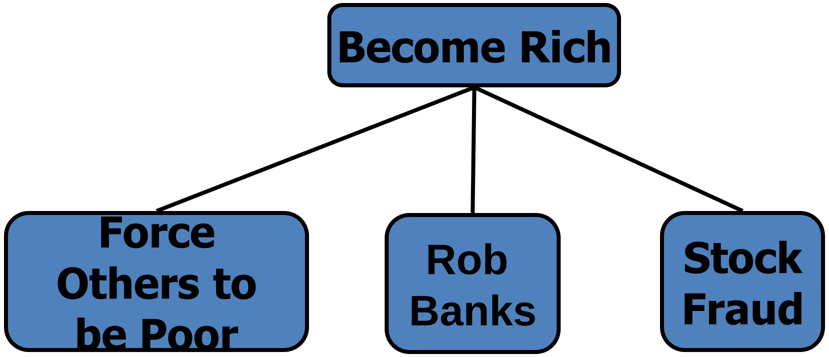
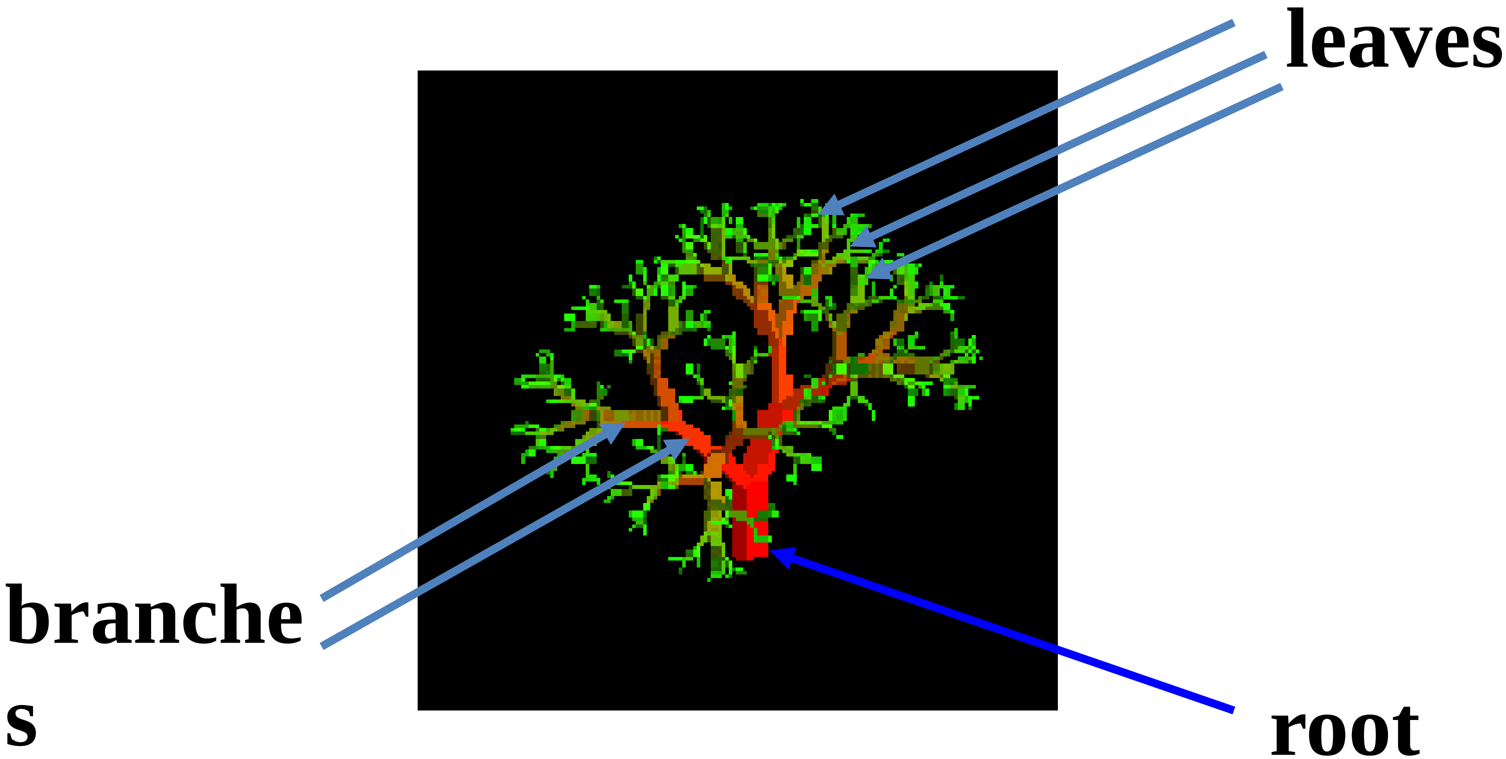
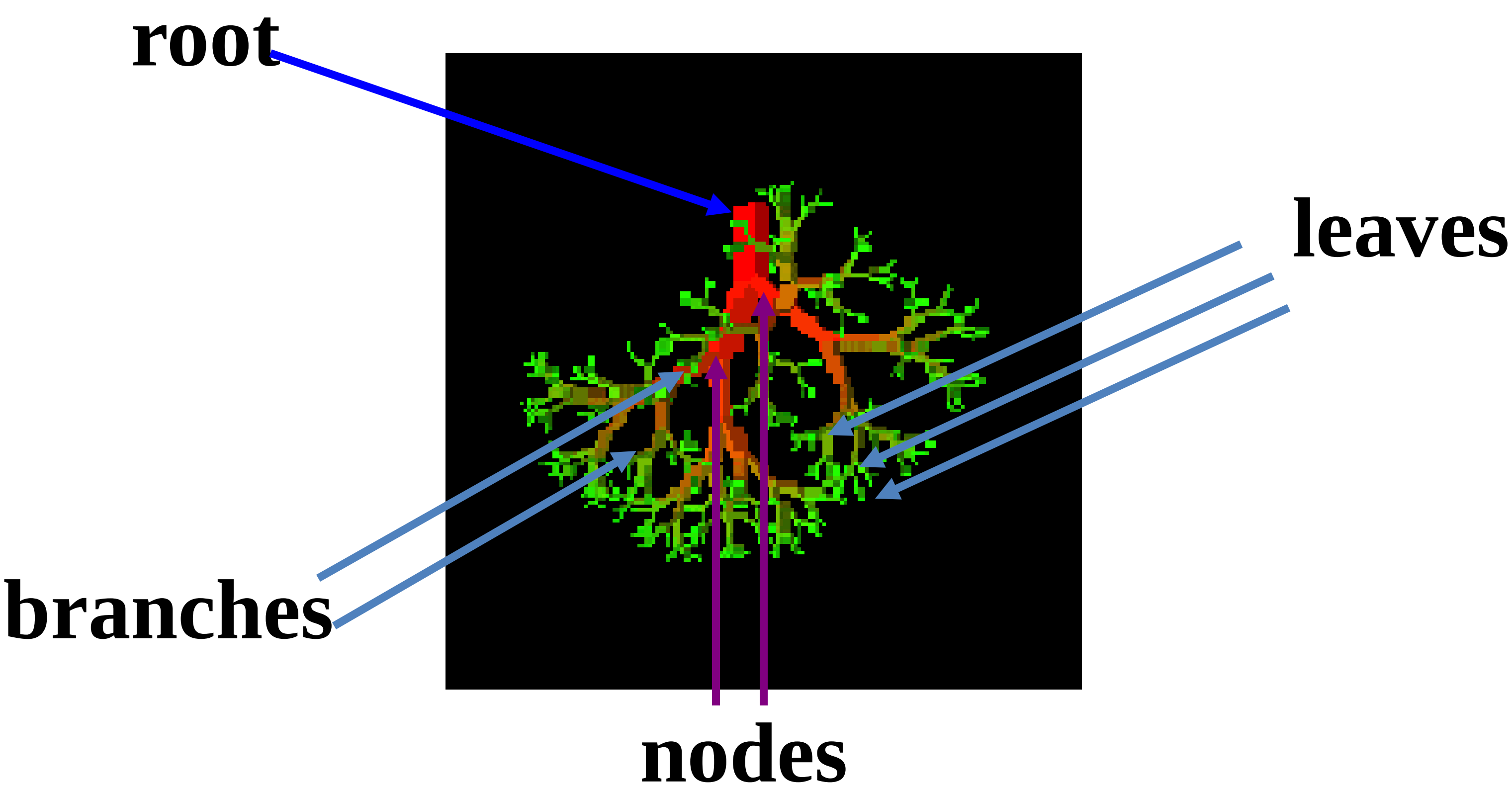
Trees and Basic Terminology (Acyclic Graph)



Nature View of a Tree

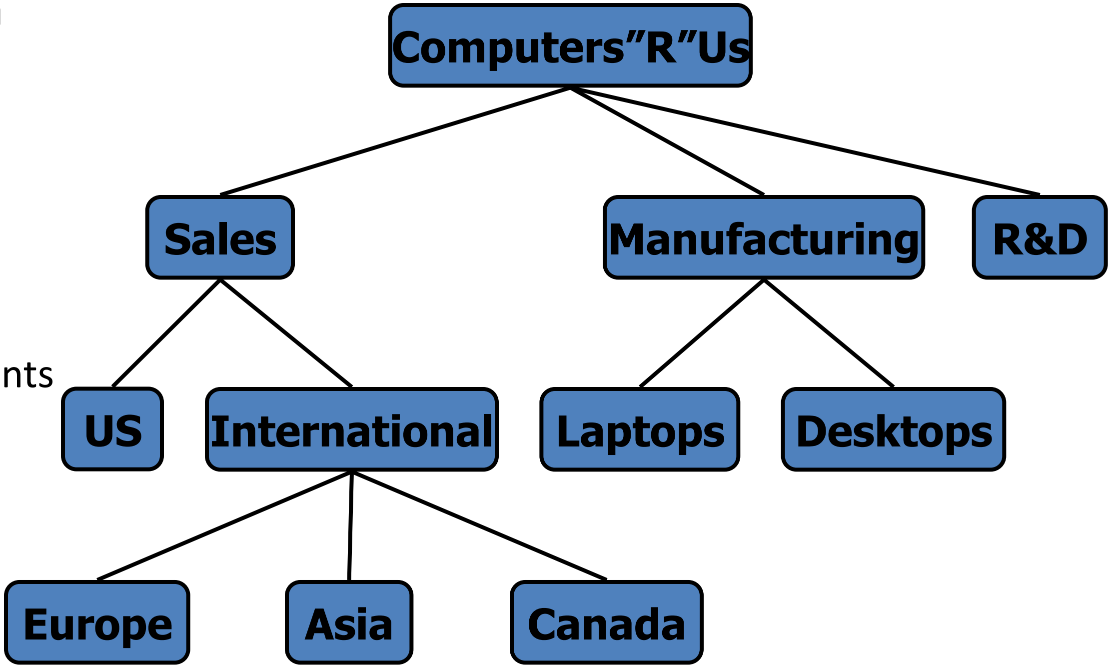


Computer Scientist’s View



What is a Tree

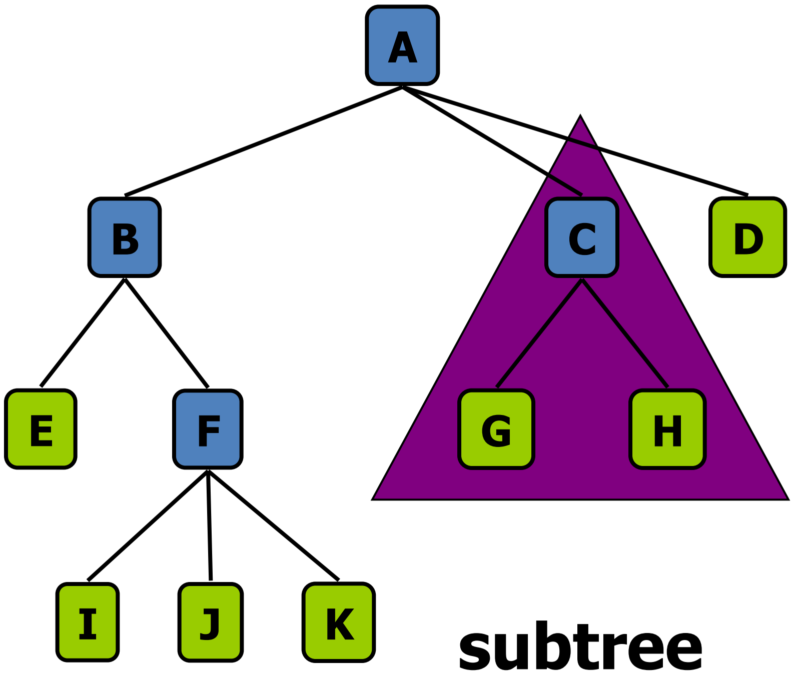
- A tree is a finite nonempty set of elements.
- It is an abstract model of a hierarchical structure.
- consists of nodes with a parent-child relation.
- Applications:
  - Organization charts
  - File systems
  - Programming environments



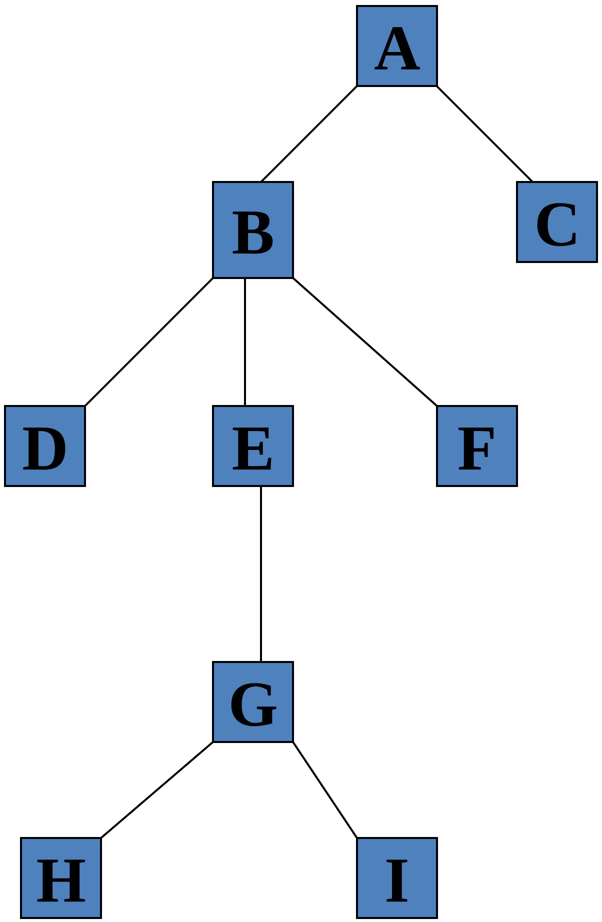
Tree Terminology

- **Root**: node without parent (A)
- **Siblings**: nodes share the same parent
- **Internal node**: node with at least one child (A, B, C, F)
- **External node** (leaf): node without children (E, I, J, K, G, H, D)
- **Ancestors** of a node: parent, grandparent, grand-grandparent, etc.
- **Descendant** of a node: child, grandchild, grand-grandchild, etc.
- **Depth** of a node: number of ancestors
- **Height** of a tree: maximum depth of any node (3)
- **Degree** of a node: the number of its children
- **Degree** of a tree: the maximum number of its node.

- **Subtree: tree consisting of a node and its descendants**



Tree Properties



Property	Value
Number of nodes	
Height	
Root Node	
Leaves	
Interior nodes	
Ancestors of H	
Descendants of B	
Siblings of E	
Right subtree of A	
Degree of this tree	

Tree ADT

- We use positions to abstract nodes
- Generic methods:
  - integer **size**()
  - boolean **isEmpty**()
  - objectIterator **elements**()
  - positionIterator **positions**()
- Accessor methods:
  - position **root**()
  - position **parent**(p)
  - positionIterator **children**(p)

- **Query methods:**
  - **boolean isInternal(p)**
  - **boolean isExternal(p)**
  - **boolean isRoot(p)**
- **Update methods:**
  - **swapElements(p, q)**
  - **object replaceElement(p, o)**
- **Additional update methods may be defined by data structures implementing the Tree ADT**

Basic Tree Concepts

- A tree consists of finite set of elements, called nodes, and a finite set of directed lines called branches, that connect the nodes.
- The number of branches associated with a node is the degree of the node.

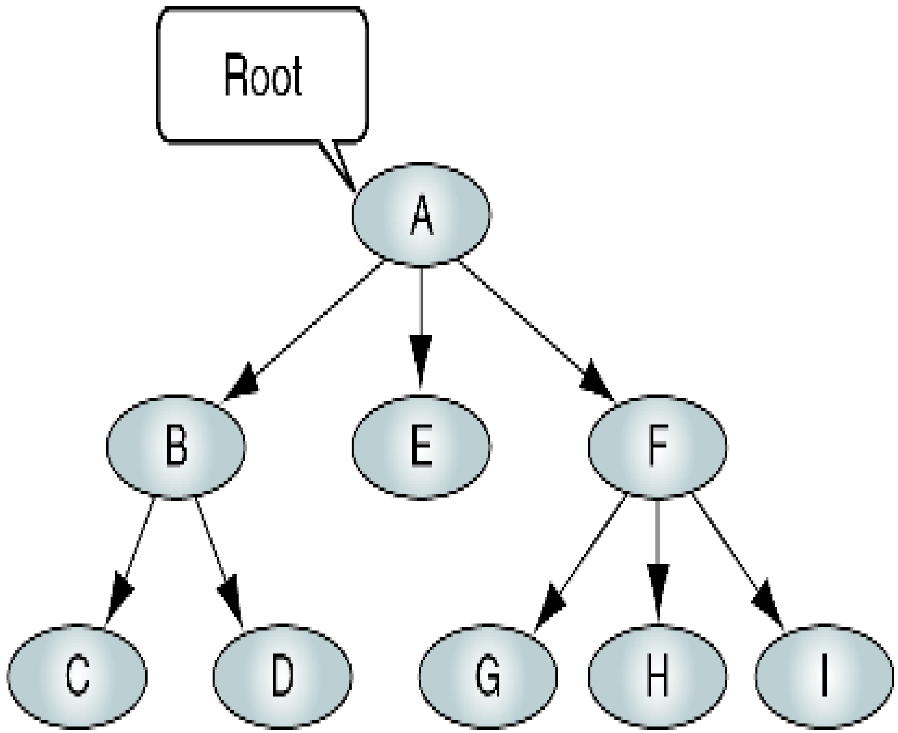


FIGURE 6-1 Tree

Basic Tree Concepts

- When the branch is directed toward the node, it is indegree branch.
- When the branch is directed away from the node, it is an outdegree branch.
- The sum of the indegree and outdegree branches is the degree of the node.
- If the tree is not empty, the first node is called the root.

Basic Tree Concepts

- The indegree of the root is, by definition, zero.
- With the exception of the root, all of the nodes in a tree must have an indegree of exactly one; that is, they may have only one predecessor.
- All nodes in the tree can have zero, one, or more branches leaving them; that is, they may have outdegree of zero, one, or more.

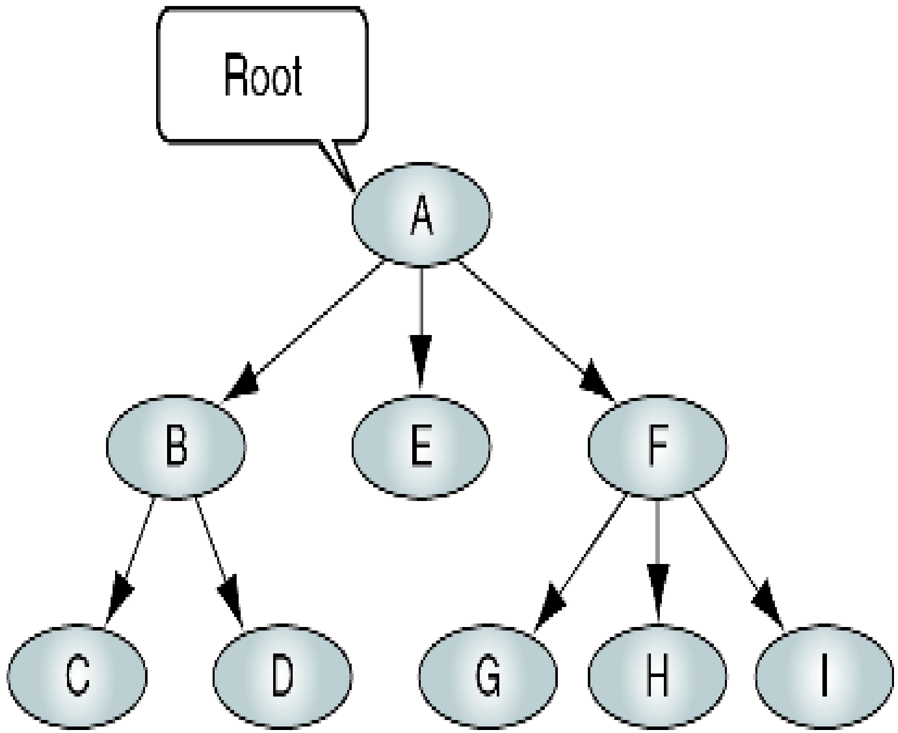


FIGURE 6-1 Tree

Basic Tree Concepts

- A leaf is any node with an outdegree of zero, that is, a node with no successors.
- A node that is not a root or a leaf is known as an internal node.
- A node is a parent if it has successor nodes; that is, if it has outdegree greater than zero.
- A node with a predecessor is called a child.

Basic Tree Concepts

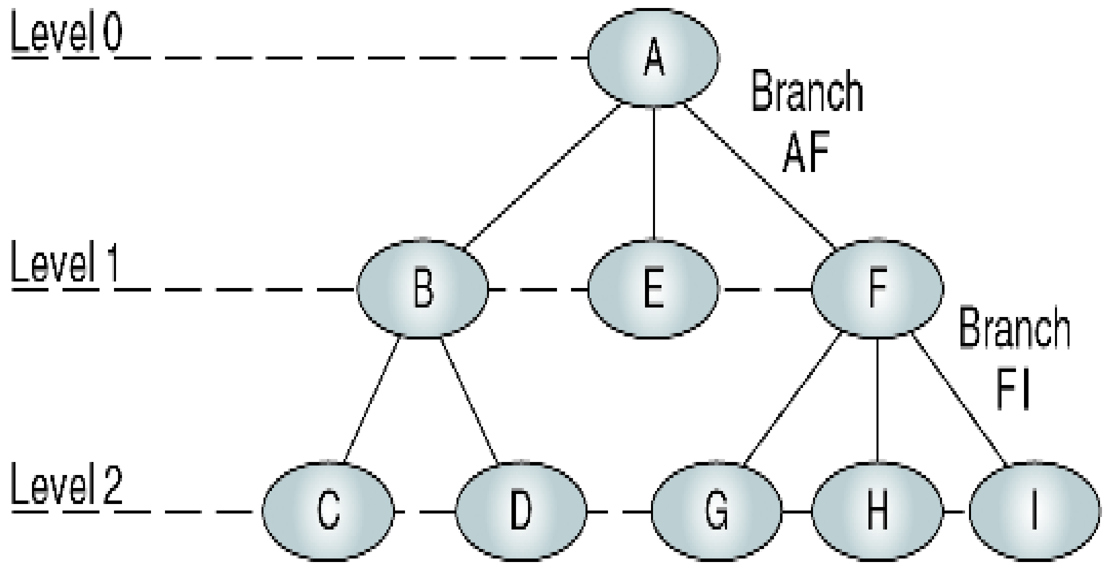
- Two or more nodes with the same parents are called siblings.
- An ancestor is any node in the path from the root to the node.
- A descendant is any node in the path below the parent node; that is, all nodes in the paths from a given node to a leaf are descendants of that node.

Basic Tree Concepts

- A path is a sequence of nodes in which each node is adjacent to the next node.
- The level of a node is its distance from the root. The root is at level 0, its children are at level 1, etc. …

Basic Tree Concepts

- The height of the tree is the level of the leaf in the longest path from the root plus 1. By definition the height of any empty tree is -1.
- A subtree is any connected structure below the root. The first node in the subtree is known is the root of the subtree.



Root: A	Siblings: {B,E,F}, {C,D}, {G,H,I}
Parents: A, B, F	Leaves: C,D,E,G,H,I
Children: B, E, F, C, D, G, H, I	Internal nodes: B,F

FIGURE 6-2 Tree Nomenclature

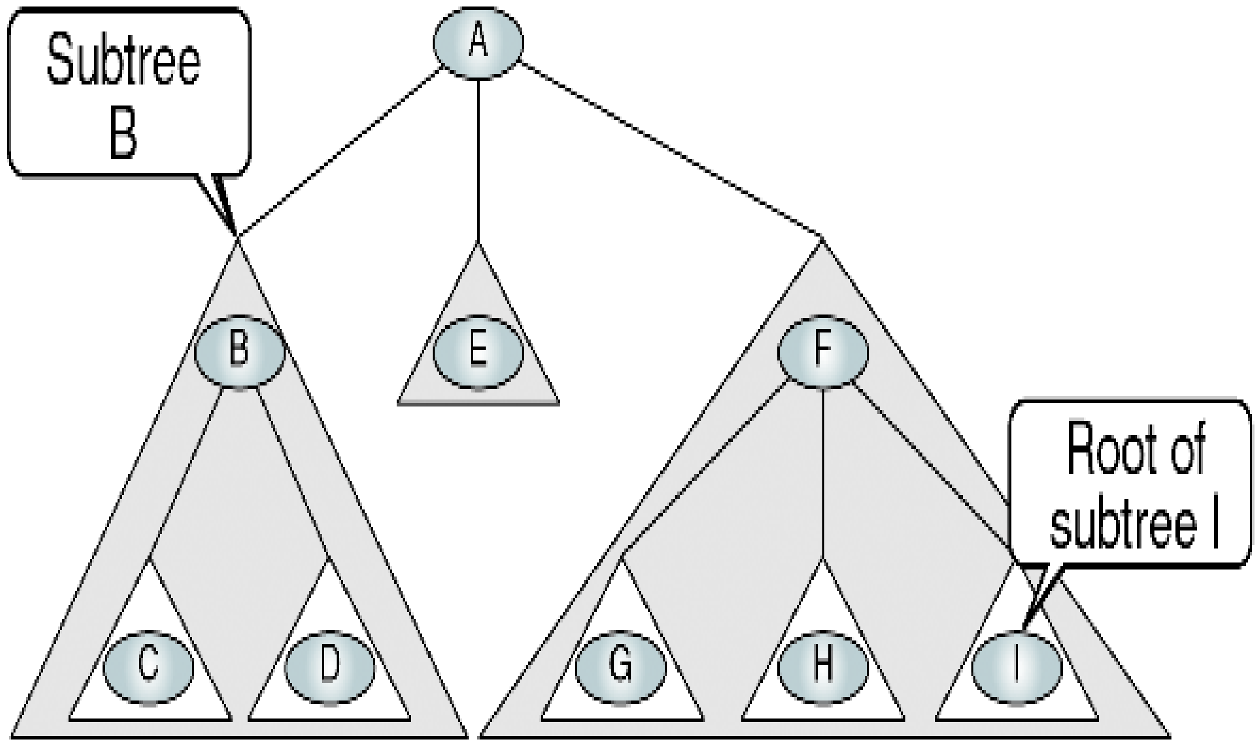


FIGURE 6-3 Subtrees

Recursive definition of a tree

- A tree is a set of nodes that either:
- is empty or
- has a designated node, called the root, from which hierarchically descend zero or more subtrees, which are also trees.

Tree Representation

- General Tree – organization chart format
- Indented list – bill-of-materials system in which a parts list represents the assembly structure of an item

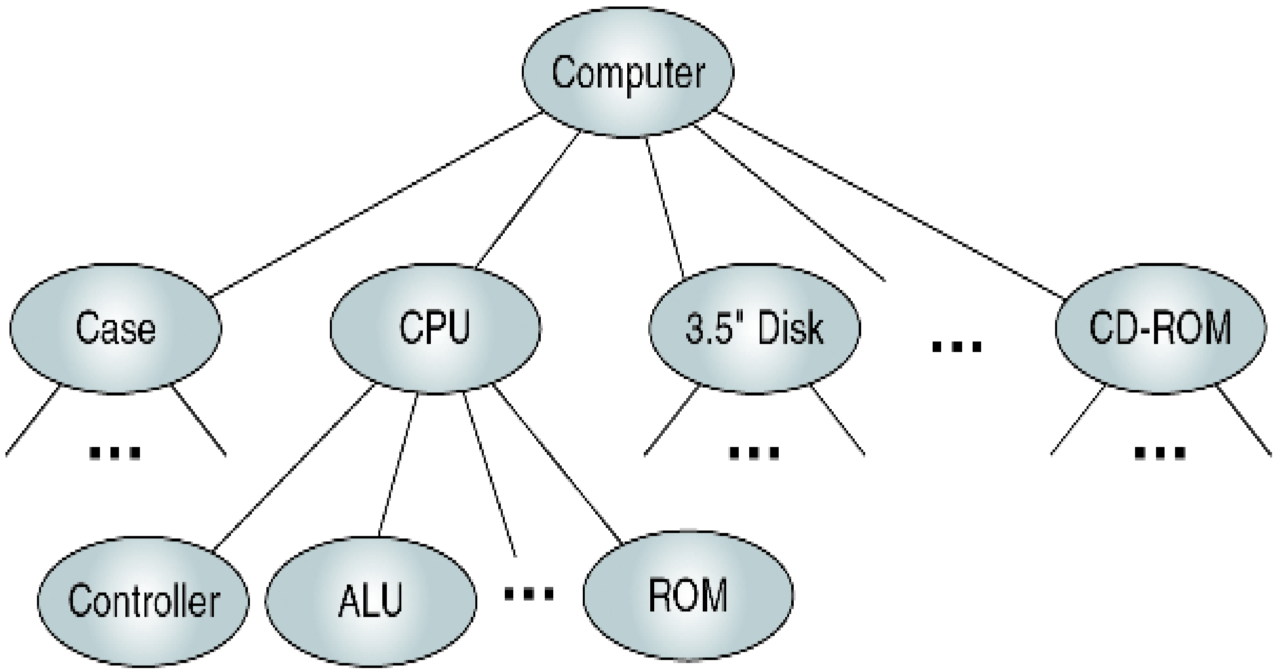


FIGURE 6-4 Computer Parts List as a General Tree

Part number	Description
301	Computer
301-1	Case
...	...
301-2	CPU
301-2-1	Controller
301-2-2	ALU
...	...
301-2-9	ROM
301-3	3.5" Disk
...	...
301-9	CD-ROM
...	...

TABLE 6-1 Computer Bill of Materials

Parenthetical Listing

- Parenthetical Listing – the algebraic expression, where each open parenthesis indicates the start of a new level and each closing parenthesis completes the current level and moves up one level in the tree.

Parenthetical Listing

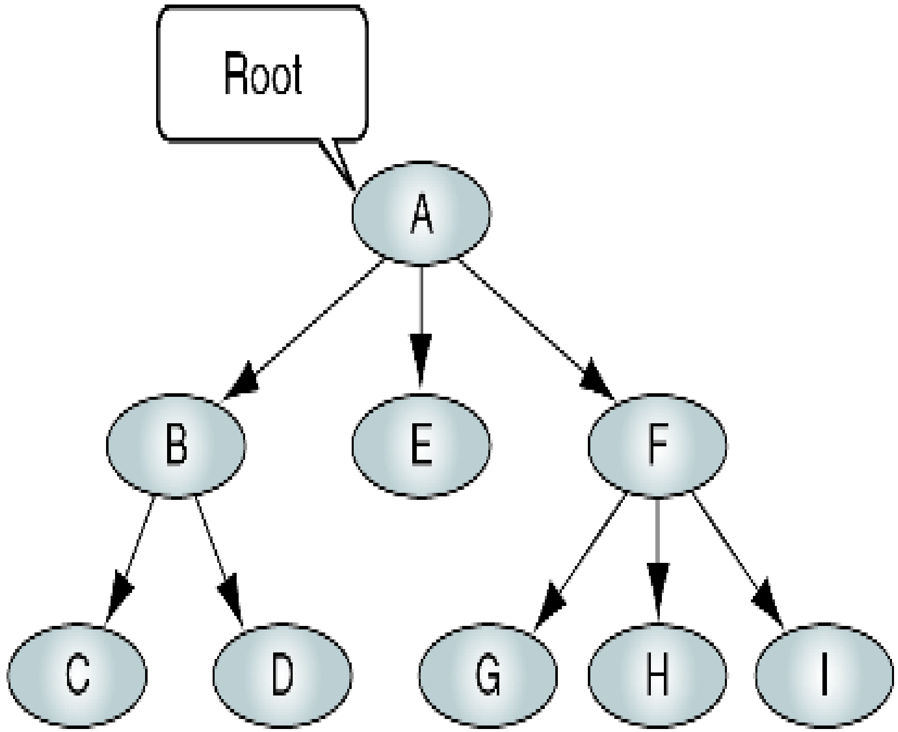


FIGURE 6-1 Tree

A (B (C D) E F (G H I))

ALGORITHM 6-1 Convert General Tree to Parenthetical Notation

```
Algorithm ConvertToParen (root, output)
Convert a general tree to parenthetical notation.
   Pre  root is a pointer to a tree node
   Post output contains parenthetical notation
1 Place root in output
2 if (root is a parent)
   1  Place an open parenthesis in the output
   2  ConvertToParen (root's first child)
   3  loop (more siblings)
      1  ConvertToParen (root's next child)
```

continued

ALGORITHM 6-1 Convert General Tree to Parenthetical Notation *(continued)*

```
     4  end loop
     5  Place close parenthesis in the output
  3 end if
  4 return
end ConvertToParen
```

Intuitive Representation of Tree Node

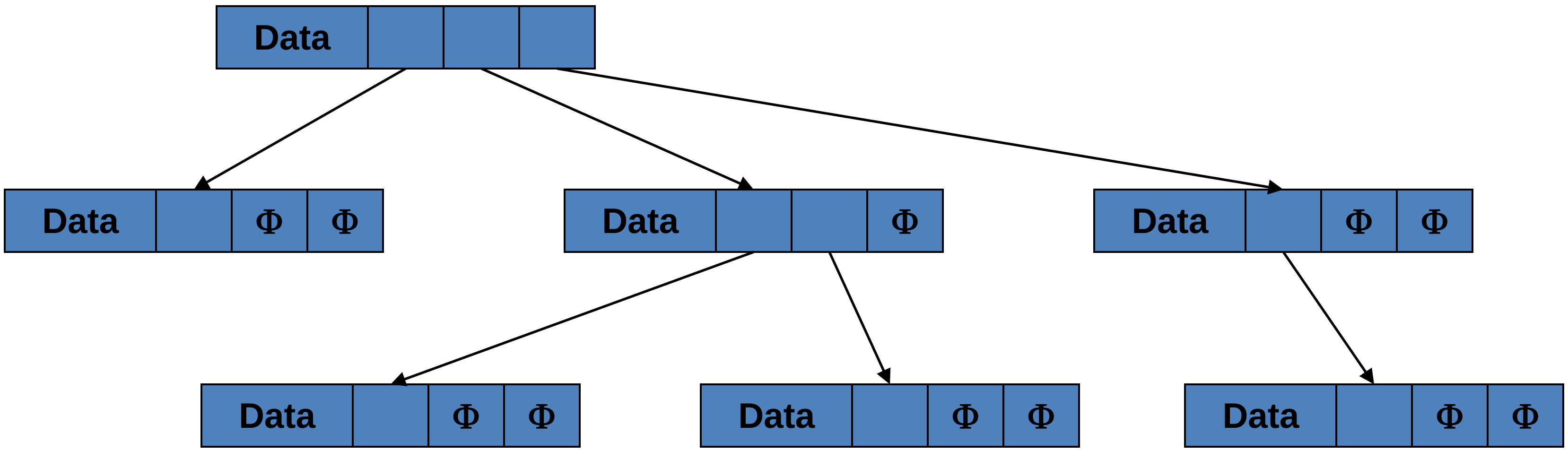
- **List Representation**
  - **(A (B (E (K, L), F), C (G), D (H (M), I, J)))**
  - **The root comes first, followed by a list of links to sub-trees**

How many link fields are needed in such a representation?

Data	Link 1	Link 2	…	Link n

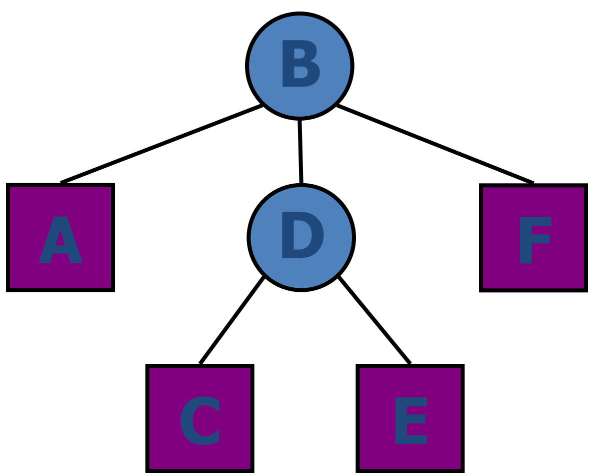
Trees

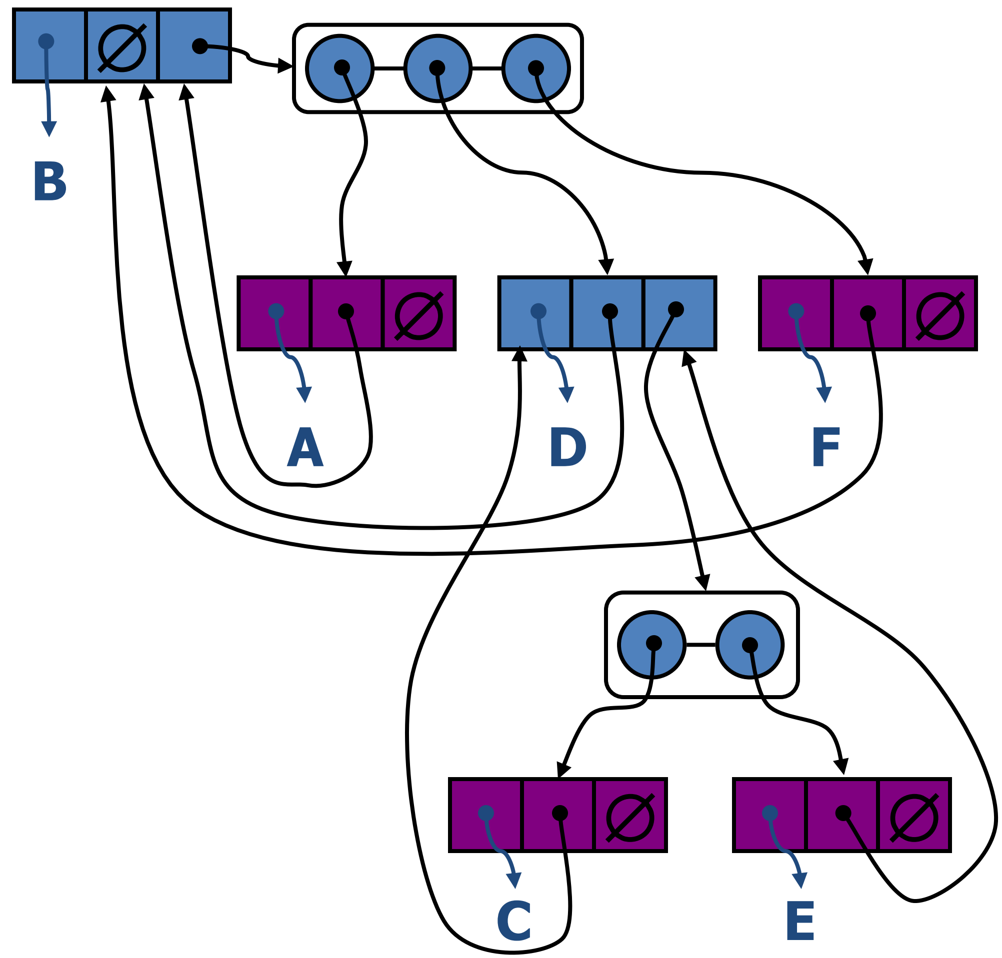
- Every tree node:
  - object – useful information
  - children – pointers to its children



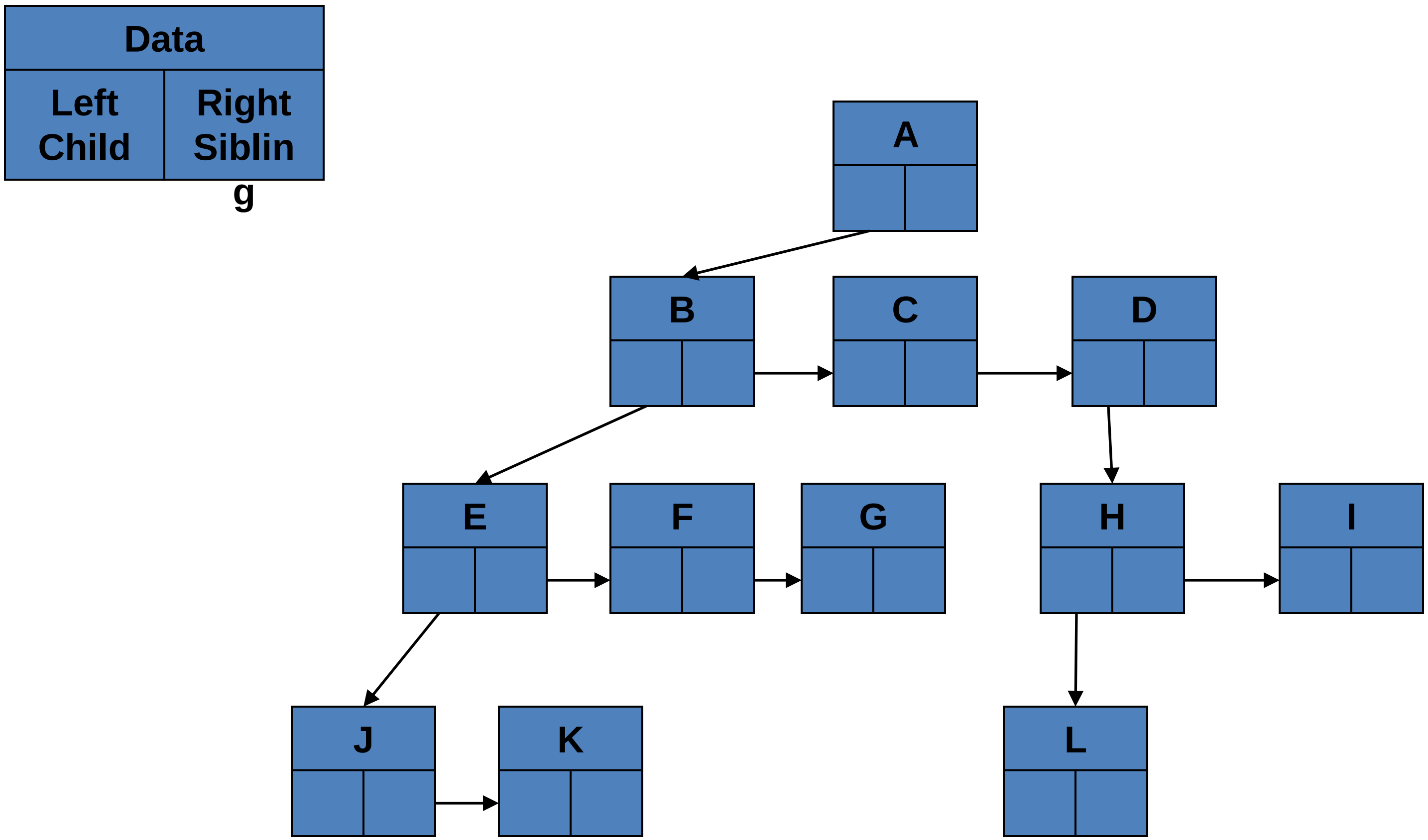
A Tree Representation

- A node is represented by an object storing
  - Element
  - Parent node
  - Sequence of children nodes





Left Child, Right Sibling Representation



Tree Traversal

- Two main methods:
  - Preorder
  - Postorder
- Recursive definition

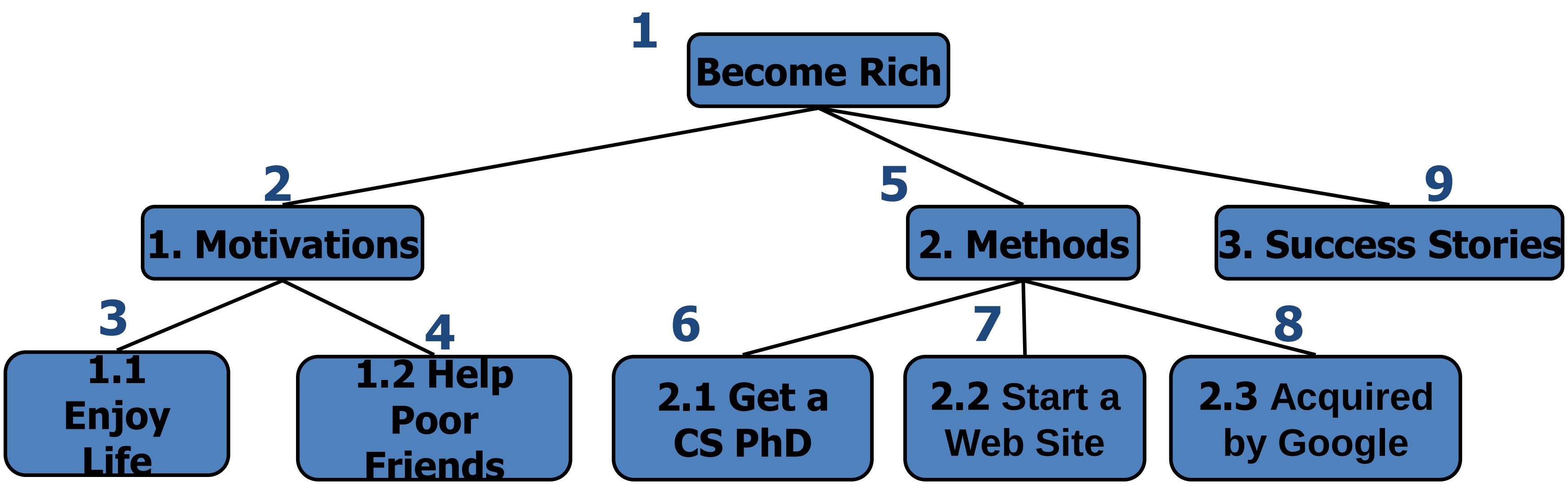
- Preorder:
  - visit the root
  - traverse in preorder the children (subtrees)

- Postorder
  - traverse in postorder the children (subtrees)
  - visit the root

Preorder Traversal

- A traversal visits the nodes of a tree in a systematic manner
- In a preorder traversal, a node is visited before its descendants
- Application: print a structured document

Algorithm *preOrder*(*v*)
    visit(*v*)
    for each child *w* of *v*
        preorder (*w*)



Postorder Traversal

- In a postorder traversal, a node is visited after its descendants
- Application: compute space used by files in a directory and its subdirectories

Algorithm *postOrder*(*v*)
  for each child *w* of *v*
    postOrder (*w*)
  visit(*v*)