

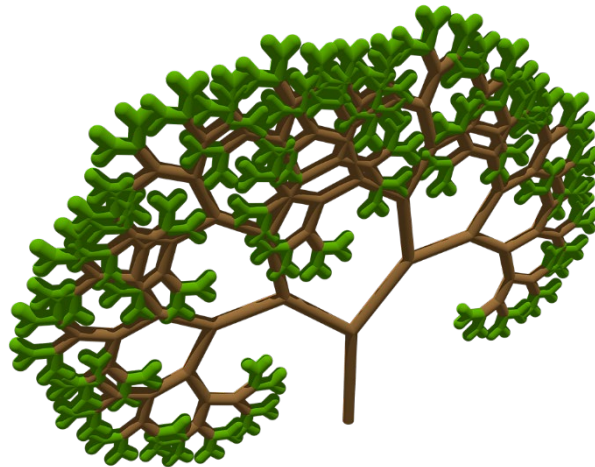


Algoritmos Estructuras de Datos I

Facultad de Ciencias Exactas y Tecnología
Universidad Nacional de Tucumán

2024

Árbol Binario

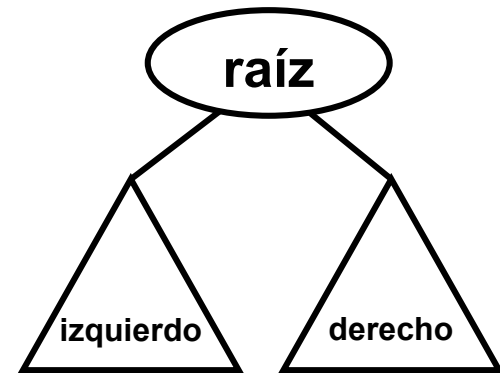


ARBOL BINARIO

Los arboles binarios son un caso particular de arboles tales que en ellos, cada nodo tiene a lo sumo 2 hijos.

Un árbol binario es un conjunto finito T tal que:

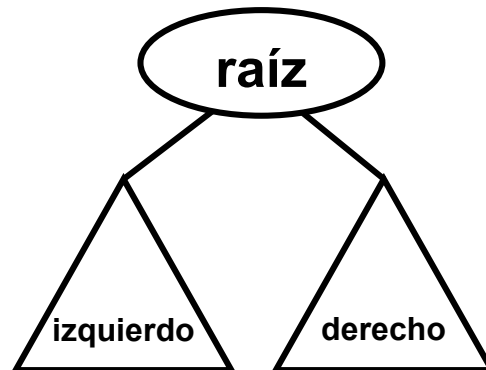
- esta vacío
o
- está compuesto de una *raíz* y dos subárboles binarios llamados *subárbol izquierdo* y *subárbol derecho*.



ARBOL BINARIO

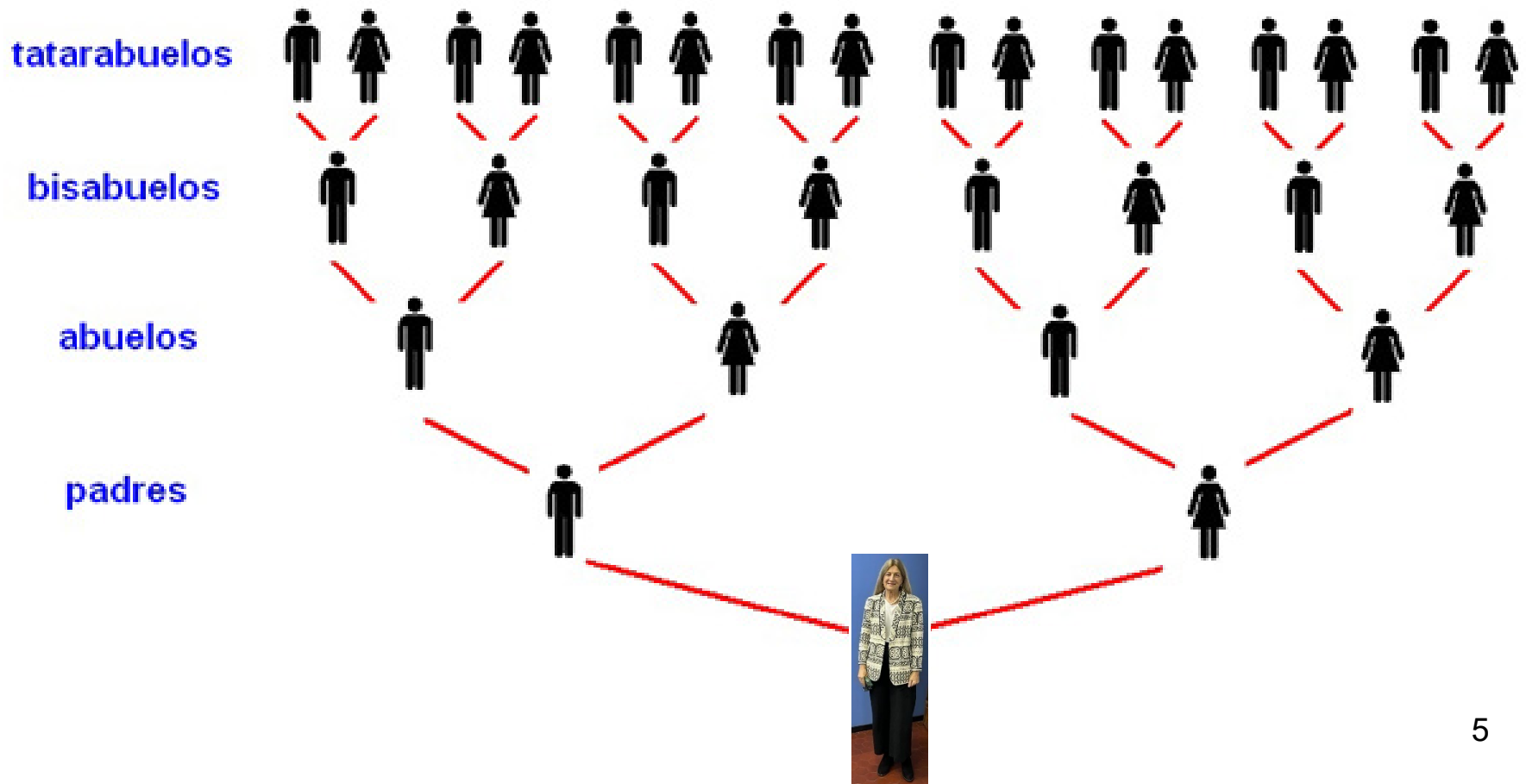
Hay numerosas aplicaciones de arboles binarios como:

- los arboles genealógicos,
- los arboles de expresión,
- los arboles que representan partidos entre dos equipos en un campeonato,
- los arboles para codificar, los arboles de decisión,
- y muchas mas.



ARBOL BINARIO

Ejemplo: árbol genealógico



ARBOL BINARIO(ITEM) - AB(ITEM)

Especificación algebraica - OPERACIONES:

A) Syntax:

ABVACIO	: $\rightarrow AB$
ARMARAB	: $AB \times ITEM \times AB \rightarrow AB$
ESABVACIO	: $AB \rightarrow \text{BOOLEAN}$
IZQUIERDO	: $AB \rightarrow AB$
RAIZ	: $AB \rightarrow ITEM \cup \{\text{indefinido}\}$
DERECHO	: $AB \rightarrow AB$
PERTENECE	: $AB \times ITEM \rightarrow \text{BOOLEAN}$

ARBOL BINARIO(ITEM) - AB(ITEM)

B) Semántica: $\forall \text{ izq, der} \in \text{AB}, \quad \forall x, y \in \text{ITEM}$

ESABVACIO(ABVACIO) \equiv TRUE

ESABVACIO(ARMARAB(izq,x,der)) \equiv FALSE

IZQUIERDO(ABVACIO) \equiv ABVACIO

IZQUIERDO(ARMARAB(izq,x,der)) \equiv izq

RAIZ(ABVACIO) \equiv indefinido

RAIZ(ARMARAB(izq,x,der)) \equiv x

DERECHO(ABVACIO) \equiv ABVACIO

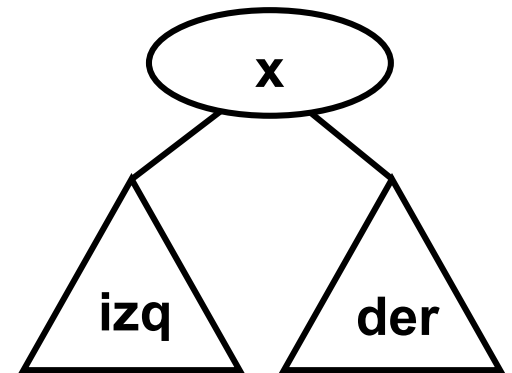
DERECHO(ARMARAB(izq,x,der)) \equiv der

PERTENECE(ABVACIO,y) \equiv FALSE

PERTENECE(ARMARAB(izq,x,der),y) \equiv $x=y$ OR

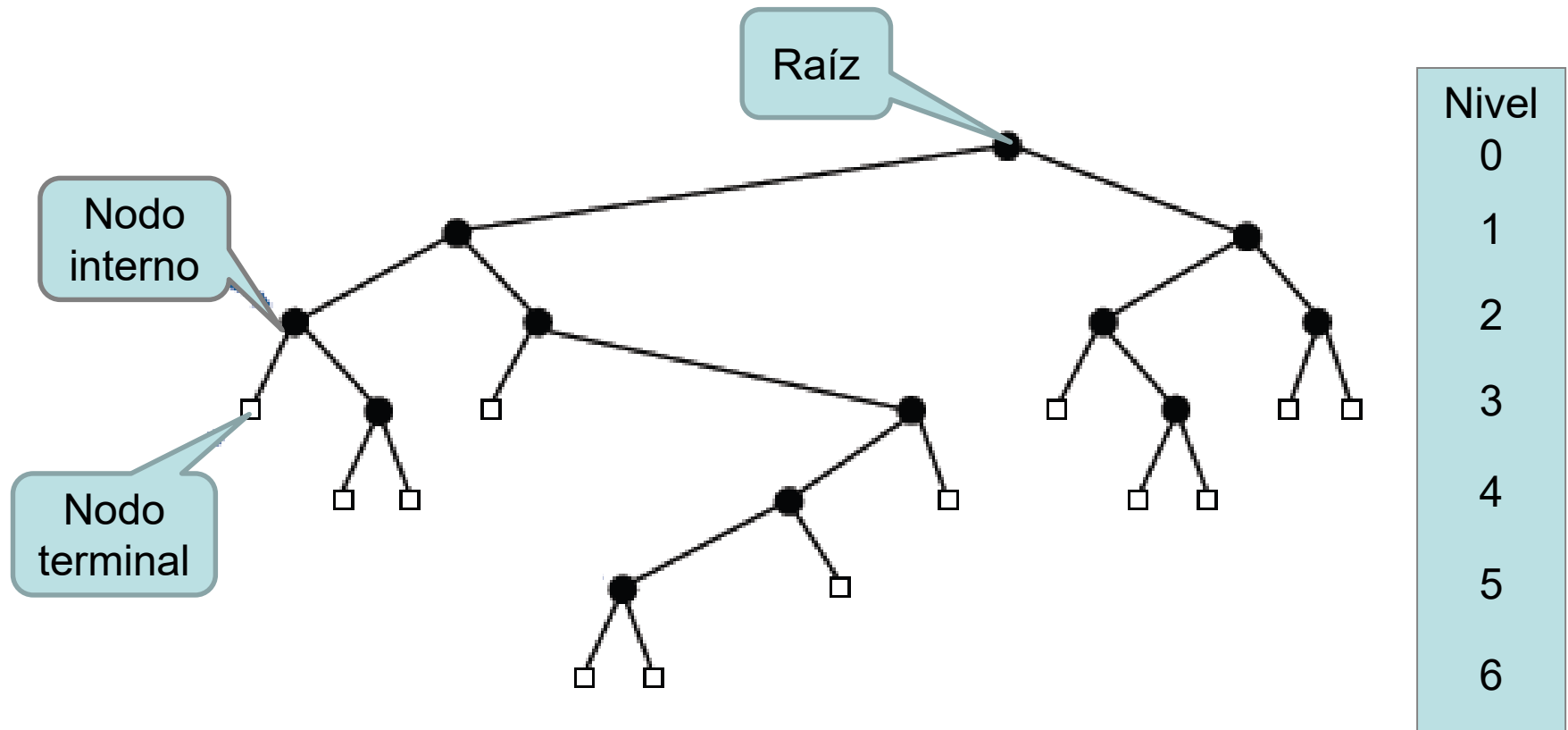
PERTENECE(izq,y) OR

PERTENECE(der,y)



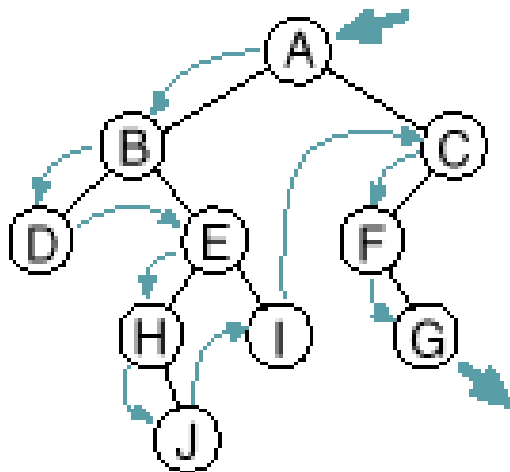
donde $=$ es la operación que me permite comparar 2 objetos del tipo item

ARBOL BINARIO

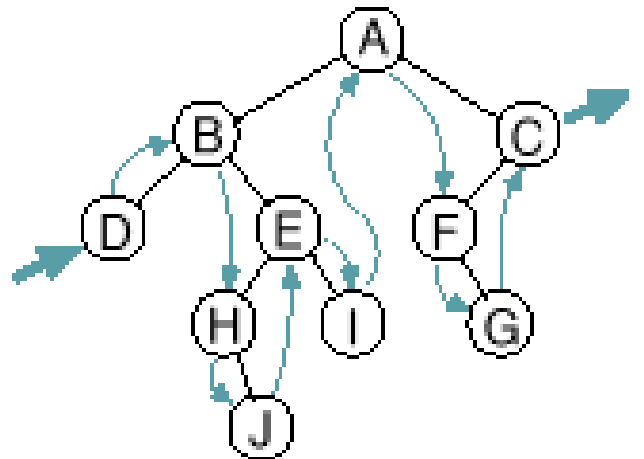


Recorrido de un AB

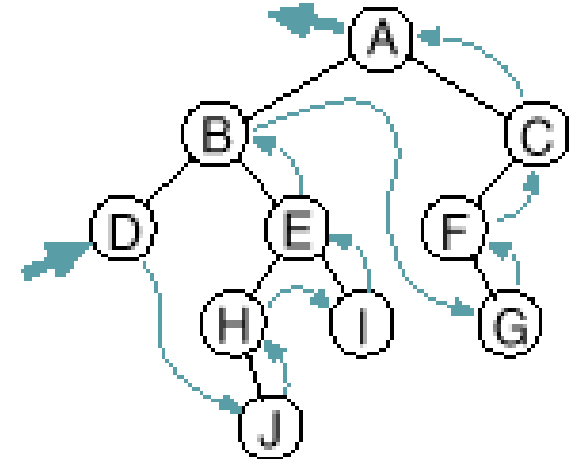
Orden Previo



Orden Simétrico



Orden Posterior



Listado en Orden Previo: **< A,B,D,E,H,J,I,C,F,G >**

Listado en Orden Simetrico: **< D,B,H,J,E,I,A,F,G,C >**

Listado en Orden Posterior: **< D,J,H,I,E,B,G,F,C,A >**

AB

Aplicación: listado en orden simétrico

Como usuario del ADT AB, algoritmo recursivo

ALGORITMO **Orden Simetrico** (T)

ENTRADA : $T \in AB(\text{ITEM})$

SALIDA : listado de los nodos de T en orden simétrico

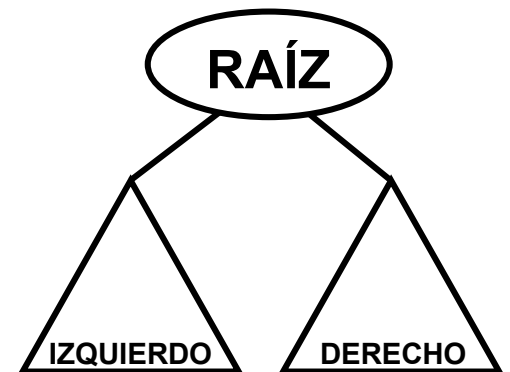
Si NOT ESABVACIO(T) entonces

Orden Simetrico (IZQUIERDO(T))

ESCRIBIR (RAIZ(T))

Orden Simetrico (DERECHO(T))

Fin



AB

Aplicación: es un árbol hoja

Como usuario del ADT AB

Funcion **Arbolhoja** (T) : $AB \rightarrow \text{BOOL}$

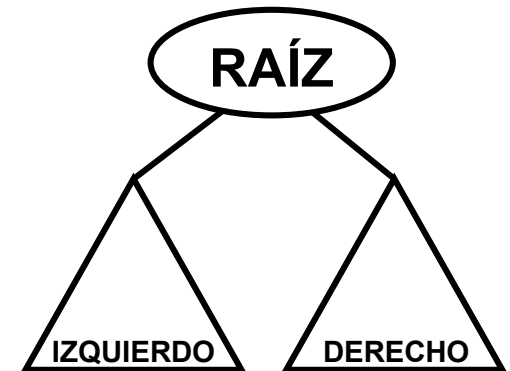
Si ESABVACIO(T) entonces

Retorna FALSE

sino

Retorna eshoja(T)

Fin



Nota: Se puede invocar con cualquier AB

AB

Aplicación: es nodo hoja

Como usuario del ADT AB

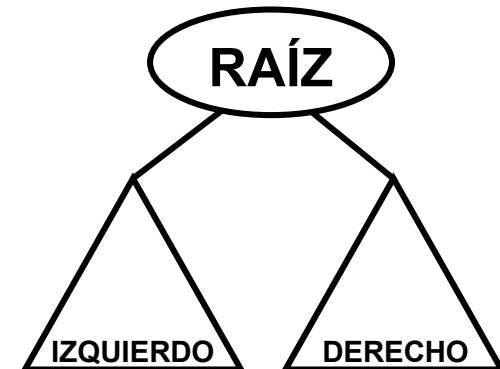
Funcion **eshoja** (T) : $AB \rightarrow \text{BOOL}$

Retorna

ESABVACIO(IZQUIERDO(T)) AND
ESABVACIO(DERECHO(T))

Fin

Nota: invocar eshoja con AB no vacío.



AB

Aplicación: numero de nodos hojas

Como usuario del ADT AB, función recursiva

Funcion **NumeroHojas** (T) : AB \rightarrow entero ≥ 0

Si ESABVACIO(T) entonces

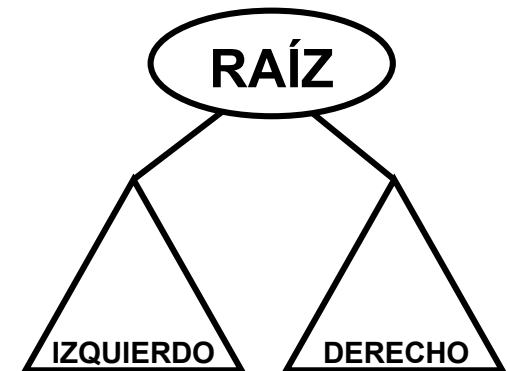
Retorna 0

sino Si eshoja (T) entonces

Retorna 1

sino

Retorna **NumeroHojas** (IZQUIERDO(T)) +
NumeroHojas (DERECHO(T))



Fin

AB

Aplicación: cuenta un item

Como usuario del ADT AB, función recursiva

Funcion **cuenta** (T, x) : AB x ITEM \rightarrow entero ≥ 0

Si ESABVACIO(T) entonces

Retorna 0

sino Si RAIZ(T) = x entonces

Retorna 1 + **cuenta** (IZQUIERDO(T),x) +
cuenta (DERECHO(T),x)

sino

Retorna **cuenta** (IZQUIERDO(T),x) +
cuenta (DERECHO(T),x)

Fin

AB

Aplicación: altura del árbol

Como usuario del ADT AB, función recursiva

Funcion **Altura** (T) : AB \rightarrow entero ≥ 0

Si ESABVACIO (T) OR eshoja (T) entonces

Retorna 0

sino

Retorna 1 + MAXIMO (**Altura**(IZQUIERDO(T)),
Altura(DERECHO(T)))

Fin

NOTA: *MAXIMO es una funcion auxiliar que determina el mayor de 2 números enteros.*

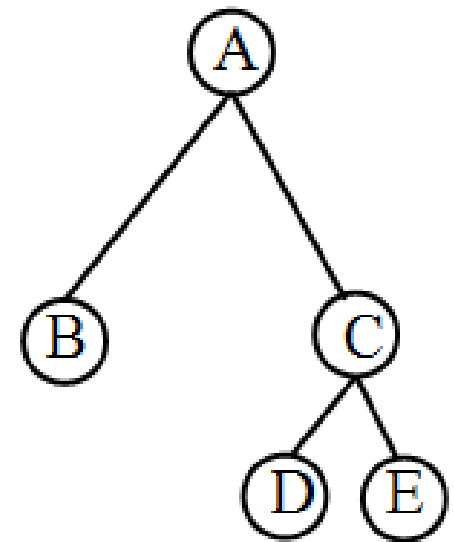
AB

Aplicación: listado de nodos por nivel

Como usuario del ADT AB, diseñar un algoritmo iterativo para un **listado por niveles** de un ab.

Ejemplo:

Listado por Nivel: < A, B, C, D, E >

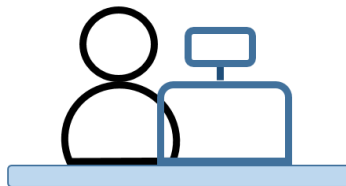


AB

Aplicación: listado de nodos por nivel

Algoritmo iterativo para un **listado por niveles** de un ab:

- Se necesita como estructura auxiliar una *fila de arboles*.
- En cada nivel $k=0,1,2,..$ se ponen en la fila todos los arboles cuya raíz se encuentre en ese nivel.
- De esa fila se va sacando de a uno los árboles, se lista la raíz y se incluye sus dos subárboles asociados al final de la fila.
- Se continúa hasta llegar a las hojas y hasta que la fila quede vacía.



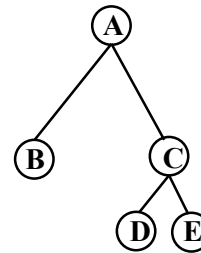
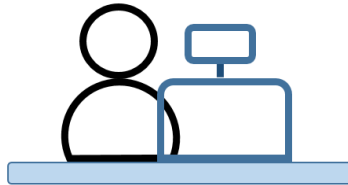
AB

Aplicación: listado de nodos por nivel

Ejemplo:

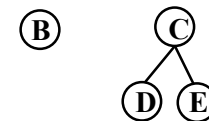
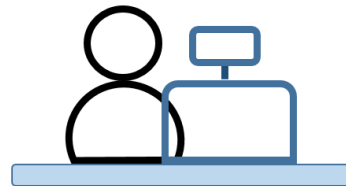
Inicial: Listado por Nivel: < >

Fila:



Después del Paso 1: Listado por Nivel: < A >

Fila:

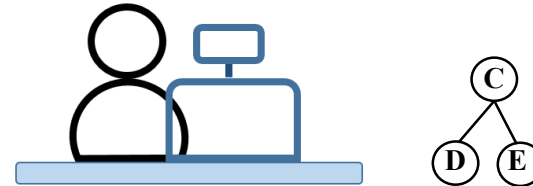


AB

Aplicación: listado de nodos por nivel

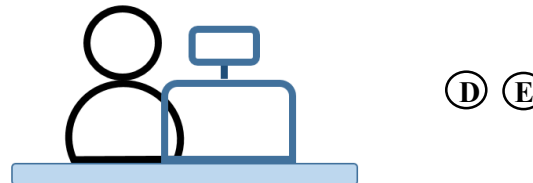
Después del Paso 2 : Listado por Nivel: < A, **B** >

Fila:



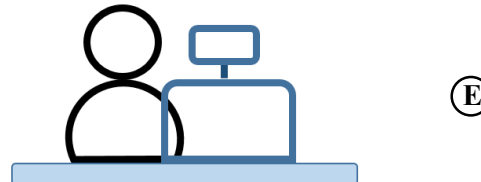
Después del Paso 3 : Listado por Nivel: < A, B, **C** >

Fila:



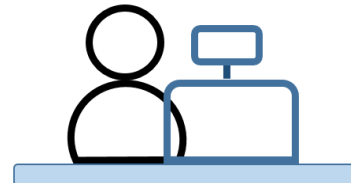
Después del Paso 4 : Listado por Nivel: < A, B, C, **D** >

Fila:



Después del Paso 5 : Listado por Nivel: < A, B, C, D, **E** >

Filavacia:



AB

Aplicación: listado de nodos por nivel

ALGORITMO **Niveles** (T)

ENTRADA : $T \in AB(\text{ITEM})$

SALIDA : listado de nodos por nivel

AUXILIAR: $F \in \text{FILA}(\text{AB})$, $\text{TAUX} \in \text{AB}(\text{ITEM})$

Si NOT ESABVACIO(T) entonces

$F \leftarrow \text{FILAVACIA}$; $\text{ENFILA}(F, T)$

Mientras NOT ESFILAVACIA(F) hacer

$\text{TAUX} \leftarrow \text{FRENTE}(F)$; $\text{DEFILA}(F)$; $\text{ESCRIBIR}(\text{RAIZ}(\text{TAUX}))$

SI NOT ESABVACIO(IZQUIERDO(TAUX)) entonces

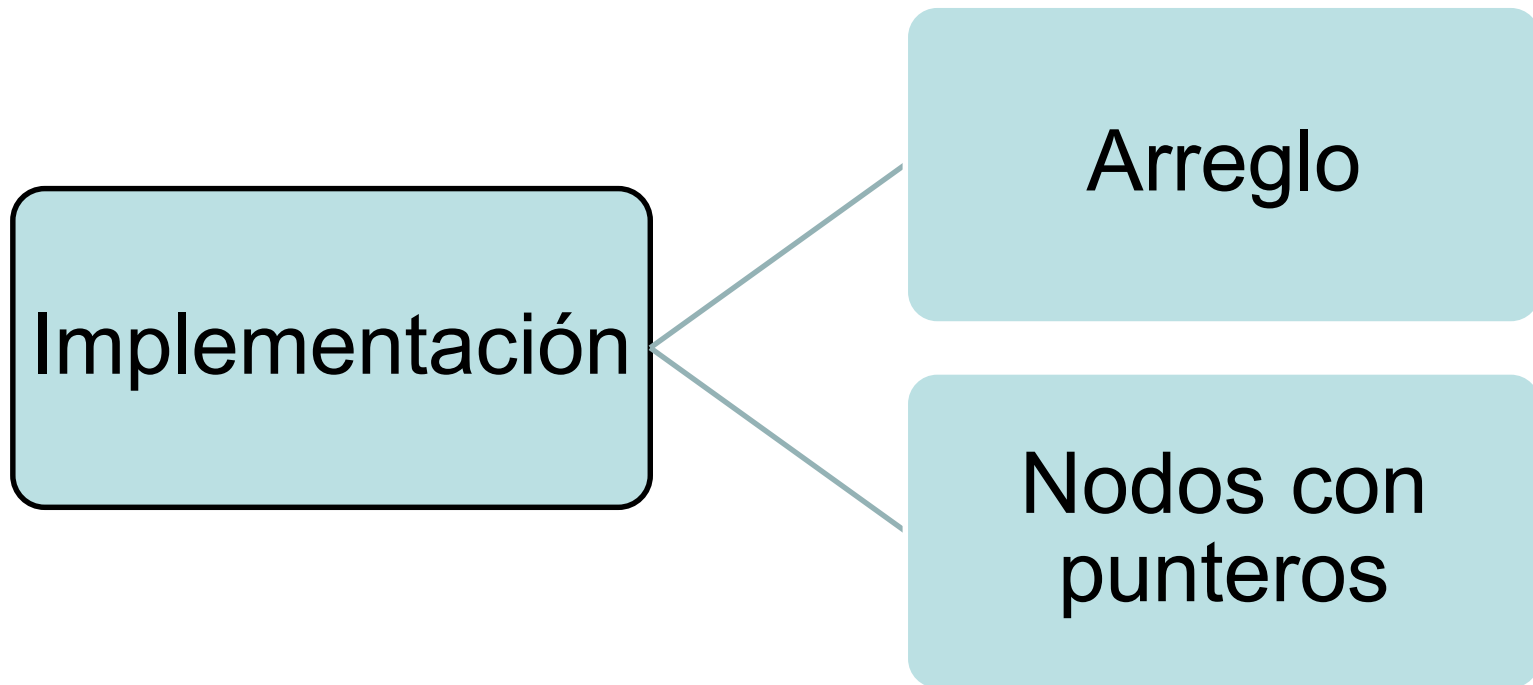
$\text{ENFILA}(F, \text{IZQUIERDO}(\text{TAUX}))$

SI NOT ESABVACIO(DERECHO(TAUX)) entonces

$\text{ENFILA}(F, \text{DERECHO}(\text{TAUX}))$

Fin

Arbol Binario



Ventajas y Desventajas?
Costos?

Implementación AB

Arreglo:

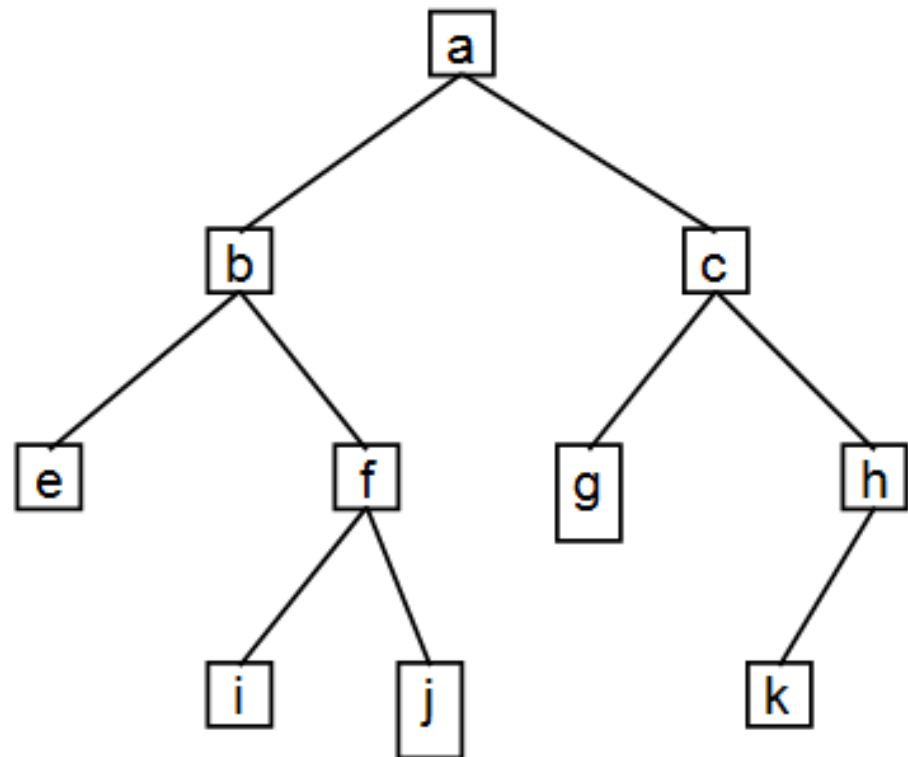
item izq der

0	a	1	2
1	b	3	4
2	c	5	6
3	e	-1	-1
4	f	7	8
5	g	-1	-1
6	h	9	-1
7	i	-1	-1
8	j	-1	-1
9	k	-1	-1
10			

IndiceRaiz=0

IndiceLibre=10

Ejemplo:

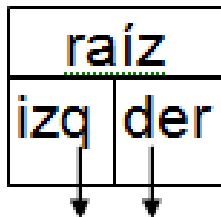


Costo de las operaciones?

Implementación AB

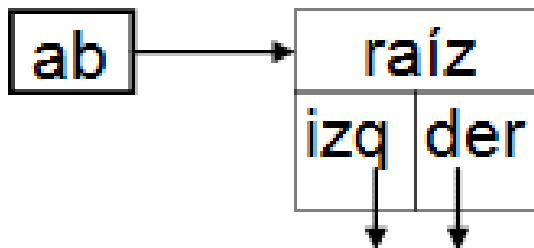
Nodos con punteros:

Nodo:

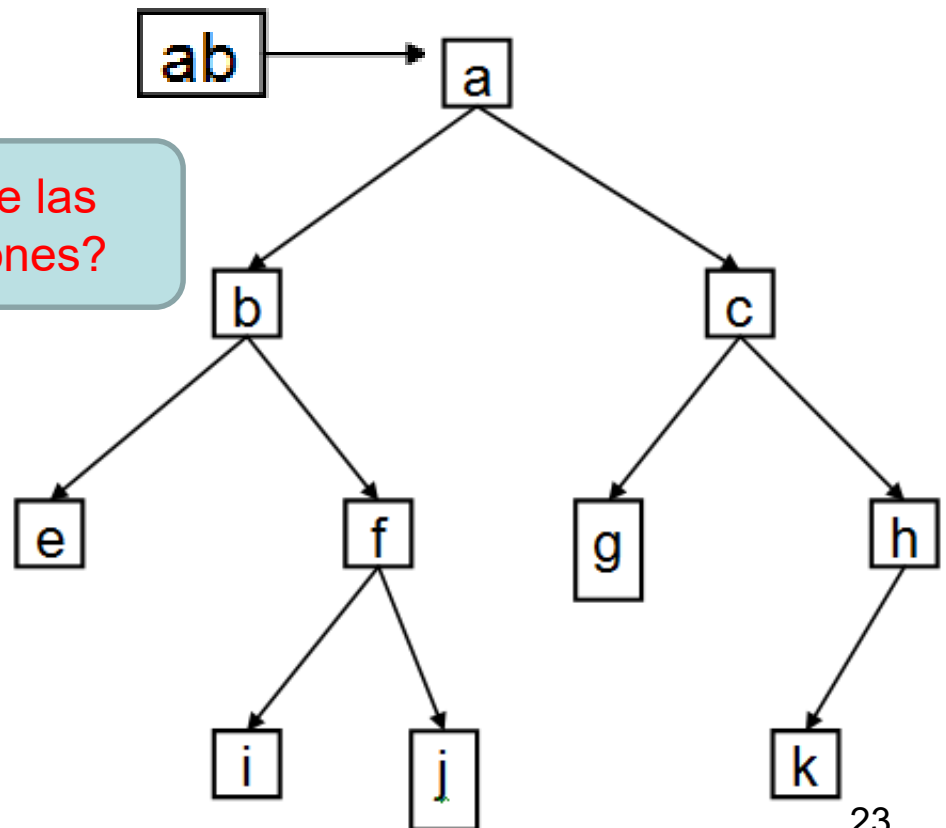


Costo de las operaciones?

Arbol Binario:



Ejemplo:



Implementación AB

En la representación del árbol con punteros, hay muchos punteros nulos. Si hay m nodos en un árbol binario, hay $2m$ enlaces de los cuales $m+1$ enlaces son vacíos, de manera que la mitad del espacio reservado para punteros está desperdiciado.

Perlis y Thornton idearon un procedimiento para aprovechar estos enlaces, usándolos para señalar otros nodos del árbol.

Este nuevo apuntador se llama *hilo* o *hilván* o *hebra*. Un árbol binario con hilvanes se denomina *hilvanado*.

Implementación AB

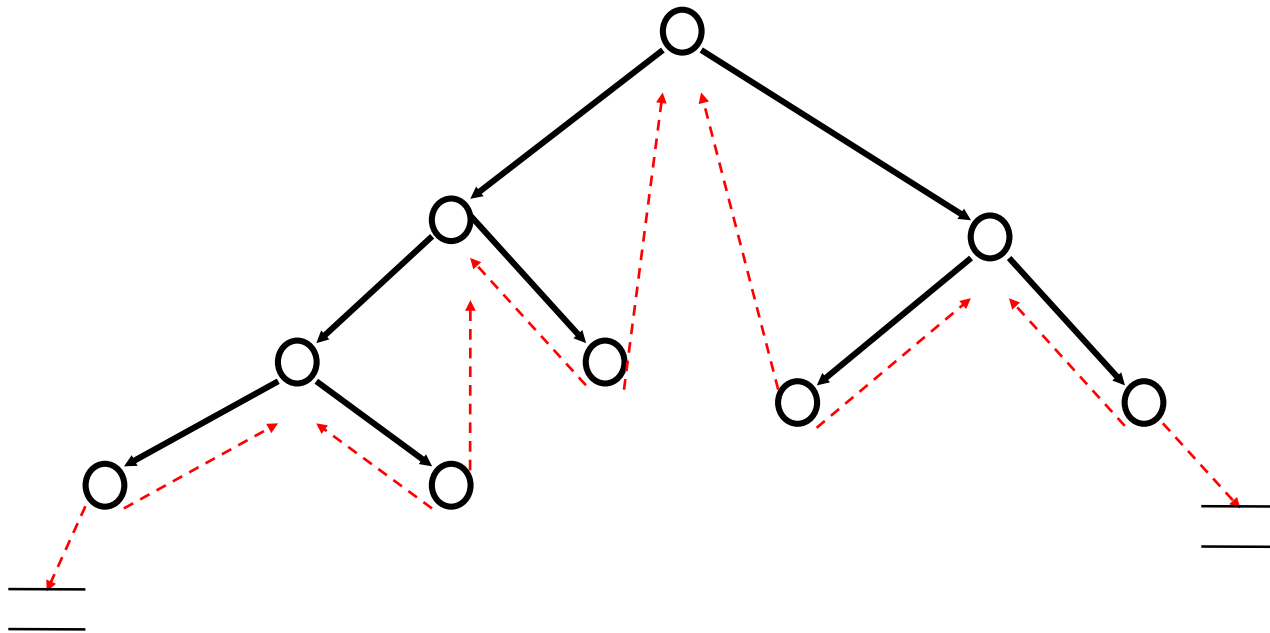
Por ejemplo: Se podría simplificar el recorrido en orden simétrico en el árbol con:

- Un enlace nulo a izquierda de un nodo puede ser reemplazado por un enlace al nodo que se listaría antes del nodo en orden simétrico.
- Un enlace nulo a derecha puede ser reemplazado por un puntero al nodo siguiente en orden simétrico.

Para implementar un árbol hilvanado se debe incluir en cada nodo un campo que indique si cada puntero es un verdadero puntero o es un hilván.

Implementación AB

Por ejemplo:



Formas de un AB

Desequilibrado o sesgado (skewed):

Por ejemplo:

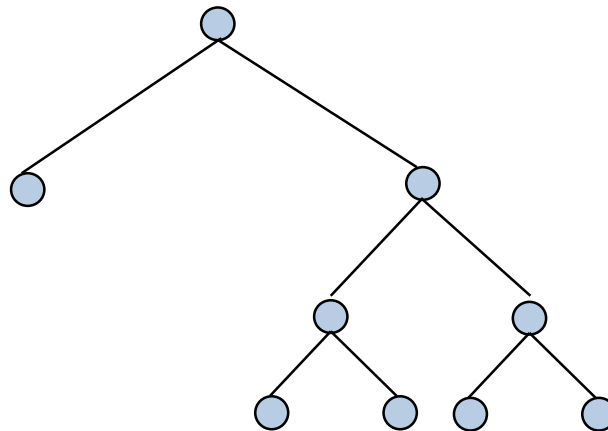


- Este tipo de árbol tiene altura ($n-1$) para n nodos.

Formas de un AB

Estrictamente Binario: si todo nodo del árbol, tiene dos hijos o es una hoja.

Por ejemplo:

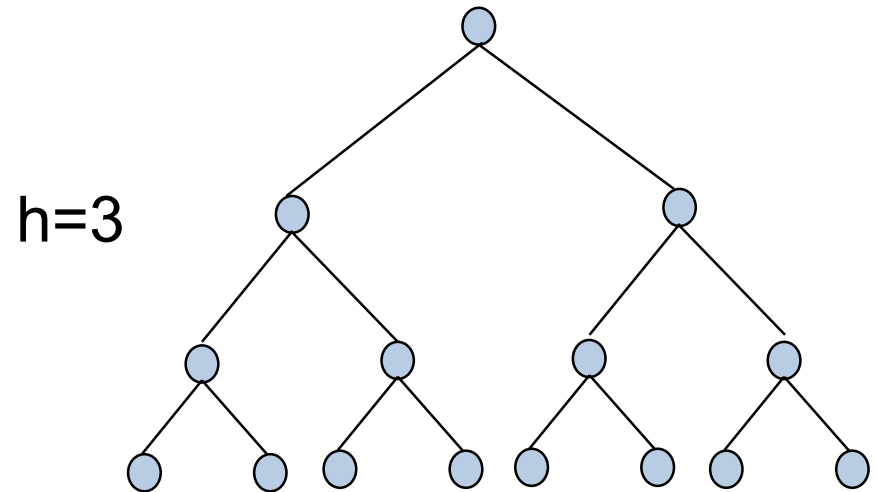
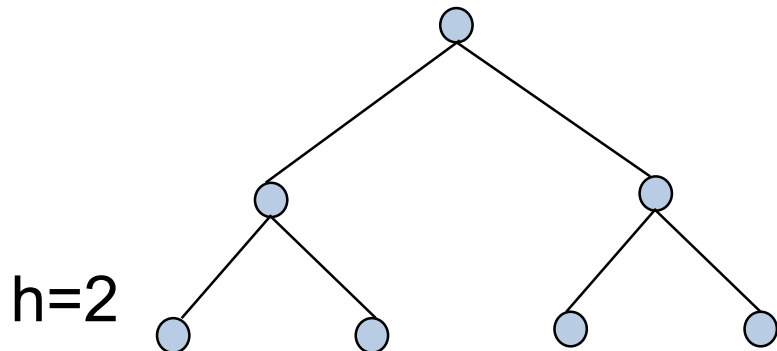
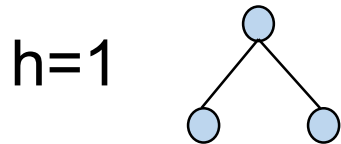


- El numero de hojas de un ab estrictamente binario es el numero de nodos internos mas uno.

Formas de un AB

Completo: es un ab estrictamente binario que tiene todas sus hojas en el mismo nivel.

Por ejemplo:



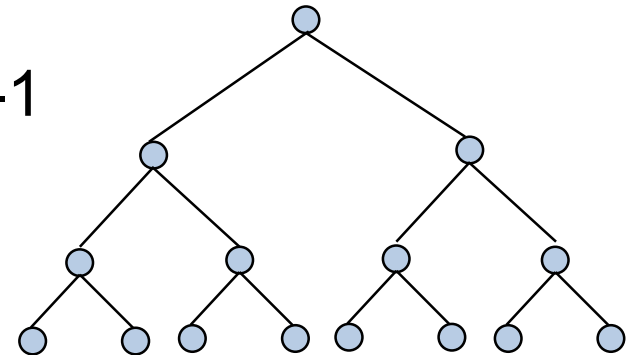
Formas de un AB

Un **árbol binario completo** de altura h tiene:

- Número de nodos en el nivel k : 2^k
- Número total de nodos:

$$n = 2^0 + 2^1 + 2^2 + \dots + 2^h = \sum 2^i = 2^{h+1} - 1$$

- Número de hojas : 2^h
- Número de nodos no terminales : $2^h - 1$
- Altura: $h = \log_2 (n+1) - 1$

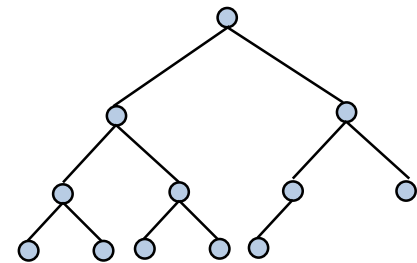
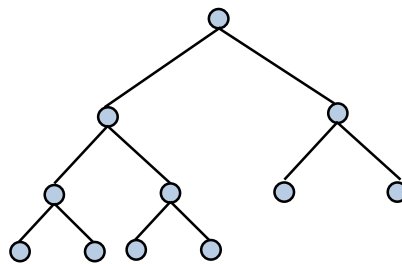
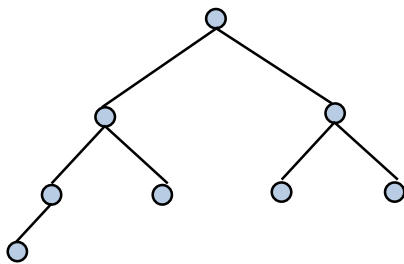


Un ab completo es el ab que tiene el número máximo de nodos para una dada altura.

Formas de un AB

Casi completo - cuasi completo:

Por ejemplo:



- En el ultimo nivel los nodos se ubican de izquierda a derecha.
- Número de nodos del penúltimo nivel = 2^h .
- Número de nodos del ultimo nivel es como mínimo 1 y como máximo 2^h .

Aplicación de AB: Árbol de Expresión

Se puede usar arboles binarios para representar *expresiones aritméticas*.

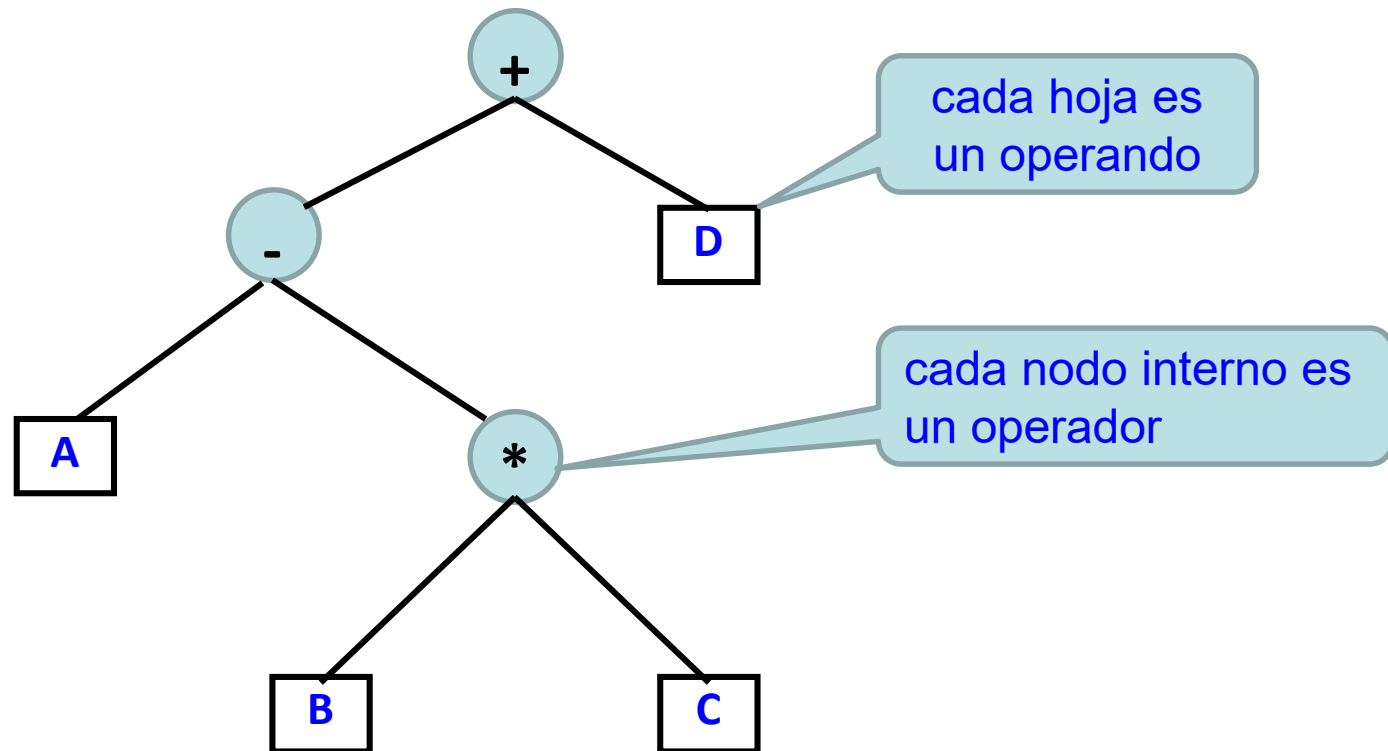
Se representa cada operando como una hoja y cada operador como un nodo no terminal cuyos hijos izquierdo y derecho son los operandos asociados al mismo.

En este caso el tipo de la información asociada a los nodos de un mismo árbol es de distinto tipo:

- los *nodos internos* son *operadores*,
- las *hojas* son *operandos*.

Aplicación de AB: Árbol de Expresión

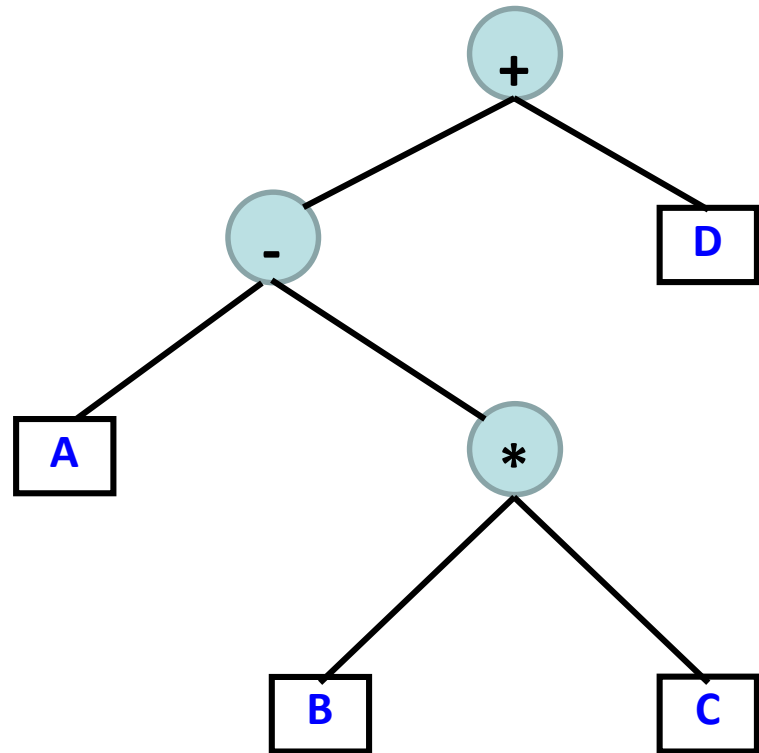
Así por ejemplo la expresión : $A - B * C + D$ se puede representar con:



Aplicación de AB: Árbol de Expresión

Un listado *orden previo* del árbol binario de expresión produce la expresión aritmética *prefija*.

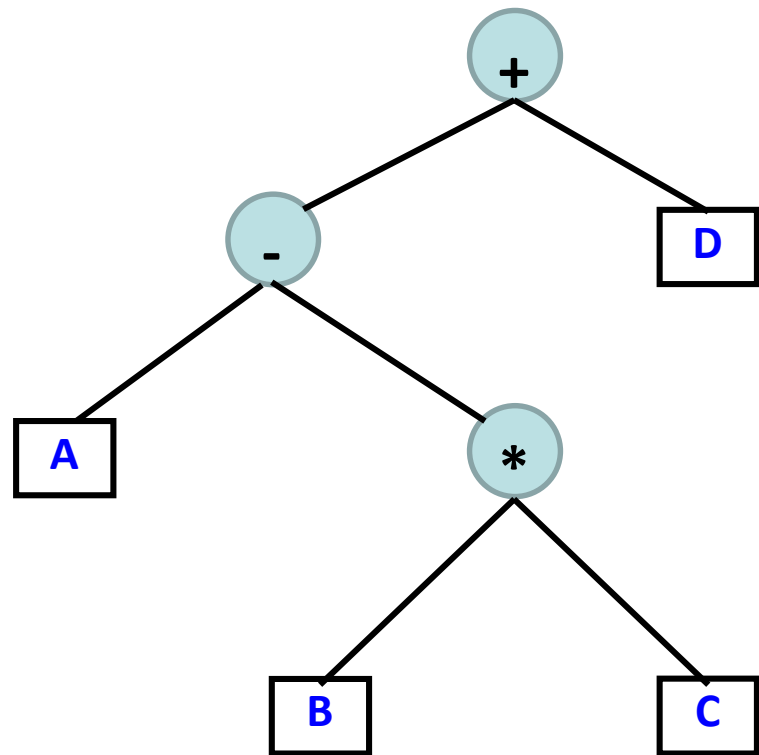
Ejemplo: + - A * B C D



Aplicación de AB: Árbol de Expresión

Un listado en **orden simétrico** de un árbol binario de expresión produce la expresión aritmética **infija sin paréntesis**.

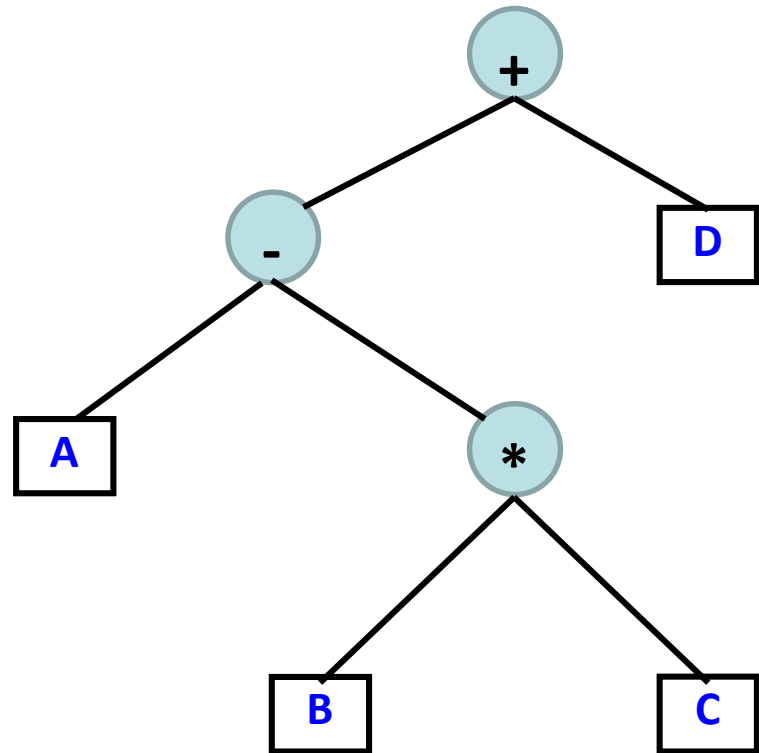
Ejemplo: $A - B * C + D$



Aplicación de AB: Árbol de Expresión

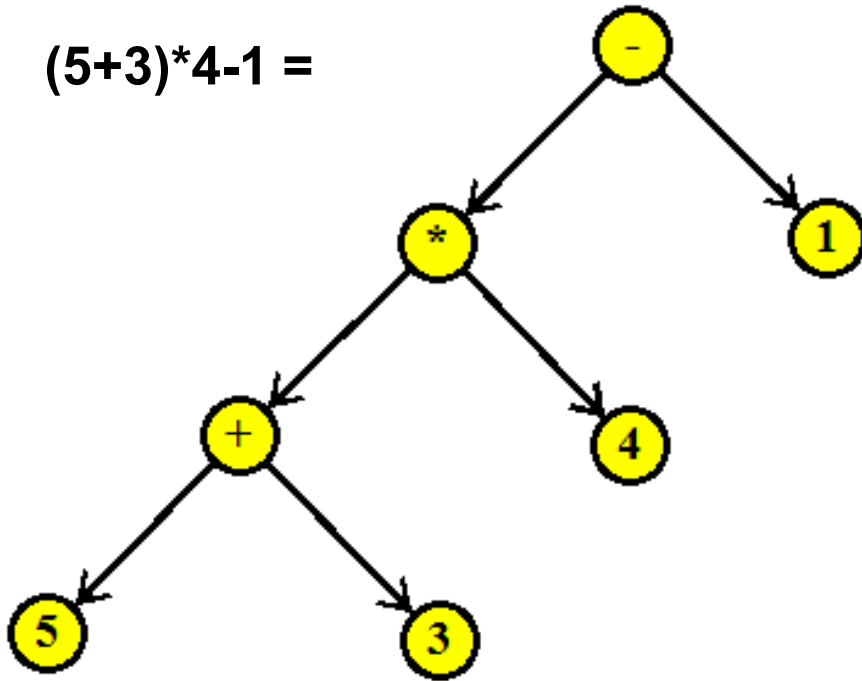
Un listado *orden posterior* del árbol binario de expresión produce la expresión aritmética *postfija*.

Ejemplo: $A B C * - D +$

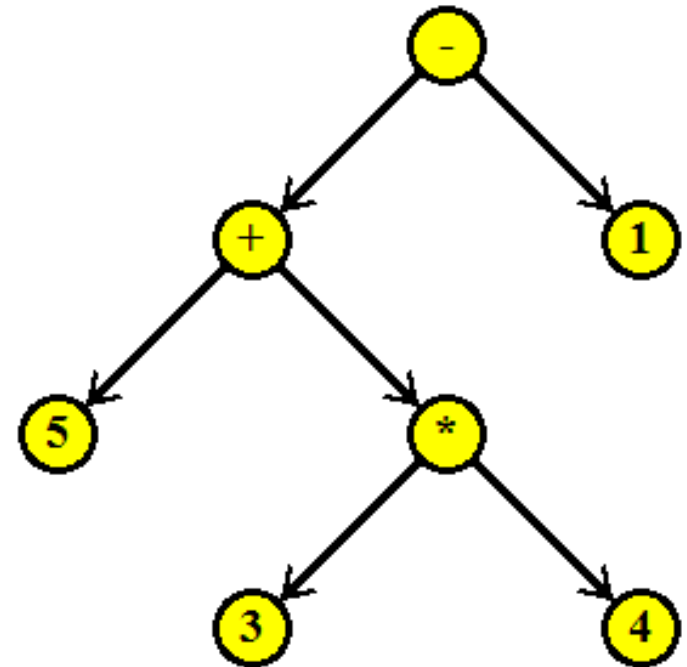


Aplicación de AB: Árbol de Expresión

$(5+3)*4-1 =$



$5+3*4-1 =$



Aplicación de AB: Árbol de Expresión

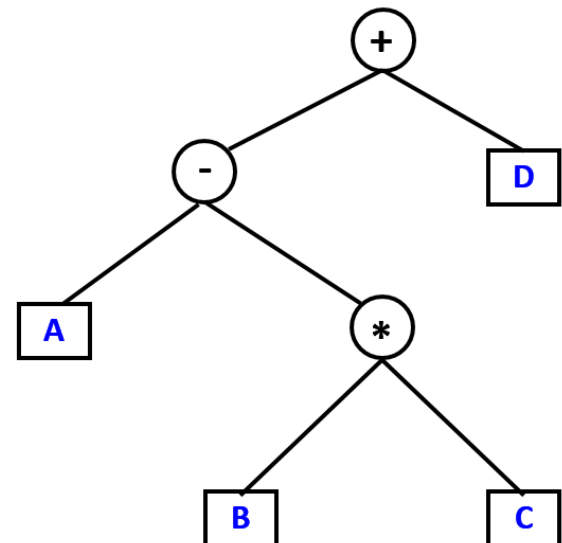
Construcción de un árbol de expresión a partir de una expresión aritmética infija:

- 1) Se pasa la expresión aritmética de infija a postfija
- 2) Se construye el ab con ayuda de una *pila de arboles*.

Ejemplo:

Expresión Infija: **A - B * C + D**

Expresión Posfija: **A B C * - D +**



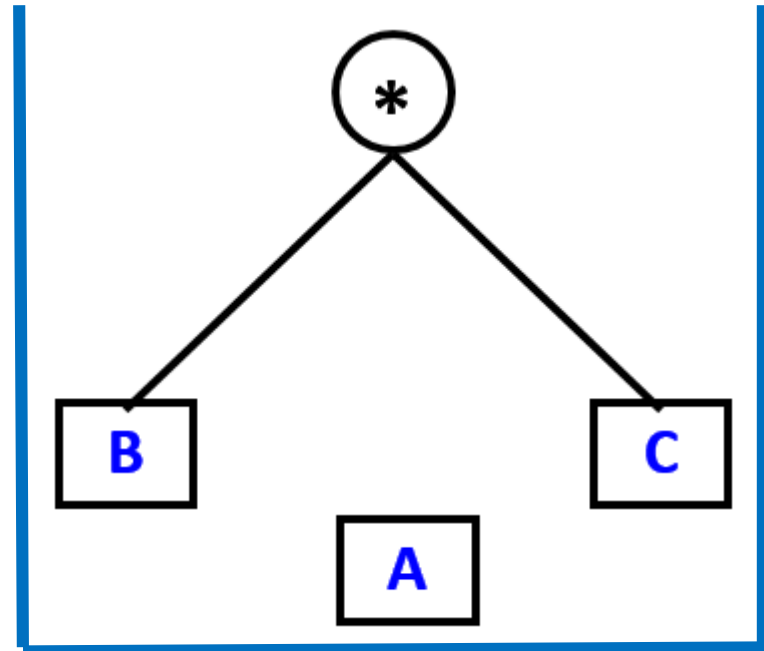
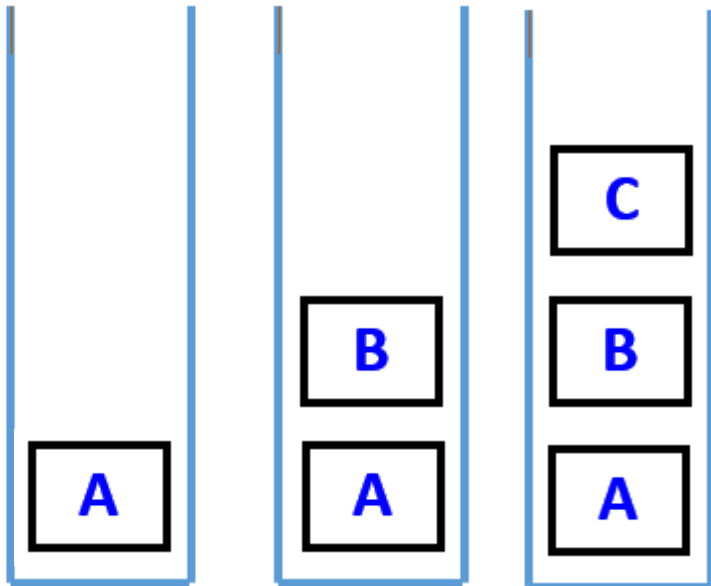
Ejemplo:
Árbol de Expresión: **A B C * - D +**

A

B

C

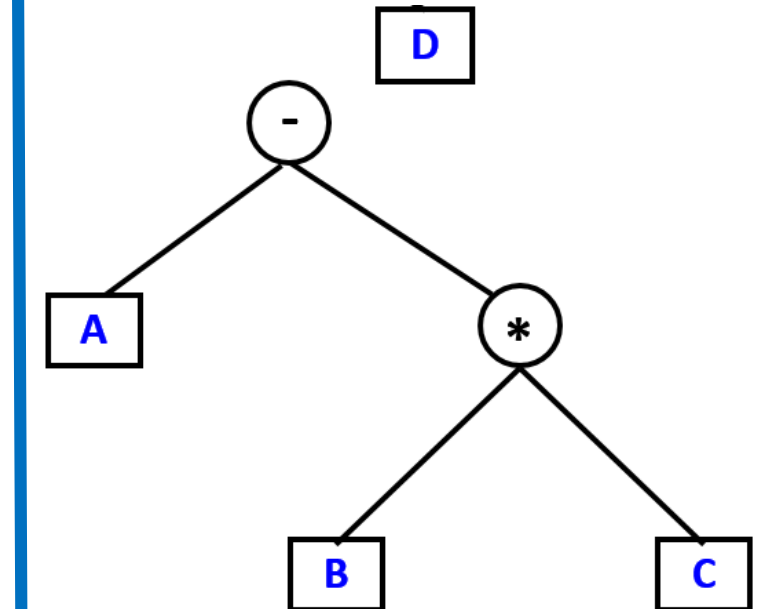
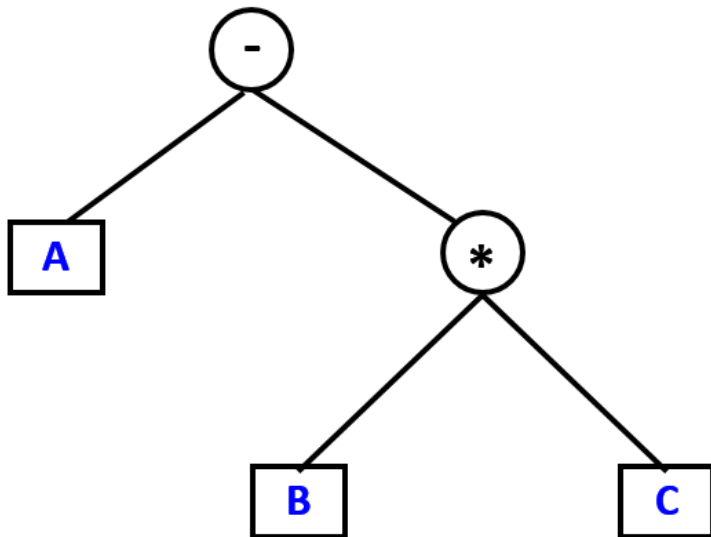
*



Ejemplo: Árbol de Expresión: **A B C * - D +**

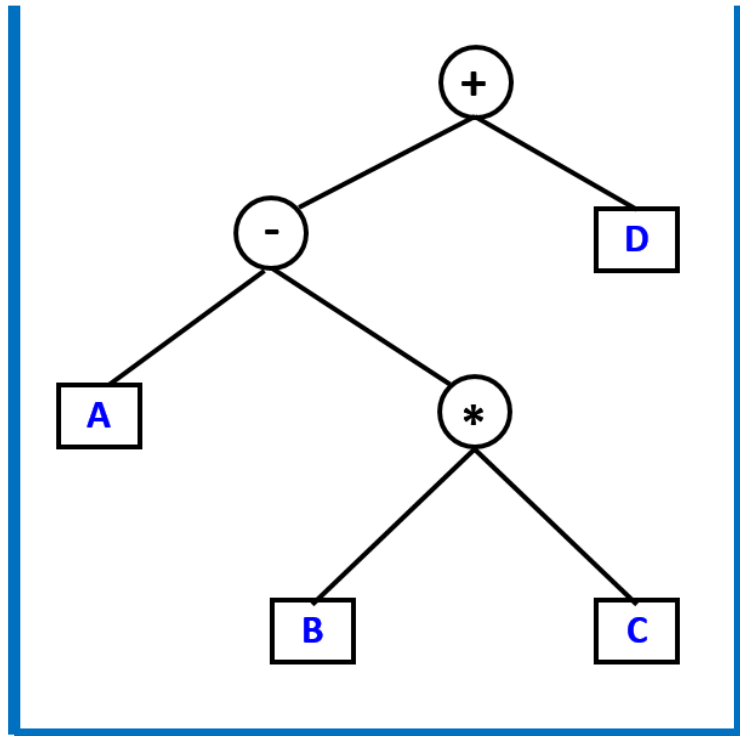
-

D



Ejemplo: Árbol de Expresión: **A B C * - D +**

+



Aplicación de AB: Árbol de Decisión

Un **árbol de decisión** es un árbol binario con raíz que se usa para representar el funcionamiento de un algoritmo con todos los datos posibles.

Cada nodo interno del árbol contiene una prueba que se aplica a los datos.

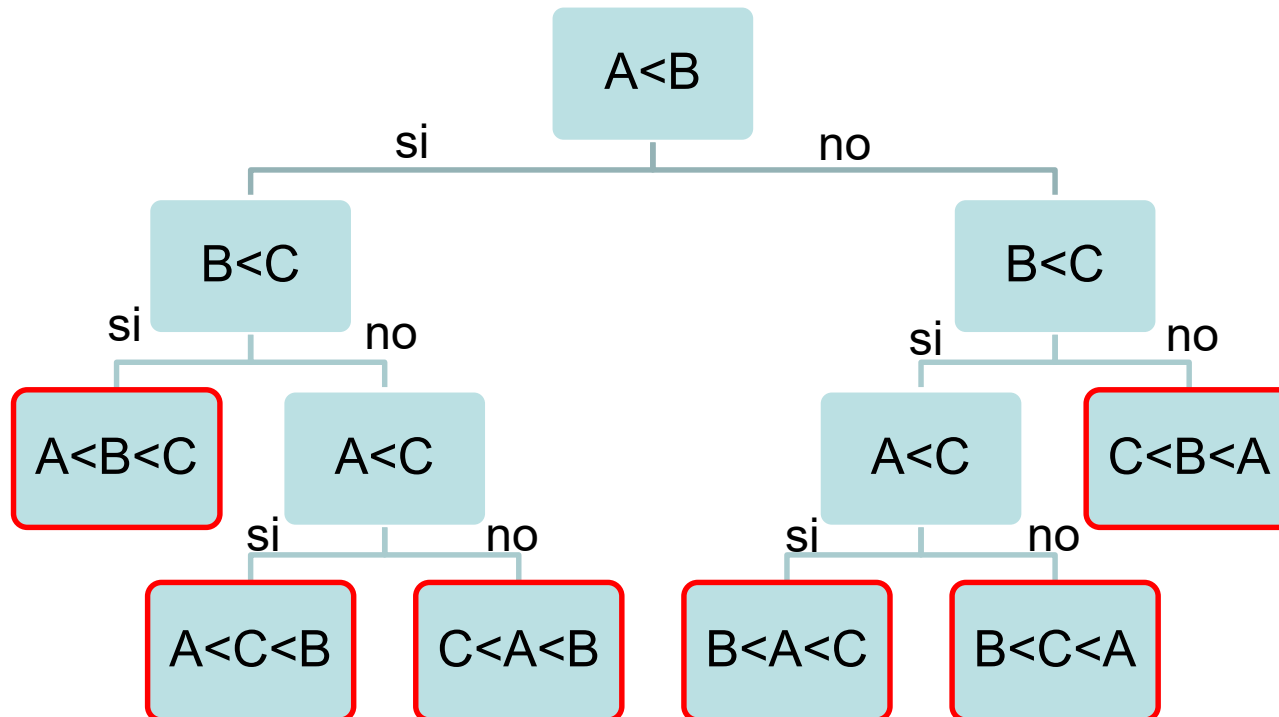
Cada hoja contiene una salida llamada veredicto.

Un camino en el árbol comienza en la raíz y hace la pregunta que se encuentra allí, si la respuesta es “sí” sigue por el subárbol izquierdo, sino sigue por el subárbol derecho. Se sigue recursivamente y el camino termina cuando se alcanza una hoja con el resultado.

Aplicación de AB: Árbol de Decisión

Ejemplo 1:

Ordenar 3 datos: A,B,C, distintos entre sí.



Aplicación de AB: Árbol de Decisión

Ejemplo 2:

Ordenar 3 datos: A,B,C, distintos entre sí.

