



# Taller de Programación



# ÁRBOLES BINARIOS DE BÚSQUEDA- Características



Un **árbol binario de búsqueda** es una estructura de datos jerárquica. Está formada por nodos, donde cada nodo tiene a lo sumo dos hijos y mantienen un orden. El nodo principal del árbol se denomina raíz y los nodos que no tienen hijos se denominan hojas del árbol.

Características: **homogénea – dinámica – no lineal - acceso secuencial**

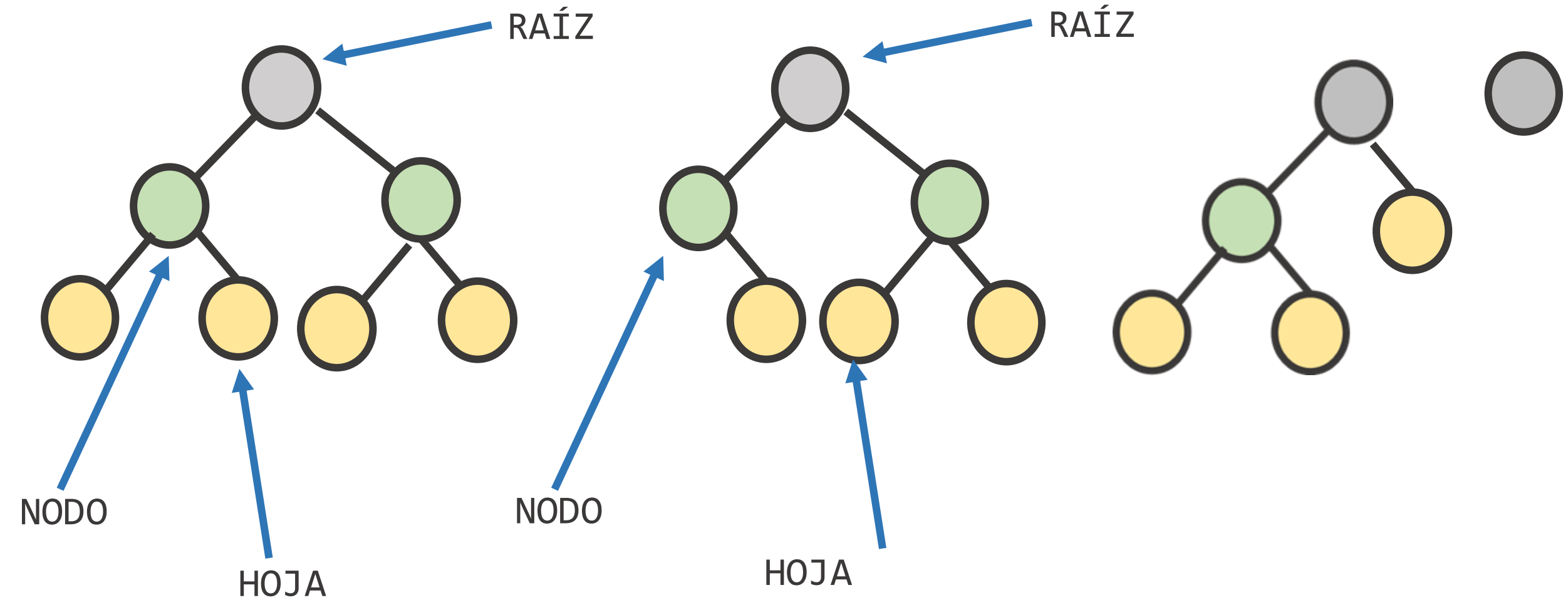
Operaciones: **crear – imprimir – buscar – mínimo – máximo - eliminar**

Cómo se ven  
gráficamente?

Cómo se  
declaran?

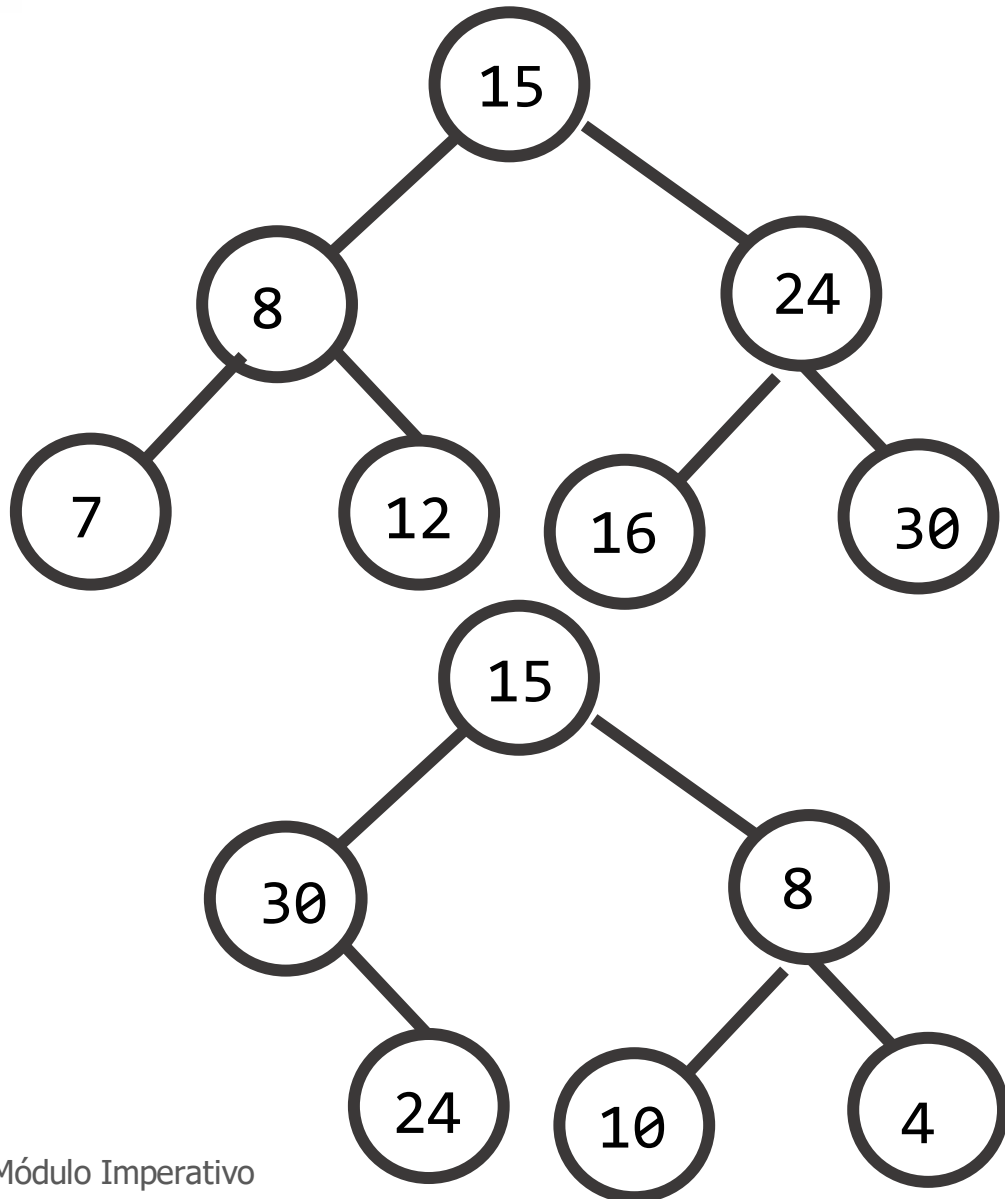


# ÁRBOLES BINARIOS DE BÚSQUEDA- Gráficamente





# ÁRBOLES BINARIOS DE BÚSQUEDA- Declaración



Programa arboles;

Type

```
arbol = ^nodo;
tipo = ...;
nodo = record
    dato: tipo;
    HI: arbol;
    HD: arbol;
end;
```

Var

**a:arbol;**

Begin

...

End.



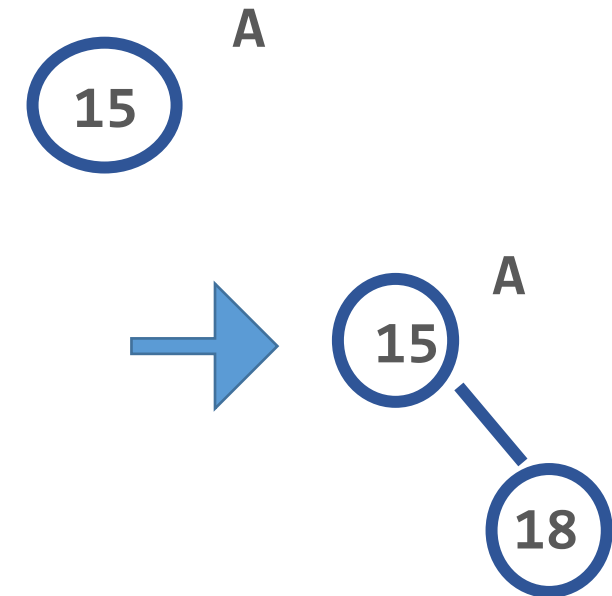
# ÁRBOLES BINARIOS DE BÚSQUEDA- Creación



Suponga que se leen los siguientes valores y se quiere crear un ABB (15, 18, 22, 16, 7) **Cómo cree que quedará el árbol?**

**15** Como el árbol es vacío, se genera un nodo nuevo y el valor 15 ocupará la raíz del árbol.

**18** Como el árbol NO es vacío, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja. Siempre debo generar un espacio para el nuevo dato.





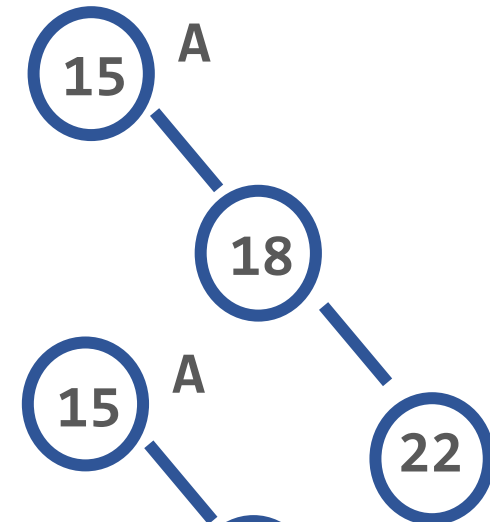
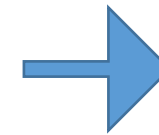
# ÁRBOLES BINARIOS DE BÚSQUEDA- Creación



Suponga que se leen los siguientes valores y se quiere crear un ABB (15, 18, 22, 16, 7) **Cómo cree que quedará el árbol?**

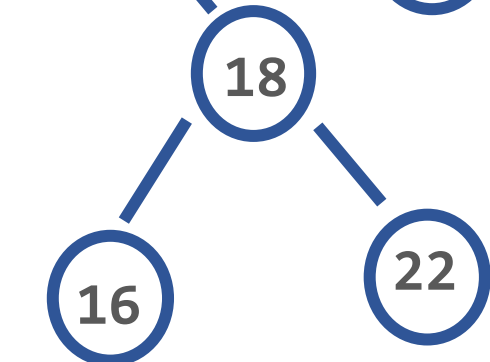
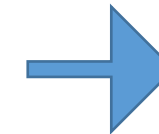
22

Como el árbol NO es vacío, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja. Siempre debo generar un espacio para el nuevo dato.



16

Como el árbol NO es vacío, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja. Siempre debo generar un espacio para el nuevo dato.





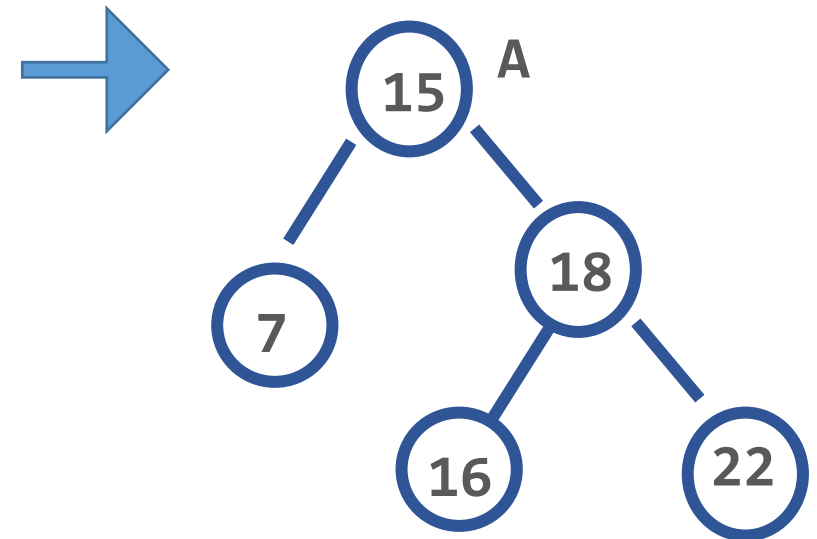
# ÁRBOLES BINARIOS DE BÚSQUEDA- Creación



Suponga que se leen los siguientes valores y se quiere crear un ABB (15, 18, 22, 16, 7) **Cómo cree que quedará el árbol?**

7

Como el árbol NO es vacío, tengo que recorrer desde la raíz hasta el lugar correspondiente respetando el orden. Siempre se inserta en una hoja. Siempre debo generar un espacio para el nuevo dato.



**Cómo cree que quedará el árbol si ahora se lee el 10, 15?**

**Cómo se implementa?**



# ÁRBOLES BINARIOS DE BÚSQUEDA- Creación

Programa arboles;

Type

arbol = ^nodo;

nodo = record

dato: integer;

HI: arbol;

HD: arbol;

end;

Var

abb:arbol; x:integer;

Begin

abb:=nil;

read (x);

while (x<>0)do

begin

crear(abb,x);

read(x);

end;

End.

Procedure crear (**var** A:árbol; num:integer);

Begin

if (A = nil) then

begin

new(A);

A^.dato:= num; A^.HI:= nil; A^.HD:= nil;

end

else

if (num < A^.dato) then **crear**(A^.HI,num)

else **crear**(A^.HD,num)

End;

Cómo funciona?





# ÁRBOLES BINARIOS DE BÚSQUEDA- Creación


Supongamos que recibe 15 7

```
Procedure crear (var A:arbol; num:integer);
Begin
  if (A = nil) then
    begin
      new(A);
      A^.dato:= num; A^.HI:= nil; A^.HD:= nil;
    end
  else
    if (num < A^.dato) then crear(A^.HI,num)
    else crear(A^.HD,num)
  End;
```

Cómo se imprime?


a = nil num=15





crear a=nil, num=15,  A

a = 15 num=7



crear a=15, num=7,  A  
Se invoca con el hijo izquierdo

crear a= nil,num=7,   A



# ÁRBOLES BINARIOS DE BÚSQUEDA- Impresión

```
Procedure enOrden ( a : arbol );
```

```
begin
```

```
  if ( a<> nil ) then begin
```

```
    1  enOrden (a^.HI);
```

```
    2  write (a^.dato);
```

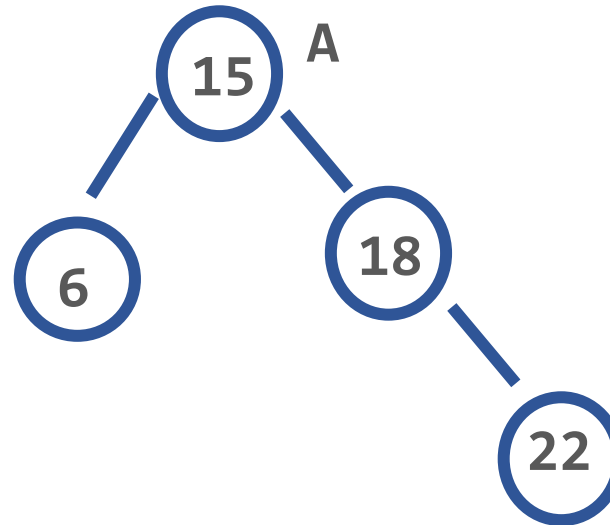
```
    3  enOrden (a^.HD);
```

```
  end;
```

```
end;
```

Cómo son los llamados  
recursivos?

Cuántas instancias  
recursivas se crearon?



a=15	
HI=6	1 2 3
HD=18	

Imprime 15

a=6	
HI=nil	1 2 3
HD=nil	

Imprime 6

a=18	
HI=nil	1 2 3
HD=nil	

Imprime 18

a=nil	No
HI=nil	hace
HD=nil	nada

a=nil	No
HI=nil	hace
HD=nil	nada

a=nil	No
HI=nil	hace
HD=nil	nada

a=22	
HI=nil	1 2 3
HD=nil	

Imprime 22

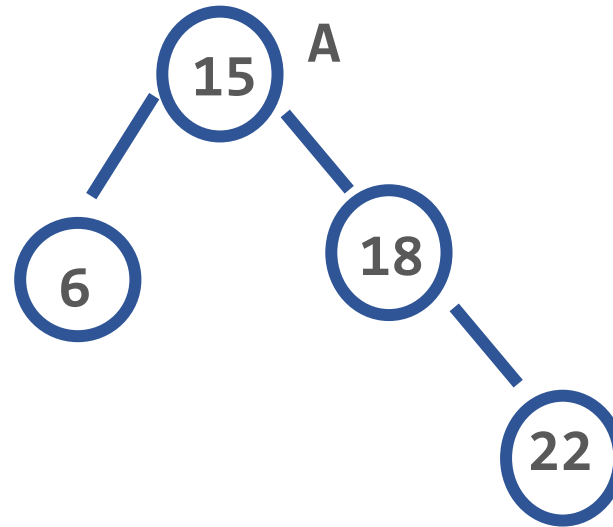
a=nil	No
HI=nil	hace
HD=nil	nada

a=nil	No
HI=nil	hace
HD=nil	nada



# ÁRBOLES BINARIOS DE BÚSQUEDA- Impresión

```
Procedure preOrden ( a : arbol );  
begin  
  if ( a<> nil ) then begin  
    1 write (a^.dato);  
    2 preOrden (a^.HI);  
    3 preOrden (a^.HD);  
  end;  
end;
```



Qué imprime?

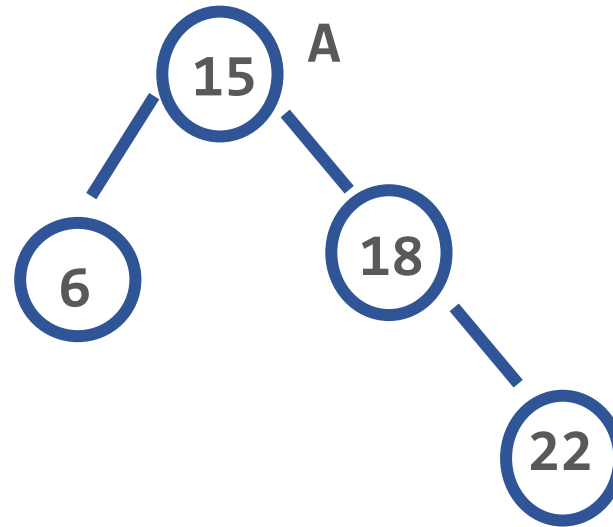
Cómo son los  
llamados recursivos?

Cuántas instancias  
recursivas se  
crearon?



# ÁRBOLES BINARIOS DE BÚSQUEDA- Impresión

```
Procedure posOrden ( a : arbol );  
begin  
  if ( a<> nil ) then begin  
1    posOrden (a^.HI);  
2    posOrden (a^.HD);  
3    write (a^.dato);  
  
  end;  
end;
```



Qué imprime?

Cómo son los  
llamados recursivos?

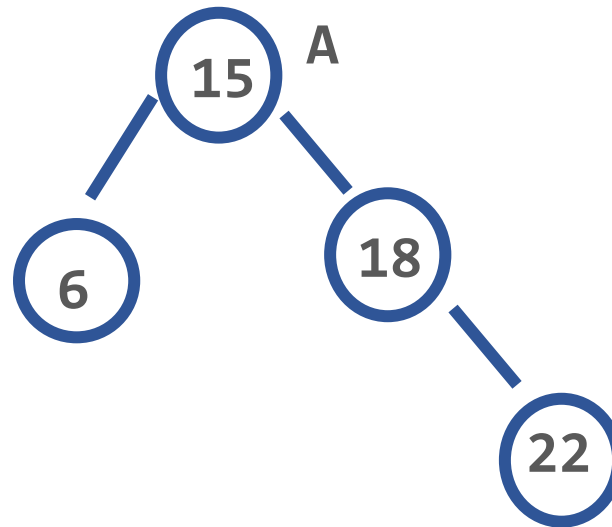
Cuántas instancias  
recursivas se  
crearon?



# ÁRBOLES BINARIOS DE BÚSQUEDA- Búsqueda



Suponga que se tiene un ABB como buscamos un elemento?



Cómo aprovecho el orden del ABB?

Dado un árbol y un valor  $x$ , esta operación retorna un puntero al nodo en el árbol  $A$  que tiene valor  $x$  o Nil si no existe.



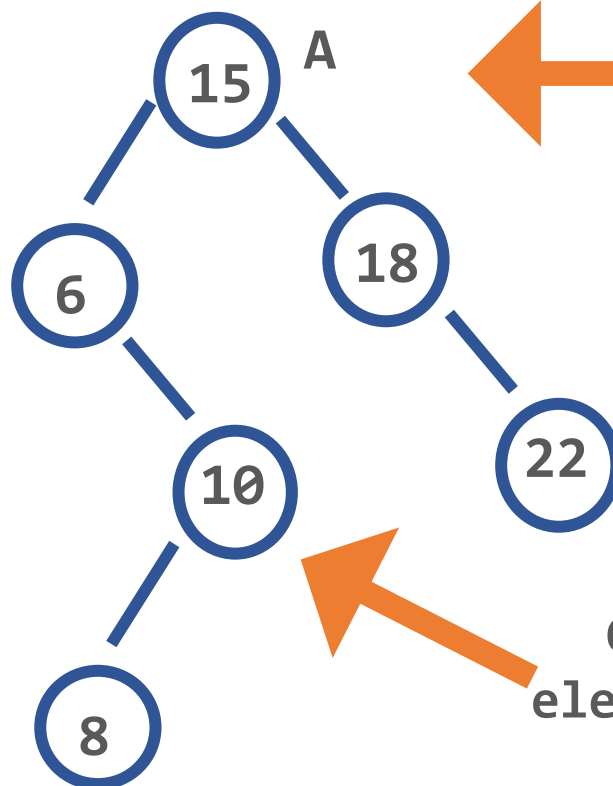
# ÁRBOLES BINARIOS DE BÚSQUEDA- Búsqueda



Suponga que se tiene un ABB, ¿cómo buscamos un elemento?

Supongamos que  
se busca el  
valor 10

Como  $6 <> 10$ , y  
 $10 > 6$ , entonces  
busco por su hijo  
derecho



Como  $15 <> 10$ , y  $10 < 15$ ,  
entonces busco por su hijo  
izquierdo

Como  $10 = 10$ , encontré el  
elemento la búsqueda termina y  
retorno el nodo

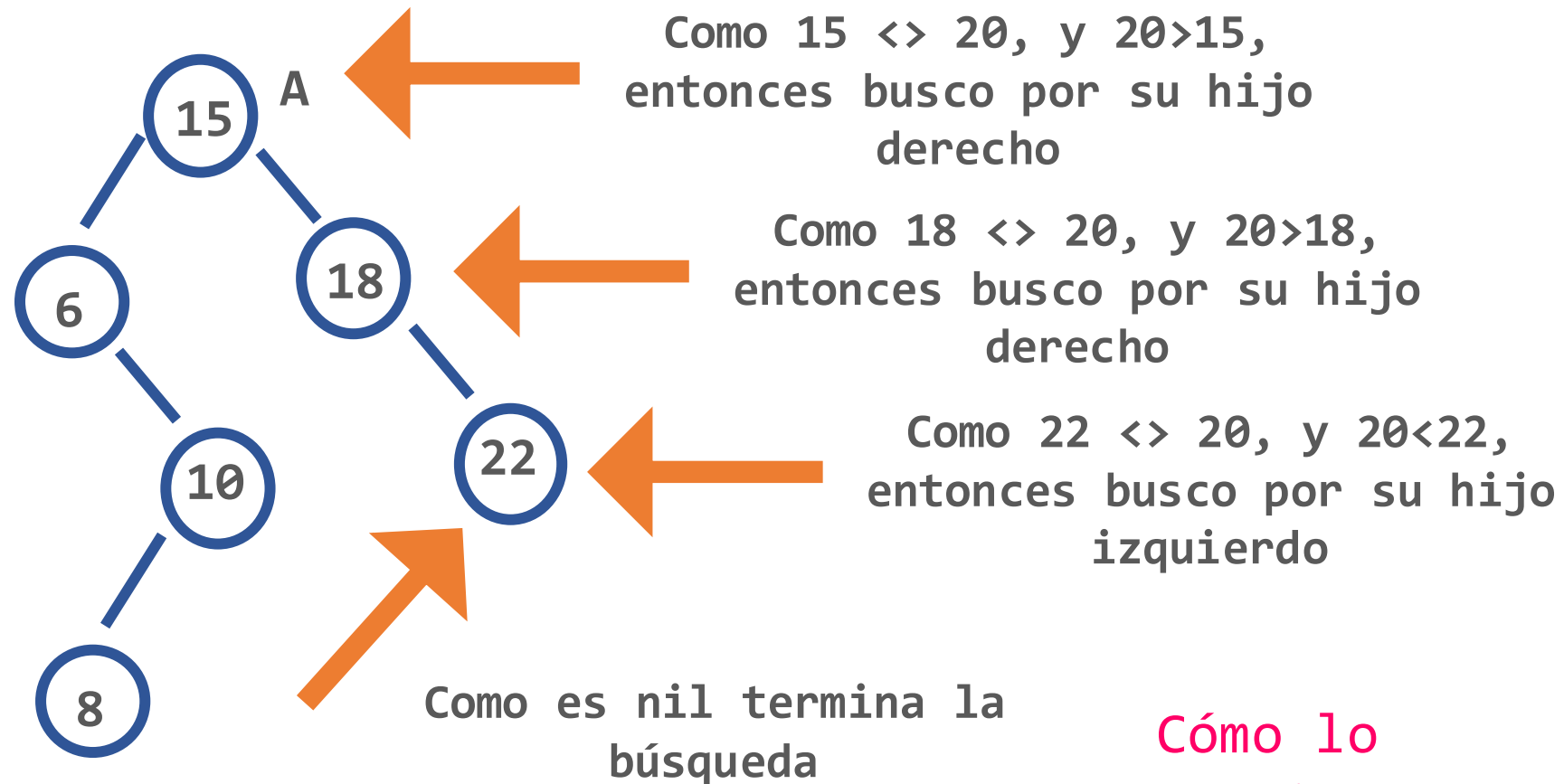


# ÁRBOLES BINARIOS DE BÚSQUEDA- Búsqueda



Suponga que se tiene un ABB, ¿cómo buscamos un elemento?

Supongamos que se busca el valor 20



Cómo lo escribo?



# ÁRBOLES BINARIOS DE BÚSQUEDA- Búsqueda

```
Function Buscar (a:arbol; x:elemento): arbol;  
begin  
  if (a=nil) then  
    Buscar:=nil  
  else if (x = a^.dato) then Buscar:=a  
    else  
  
      if (x < a^.dato) then  
        Buscar:= Buscar(a^.hi ,x)  
      else  
        Buscar:=Buscar(a^.hd ,x)  
    end;  
end;
```





# ÁRBOLES BINARIOS DE BÚSQUEDA- Conclusiones

Un **árbol binario de búsqueda** agrega los elementos por sus hojas. Dichos elementos quedan ordenados. Esta operación lleva un tiempo de ejecución de  $O(\log n)$ .

Qué ocurre en vectores y listas si quiero generar la estructura ordenada?



En un **árbol binario de búsqueda** la búsqueda de un elemento es de  $O(\log n)$ .

Qué ocurre en vectores y listas si quiero buscar un elemento?

En qué caso no se cumple que el tiempo de búsqueda sea  $O(\log n)$ ?

Si los valores leídos son 7, 10, 22, 44. ¿Cómo queda formado el árbol?