PROGRAMA EDUCATIVO INFORMATICA ADMINISTRATIVA

UNIDAD DE APRENDIZAJE ADMINISTRACION DE BASES DE DATOS

Unidad de competencia III

Manejar las estructuras dinámicas en memoria secundaria

Arboles

ELABORACION
ADRIAN TRUEBA ESPINOSA

PRESENTACIÓN DEL CURSO

El conocimiento de las estructuras básicas de los datos es de vital importancia para el desarrollo de programas eficientes en cuanto a la optimización de memoria y tiempo de ejecución de una aplicación, Se busca, con este programa, que el alumno desarrolle el criterio de elegir las mejores estructuras de datos que le permitan manejar información de una manera eficiente y óptima.

CONTENIDO DEL CURSO

Unidad 1: Manipular con eficiencia los tipos y valores de información así como el manejo de arreglos y registros

Unidad 2: Manejar las Estructuras dinámicas en memoria central.

Unidad 3: Manejar las Estructuras dinámicas en memoria secundaria

METAS A ALCANZAR

Que el alumno conozca los elementos teóricos y prácticos de la estructura de datos “Arboles”

- Conceptos
- Terminología
- Métodos gráficos de representación
- Recorridos

OBJETIVO DEL MATERIAL DIDÁCTICO

Crear las competencias para la comprensión y manejo de la estructura de datos arboles

METODOLOGÍA DEL CURSO

El curso se desarrollará bajo el siguiente proceso de estudio:

1. Exposición de parte del profesor mediante la utilización de este material en diapositivas.
2. Control de lecturas selectas que el profesor asignará para complementar la clase.
3. Investigación de temas, conceptos, terminología y métodos gráficos de representación de los arboles
4. Participación en clases
5. Prácticas de laboratorio

UTILIZACIÓN DEL MATERIAL DE DIAPOSITIVAS

El material didáctico visual es una herramienta de estudio que sirve como una guía para que el alumno repase los temas más significativos de “Las bases de datos lógicas”,.

Unidad de competencia III

Manejar las estructuras dinámicas en memoria secundaria

Arboles

Árboles Binarios

INTRODUCCION

Las estructuras dinámicas son las que en la ejecución varia el número de elementos y uso de memoria)

Entre ellas están:

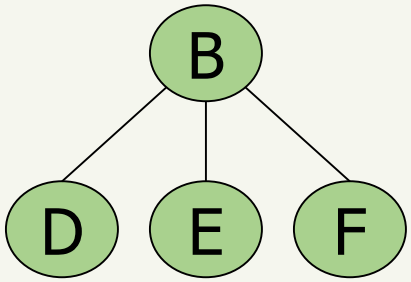
Lineales (listas enlazadas, pilas y colas)

No lineales (arboles binarios y grafos o redes)

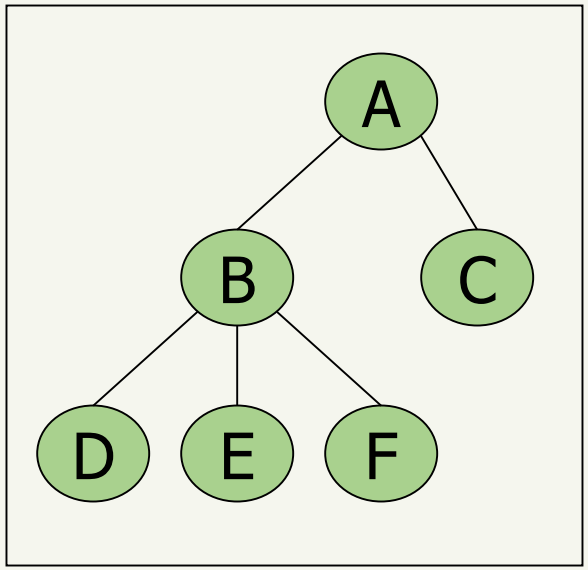
- Las listas enlazadas son estructuras lineales
  - Son flexibles pero son secuenciales, un elemento detrás de otro
- Los árboles
  - Junto con los grafos son estructuras de datos no lineales
  - Superan las desventajas de las listas
  - Sus elementos se pueden recorrer de distintas formas, no necesariamente uno detrás de otro
- Son **muy útiles** para la búsqueda y recuperación de información

CONCEPTO

- Estructura que organiza sus elementos formando jerarquías: PADRES E HIJOS
  - Los elementos de un árbol se llaman **nodos**
  - Si un nodo **p** tiene un enlace con un nodo **m**,
    - **p** es el padre y **m** es el hijo
    - Los hijos de un mismo padre se llaman: **hermanos**
  - Todos los nodos tienen al menos un padre, menos la raíz: A
  - Si no tienen hijos se llaman **hoja**: D, E, F y C
  - Un subárbol de un árbol
    - Es cualquier nodo del árbol junto con todos sus descendientes



A es Padre de B y C, también son hijos de A y son hermanos
B es Padre de D, E, F hijos de B y son hermanos

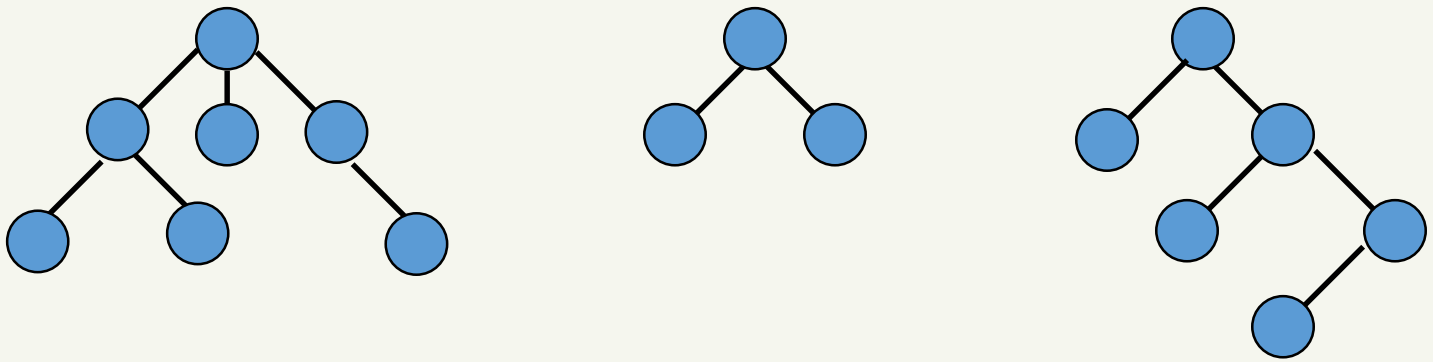


TERMINOLOGIA

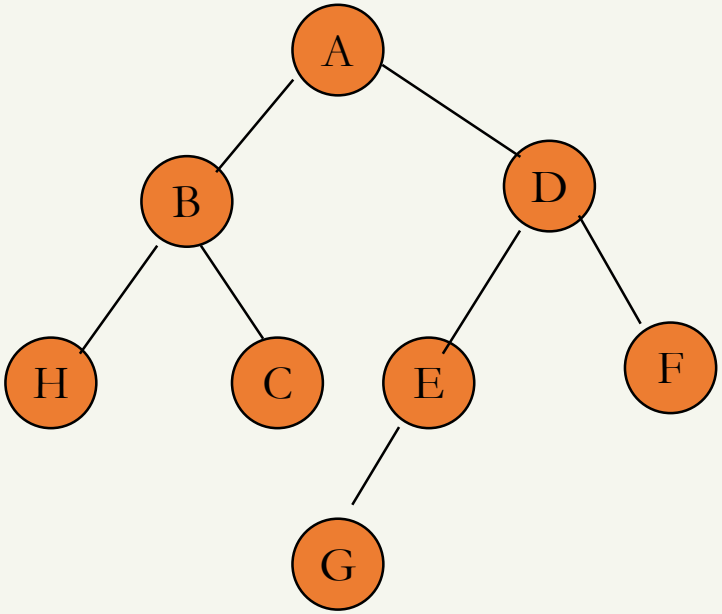
- **Camino**: Secuencia de nodos conectados dentro de un árbol
- **Longitud del camino**: es el numero de nodos menos 1 en un camino
- **Altura del árbol**: es el nivel mas alto del árbol
  - Un árbol con un solo nodo tiene altura 1
- **Nivel(profundidad) de un nodo**: es el numero de nodos entre el nodo y la raíz.
- **Nivel de un árbol**
  - Es el numero de nodos entre la raíz y el nodo mas profundo del árbol, la altura del un árbol entonces
- **Grado(aridad) de un nodo**: es numero de hijos del nodo
- **Grado(aridad) de un árbol**: máxima aridad de sus nodos

¿Qué es un Árbol?

- Es una estructura de datos jerárquica.
- La relación entre los elementos es de uno a muchos.

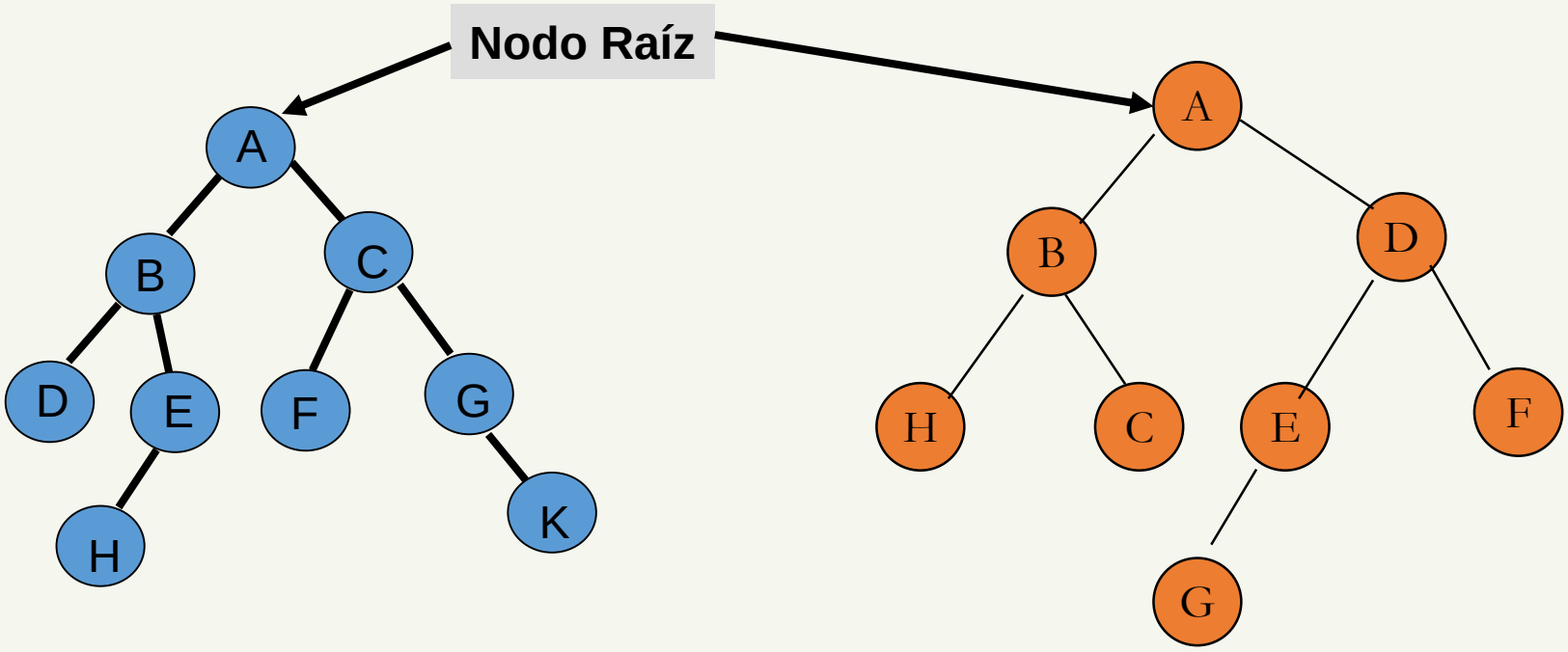


Árbol con 8 nodos

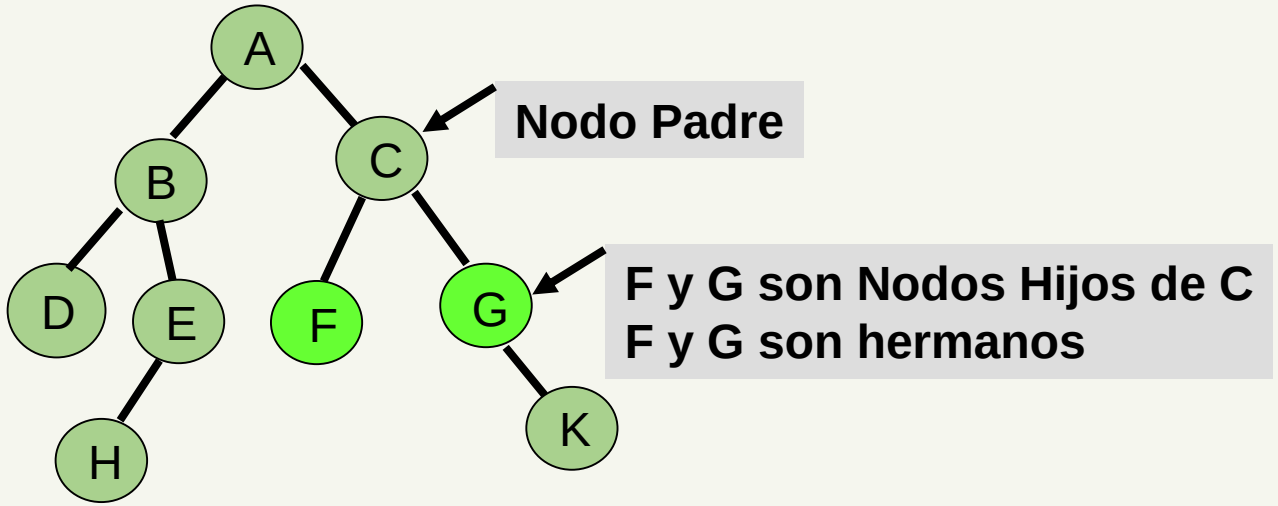


Terminología

- Nodo: Cada elemento en un árbol.
- Nodo Raíz: Primer elemento agregado al árbol
- *Es el nodo que no es apuntado por ningún otro nodo*.

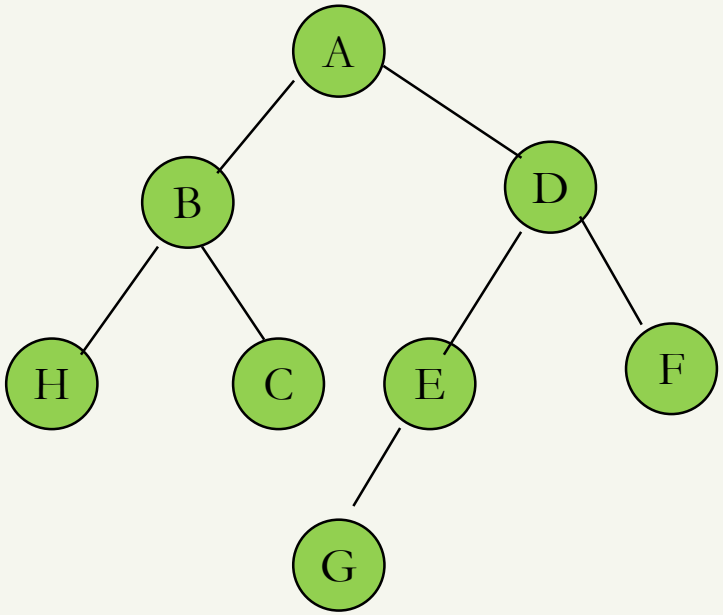


- **Nodo Padre: Se le llama así al nodo predecesor de un elemento.**
- **Nodo Hijo: Es el nodo sucesor de un elemento.**
- **Hermanos: Nodos que tienen el mismo nodo padre.**



Padre

X es el padre de Y si X apunta a Y



A es el padre de B y de D

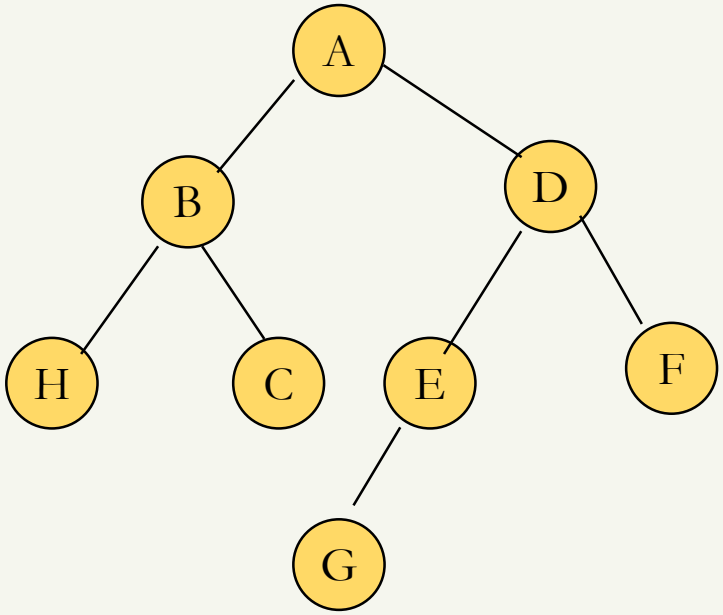
B es el padre de H y C

H no es padre de nadie

A no es el padre de H!

Hijo

Y es el hijo de X si X apunta a Y

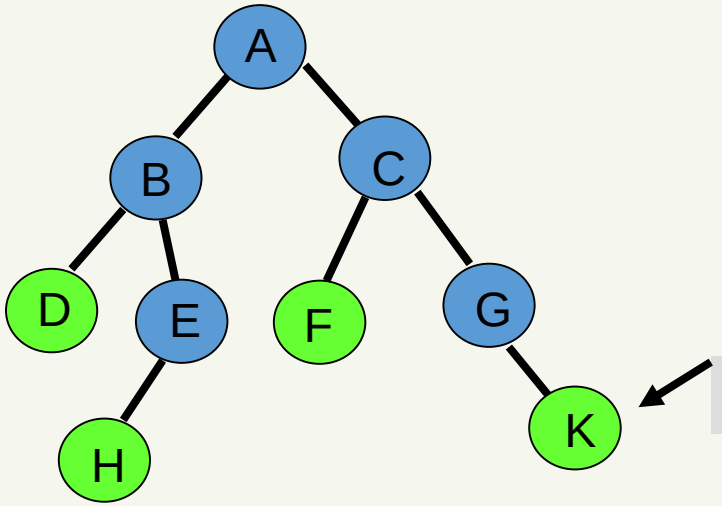


B y D son hijos de A

H y C son hijos de B

H no es un hijo de A

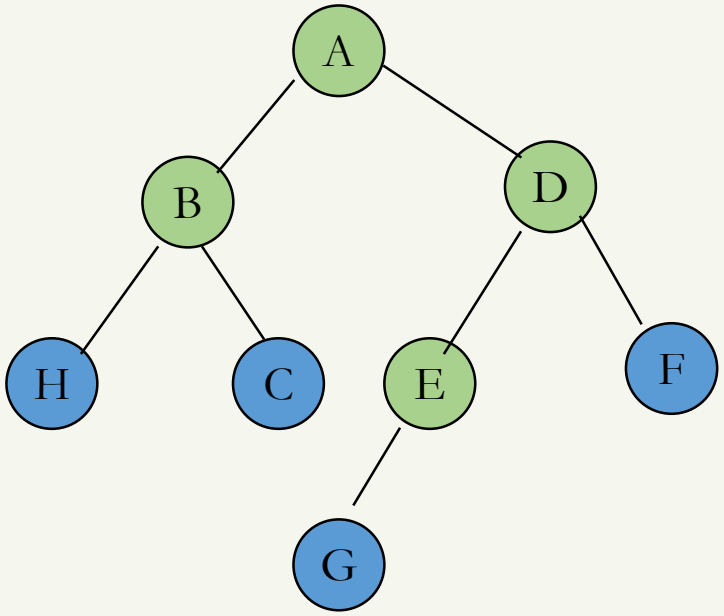
- Nodo Hoja: Aquel nodo que no tiene hijos.



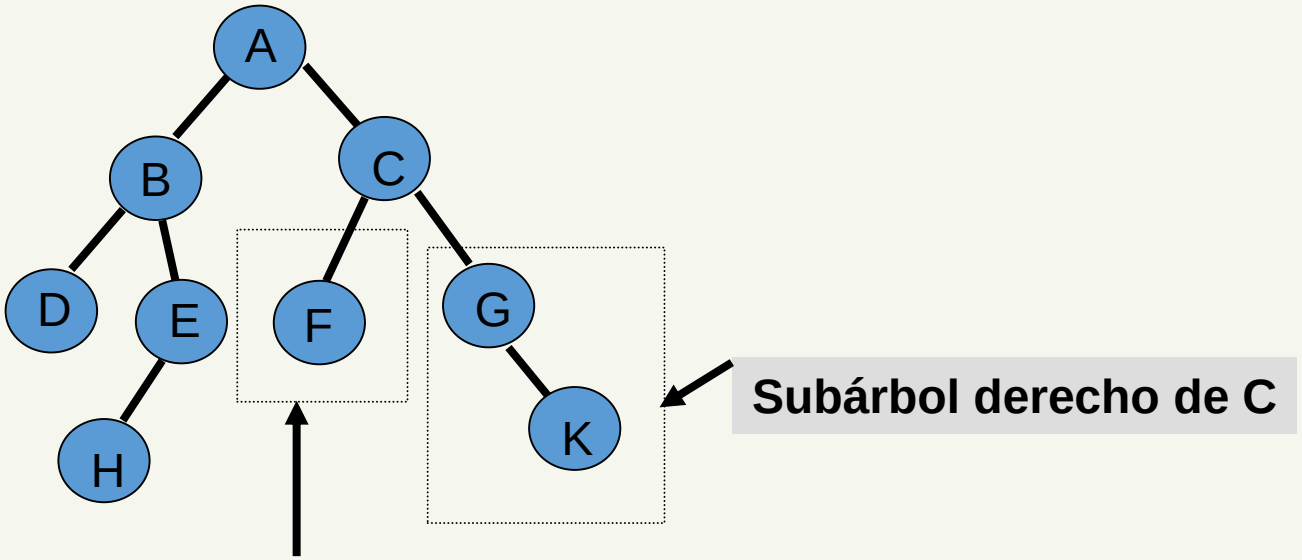
D, H, F y K son Nodos Hojas

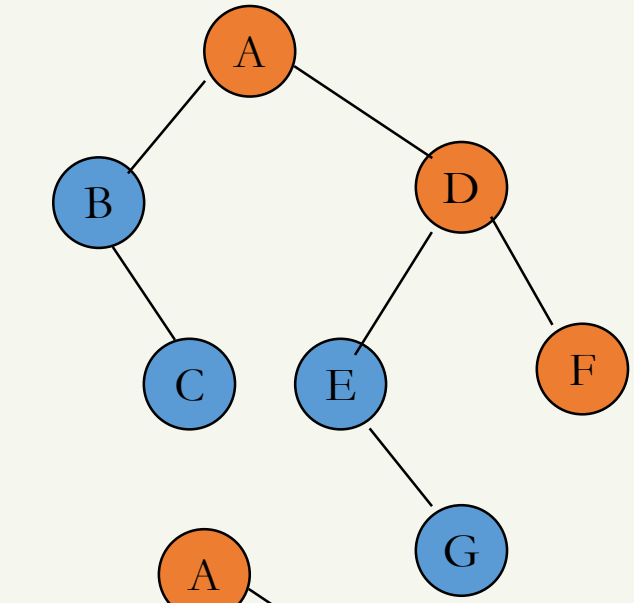
Nodo no terminal

Es un nodo que no es hoja

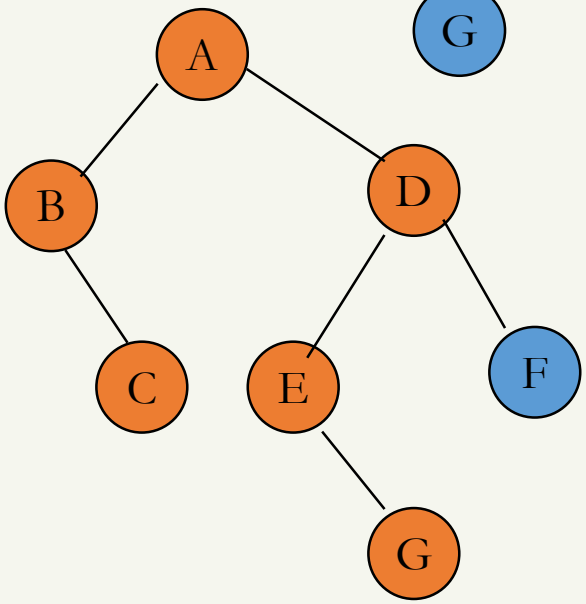


- Subárbol: Todos los nodos descendientes por la izquierda o derecha de un nodo.





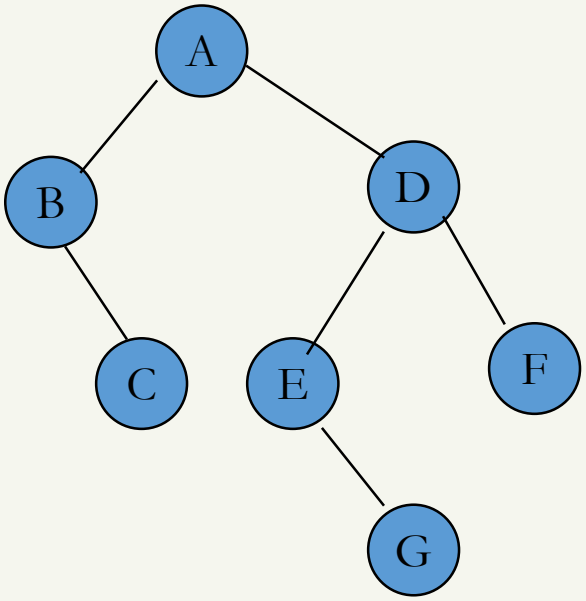
El camino A->D->F se presenta en el árbol



El camino G->E->D->A->B->C no se da en el camino

Longitud

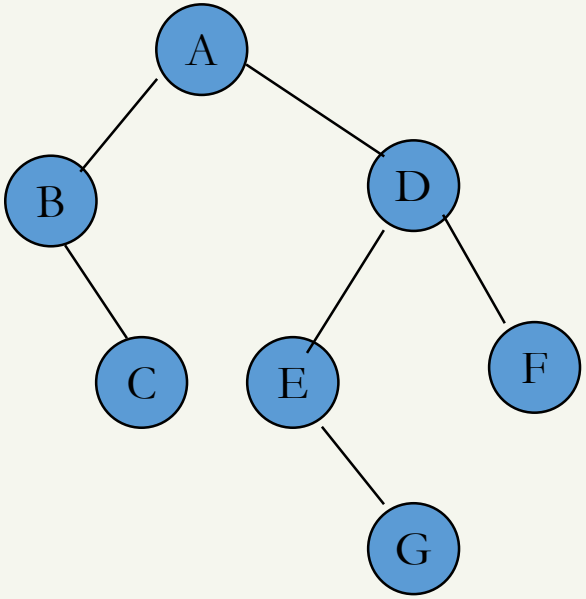
Es el número de nodos que se deben recorrer para pasar de un nodo a otro



La longitud entre A y G es 3
La longitud entre A y A es 0
La longitud entre D y F es 1

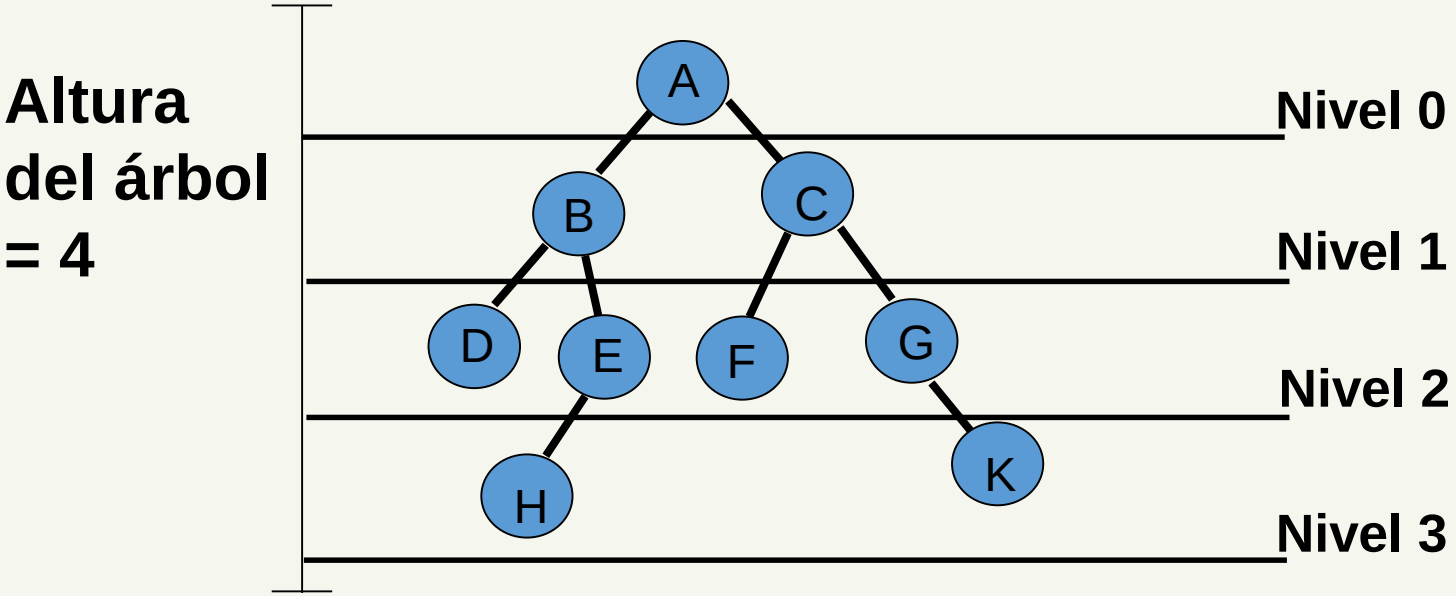
Ancestro

Un nodo X es ancestro de Y si existe una camino ente X y Y



Es G un ancestro de B?
Es B un ancestro de E?
Es A un ancestro de F?

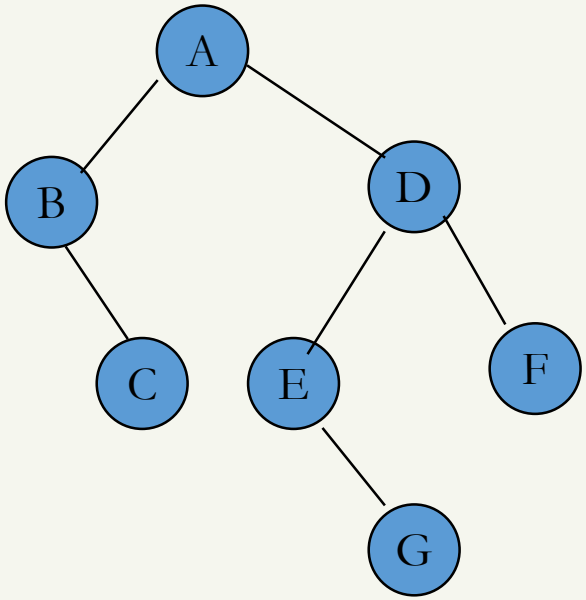
Altura y Niveles



La Altura es la cantidad de niveles.

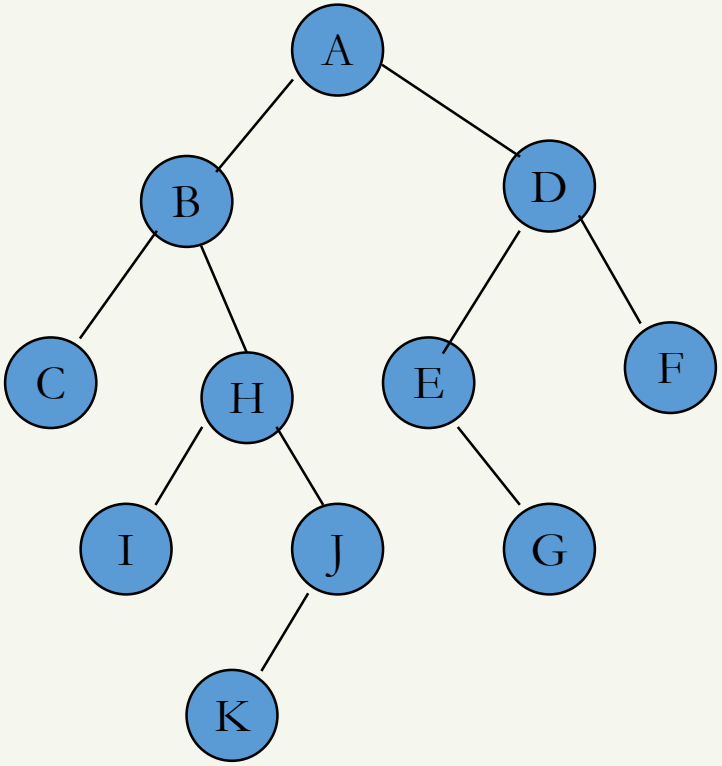
Grado de un nodo

Es el número de hijos
El grado de un nodo terminal siempre es 0
En un árbol binario el grado de cada nodo varia entre 0 y 2



El grado de A es 2
El grado de D es 2
El grado de E es 1
El grado de C, G y F es 0

Ejercicio



Cuántos nodos tiene
Cuál es el nodo raíz
Cuáles son los nodos no terminales
Cuáles son las hojas
Cuál es el grado del nodo E
Cuál es el nivel del nodo I
Cuál es la longitud entre A y K
Se presente el camino B-A-D-F
Cuál es la altura del árbol

Árbol binario

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

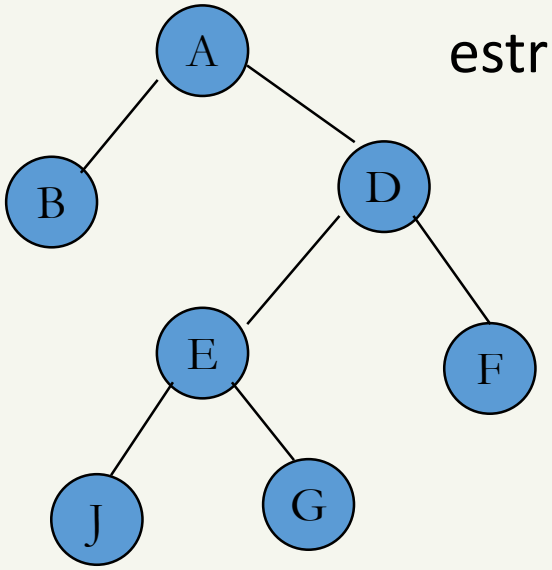
El primer subconjunto contiene un elemento único llamado raíz del árbol. Los otros dos subconjuntos son por si mismos árboles binarios y se les conoce como subárboles izquierdo y derecho del árbol original.

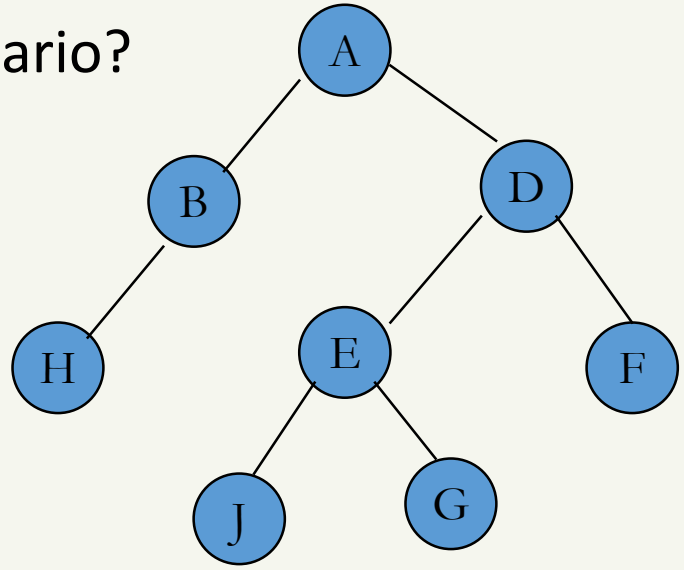
Un subárbol izquierdo o derecho puede estar vacío. Cada elemento de un árbol binario se denomina nodo del árbol

Árbol estrictamente binario

Si cada nodo que no es una hoja en un árbol binario tiene subárboles izquierdo y derecho que no están vacíos, se clasifica como árbol estrictamente binario

Es este un árbol estrictamente binario?





Árbol binario completo

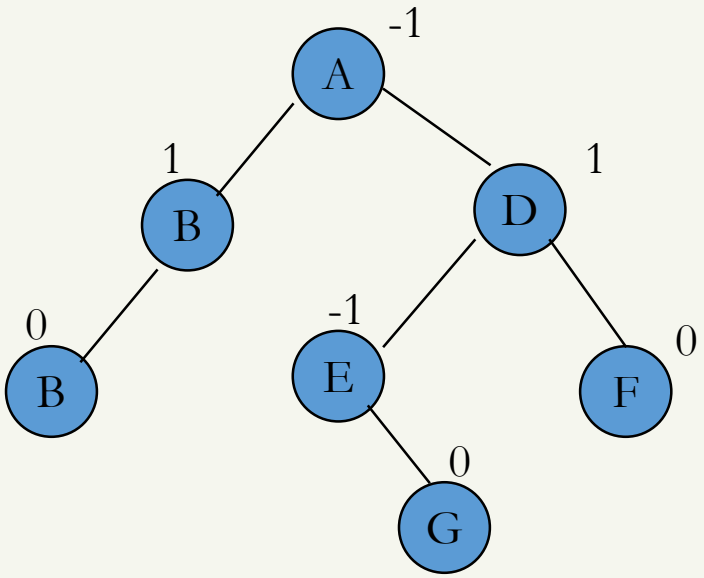
Un árbol binario completo tiene 2^l nodos en cada nivel l, donde l varia entre 0 y d

$$totalNodos = 2^0 + 2^1 + 2^2 + \ldots + 2^d = \sum_{j=0}^{d} 2j = 2^{d+1} - 1$$

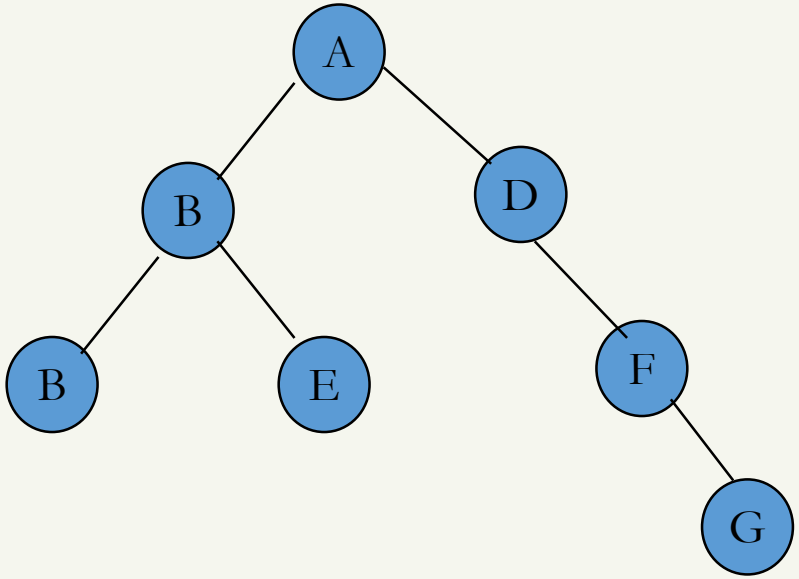
- Cuántos nodos no terminales tiene un árbol binario completo?
- Cuál es la profundidad de un árbol binario completo con T nodos?

Árbol balanceado

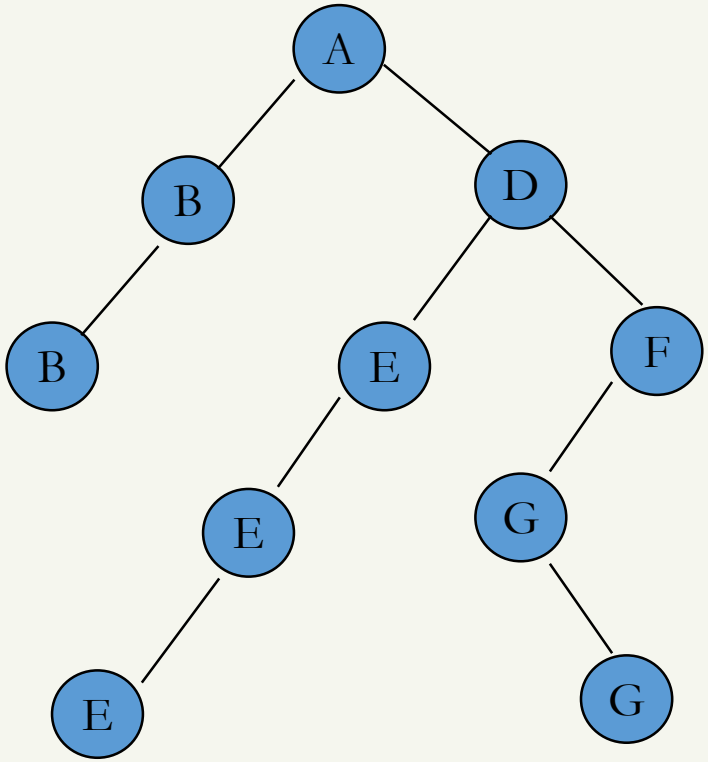
Un árbol binario balanceado o AVL, es aquel en el que el balance para cada nodo es -1, 0 ó 1



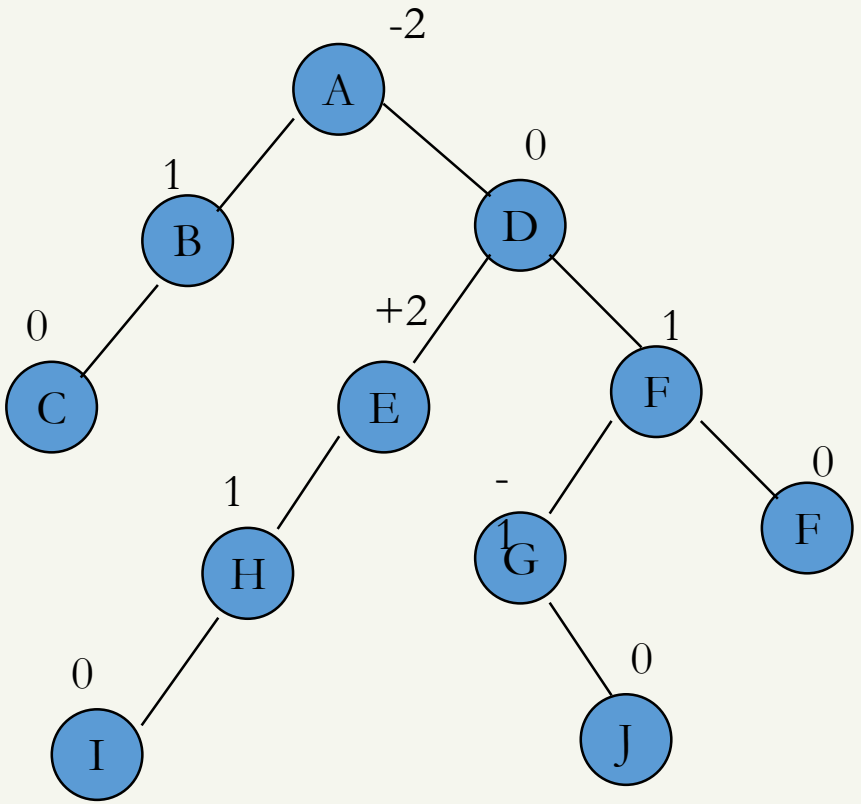
Se indican los balances para cada nodo



Indique los balances de cada nodo, y determine si el árbol es o no, AVL



Indique los balances de cada nodo, y determine si el árbol es o no, AVL



Árboles

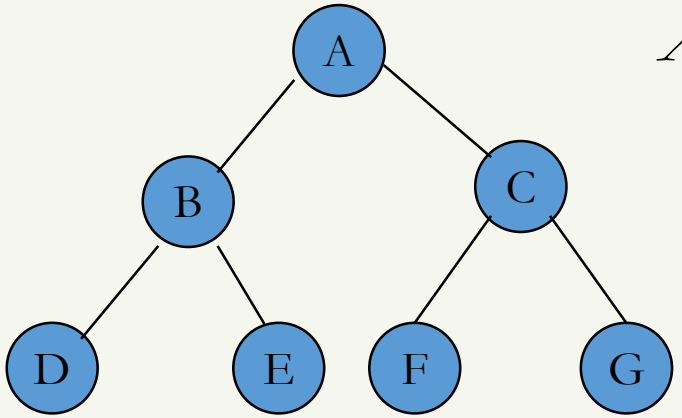
Formas de recorrer un árbol

- *Preorden*
- *Inorden*
- *posorden*

Preorden

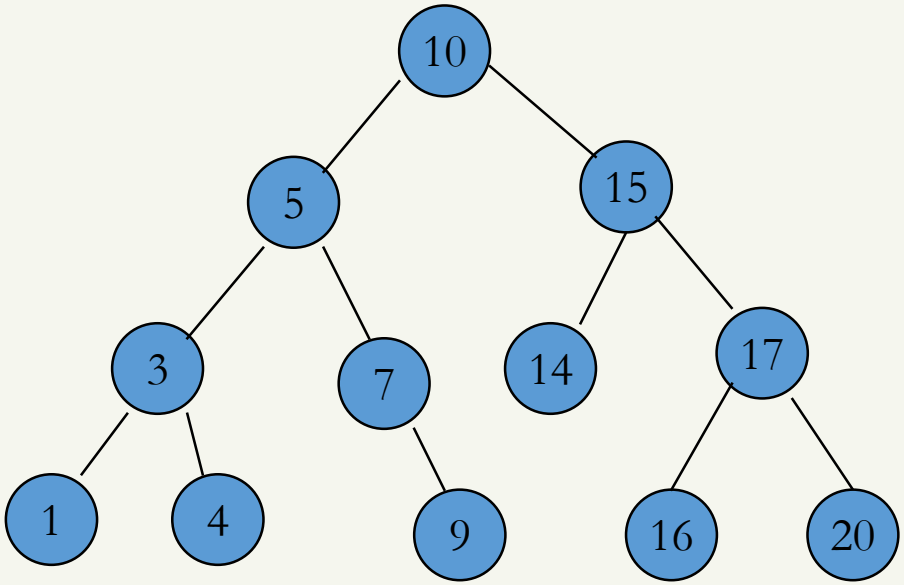
Examinar el dato del nodo raíz
Recorrer el árbol izquierdo en preorden
Recorrer el árbol derecho en preorden

Recorrer el árbol en preorden



ABDECFG

Recorrer el árbol en preorden

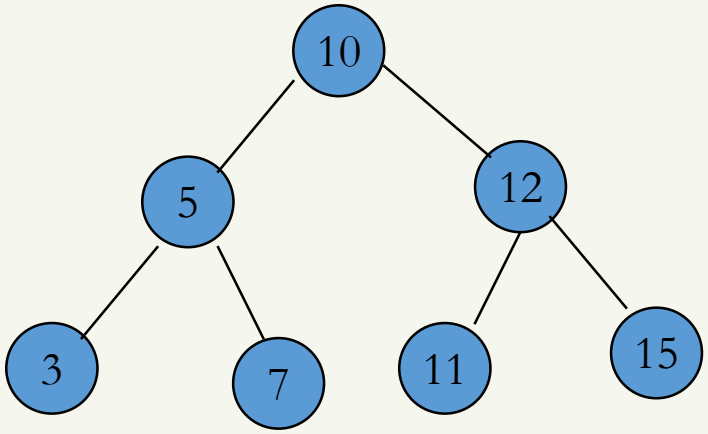


10-5-3-1-4-7-9-
15-14-17-16-20

Árboles

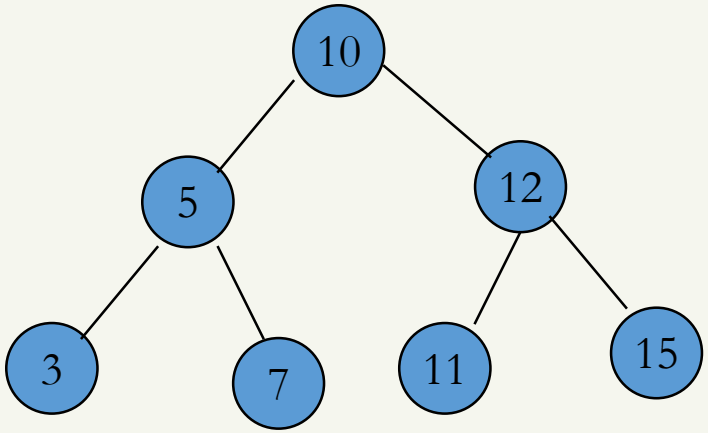
Inorden

Recorrer el árbol izquierdo en inorden
Examinar el dato del nodo raíz
Recorrer el árbol derecho en inorden

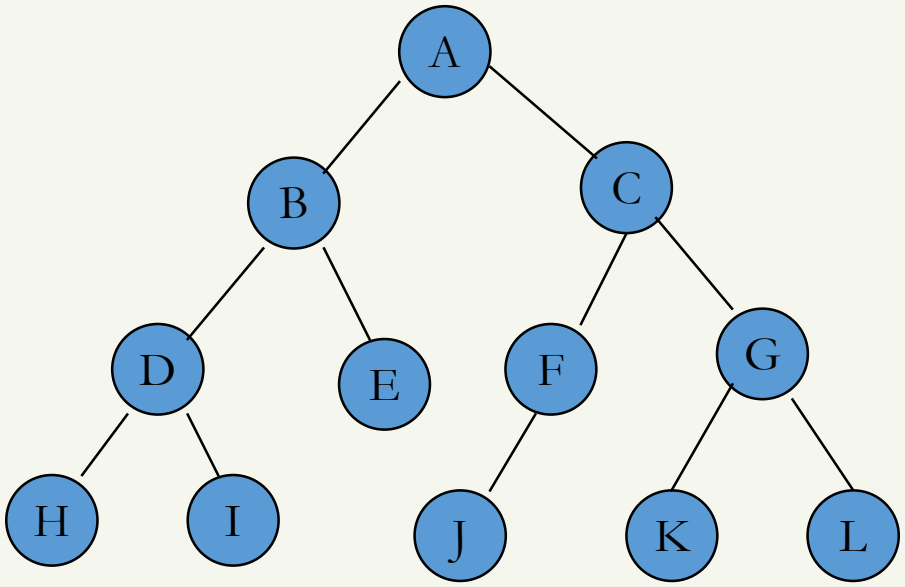


Recorrer el árbol en inorden

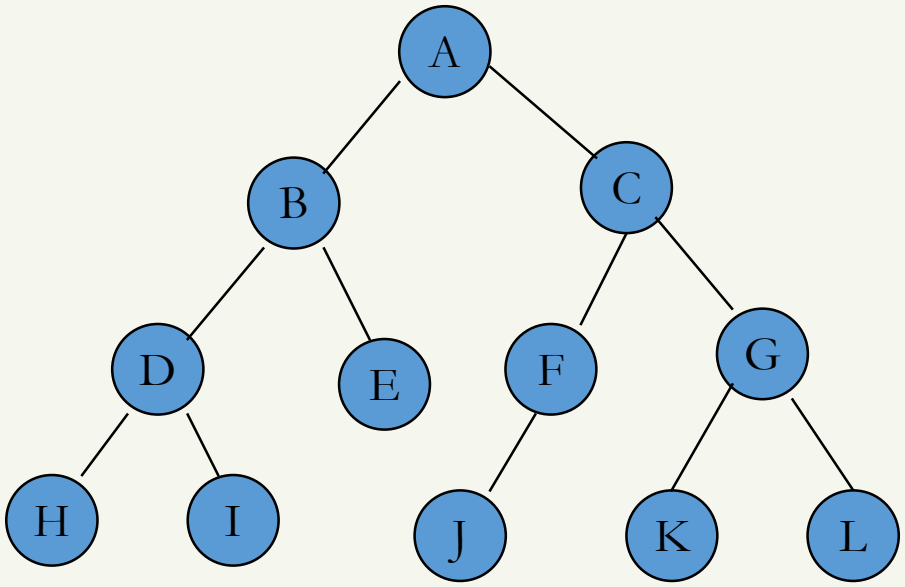
Árboles



Árboles



Árboles



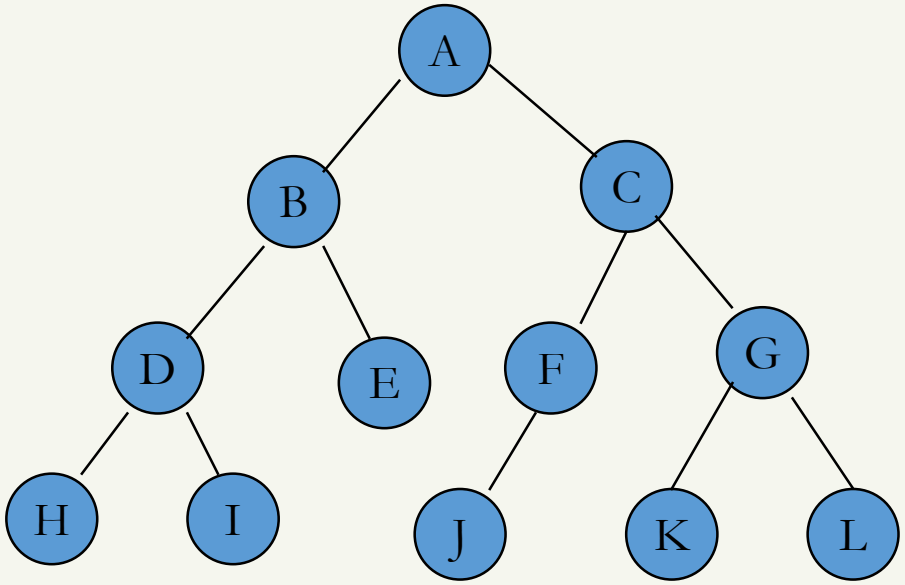
H D I B E A J F C K G L

Árboles

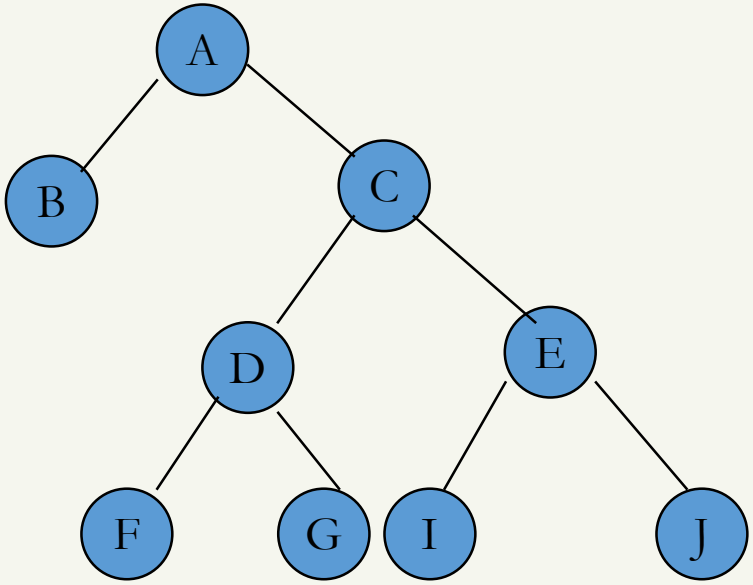
Posorden

Recorrer el árbol izquierdo en posorden
Recorrer el árbol derecho en posorden
Examinar el dato del nodo raíz

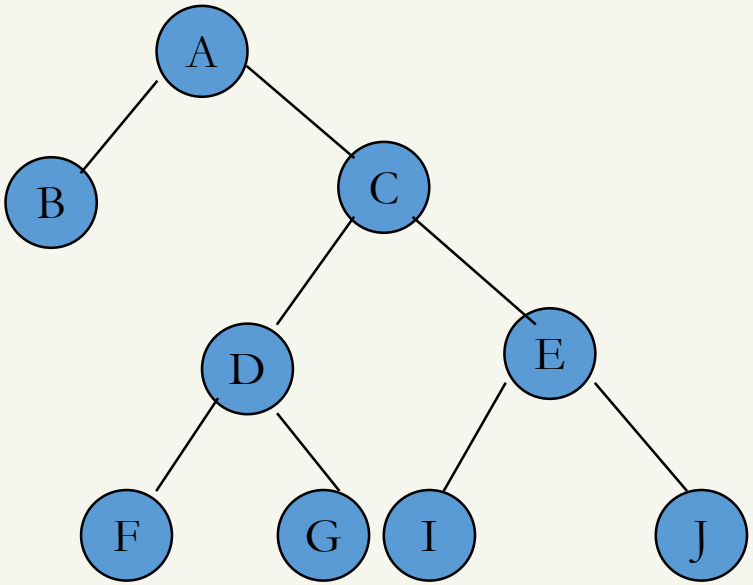
-Recorrer el árbol en posorden



H I D E B J F K L G C A

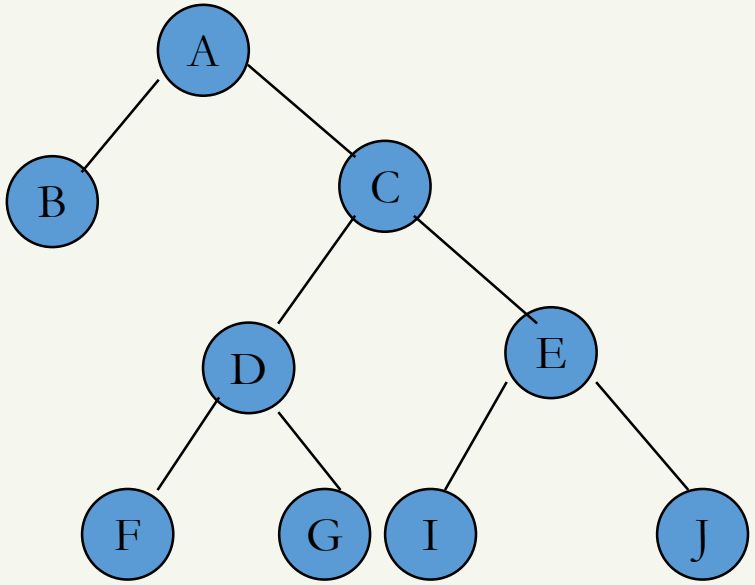


Muestre el resultado de recorrer el árbol en preorden, inorden y posorden



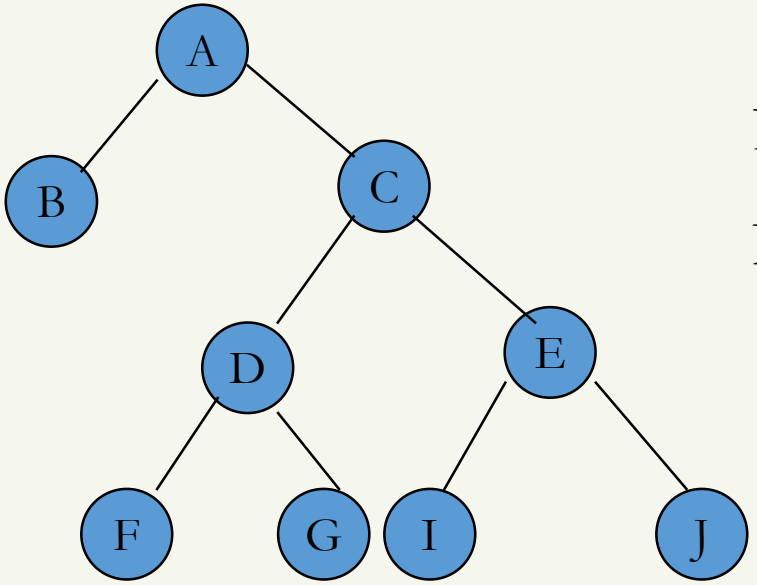
Preorden

A B C D F G E I J



Inorden

B A F D G C I E J



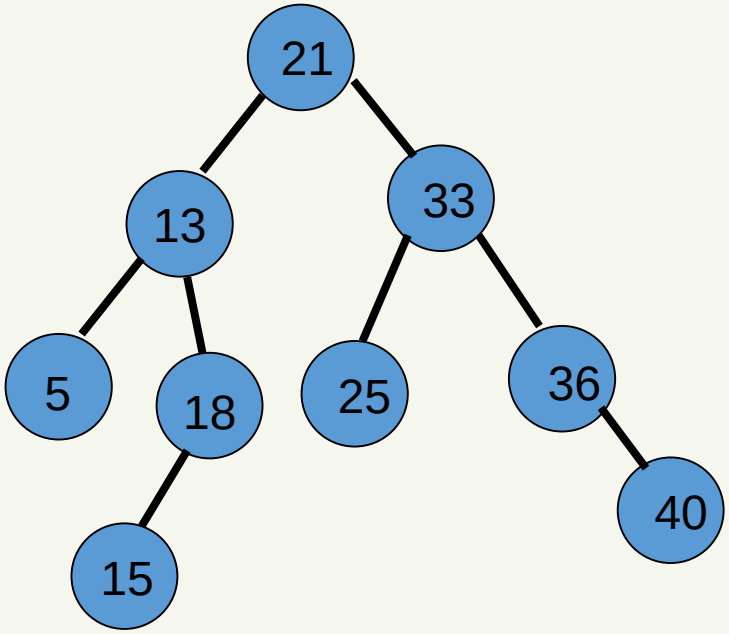
Posorden

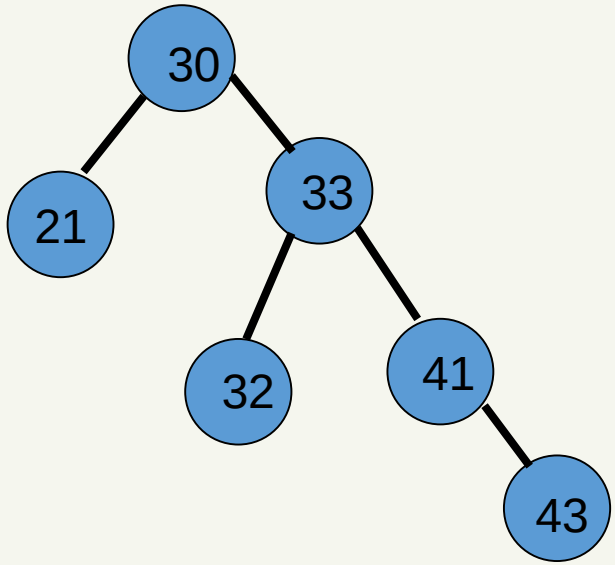
B F G D I J E C A

Árbol Binario de Búsqueda (ABB)

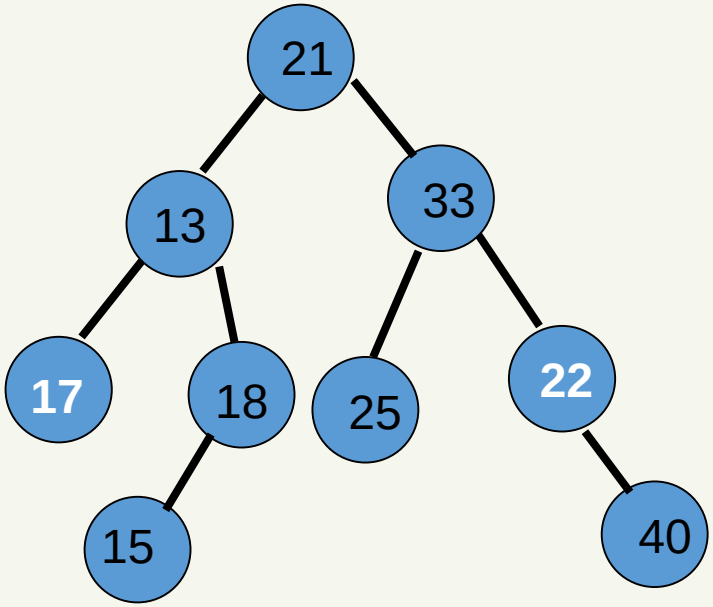
- Este tipo de árbol permite almacenar información ordenada.
- Reglas a cumplir:
  - Cada nodo del árbol puede tener 0, 1 ó 2 hijos.
  - Los descendientes izquierdos deben tener un valor menor al padre.
  - Los descendientes derechos deben tener un valor mayor al padre.

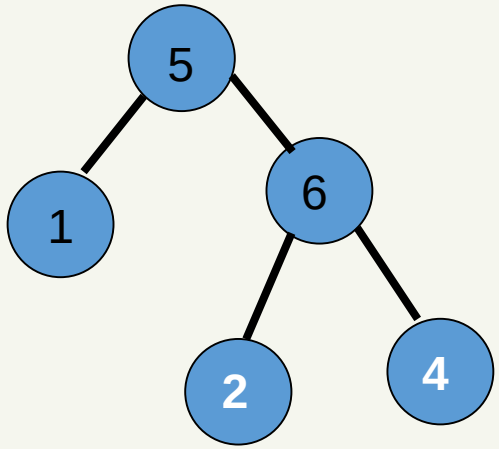
Ejemplos de ABB…





¿Por qué no son ABB?





Implementación de un ABB…

```
class NodoArbol
{
    public:
        int info;
        NodoArbol *izq, *der;
        NodoArbol( );
        NodoArbol(int dato);
};
NodoArbol(void) { izq = der = NULL; }
NodoArbol(int dato) { info = dato; izq = der = NULL; }
```

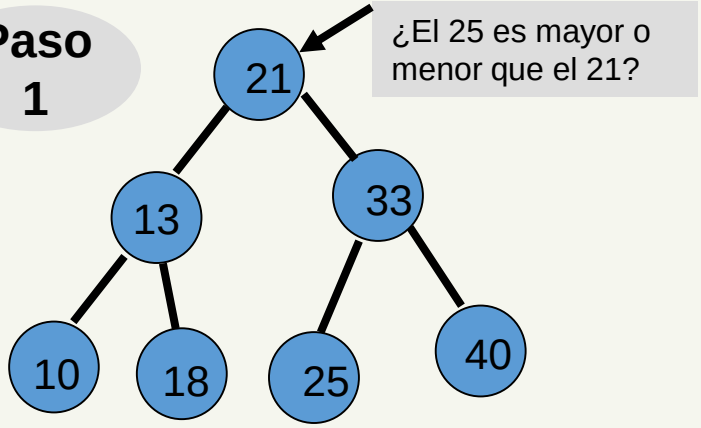
Continuación…

```
class ABB
{
    private:
        NodoArbol *raiz;
    public:
        ABB( );      // constructor
        ~ABB( );     // destructor
        //otros métodos
};
```

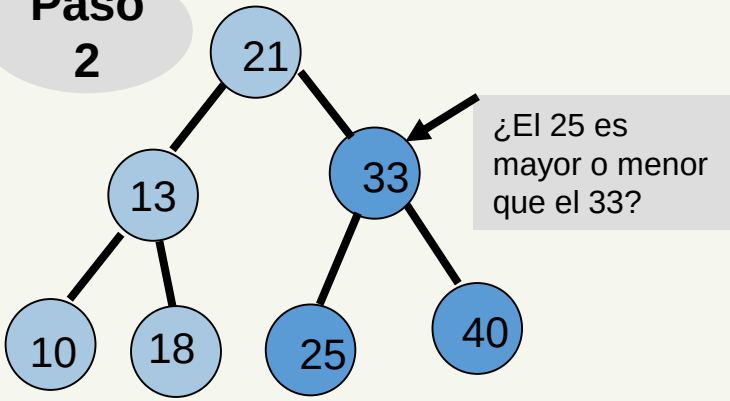
Proceso para buscar un nodo...

Buscar el 25

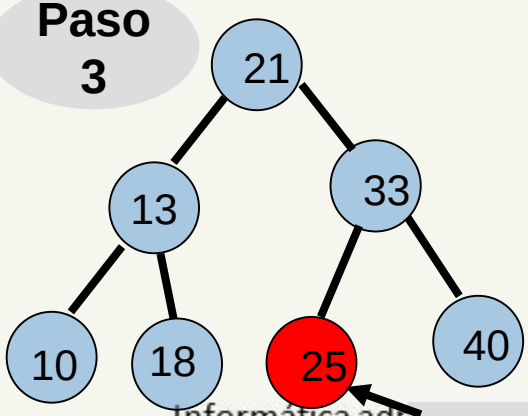
Paso 1



Paso 2



Paso 3



Implementación de la búsqueda

```
...
p=raiz;
while (p != NULL)
{ if (p->info == valor)
        return  p;
   else
      p=(p->info > valor? p->izq: p->der);
}
return NULL;
...
```

P contiene la dirección del nodo que tiene el valor buscado

No se encontró el valor por lo que se regresa un NULL

Equivalente a:

```
if ( p -> info > valor )
     p = p -> izq;
else p = p-> der;
```

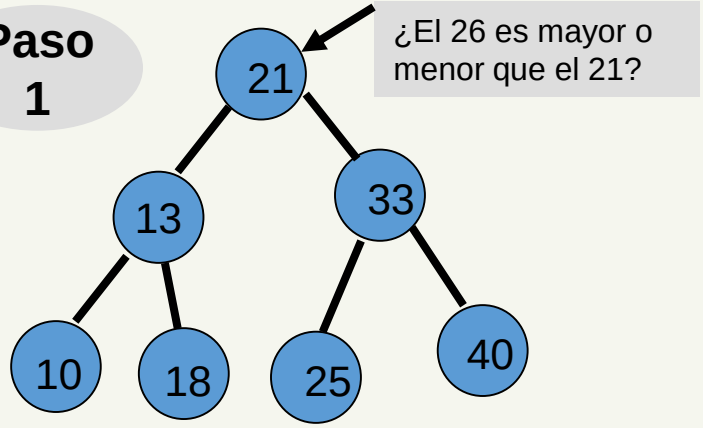
Proceso para agregar nodos...

- Reglas:
  - El valor a insertar no existe en el árbol.
  - El nuevo nodo será un Nodo Hoja del árbol.
- Procedimiento
  1. Buscar el Nodo Padre del nodo a agregar.
  2. Agregar el nodo hoja.

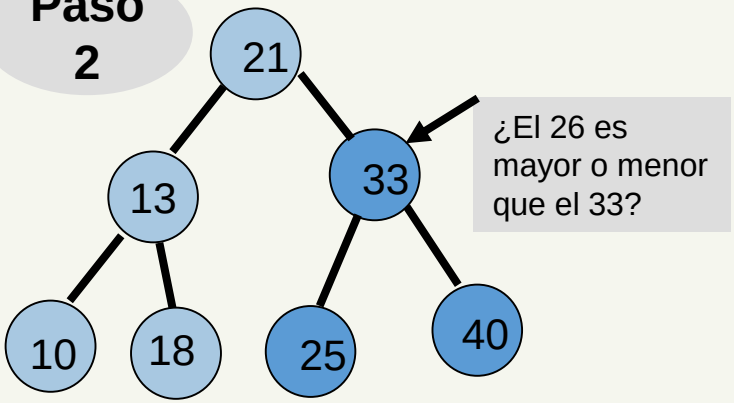
Ejemplo

Agregar el valor 26

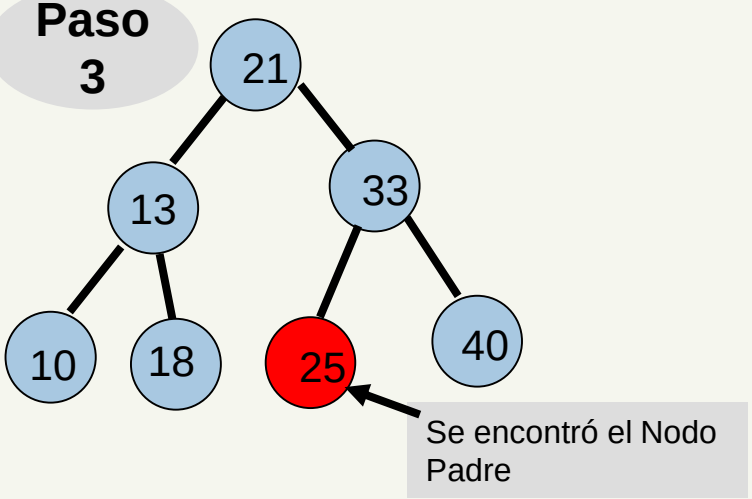
Paso 1



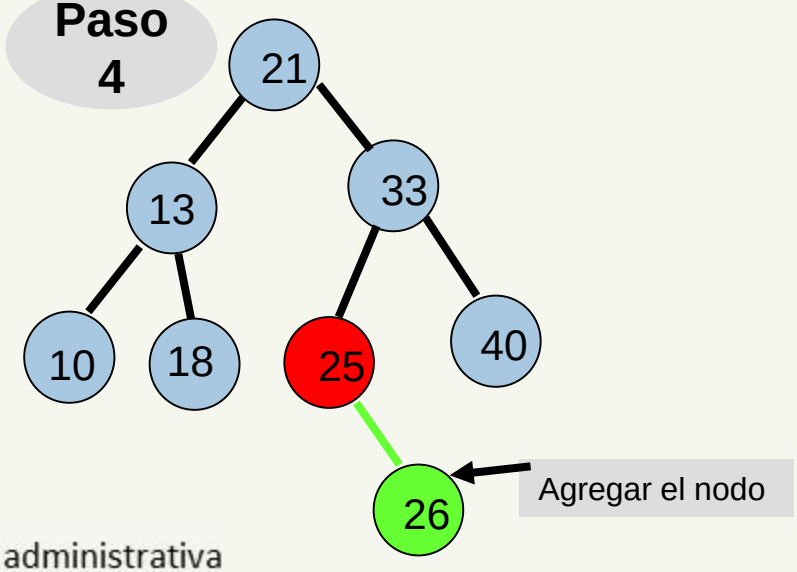
Paso 2



Paso 3

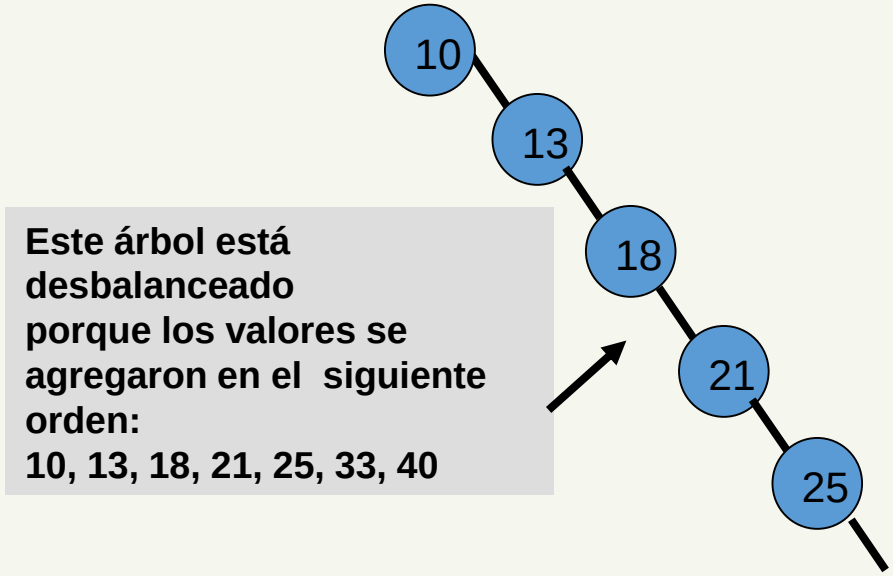


Paso 4



Comentarios importantes....

- El orden de inserción de los datos, determina la forma del ABB.
- ¿Qué pasará si se insertan los datos en forma ordenada?
- La forma del ABB determina la eficiencia del proceso de búsqueda.
- Entre menos altura tenga el ABB, más balanceado estará, y más eficiente será.



Este árbol está desbalanceado porque los valores se agregaron en el siguiente orden:
10, 13, 18, 21, 25, 33, 40

Implementación....

```
bool ABB::Insertar(int valor)
{
    NodoArbol *nuevo, *actual, *anterior;
    nuevo = new NodoArbol(valor);
    actual = raiz;
    anterior = NULL;
    while ( actual != NULL )
    {
        if ( valor == actual -> info ) return 0;
        anterior = actual;
        actual = (actual->info > valor ? actual->izq : actual->der);
    }
    if(anterior==NULL)
        raiz=nuevo;
    else {
            if ( anterior -> info > valor )
                anterior -> izq = nuevo;
            else
                anterior -> der = nuevo;
    }
    return 1;
}
```

Busca el Nodo Padre.
Al final, Anterior será el padre del nuevo nodo.

Agrega el nodo como nodo hoja.
Si Anterior es igual a NULL quiere decir que el árbol está vacío por lo que el nodo a agregar será la raíz.

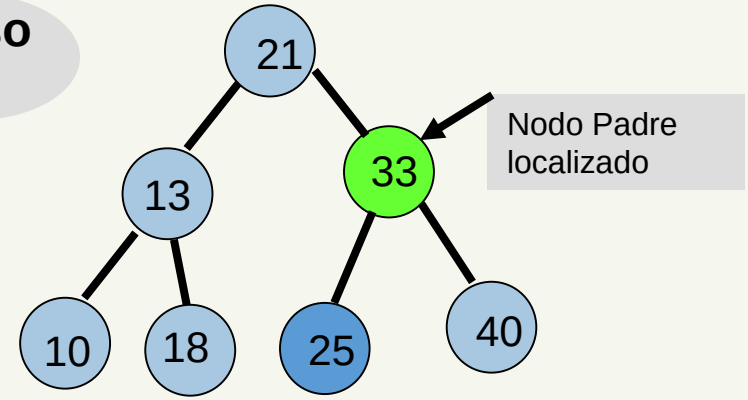
Proceso para eliminar un nodo

- Si el nodo a eliminar es un:
  - Nodo hoja
    - Buscar el Nodo Padre del nodo a borrar.
    - Desconectarlo.
    - Liberar el nodo.
  - Nodo con un hijo
    - Buscar el Nodo Padre del nodo a borrar.
    - Conectar el hijo con el padre del nodo a borrar.
    - Liberar el nodo.
  - Nodo con dos hijos
    - Localizar el nodo predecesor o sucesor del nodo a borrar.
    - Copiar la información.
    - Eliminar el predecesor o sucesor según sea el caso.

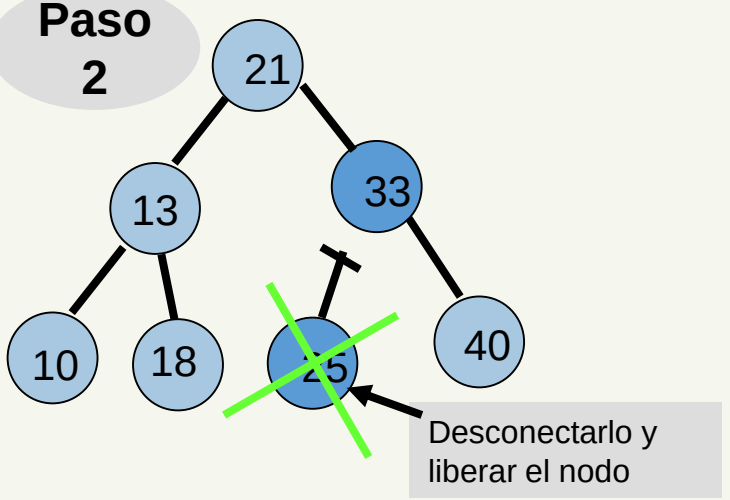
Caso: Eliminar Nodo hoja

Eliminar el valor 25

Paso 1



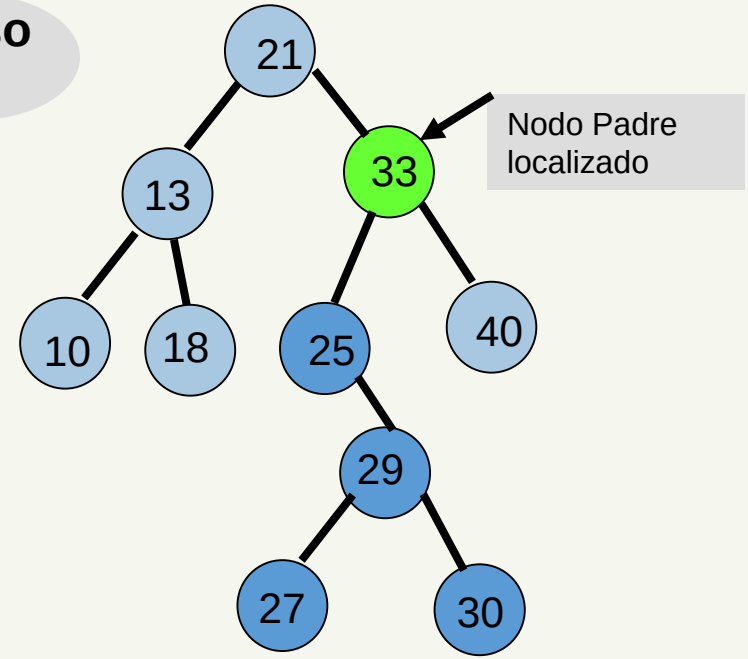
Paso 2



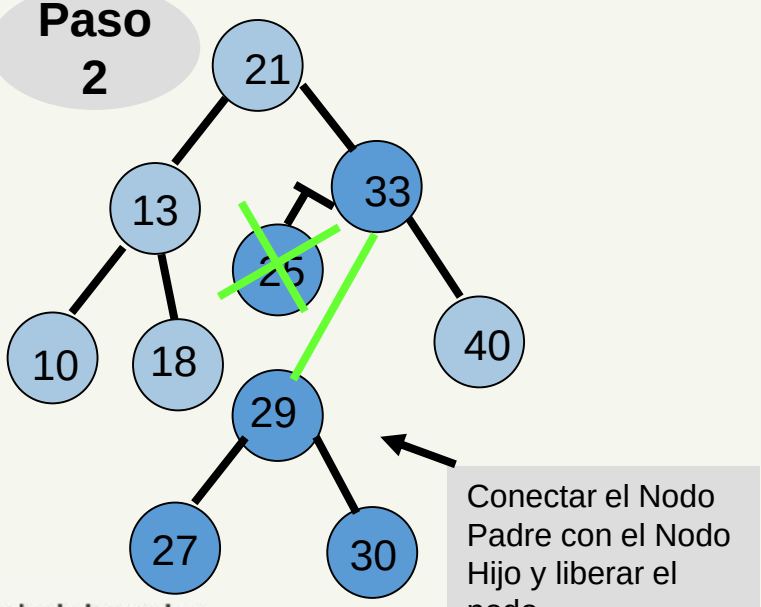
Caso: Eliminar Nodo con un hijo

Eliminar el valor 25

Paso 1



Paso 2

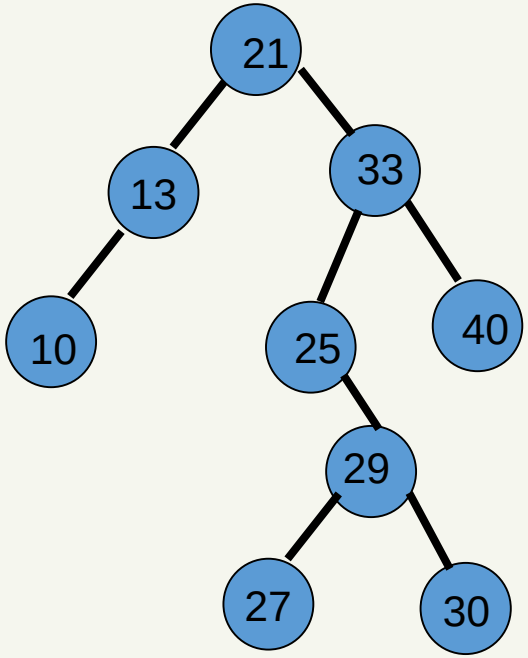


Caso: Eliminar nodo con dos hijos

1. Localizar el nodo predecesor o sucesor del nodo a borrar.
   - El PREDECESOR es “el Mayor de los Menores”.
   - El SUCESOR es “el Menor de los Mayores”.
   - Para la implementación es igual de eficiente programar la búsqueda del predecesor que del sucesor.
2. El valor del Predecedor (o sucesor) se copia al nodo a borrar.
3. Eliminar el nodo del predecesor (o sucesor según sea el caso).

Predecesor

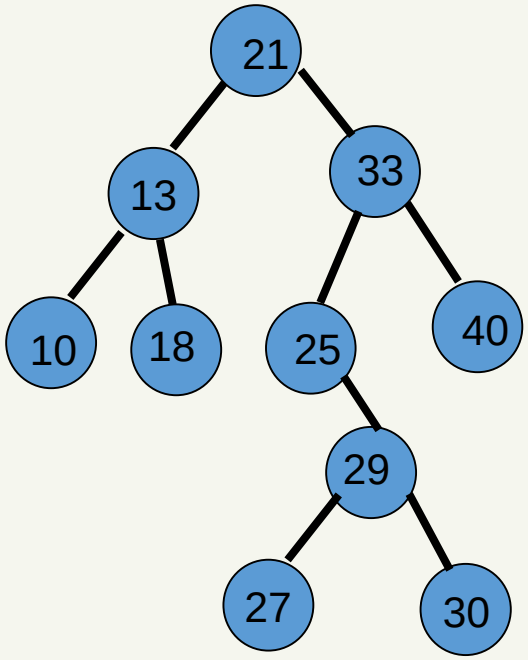
Uno a la IZQUIERDA y todo a la DERECHA



El predecesor de:	Es:
33	30
21	13
29	27

Sucesor

Uno a la DERECHA y todo a la IZQUIERDA



El sucesor de:	Es:
21	25
33	40
29	30

Implementación del….

PREDECESOR

```
P = actual -> izq;
while( p -> der != NULL)
   p=p->der;
return  p;
```

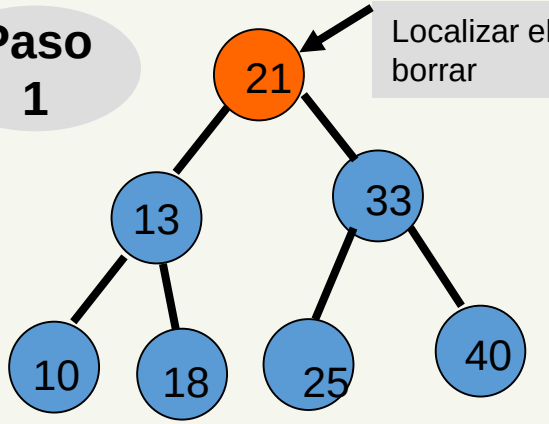
actual apunta al nodo a borrar

SUCESOR

```
P = actual -> der;
While (p -> izq != NULL )
   p=p->izq;
return p;
```

Caso: Eliminar Nodo con dos hijos

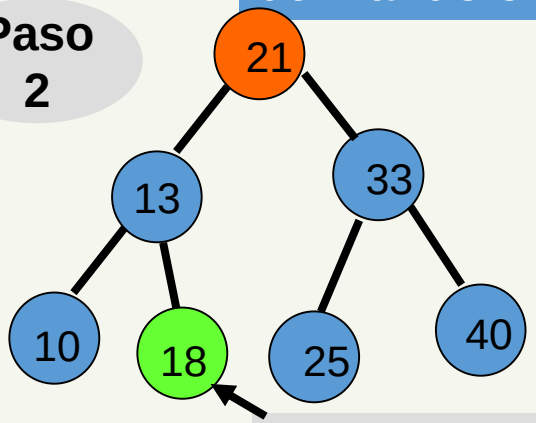
Paso 1



Localizar el valor a borrar

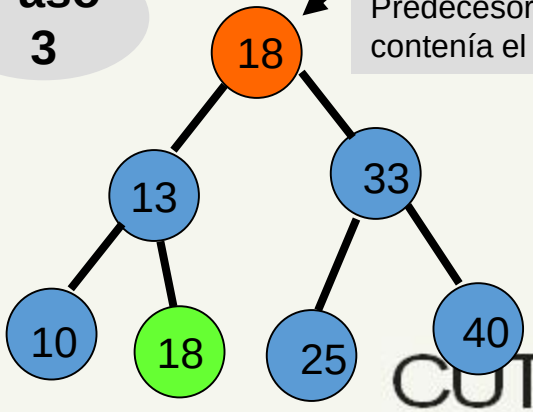
Eliminar el valor 21 utilizando el predecesor

Paso 2



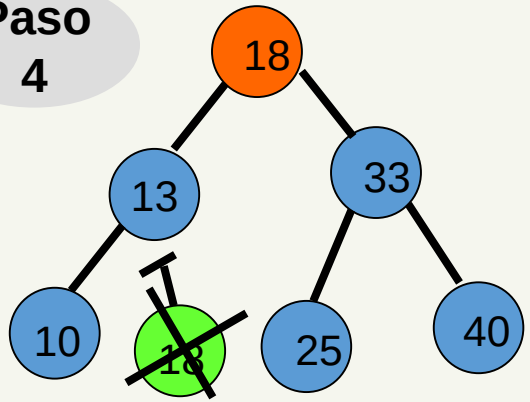
Localizar el Predecesor

Paso 3



Copiar el valor del Predecesor al nodo que contenía el valor a borrar

Paso 4

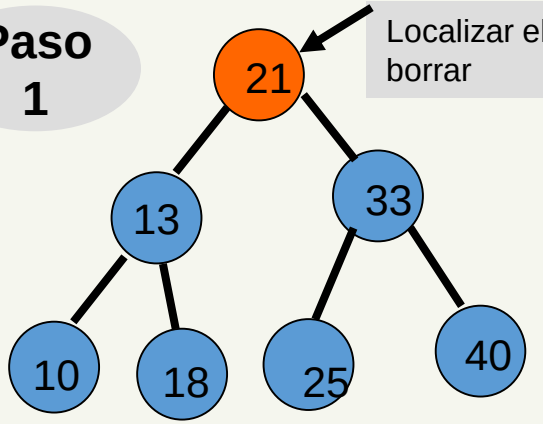


Desconectar y liberar el nodo del Predecesor

Caso: Eliminar Nodo con dos hijos

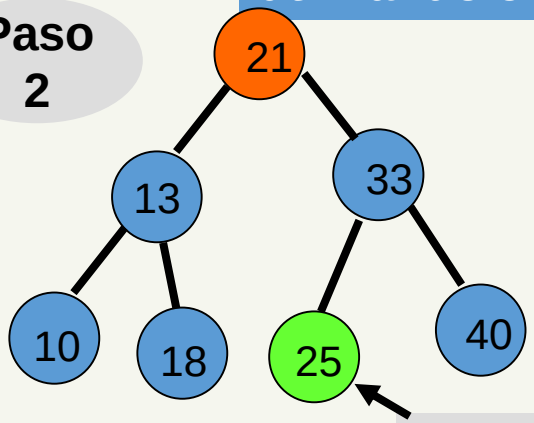
Eliminar el valor 21 utilizando el Sucesor

Paso 1



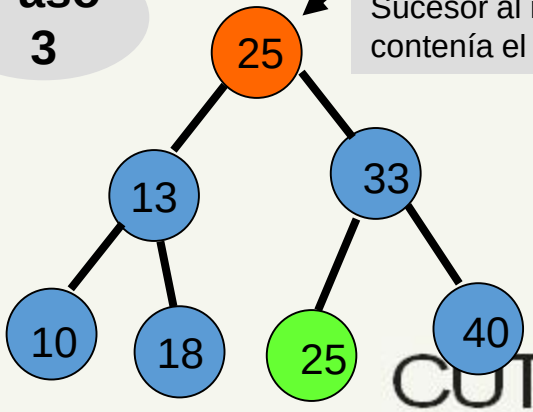
Localizar el valor a borrar

Paso 2



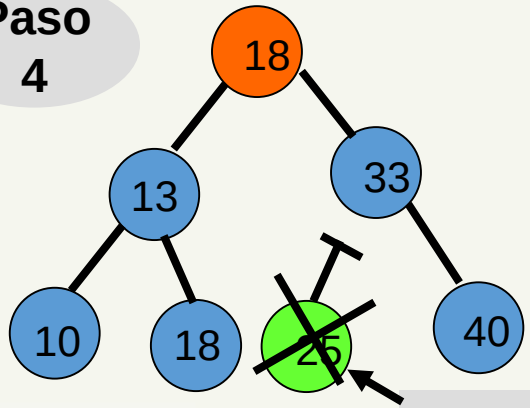
Localizar el Sucesor

Paso 3



Copiar el valor del Sucesor al nodo que contenía el valor a borrar

Paso 4



Desconectar y liberar el nodo del Sucesor

Bibliografía

Cairo O. y Guardati S., 2003. Estructura de datos, 3ra Edición Mc Graw Hill

Allen Weiss Mark , 2013. Estructura de datos en Java 4ta Edición PEARSON

MARTI O.N., VERDEJO A. Y ORTEGA M. 2013 ESTRUCTURAS DE DATOS Y METODOS ALGORITMICOS 2DA EDICIÓN , GARCETA GRUPO EDITORIAL, 2013