Arboles Binarios

Estructuras de Datos

1

Las estructuras dinámicas son las que en la ejecución varia el número de elementos y uso de memoria a lo largo del programa)

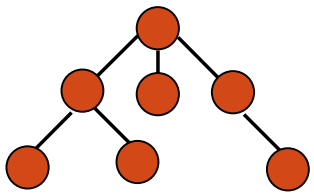
Entre estas tenemos:

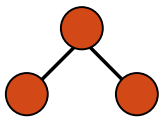
Lineales (listas enlazadas, pilas y colas)
No lineales (árboles binarios y grafos o redes)

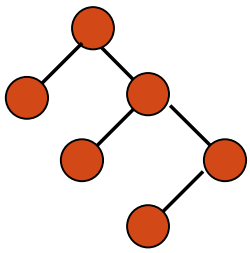
2

¿Qué es un Árbol?

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



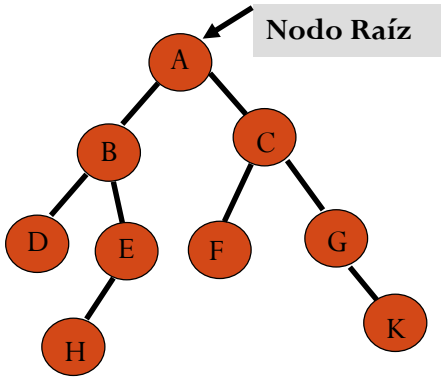




3

Terminología

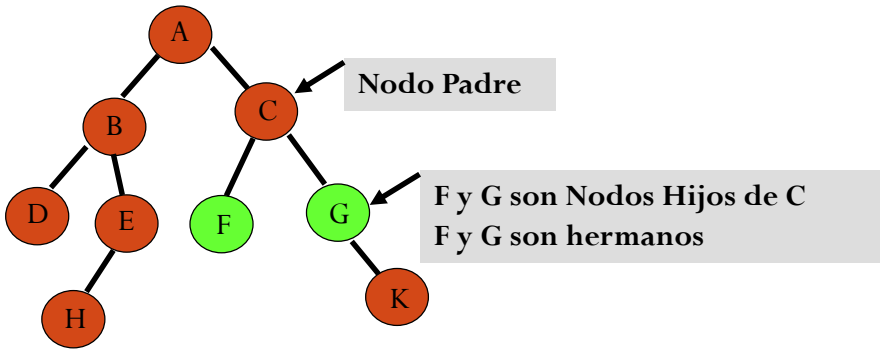
- Nodo: Cada elemento en un árbol.
- Nodo Raíz: Primer elemento agregado al árbol.



4

Más terminología

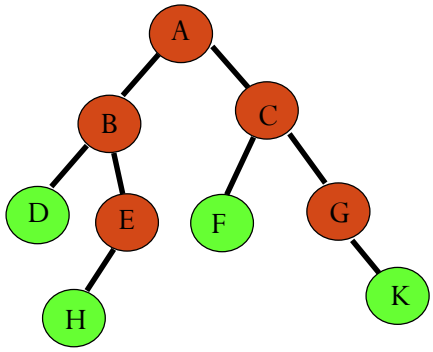
- Nodo Padre: Se le llama así al nodo predecesor de un elemento.
- Nodo Hijo: Es el nodo sucesor de un elemento.
- Hermanos: Nodos que tienen el mismo nodo padre.



5

Más terminología

- Nodo Hoja: Aquel nodo que no tiene hijos.

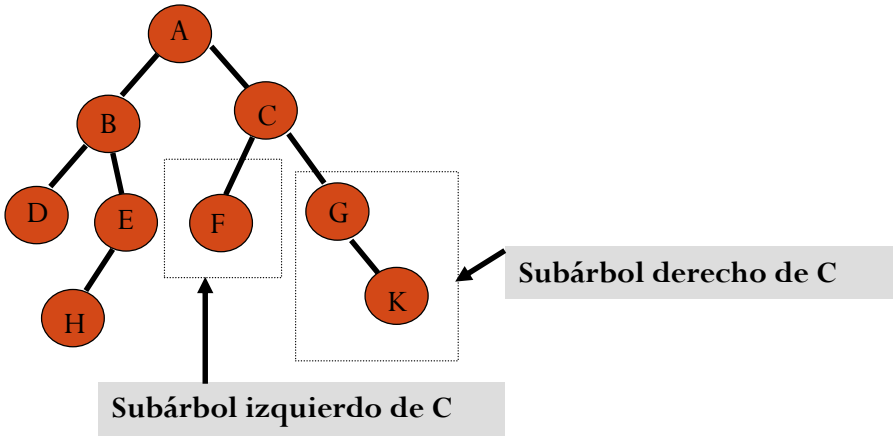


D, H, F y K son Nodos Hojas

6

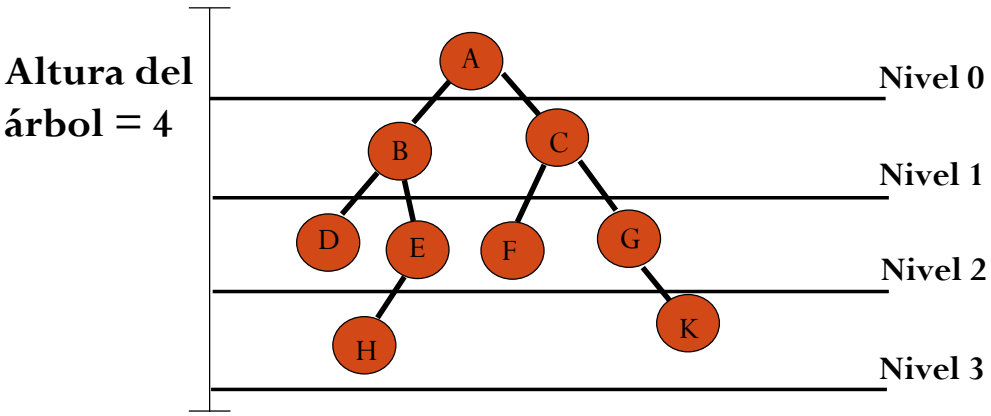
Más terminología

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



7

Altura y Niveles



La Altura es la cantidad de niveles.

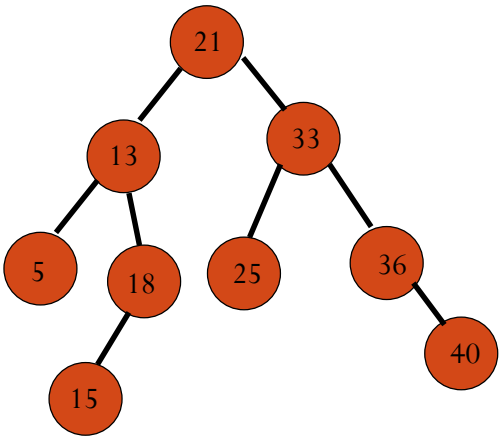
8

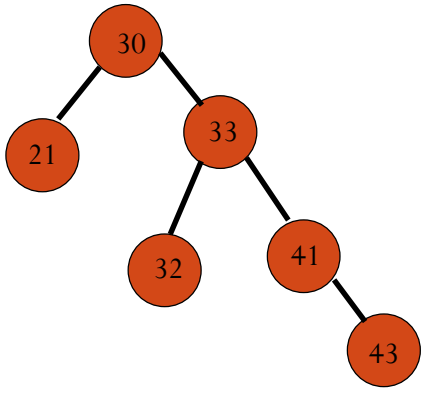
Árbol Binario de Búsqueda (ABB)

- 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.

9

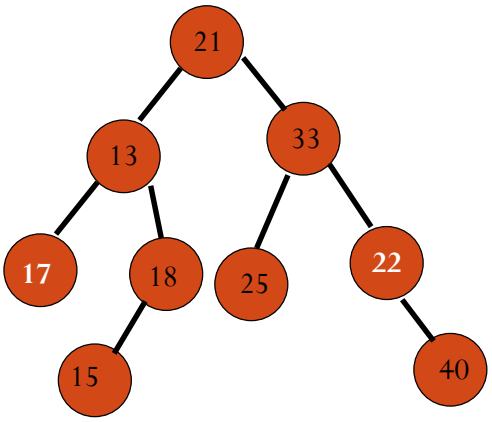
Ejemplos de ABB...

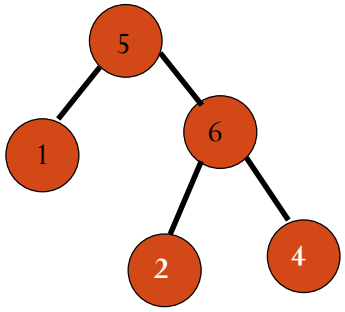




10

¿Por qué no son ABB?





11

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; }
```

12

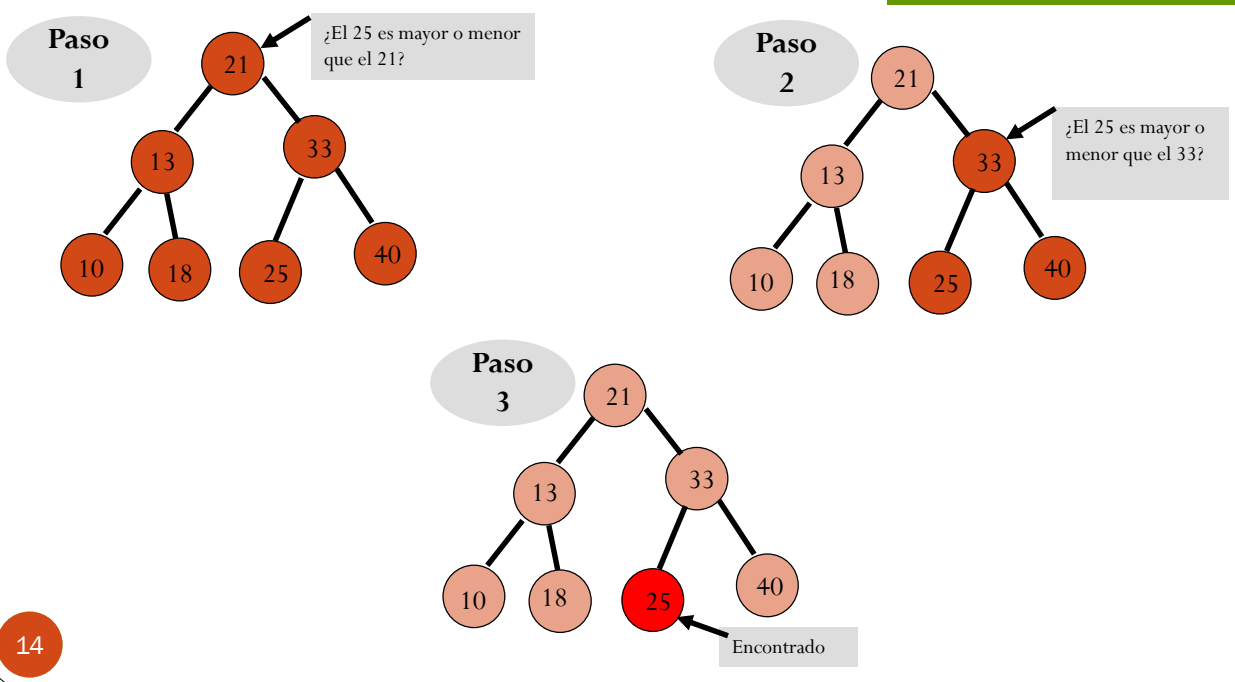
Continuación…

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

13

Proceso para buscar un nodo...

Buscar el 25



14

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;
```

15

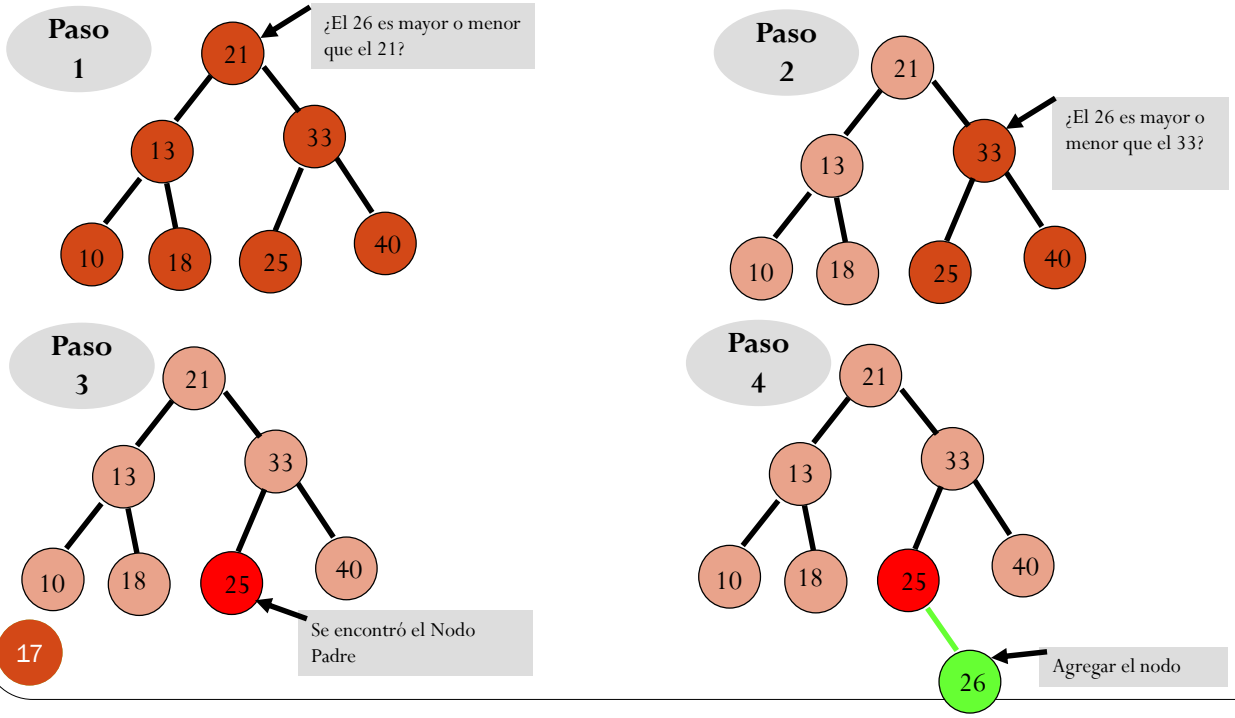
Proceso para agregar nodos...

- 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.

16

Ejemplo

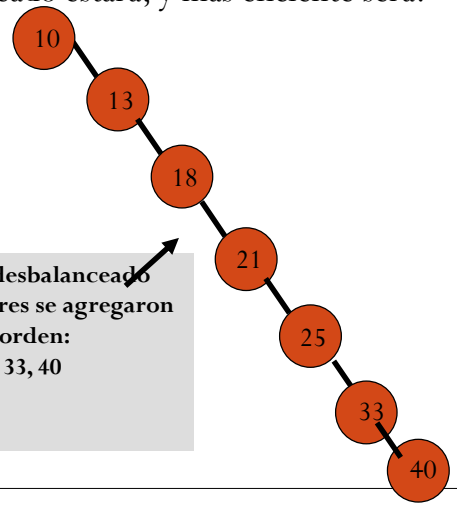
Agregar el valor 26



17

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

18

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.

19

Proceso para eliminar un nodo

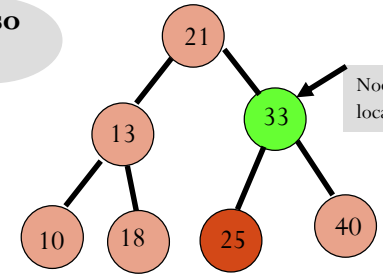
- Si el nodo a eliminar es un:
  - Nodo hoja
    - Buscar el Nodo Padre del nodo a borrar.
    - Desconectarlo.
    - Liberar el nodo.
  - Nodo con un hijo
    - Buscar el Nodo Padre del nodo a borrar.
    - Conectar el hijo con el padre del nodo a borrar.
    - Liberar el nodo.
  - Nodo con dos hijos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

20

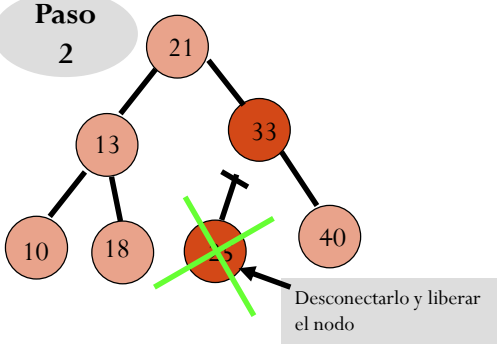
Caso: Eliminar Nodo hoja

Eliminar el valor 25

Paso 1



Paso 2

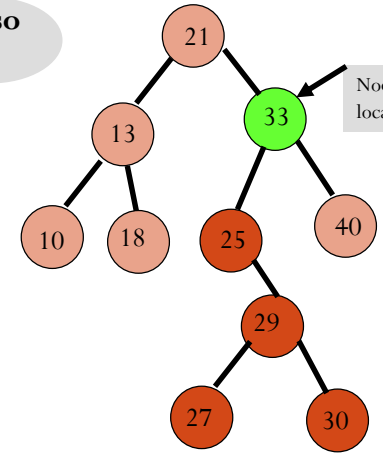


21

Caso: Eliminar Nodo con un hijo

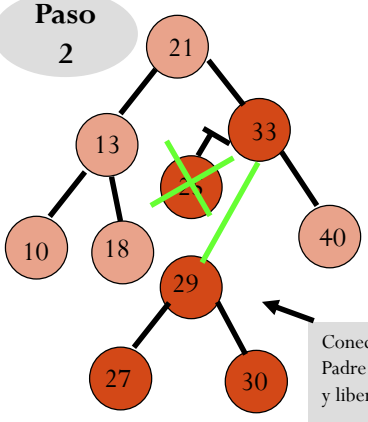
Eliminar el valor 25

Paso 1



Nodo Padre localizado

Paso 2



Conectar el Nodo Padre con el Nodo Hijo y liberar el nodo.

22

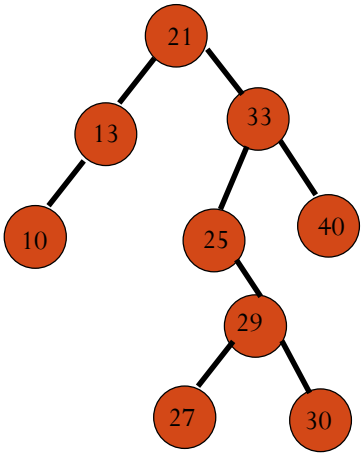
Caso: Eliminar nodo con dos hijos

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).

23

Predecesor

Uno a la IZQUIERDA y todo a la DERECHA

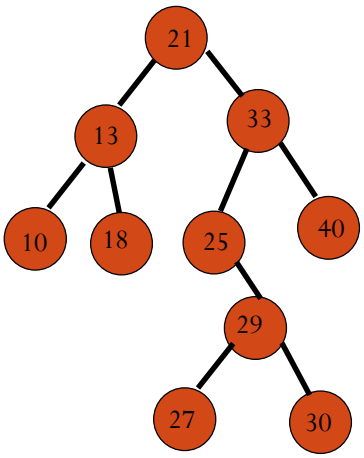


El predecesor de:	Es:
33	30
21	13
29	27

24

Sucesor

Uno a la DERECHA y todo a la IZQUIERDA



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

25

Implementación del....

PREDECESOR

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

actual apunta al nodo a borrar

SUCESOR

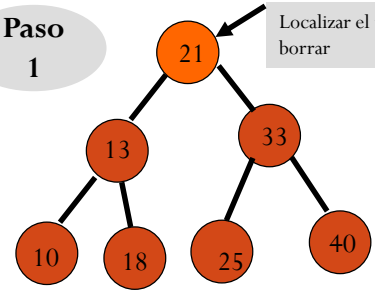
```
P = actual -> der;
While (p -> izq != NULL )
  p=p->izq;
return p;
```

26

Caso: Eliminar Nodo con dos hijos

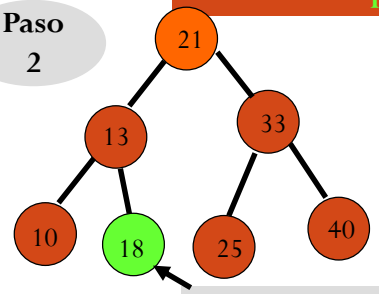
Eliminar el valor 21 utilizando el predecesor

Paso 1



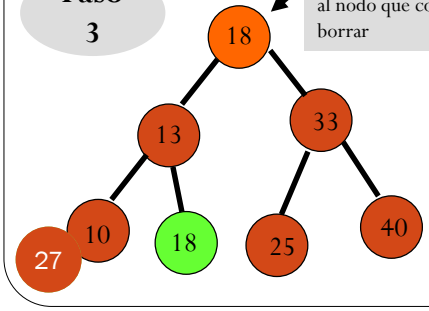
Localizar el valor a borrar

Paso 2



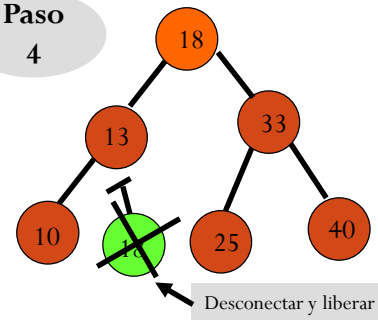
Localizar el Predecesor

Paso 3



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

Paso 4

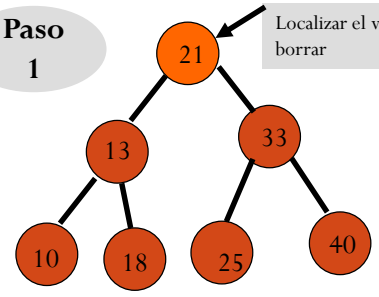


Desconectar y liberar el nodo del Predecesor

Caso: Eliminar Nodo con dos hijos

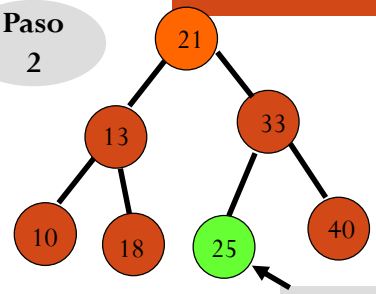
Eliminar el valor 21 utilizando el Sucesor

Paso 1



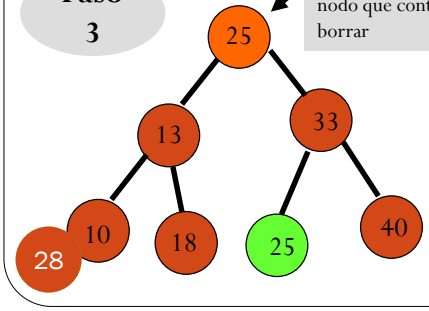
Localizar el valor a borrar

Paso 2



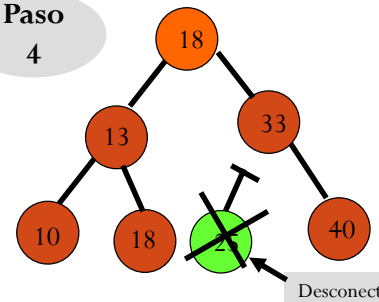
Localizar el Sucesor

Paso 3



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

Paso 4



Desconectar y liberar el nodo del Sucesor