UNIVERSIDAD FRANCISCO DE VITORIA
VINCE IN BONO MALUM

# Estructuras de Datos y Algoritmos

## Tema 5.1. Árboles.
## Árboles binarios y generales

Prof. Dr. P. Javier Herrera

# Contenido

1. Introducción
2. Terminología
3. Árboles binarios
4. Árboles generales

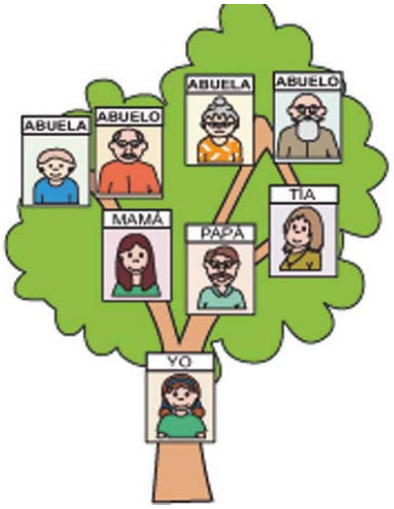
# 1. Introducción

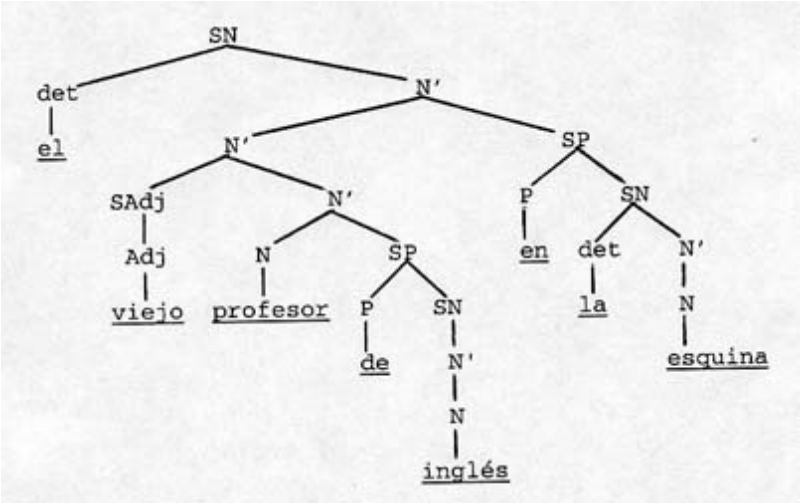
- **Árboles**: Estructuras no lineales
  - Idea formal: Un árbol es un grafo no orientado, conexo y acíclico con un vértice destacado llamado raíz.
  - Idea intuitiva: Los árboles expresan relaciones jerárquicas en las que:
    - Cada elemento (excepto el primero) tiene un único superior inmediato
    - Cada elemento tiene varios siguientes inmediatos

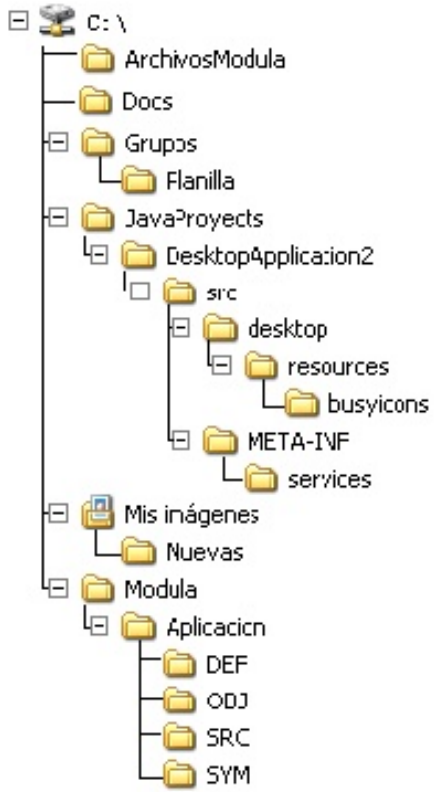
- La estructuras arbóreas son una **generalización de las estructuras lineales** vistas en capítulos anteriores:
  - Listas: cada elemento tiene un único sucesor.
  - Árboles: los elementos pueden tener mas de un sucesor.

# 1. Introducción

- Permiten representar la noción de jerarquía. Ejemplos:
  - Árboles genealógicos: familiares, lingüísticos
  - Árboles sintácticos
  - Estructura de los directorios

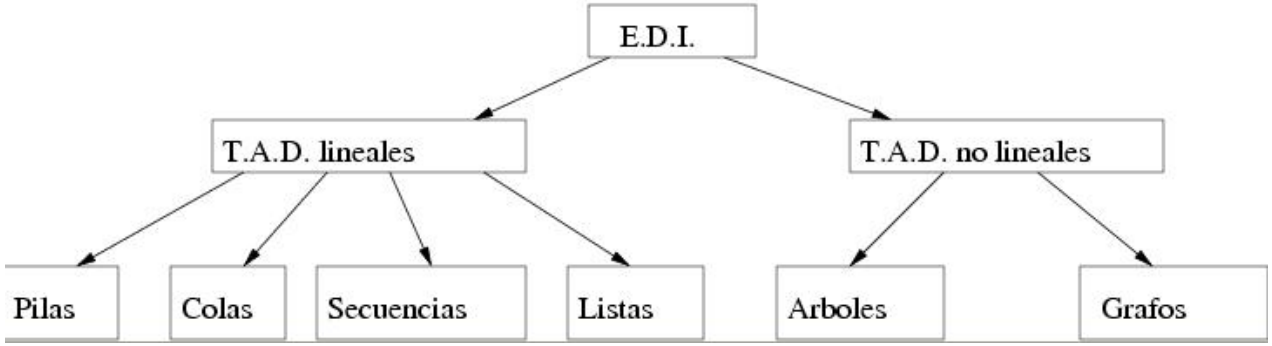


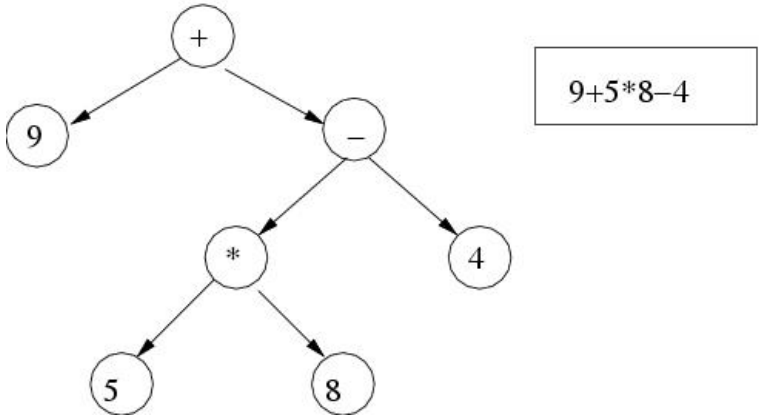



# 1.1 Aplicaciones

- Representación de estructuras anidadas:




- Representación de expresiones:



# 2. Terminología

- La terminología de las estructuras arbóreas se basa a veces en terminología genealógica (“padre”, “hijo”, “primogénito”, “hermano”), o en terminología botánica (“raíz”, “hojas”, “ramas”, etc.).

- **Hoja**: Árbol con un solo elemento.

- **Camino**: Secuencia $A_1,\ldots, A_s$ de árboles tal que, para todo $i \in \{1..s-1\}$, $A_{i+1}$ es subárbol de $A_i$.
  - Se denomina **longitud** del camino a $s-1$.
  - Consideraremos que existe un camino de longitud $0$ de todo subárbol a sí mismo.

# 2. Terminología

- **Ascendiente/Descendiente**: Diremos que el subárbol $A_1$ es ascendiente del subárbol $A_2$ (y que $A_2$ es descendiente de $A_1$), si existe un camino $A_1$,..., $A_2$.

- **Padre**: Se denomina así al primer ascendiente directo, si existe, de un subárbol. Es único y sólo la raíz no tiene padre.

- **Hijos**: Son los primeros descendientes directos de un árbol.

- **Hermanos**: Subárboles con el mismo padre.

# 2. Terminología

- **Altura de un árbol**: Longitud más uno del camino que va desde el árbol original hasta la hoja más lejana. La altura de una hoja es $1$.

- **Profundidad de un subárbol**: Longitud del (único) camino desde el árbol original a dicho subárbol.

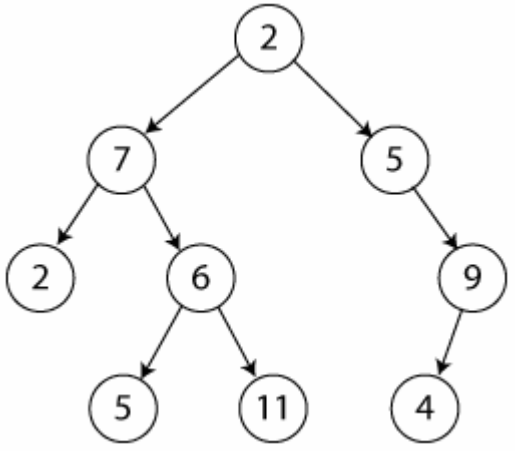
- **Nivel**: Conjunto de subárboles situados a la misma profundidad.

- **Grado de un árbol**: Es el número máximo de hijos que pueden tener sus subárboles.

# 3. Árboles binarios

- Un árbol binario es un conjunto de elementos del mismo tipo tal que:
  - O bien es el conjunto vacío, en cuyo caso se denomina **árbol vacío**.
  - O bien no es vacío, en cuyo caso existe un elemento distinguido llamado **raíz**, y el resto de los elementos se distribuyen en dos subconjuntos disjuntos, cada uno de los cuales es un árbol binario, llamados respectivamente **subárbol izquierdo** y **subárbol derecho**.

- Todo nodo tiene siempre **dos hijos**, aunque uno de ellos o ambos pueden ser vacíos. Los dos hijos se distinguen entre si y se conocen como *hijo izquierdo* e *hijo derecho*.



# 3.1 Operaciones básicas

- El TAD de los árboles binarios cuenta con las siguientes operaciones:
  - crear el árbol vacío,
  - construir un árbol a partir de un elemento y dos árboles,
  - consultar la raíz,
  - calcular el hijo izquierdo,
  - calcular el hijo derecho, y
  - determinar si un árbol es vacío.

# 3.2 Recorridos de árboles binarios

- Recorrer un árbol consiste en “visitar” exactamente una vez todos los elementos del árbol en cierto orden y haciendo algo con ellos.

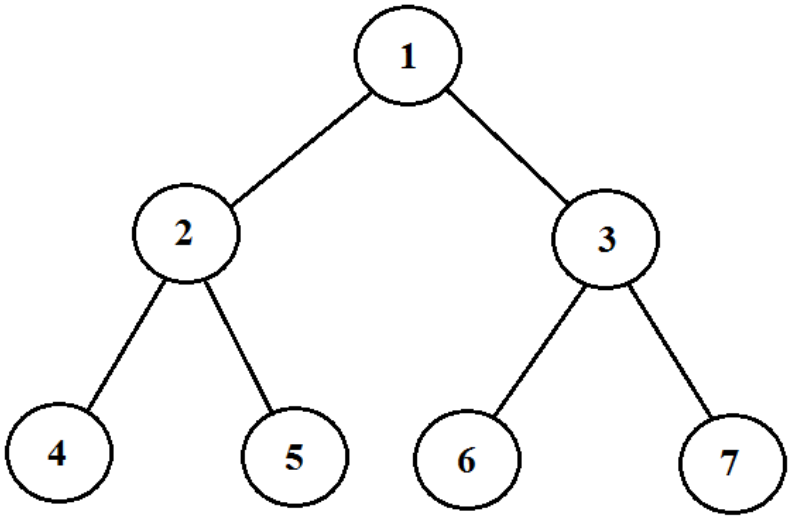
- Dos tipos:
  - Basados en la relación padre-hijo: **recorridos en profundidad**
  - Basados en la distancia del nodo a la raíz: **recorrido por niveles**

# 3.2.1 Recorridos en profundidad

***PREORDEN*** (raíz, izquierdo, derecho)

- Para recorrer un árbol binario no vacío en preorden, hay que realizar las siguientes operaciones recursivamente en cada nodo, comenzando con el nodo de raíz:
  1. Visite la raíz
  2. Atraviese el subárbol izquierdo
  3. Atraviese el subárbol derecho

- Ejemplo: 1 2 4 5 3 6 7

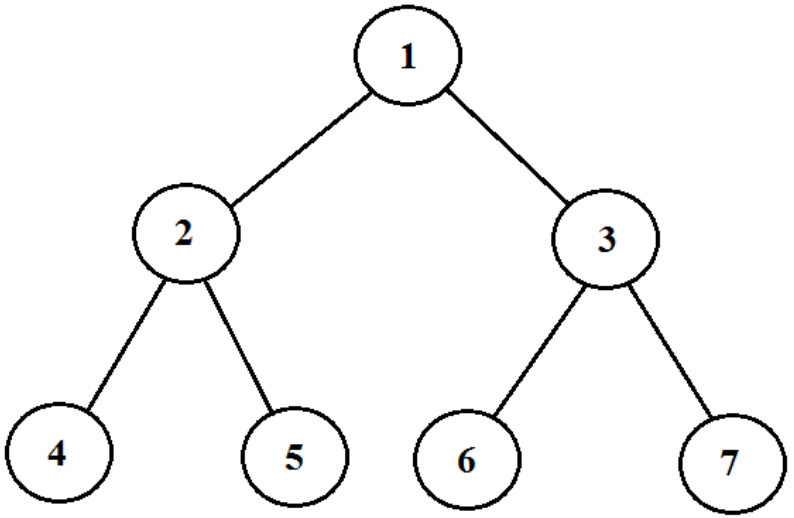


# 3.2.1 Recorridos en profundidad

***INORDEN*** (izquierdo, raíz, derecho)

- Para recorrer un árbol binario no vacío en inorden (simétrico), hay que realizar las siguientes operaciones recursivamente en cada nodo:
  1. Atraviese el subárbol izquierdo
  2. Visite la raíz
  3. Atraviese el subárbol derecho

- Ejemplo: 4 2 5 1 6 3 7

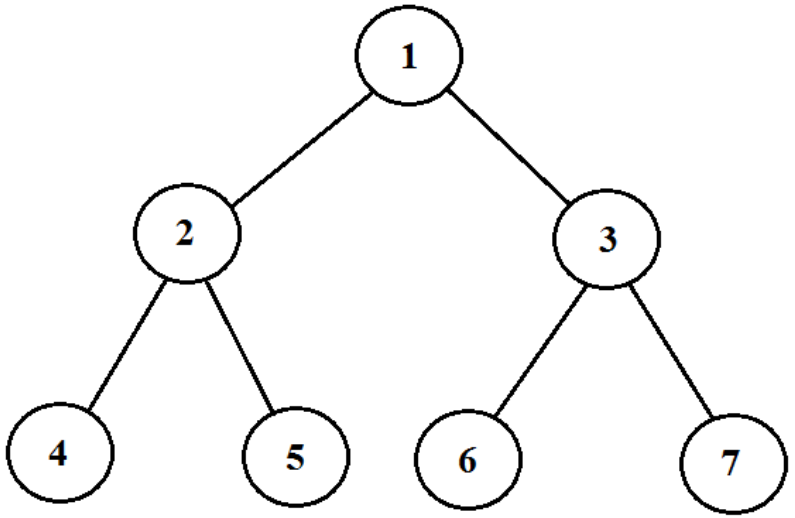


# 3.2.1 Recorridos en profundidad

***POSTORDEN*** (izquierdo, derecho, raíz)

- Para recorrer un árbol binario no vacío en postorden, hay que realizar las siguientes operaciones recursivamente en cada nodo:
  1. Atraviese el subárbol izquierdo
  2. Atraviese el subárbol derecho
  3. Visite la raíz

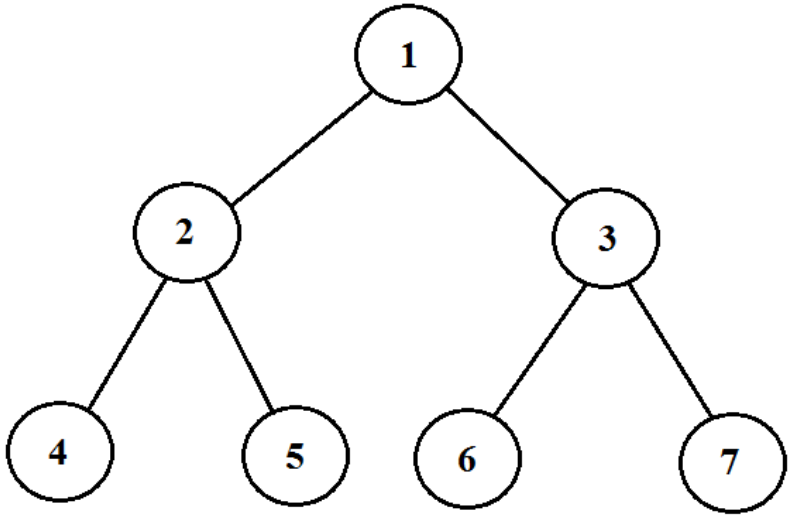
- Ejemplo: 4 5 2 6 7 3 1



# 3.2.2 Recorridos por niveles

- Los árboles también pueden recorrerse por niveles (de nivel en nivel), donde visitamos cada nodo en un nivel antes de ir a un nivel inferior.

- Ejemplo: 1 2 3 4 5 6 7

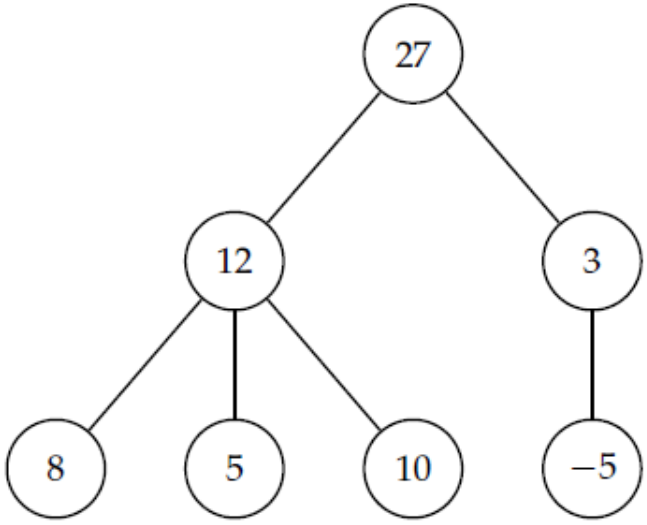


# 4. Árboles generales

- El **número de hijos** de cada nodo es **variable**, desde cero en el caso de una hoja hasta cierto número máximo que se llama **grado del árbol**.

- Cuando el grado es $n$, también se dice que el árbol es ***n*-ario**, pero esto no significa que todos los nodos tienen exactamente $n$ hijos, sino que $n$ es el número máximo de hijos.

- Ejemplo: Árbol 3-ario



# 4. Árboles generales

- Los árboles binarios no son el caso particular de árboles $n$-arios con $n = 2$.
  - Un árbol binario puede ser vacío mientras que **un árbol general nunca es vacío**.
  - Cuando un nodo en un árbol general tiene un único hijo, no se puede decir si es el izquierdo o el derecho porque esto no tiene sentido; en cambio, cuando un nodo en un árbol binario tiene un "único" hijo porque el otro es vacío, se puede distinguir si el hijo no vacío es el izquierdo o el derecho.

- **Todo árbol general se puede representar mediante un árbol binario**. Los árboles binarios adquieren especial importancia cuando tienen propiedades relacionadas con un orden sobre los elementos, como veremos en temas siguientes.

# 4.1 Operaciones básicas

- El TAD de los árboles generales cuenta con al menos las siguientes operaciones:

  - construir un árbol a partir de un elemento y una sucesión de árboles del mismo tipo,
  - construir sucesiones de árboles generales, incluyendo la sucesión vacía,
  - consultar la raíz,
  - calcular la sucesión de hijos,
  - calcular el número de hijos,
  - calcular el hijo $i$-ésimo, y
  - determinar si un árbol es una hoja.

# Bibliografía


- Martí, N., Ortega, Y., Verdejo, J.A. *Estructuras de datos y métodos algorítmicos*. Ejercicios resueltos. Pearson/Prentice Hall, 2003. Capítulo 6

- Peña, R.; *Diseño de programas. Formalismo y abstracción*. Tercera edición. Prentice Hall, 2005. Capítulo 7


(Estas transparencias se han realizado a partir de aquéllas desarrolladas por los profesores Clara Segura, Alberto Verdejo y Yolanda García de la UCM, y la bibliografía anterior)