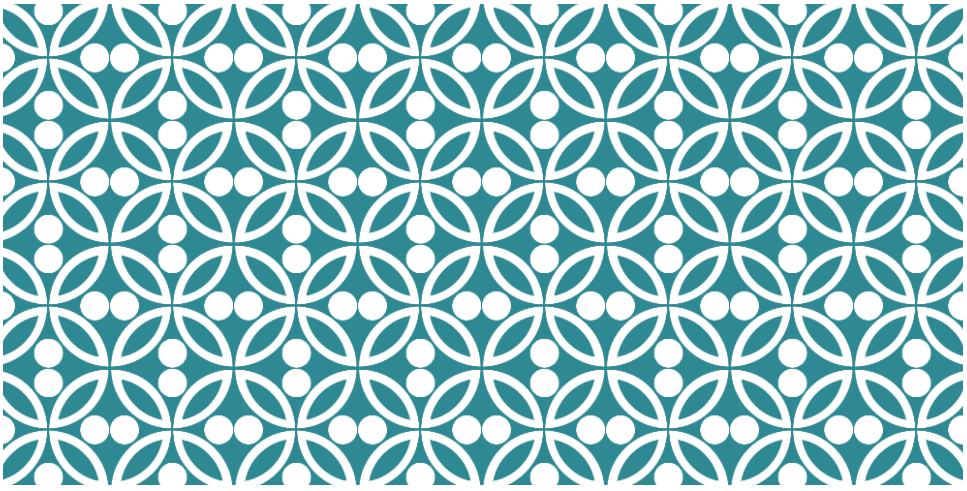
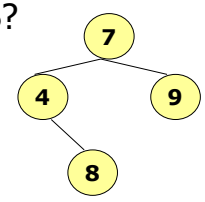


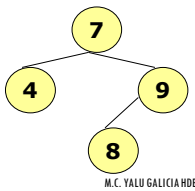
PARTE II: ÁRBOLES BINARIOS DE BÚSQUEDA

- Árboles binarios equilibrados
- Árboles AVL

ÁRBOLES BINARIOS DE BÚSQUEDA (ABB)

- ABB = Árbol binario ordenado según uno o más criterios
- Cada nodo tiene dos hijos:
  - el subárbol **izquierdo** es el árbol vacío o es un subárbol que contiene nodos cuya clave es menor que la suya
  - el subárbol **derecho** es el árbol vacío o es un subárbol que contiene nodos cuya clave es mayor que la suya
- ¿Cuál de estos dos árboles binarios de enteros es un ABB?





ÁRBOLES BINARIOS DE BÚSQUEDA (ABB)

- Utilidad
  - Almacenar estructuras lineales (que normalmente serían listas) mejorando la complejidad de las búsquedas
    - En el caso peor
    - En el caso medio
- Motivación: AB no ordenados son de poco interés
  - La falta de ordenación en un AB hace injustificable una estructura enlazada de árbol, prefiriéndose una lista
- Problema
  - un ABB puede llegar a degenerar en una lista
- Solución: equilibrio en ABB

ABB: DESCRIPCIÓN

- El nodo se compone de clave, información y referencias a los subárboles izquierdo y derecho
  - <arbol>::= <<nulo>> | <nodo>
  - <nodo>::= <clave> <info> <izq> <der>
  - <clave>::= <<dato>> {<<dato>>}
  - <info>::={<<dato>>}
  - <izq>::=<arbol>
  - <der>::=<arbol>
- El proceso de inserción es el encargado de garantizar que se cumple la condición de ABB

ABB: OPERACIONES

Creación de un árbol	`crearArbol (nombreArbol)`
Comprobación del estado	`arbolVacio(nombreArbol) → Booleano` `arbolVacio(referenciaNodo) → Booleano`
Inserción de nodos	`insertar(nombreArbol,valorClave,valorInfo)`
Borrado de nodos	`borrar(nombreArbol, valorClave)`
Búsqueda de un nodo	`buscar(nombreArbol, valorClave)→referenciaNodo`
Recorrido del árbol	`recorrer(nombreArbol,tipoRecorrido)`

Acceso a los nodos	`clave(referenciaNodo) → Clave` `info(referenciaNodo) → Informacion` `izq(referenciaNodo) → enlace` `der(referenciaNodo) → enlace` *`eshoja(referenciaNodo) → Booleano`*
Modificación de los nodos	`asignarClaver(referenciaNodo, valorClave)` `asignarInfo(referenciaNodo, valorInformacion)` `asignarIzq(referenciaNodo, valorEnlace)` `asignarDer(referenciaNodo, valorEnlace`

5

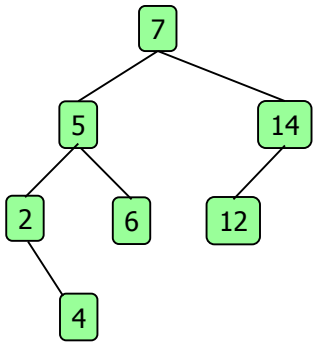
ABB: INSERCIÓN

- Los nodos se insertan siempre como nodos hoja
- El algoritmo de inserción garantiza para cada nodo del árbol que:
  - Su subárbol izquierdo contiene claves menores
  - Su subárbol derecho contiene claves mayores
- Funcionamiento:
  - Si el árbol está vacío, se inserta el nodo en la raíz
  - si no, se va recorriendo el árbol:
    - En cada nodo se decide si hay que insertar a la derecha o la izquierda
    - Si el subárbol en el que hay que insertar es vacío, se inserta el nuevo elemento
    - Si el subárbol en que hay que insertar no es vacío hay que recorrerlo hasta encontrar el lugar que le corresponde al nodo en ese subárbol
  - Es un algoritmo recursivo

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 6

TAD ABB: EJEMPLO DE INSERCIÓN

- Insertar 7, 5, 2, 14, 12, 6, 4
  - Insertar 7
  - Insertar 5
  - Insertar 2
  - Insertar 14
  - Insertar 12
  - Insertar 6
  - Insertar 4



ACTIVIDAD INDIVIDUAL

- Insertar en un ABB los elementos:
  - 1, 4, 5, 6, 8, 12, 20
- Insertar los elementos en un ABB:
  - 8, 4, 2,1,3,12,10, 9, 11, 6, 7, 5, 14, 15, 13

¿ y el algoritmo?

ABB: BÚSQUEDA

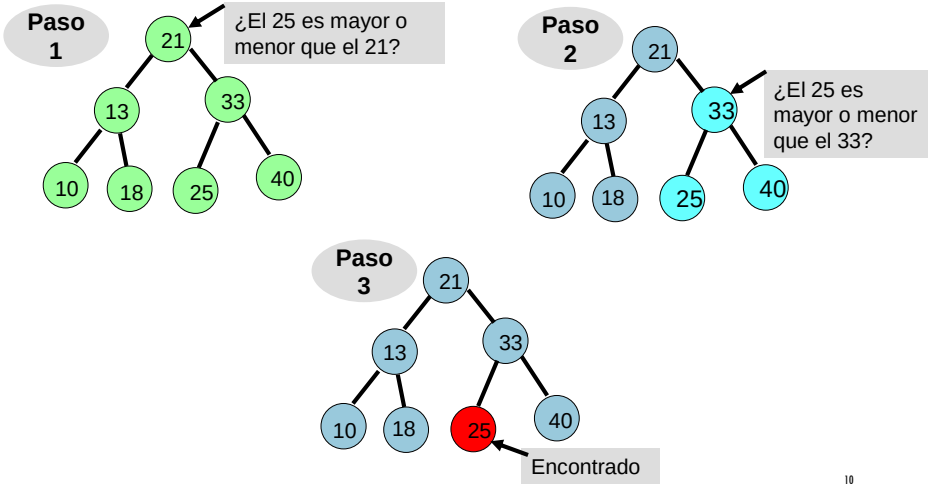
- Funcionamiento:
  - se va recorriendo el árbol
  - si el nodo actual no es el buscado se decide si hay que buscar a la derecha o la izquierda
  - el algoritmo termina al encontrar el nodo o llegar al árbol vacío
- Puede desarrollarse:
  - como algoritmo recursivo del nodo del árbol
  - como algoritmo iterativo del árbol

¿ y el algoritmo?

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 9

PROCESO PARA BUSCAR UN NODO...

Buscar el 25



10

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

Equivalente a:
if (p. info > valor)
    p = p.izq;
else p = p.der;

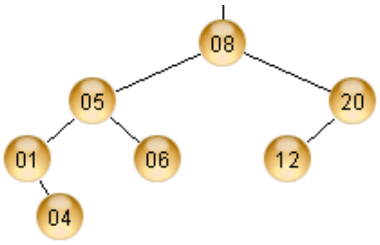
No se encontró el valor por lo que se regresa un NULL

ABB: BORRADO

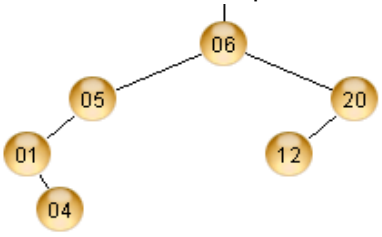
- Funcionamiento:
  - Buscar el nodo a borrar
  - Si es un nodo hoja, basta con que su padre haga referencia al árbol vacío
  - Si no es nodo hoja hay que sustituirlo por otro
    - 1. El nodo a borrar sólo tiene un hijo: sustituirlo por su hijo
    - 2. El nodo a borrar tiene dos hijos, sustituirlo por:
      - El mayor de su subárbol izquierdo o
      - El menor de su subárbol derecho
  - Si el nodo a borrar es la raíz, hay que variarla aplicando el caso que corresponda

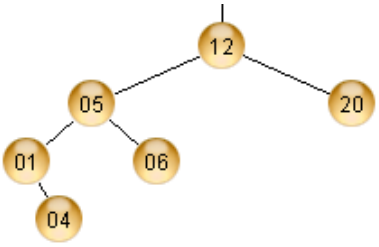
ABB: EJEMPLO DE BORRADO

- Borrar 8



Sustituir por 6 ó Sustituir por 12

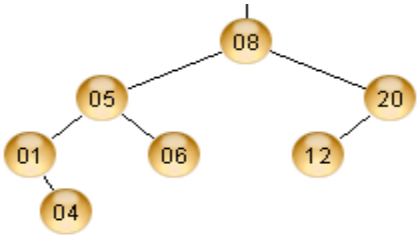




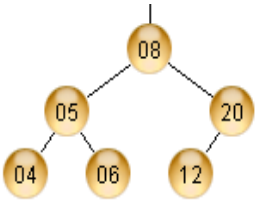
13

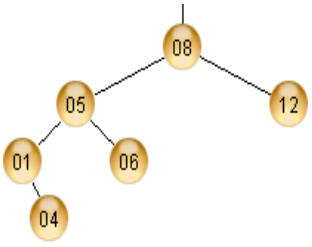
ABB: EJEMPLO DE BORRADO

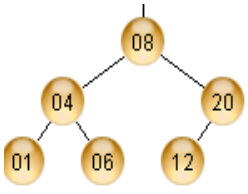
- Borrar 1
- Borrar 20
- Borrar 5



Borrar 1 Borrar 20 Borrar 5



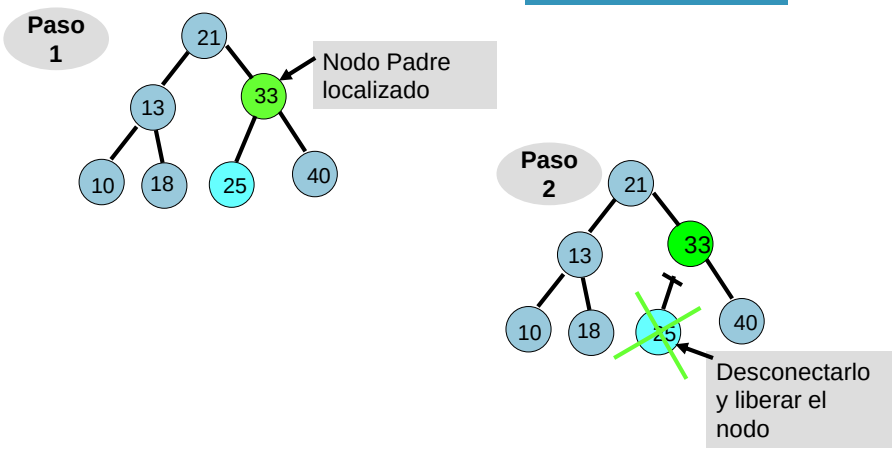




14

CASO: ELIMINAR NODO HOJA

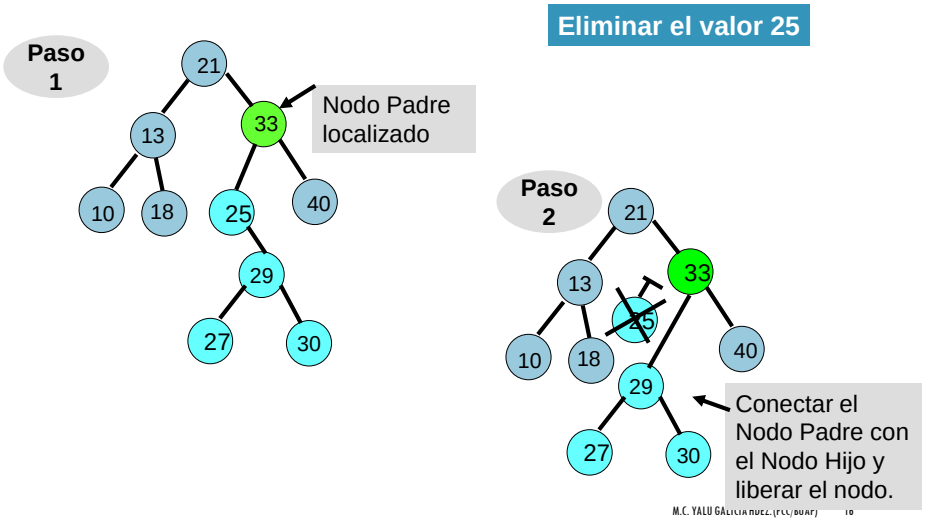
Eliminar el valor 25



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 15

CASO: ELIMINAR NODO CON UN HIJO

Eliminar el valor 25



M.C. YALU GALICIA HDEZ. (FCC/BUAP) 16

ACTIVIDAD INDIVIDUAL

- Utilizando los ABB creados en la actividad anterior, borrar los siguientes elementos:
  - Primer ABB:
    - 20, 5, 1
  - Segundo ABB
    - 7, 4, 10, 14, 8

¿ y el algoritmo?

M.C. YALU GALICIA HDEZ. (FCC/BUAP) 17

QUÉ HEMOS APRENDIDO?