunda columna leídos de arriba hacia abajo reflejan el recorrido del árbol en **PREORDEN**: A, B, C, D, E, F, G.

Este tipo de estructura tiene los problemas típicos en cuanto a la limitación en el número de nodos, desperdicio de espacio, acceso a posiciones en lugar de a direcciones, etc.

Ejercicio: ¿Cómo reconstruir el árbol a partir de la Tabla de Cursores?

1.Desde el nodo raíz recorra la tabla hasta que el valor hijo izq. Y hijo der. de un nodo sean cero. Así se puede identificar el primer nodo hoja. Puede ir identificando también en la tabla aquellos nodos cuyo valor de hijo der. sea cero.

POS	VAL	HIJO IZQ	HIJO DER
1	**A**	**2**	6
2	**B**	**3**	4
3	**C**	**0**	**0**



2. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

POS	VAL	HIJO IZQ	HIJO DER
1	**A**	**2**	6
2	**B**	**3**	**4**
3	**C**	**0**	**0**
4	**D**	**0**	**5**
5	**E**	**0**	**0**

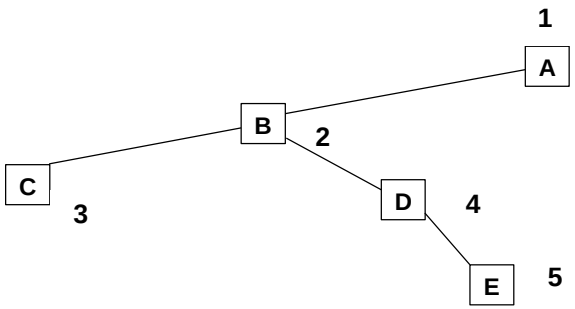
El Nodo detectado es el E.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo E, que ocupa la posición 5,
y ese valor 5 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 4, y esa fila le corresponde al nodo D.
DE MODO QUE E ES HIJO DERECHO DE D.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo D, que ocupa la posición 4,
y ese valor 4 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 2, y esa fila le corresponde al nodo B.
DE MODO QUE D ES HIJO DERECHO DE B.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo B, que ocupa la posición 2,
y ese valor 2 vuelve a aparecer en la columna 3 (HIJO IZQUIERDO) fila 1, y esa fila le corresponde al nodo A.
DE MODO QUE B ES HIJO IZQUIERDO DE A. (ESTO VERIFICA LA CONSISTENCIA).

No se retrocede más, debido a que ya se está en la Fila 1.



3. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

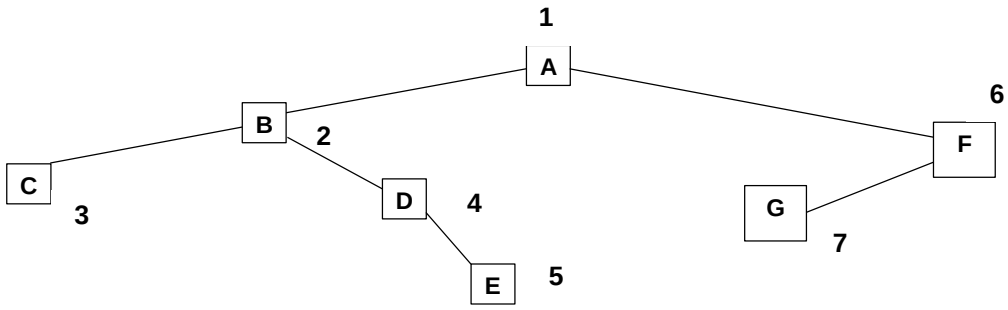
POS	VAL	HIJO IZQ	HIJO DER
1	**A**	**2**	**6**
2	**B**	**3**	**4**
3	**C**	**0**	**0**
4	**D**	**0**	**5**
5	**E**	**0**	**0**
6	**F**	**7**	**0**
7	**G**	**0**	**0**

El Nodo detectado es el G.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo G, que ocupa la posición 7,
y ese valor 7 vuelve a aparecer en la columna 3 (HIJO IZQUIERDO) fila 6, y esa fila le corresponde al nodo F.
DE MODO QUE G ES HIJO IZQUIERDO DE F.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo F, que ocupa la posición 6,
y ese valor 6 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 1, y esa fila le corresponde al nodo A.
DE MODO QUE F ES HIJO DERECHO DE A.

No se retrocede más, debido a que ya se está en la Fila 1.



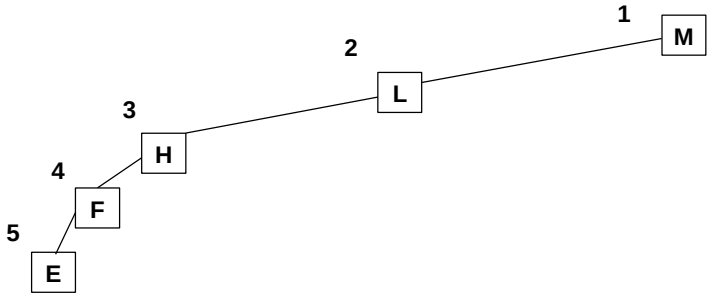
Dada la tabla que indica la representación estática mediante cursores, reconstruya el árbol.

POS	VAL	HIJO IZQ	HIJO DER
1	M	2	10
2	L	3	0
3	H	4	7
4	F	5	6
5	E	0	0
6	G	0	0
7	I	0	8
8	J	0	9
9	K	0	0
10	A	0	11
11	C	12	13
12	B	0	0
13	D	0	0

SOLUCIÓN:

1.Desde el nodo raíz recorra la tabla hasta que el valor hijo izq. Y hijo der. de un nodo sean cero. Así se puede identificar el primer nodo hoja. Puede ir identificando también en la tabla aquellos nodos cuyo valor de hijo der. sea cero.

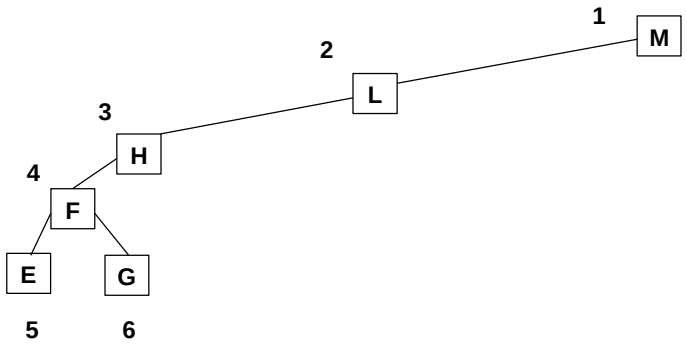
POS	VAL	HIJO IZQ	HIJO DER
1	**M**	**2**	10
2	**L**	**3**	**0**
3	**H**	**4**	7
4	**F**	**5**	6
5	**E**	**0**	**0**



2. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

El Nodo detectado es el G.
Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo G, que ocupa la posición 6, y ese valor 6 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 4, y esa fila le corresponde al nodo F. DE MODO QUE G ES HIJO DERECHO DE F.

POS	VAL	HIJO IZQ	HIJO DER
1	**M**	**2**	10
2	**L**	**3**	**0**
3	**H**	**4**	7
4	**F**	**5**	**6**
5	**E**	**0**	**0**
6	**G**	**0**	**0**



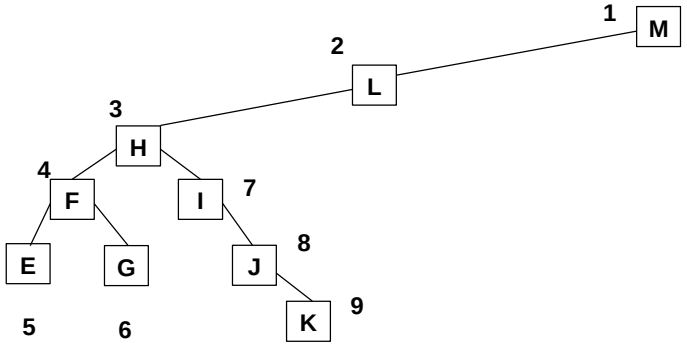
3. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

El siguiente nodo hoja que se encuentra es K cuya posición es la 9.
Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo K, que ocupa la posición 9,
y ese valor 9 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 8, y esa fila le corresponde al nodo J.
DE MODO QUE K ES HIJO DERECHO DE J.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo J, que ocupa la posición 8,
y ese valor 8 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 7, y esa fila le corresponde al nodo I.
DE MODO QUE J ES HIJO DERECHO DE I.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo I, que ocupa la posición 7,
y ese valor 7 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 3, y esa fila le corresponde al nodo H.
DE MODO QUE I ES HIJO DERECHO DE H.

POS	VAL	HIJO IZQ	HIJO DER
1	**M**	**2**	10
2	**L**	**3**	**0**
3	**H**	**4**	**7**
4	**F**	**5**	**6**
5	**E**	**0**	**0**
6	**G**	**0**	**0**
7	**I**	**0**	**8**
8	**J**	**0**	**9**
9	**K**	**0**	**0**



4. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

El siguiente nodo hoja que se encuentra es B cuya posición es la 12.

POS	VAL	HIJO IZQ	HIJO DER
1	**M**	**2**	**10**
2	**L**	**3**	**0**
3	**H**	**4**	**7**
4	**F**	**5**	**6**
5	**E**	**0**	**0**
6	**G**	**0**	**0**
7	**I**	**0**	**8**
8	**J**	**0**	**9**
9	**K**	**0**	**0**
10	**A**	**0**	**11**
11	**C**	**12**	13
12	**B**	**0**	**0**

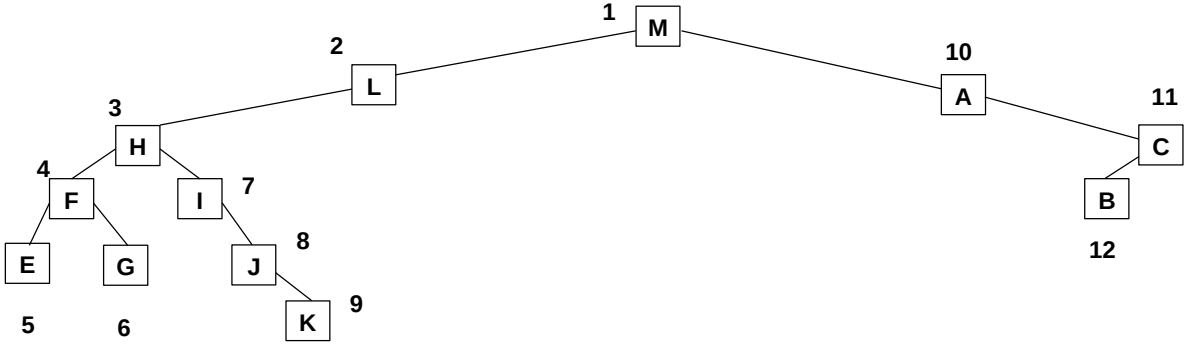
X

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo, que ocupa la posición 12, y ese valor 12 vuelve a aparecer en la columna 3 (HIJO IZQUIERDO) fila 11, y esa fila le corresponde al nodo C.
DE MODO QUE B ES HIJO IZQUIERDO DE C.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo C, que ocupa la posición 11, y ese valor 11 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 10, y esa fila le corresponde al nodo A.
DE MODO QUE C ES HIJO DERECHO DE A.

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo A, que ocupa la posición 10, y ese valor 10 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 1 (ojo) , y esa fila le corresponde al nodo M.
DE MODO QUE A ES HIJO DERECHO DE M.

No se retrocede más, debido a que ya se está en la Fila 1.

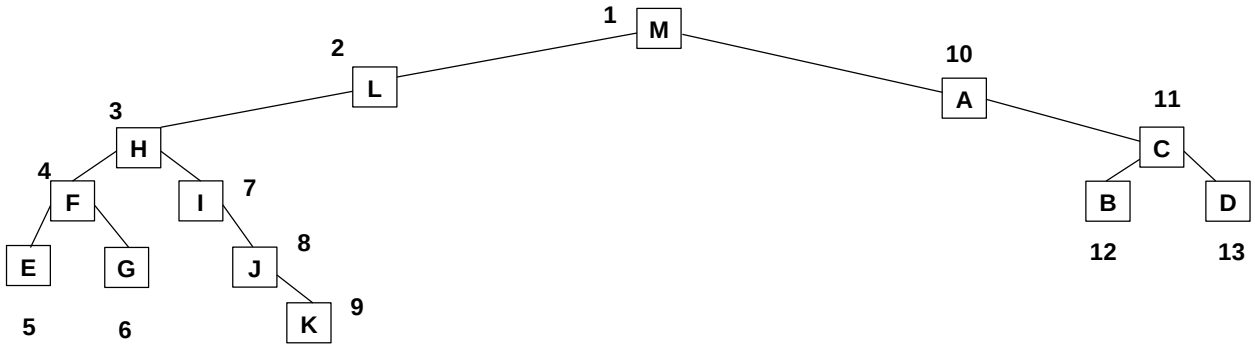


5. Lea la siguiente fila de la Tabla e identifique si ese nodo tiene hijo izq, y actúe como en el paso 1. En caso de no tenerlo (valor = 0), identifique si ese nodo tiene hijo der, y actúe como en el paso 1. En caso de no tener hijo derecho (valor = 0), entonces es nodo hoja.

El siguiente nodo hoja que se encuentra es D cuya posición es la 13.

POS	VAL	HIJO IZQ	HIJO DER
1	**M**	**2**	**10**
2	**L**	**3**	**0**
3	**H**	**4**	**7**
4	**F**	**5**	**6**
5	**E**	**0**	**0**
6	**G**	**0**	**0**
7	**I**	**0**	**8**
8	**J**	**0**	**9**
9	**K**	**0**	**0**
10	**A**	**0**	**11**
11	**C**	**12**	**13**
12	**B**	**0**	**0**
13	**D**	**0**	**0**

Ahora se revisa en la tabla para rastrear cuál es el padre de ese nodo D, que ocupa la posición 13,
y ese valor 13 vuelve a aparecer en la columna 4 (HIJO DERECHO) fila 11, y esa fila le corresponde al nodo C. DE MODO QUE D ES HIJO DERECHO DE C.



---------- FIN DE ESTA SECCIÓN