Programación II - 2011

Árboles

Árboles

Es una estructura de datos :

- homogénea.
- dinámica.
- no lineal, ya que cada nodo puede tener 0,1 o más sucesores.
- jerárquica.

Árboles

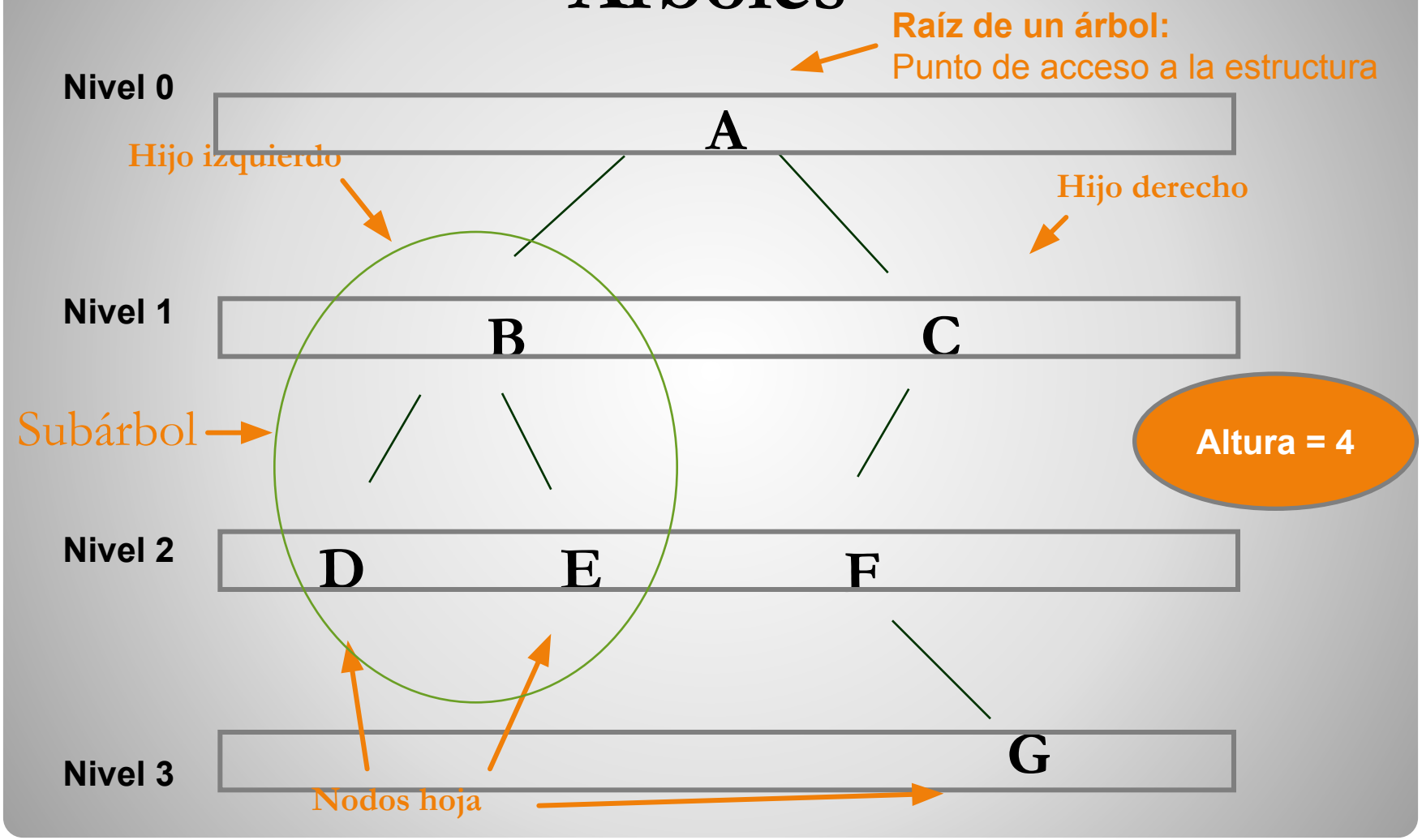
Cada elemento del árbol se relaciona con cero o más elementos a quienes llama hijos.

Si el árbol no está vacío, hay un único elemento al cual se llama raíz y que no tiene padre (predecesor), es decir, no es hijo de ningún otro.

Todo otro elemento del árbol posee un único padre y es un descendiente (hijo del hijo del hijo, etc.) de la raíz.

Cuando cada nodo tiene como máximo 2 hijos se denominan árboles BINARIOS.

Árboles



Altura del árbol Recorrido desde la raíz a la hoja mas lejana = Número de Nivel + 1

Definición

```
Program uno;
Type
   Elemento = …;
   arbol = ^nodo;
   nodo = record
         hijoIzq: arbol;
         elem: elemento;
         hijoDer: arbol;
    end;
```

Char
Integer
Boolean
Real
Enumerativos
Registros
Listas
Arreglos
Arboles

Árbol Binario de Búsqueda

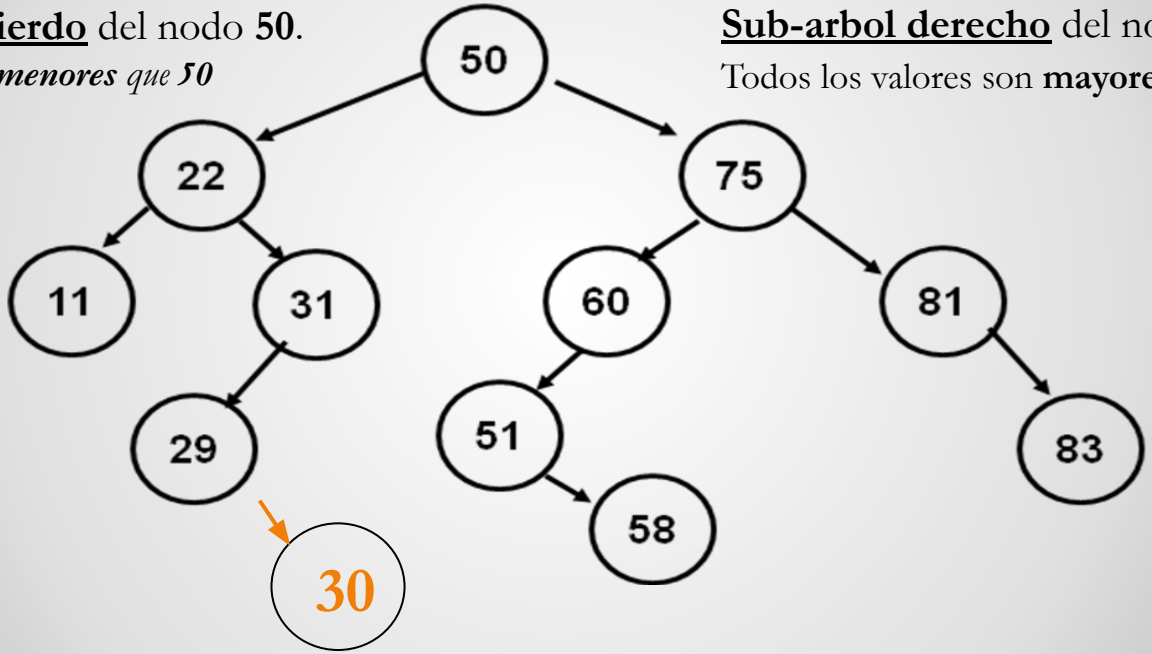
Los datos se almacenan en el árbol siguiendo un **orden.**
Cada nodo tiene un valor.

En cada nodo, todos los elementos del sub-arbol izquierdo son menores al elemento del nodo y todos los elementos del sub-arbol derecho, son mayores con respecto al elemento del nodo.

Ejemplo Árbol Binario de Búsqueda

Sub-arbol izquierdo del nodo **50**.
*Todos los valores son **menores** que **50***

Sub-arbol derecho del nodo **50**.
Todos los valores son **mayores** que **50**



*¿Dónde se debería agregar el valor **30**?*

Enunciado de Ejemplo

Se leen números enteros y se desea almacenarlos en una estructura de árbol binario de búsqueda.

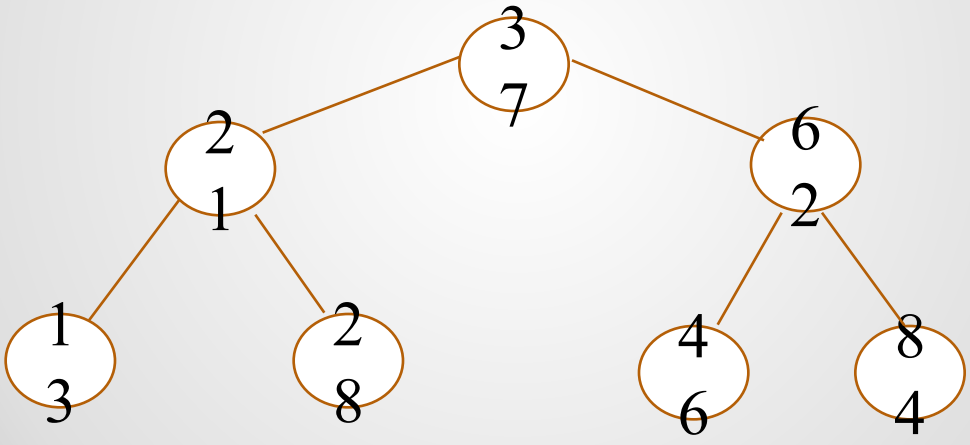
La lectura finaliza cuando se lee el número 0 (cero).

Además, los números que se leen desde teclado, vienen desordenados.

Una vez finalizada la lectura, realizar un módulo que reciba la estructura generada e informe todos aquellos números pares.

Ejemplo del ejercicio

37 21 28 62 46 13 84 0



Solución Propuesta

```
Program ejercicio;
TYPE
        arbol = ^nodo_Arbol;
        nodo_Arbol  = record
            hi  : arbol;
            dato: integer;
            hd : arbol;
        end;

VAR
    a : arbol;

Begin
    a := nil;
    generarArbol(a);
    informarNumeroPares(a);
End.
```

Solución Propuesta

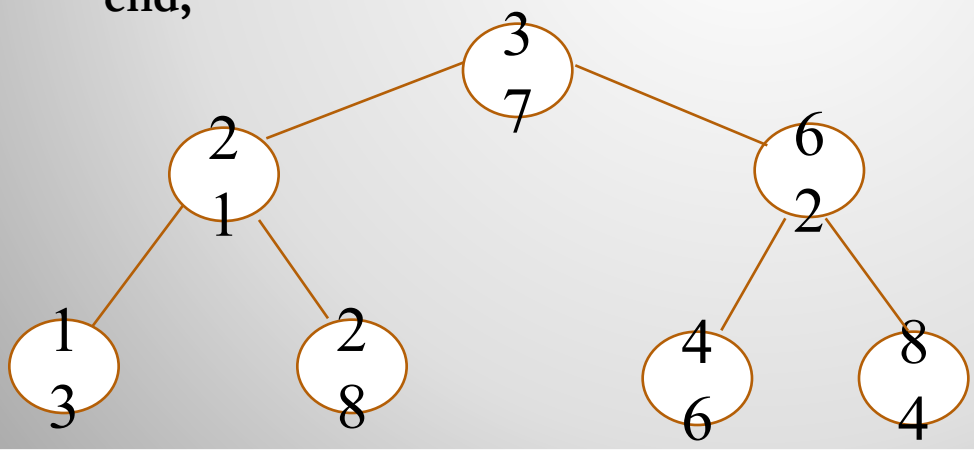
```
Procedure generarArbol(var  a : arbol);
{Proceso que genera el árbol}
Var  n:integer;
Begin
     write('Ingrese un número entero');
     readln(n);
     while (n<>0) do begin
       insertarEnArbol(a,n);
       write('Ingrese un número entero');
       readln(n);
     end;
end;
```

Solución Propuesta

```
Procedure insertarEnArbol(var a : arbol; n : integer);
{Proceso que inserta un nodo el árbol}
Begin
    if (a = nil) then begin
    new (a);
    a^.dato := n;
    a^.hi := nil;
    a^.hd:= nil;
    end
    else
      if (a^.dato > n) then
            insertarEnArbol(a^.hi, n)
      else
            insertarEnArbol(a^.hd, n);
End;
```

Solución Propuesta

```
Procedure informarNumerosPares(a : arbol);
{Proceso que recorre el árbol e informa los números pares}
Begin
    if (a <> nil) then
    begin
       if ( a^.dato mod 2 = 0 ) then
              writeln(a^.dato);
       informarNumerosPares(a^.hi);
       informarNumerosPares(a^.hd);
    end;
end;
```



Informa

28 62 46 84

¿Preguntas?