ÁRBOLES

Es una estructura de datos muy útil en muchas aplicaciones.

Veremos varias formas de árboles.

Nos interesa el tratamiento por su forma de implantación, no lo analizaremos como tipo de dato, sino por su estructura de dato.

Definición de Arbol

Formalmente, podemos definir un árbol de la siguiente forma:

- Caso base: un árbol con sólo un nodo (es a la vez raíz del árbol y hoja).
- Un nuevo árbol a partir de un nodo n_r y k árboles de raíces $n_1, n_2, n_3 \ldots n_k$ con $N_1, N_2, N_3 \ldots N_K$ elementos cada uno, puede construirse estableciendo una relación padre-hijo entre n_r y cada una de las raíces de los k árboles. El árbol resultante de $N = N_1 + N_2 + \ldots N_k$ nodos tiene como raíz el nodo n_r, los nodos $n_1, n_2, n_3, \ldots n_k$ son los hijos de n_r y el conjunto de nodos hoja está formado por la unión de los k conjuntos hojas iniciales. A cada uno de los árboles A_i se les denota ahora **subárboles** de la raíz.

ÁRBOLES BINARIOS

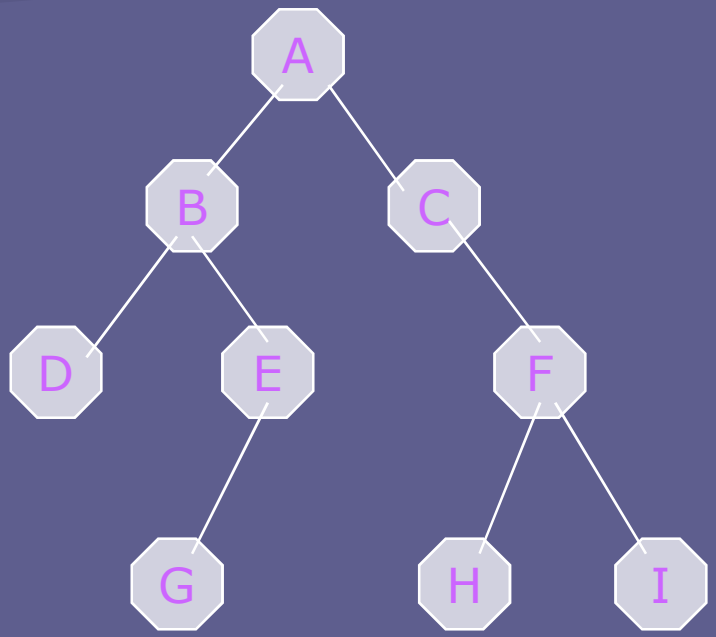
Un árbol binario es un conjunto finito de elementos que o está vacío o esta dividido en tres subconjuntos.

El primer subconjunto contiene un solo elemento llamado raíz del árbol.

Los otros dos son en sí mismos árboles binarios, y se llaman subárbol izquierdo y derecho. Cualquiera de estos puede estar vacío.

Cada elemento del árbol binario se llama nodo del árbol.

ÁRBOLES BINARIOS



Esta imagen es la representación gráfica de un árbol binario.

Si A es la raíz del árbol y B representa al subárbol izquierdo, entonces A es padre de B.

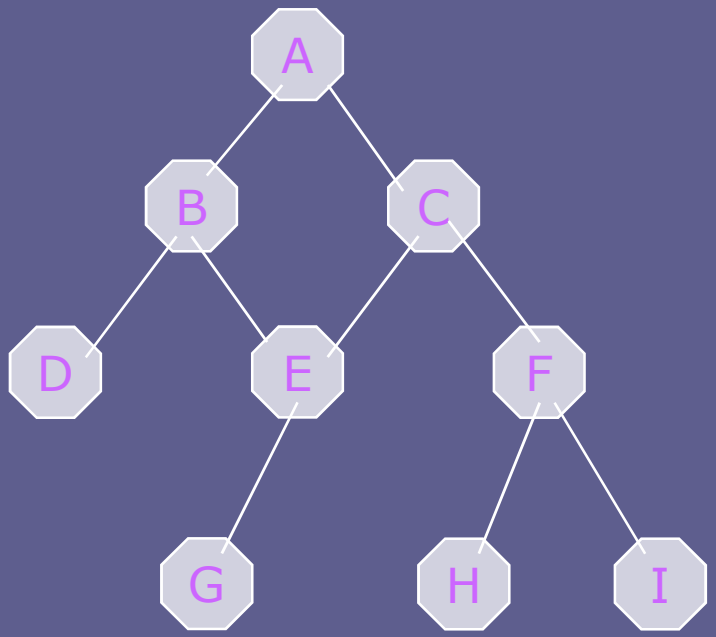
Un nodo que no tiene hijos como D, G, H, I se llama HOJA.

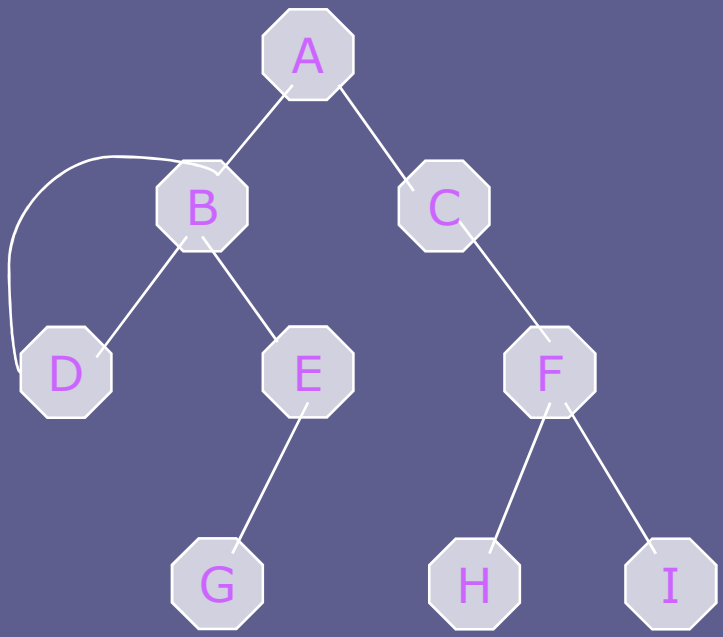
Podemos realizar alguna asociación de tipo familiar y decir que B y C son hermanos.

Y también que B es un ancestro de G, pero no lo es de H.

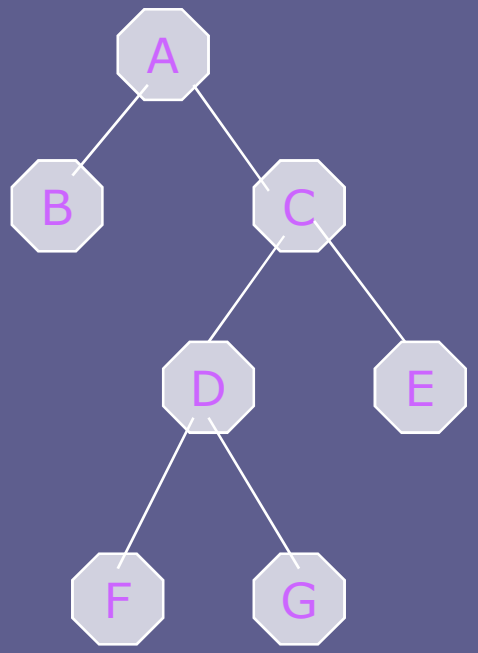
ÁRBOLES BINARIOS

Estas imágenes no representan un árbol binario.





ÁRBOLES BINARIOS



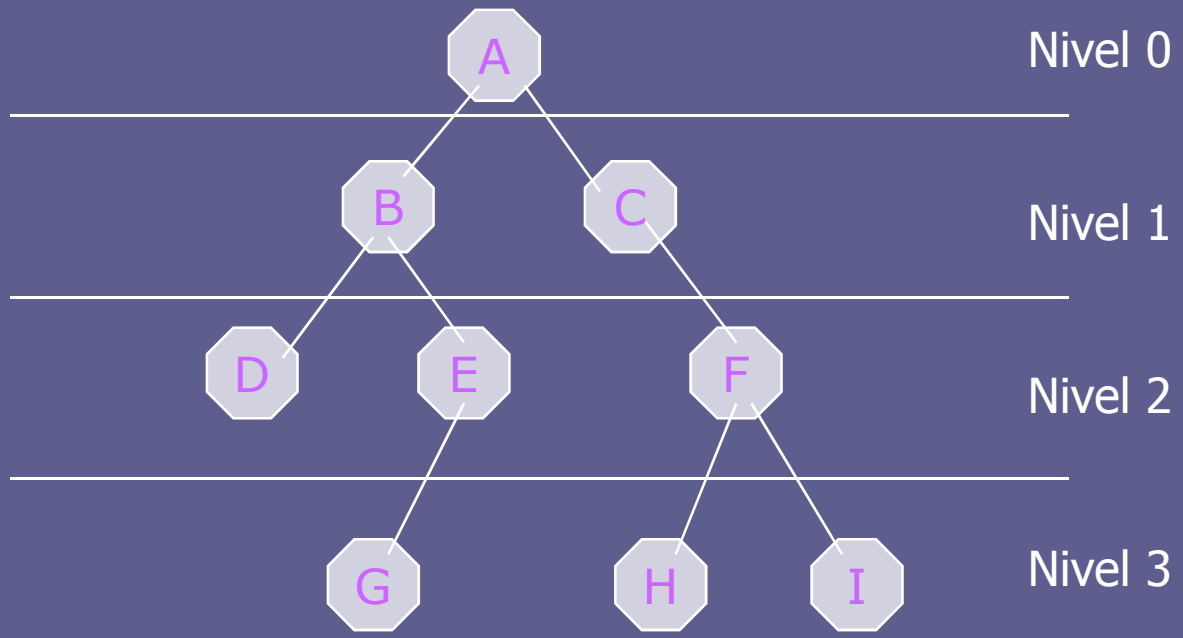
Si un nodo que no es hoja de un árbol binario, tiene subárboles izquierdo y derecho no-vacíos, el árbol se llama ESTRICTAMENTE BINARIO.

Un árbol estrictamente binario con n hojas, siempre tiene 2n – 1 nodos.

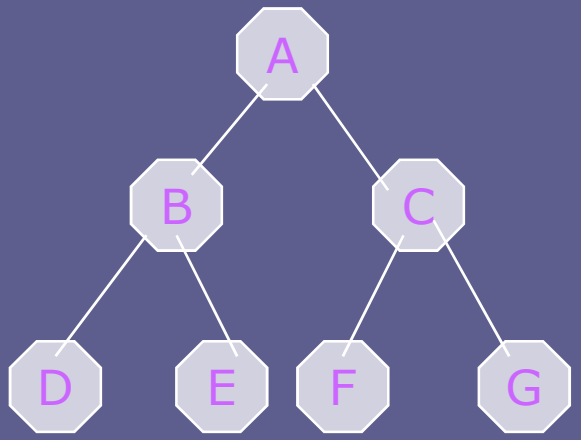
ÁRBOLES BINARIOS

El nivel de un nodo de un árbol se define de la siguiente manera:

La raíz del árbol tiene nivel 0, luego cada nodo del árbol es un nivel más que su padre



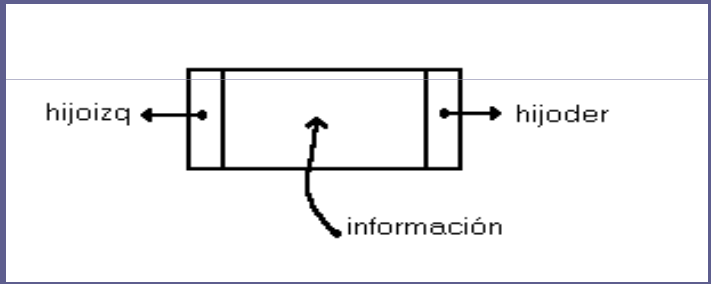
ÁRBOLES BINARIOS



Un árbol binario completo de profundidad d, es un árbol estrictamente binario cuyas hojas están todas en el nivel d.

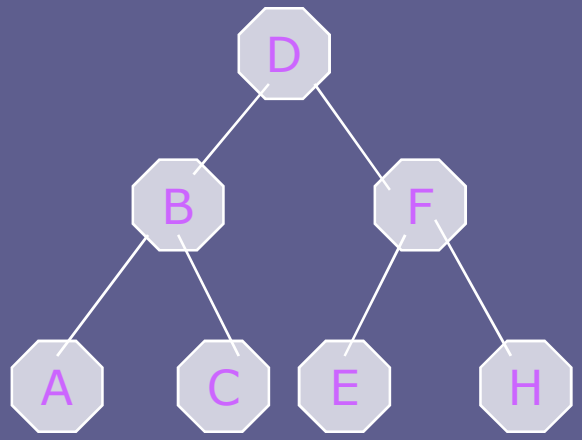
REPRESENTACION LIGADA

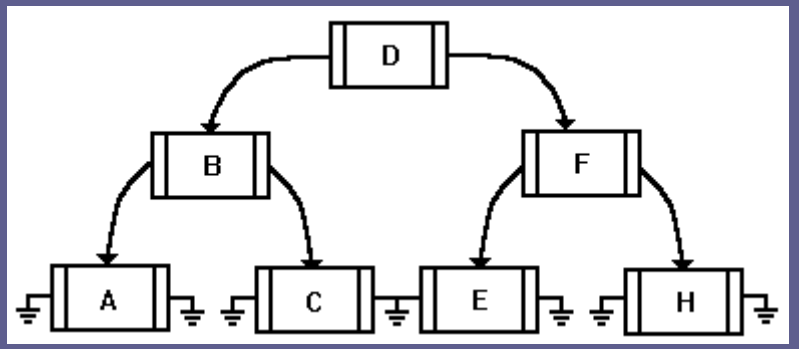
La representación más usada es la ligada o dinámica, donde cada nodo o celda tiene la siguiente forma:



REPRESENTACION LIGADA

Entonces un árbol como este se verá de la siguiente forma:

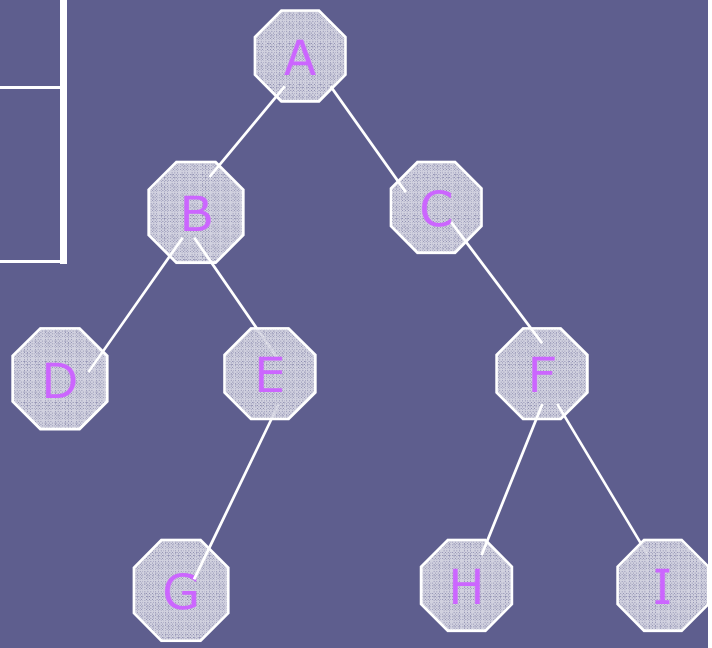




ÁRBOLES BINARIOS

Si consideramos p como un puntero a un nodo, entonces las operaciones que se pueden realizar sobre un árbol binario son:

Info(p)	Devuelve el contenido del nodo.
Left(p)	Devuelve el puntero al hijo izquierdo del nodo.
Right(p)	Devuelve el puntero al hijo derecho del nodo.



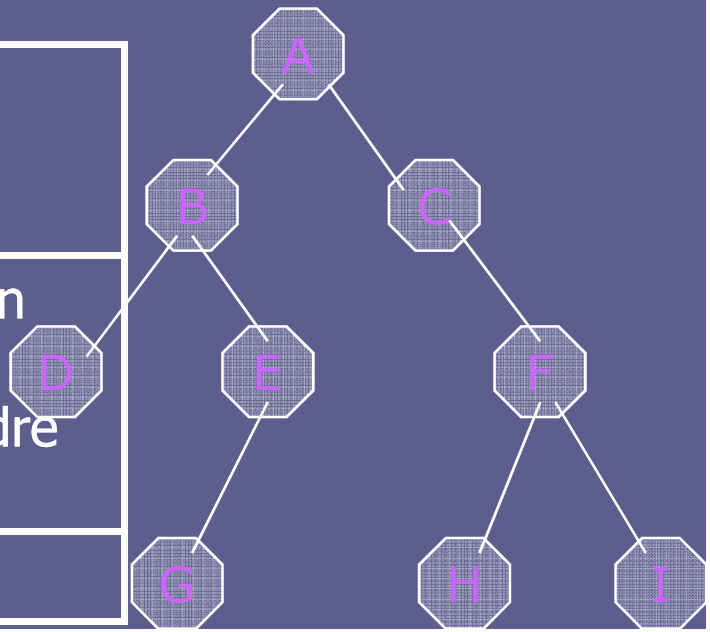
ÁRBOLES BINARIOS

Estas funciones devuelven valores nulos si no tiene hijo izquierdo o derecho o no tiene hermano, o padre.

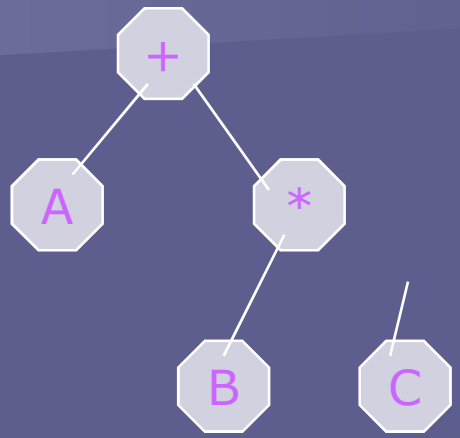
Podríamos entonces implantar las funciones isleft(p) e isright(p), que nos indican si realmente el nodo es el hijo izquierdo o derecho de p.

También podemos definir las funciones:

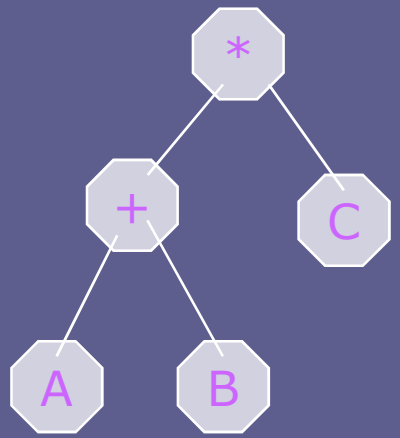
Maketree(x)	Crea un árbol binario con la información x y devuelve un puntero
Setleft(p, x)	Crea un nuevo hijo izquierdo con información x, y conectado mediante el puntero p, a un padre que no tiene hijo izquierdo
Setright(p, x)	Similar pero derecho



APLICACIONES: Representación de Fórmulas

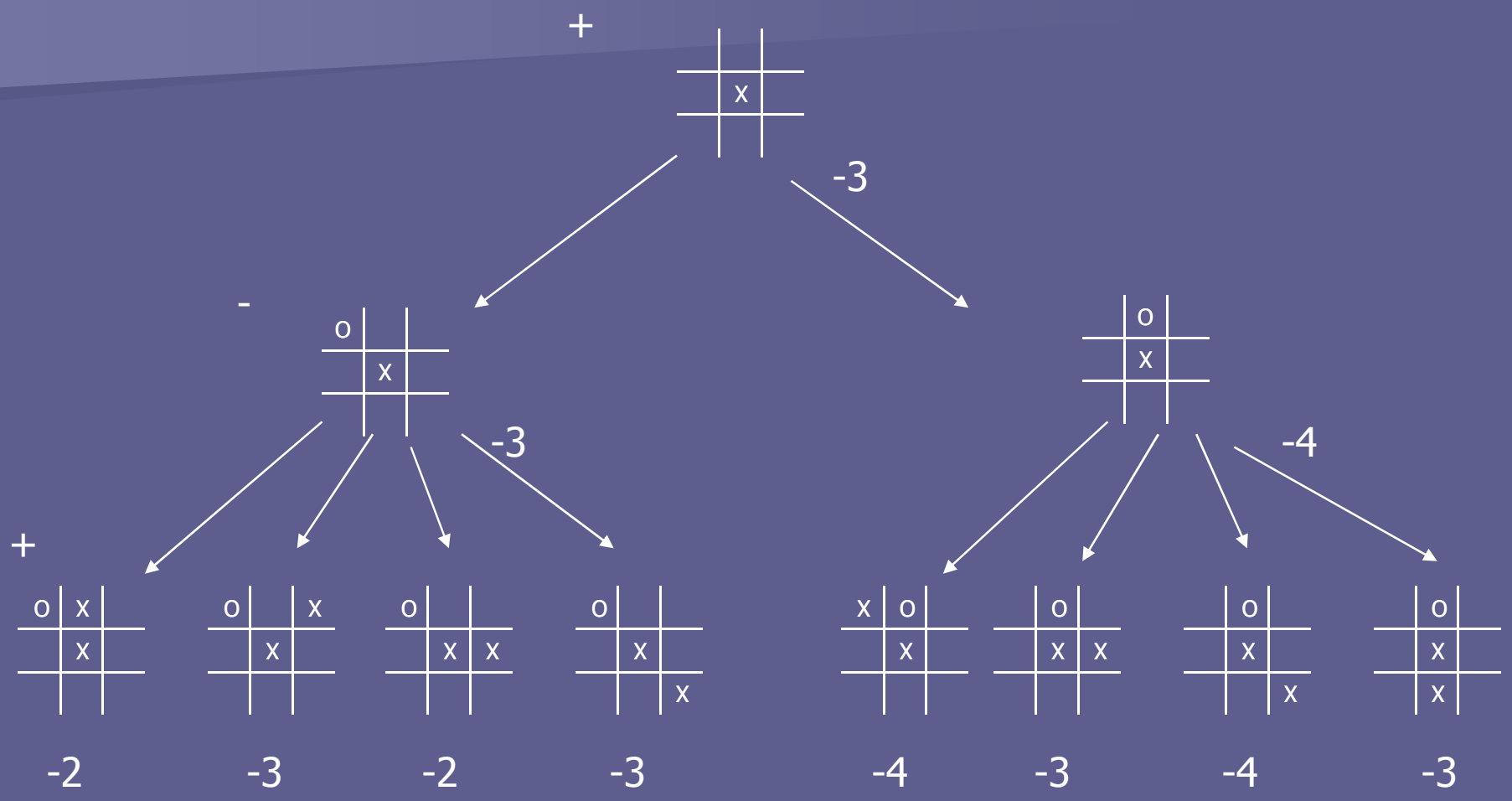


A + B * C



(A + B) * C

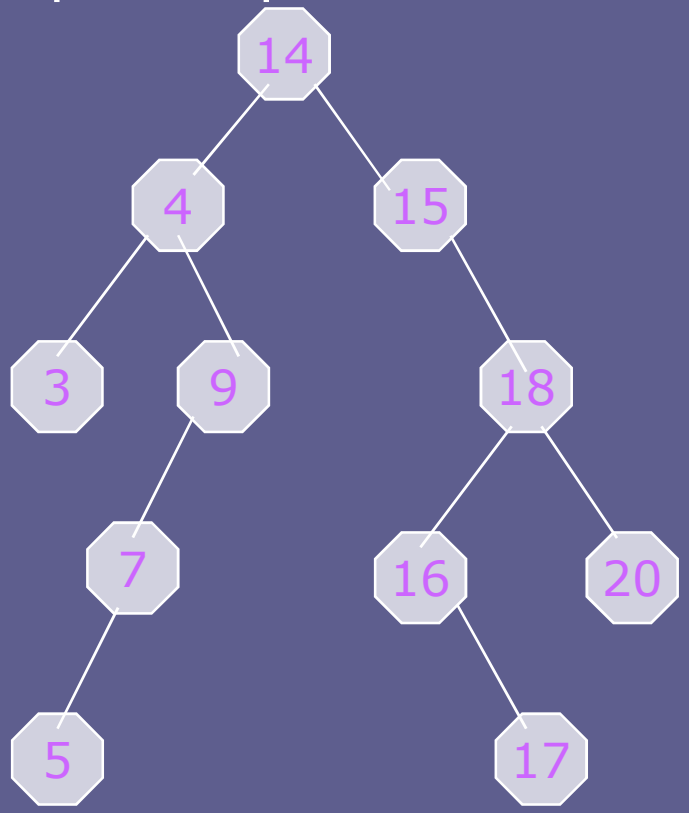
TA-TE-TI



APLICACIONES: Árboles Binarios

Un árbol binario puede ser muy útil cuando en cada punto de un proceso hay que tomar una decisión de doble opción.

Supongamos que se quiere encontrar todos los número duplicados que una persona teclea mediante un ingreso de datos.



Cada número que se ingresa debemos revisarlo contra todos los números ingresados anteriormente.

En cambio utilizando un árbol, podemos reducir considerablemente la revisión.

En este caso los datos ingresados son: 14, 15, 4, 9, 7, 18, 3, 5, 16, 4, 20, 17, 9, 14, 5

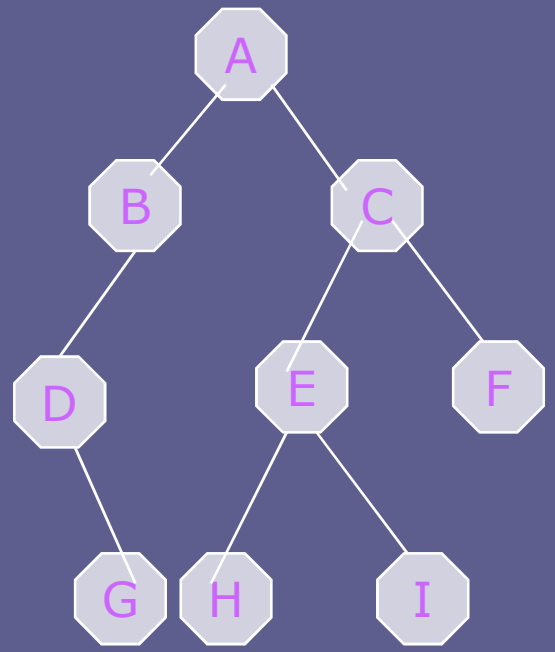
Recorrido de Árboles Binarios

Otra operación es recorrer un árbol binario, o sea, pasar a través del árbol, enumerando (imprimiendo o revisando) cada uno de sus nodos.

El orden de visita en caso de un árbol no es lineal.

Para recorrer un árbol en preorden, u orden con prioridad a la profundidad. Se hace lo siguiente:

1. Visitar la raíz,
2. Visitar el subárbol izquierdo en preorden,
3. Visitar el subárbol derecho en preorden.



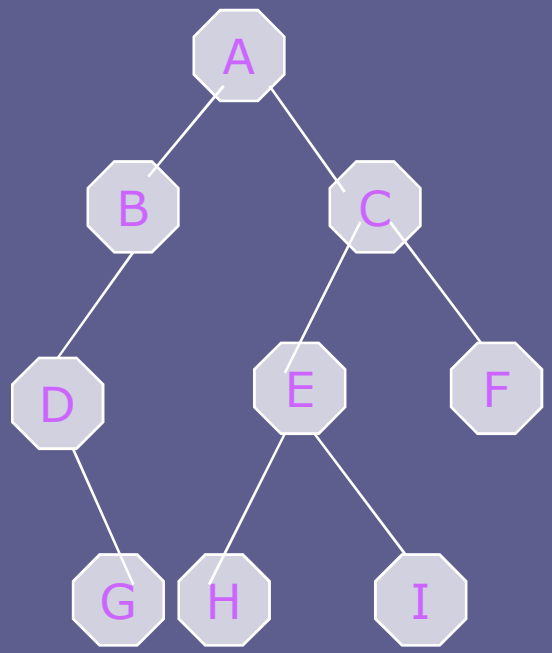
En PREORDEN: A B D G C E H I F

Recorrido de Árboles Binarios

Otra manera de recorrer el árbol es en orden, u orden simétrico.

Se hace lo siguiente:

1. Recorrer el subárbol izquierdo en orden,
2. Visitar la raíz,
3. Recorrer el subárbol derecho en orden.



En PREORDEN: A B D G C E H I F

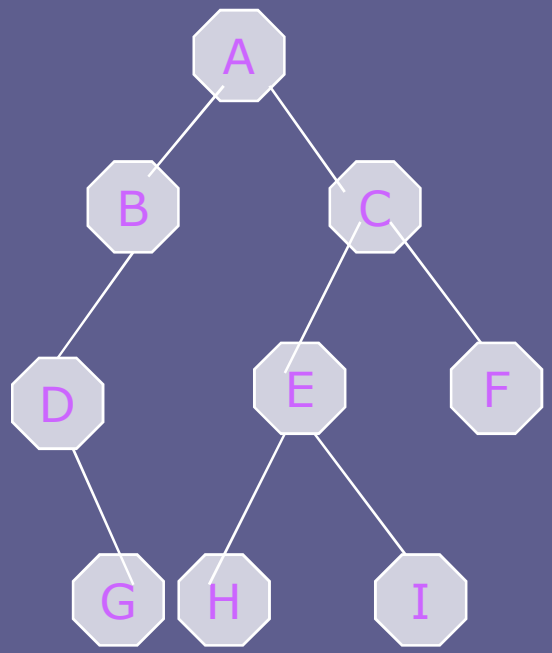
En ORDEN: D G B A H E I C F

Recorrido de Árboles Binarios

Otra manera de recorrer el árbol es en posotrden.

Se hace lo siguiente:

1. Recorrer el subárbol izquierdo en postorden,
2. Recorrer el subárbol derecho en postorden.

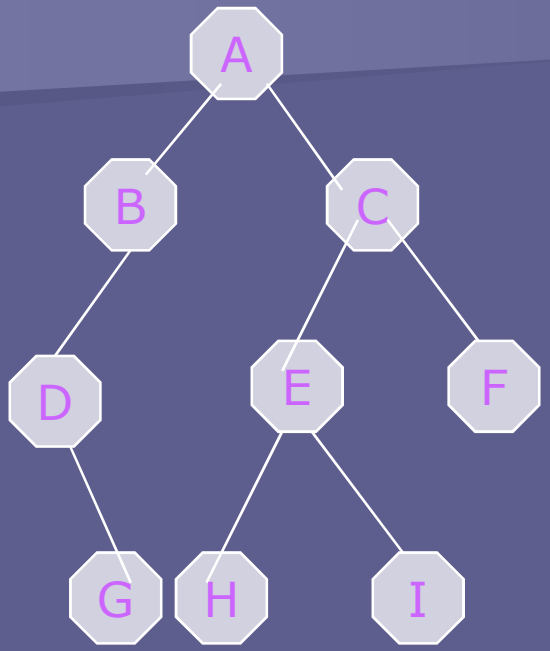


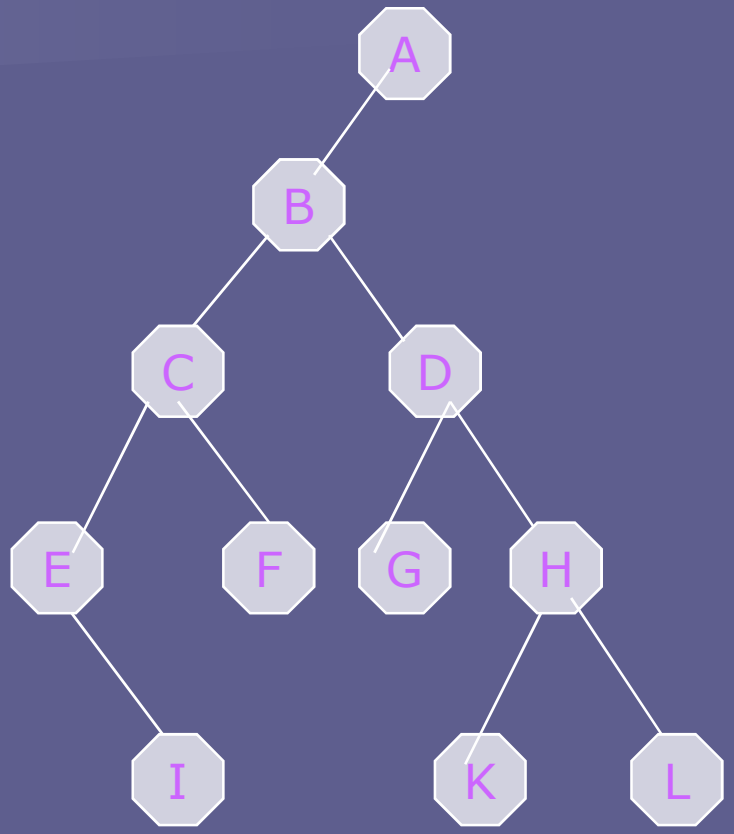
En PREORDEN: A B D G C E H I F

En ORDEN: D G B A H E I C F

En POSTORDEN: G D B H I E F C A

Recorrido de Árboles Binarios





En PREORDEN: A B D G C E H I F

En ORDEN: D G B A H E I C F

En POSTORDEN: G D B H I E F C A

En PREORDEN: A B C E I F D G H K L

En ORDEN: E I C F B G D K H L A

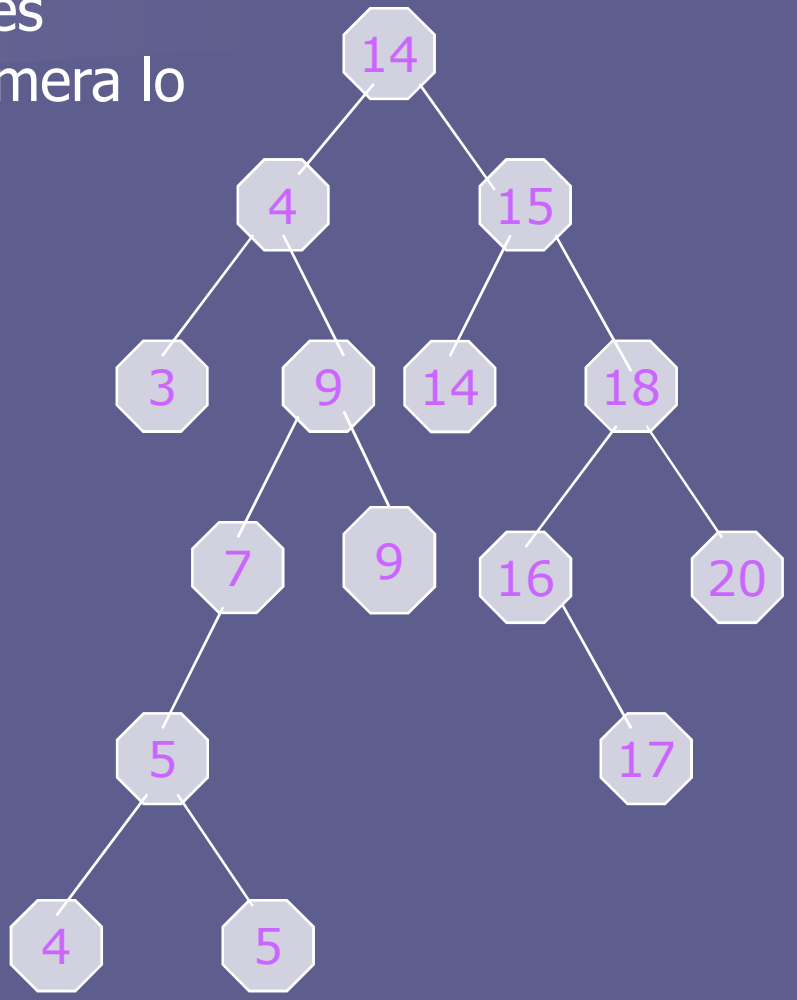
En POSTORDEN: I E F C G K L H D B A

APLICACIONES: Árboles Binarios

Muchas aplicaciones que utilizan árboles binarios proceden en 2 fases, en la primera lo construyen en la segunda lo recorren.

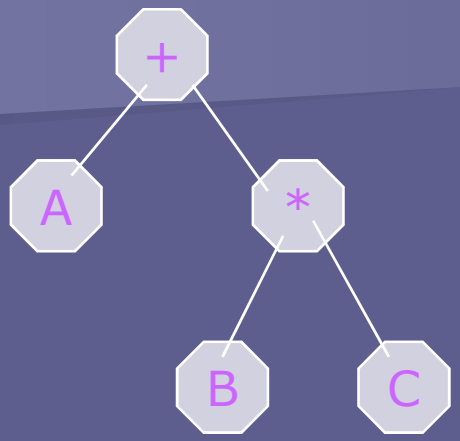
Supongamos que dada una lista de números quiero imprimirlos en orden ascendente.

Al ingresar los datos se cargan en un árbol binario, donde a la izquierda se ponen los números menores a la raíz, y a la derecha se ponen los números mayores o iguales a la raíz.



Datos Ingresados: 14 15 4 9 7 18 3 5 16 4 20 17 9 14 5

APLICACIONES: Representación de Fórmulas



A + B * C

Recorrer el árbol en PREORDEN, conduce a una expresión en forma PREFIJA.

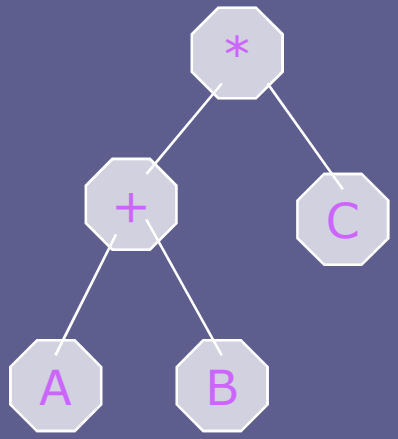
+ A * B C

* + A B C

Recorrer el árbol en POSTORDEN conduce a una expresión en forma POSTFIJA

A B C * +

A B + C *



(A + B) * C

Recorrer el árbol en ORDEN, conduce a una expresión INFIJA, con el problema de los paréntesis

A + B * C

A + B * C

TA-TE-TI