CSE-213 (Data Structure)

Lecture on Tree

Md. Jalal Uddin

Lecturer

Department of CSE

City University

Email: jalalruice@gmail.com

No: 01717011128 (Emergency Call)

City University ...creating a culture of excellence

Department of Computer Science & Engineering (CSE)
City University, Khagan, Birulia, Savar, Dhaka-1216, Bangladesh

CSC-391: Data Structures

Lecture: 11

Tree-1: Preliminaries

Lecture-11: Preliminaries on Tree

Objectives of this Lecture:

- ❖ Define Tree
- ❖ Tree terminologies
- ❖ Variations of tree:
  - General tree
  - Binary tree
  - Complete binary tree
  - Extended binary tree

What is Tree?

What is tree in Data Structure:

- In computer science, a tree is a widely used non-linear data structure consisting of nodes organized as a hierarchy, i.e., there is a hierarchical relationship between data elements of a tree.

- A tree is a finite set of one or more nodes such that:

❖ There is a specially designated node called the root.

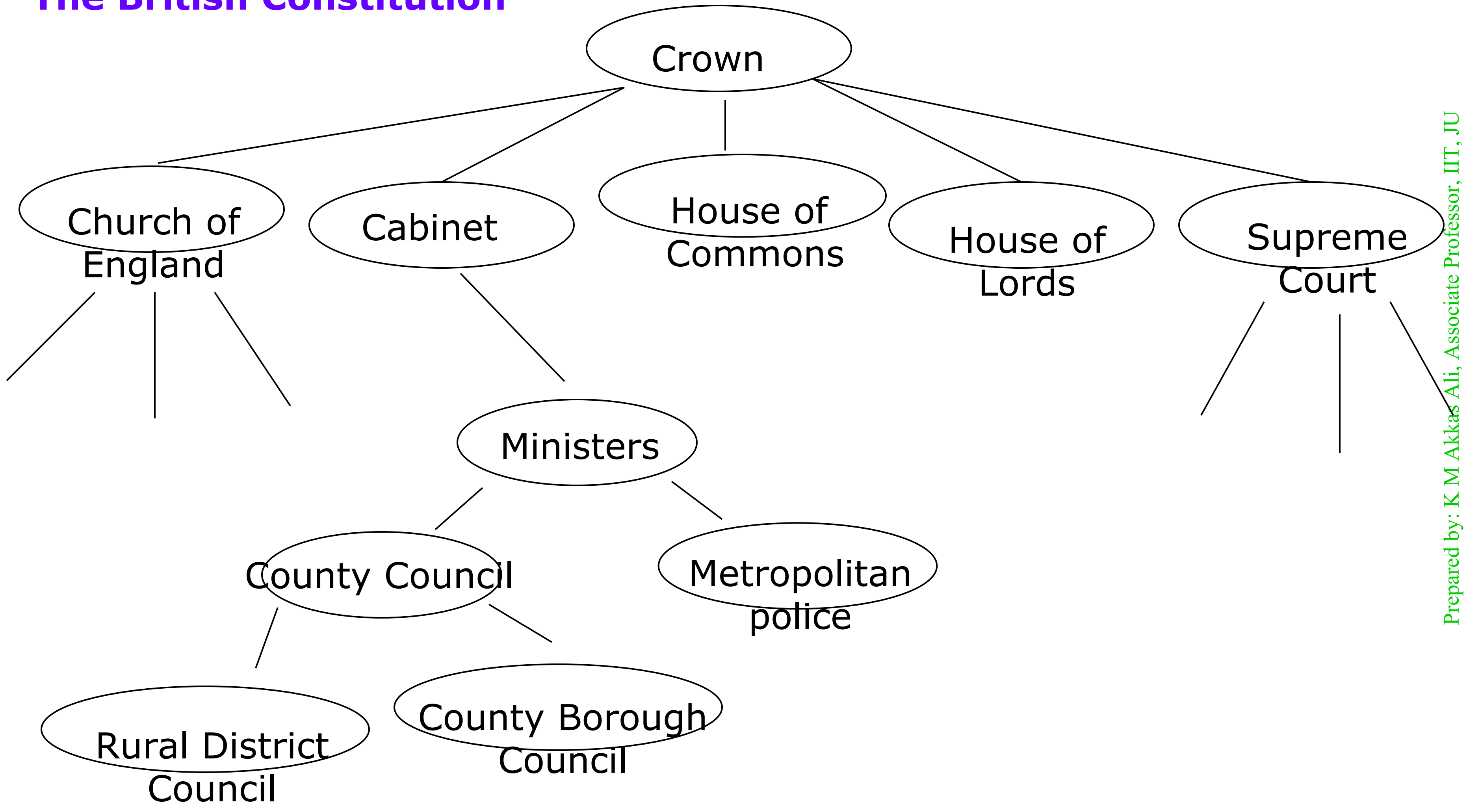
❖ The remaining nodes are partitioned into n (where n>=0) disjoint sets T_1, T_2,…, T_n, where each of these sets is a tree.

❖ We call T_1, T_2, ..., T_n the subtrees of the root.

What is Tree?

Example of Tree in Everyday Life:

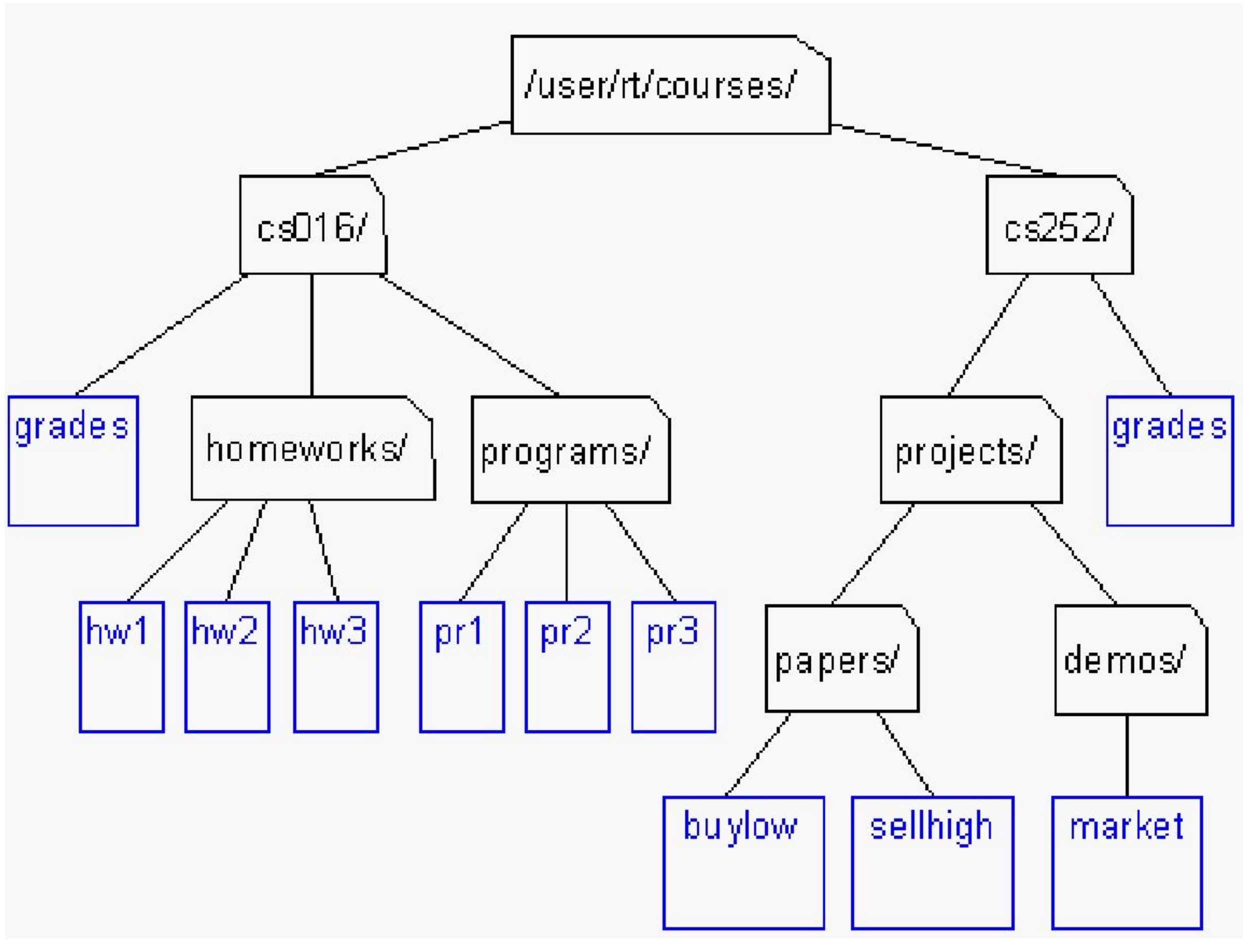
The British Constitution



What is Tree?

Example of Tree in Everyday Life:

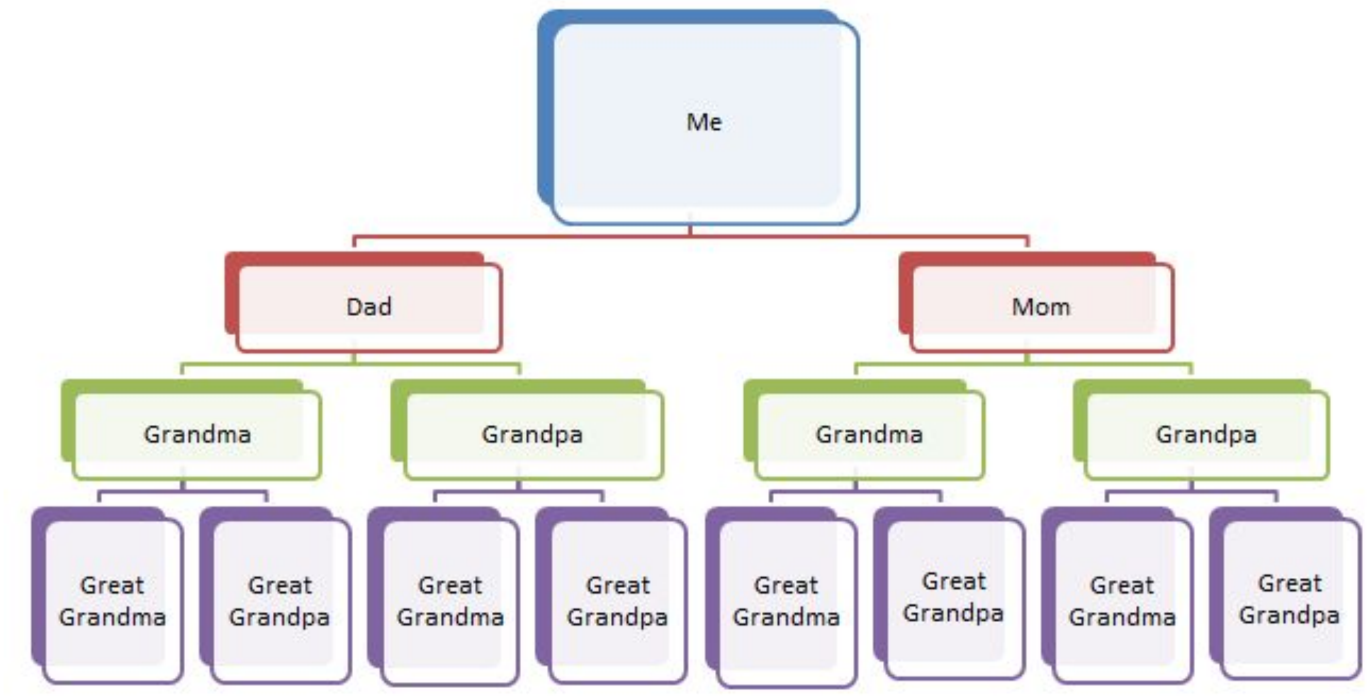
Unix / Windows file structure



What is Tree?

Example of Tree in Everyday Life:

My Family Tree



Tree Terminology

- Terminology describing family relationships is frequently used to describe relationships between the nodes of a tree.

- Consider the tree shown on the figure.

Node:

❑ Finite set of elements of a tree are called nodes.
  – Total node here is 13.

Children:

❑ Nodes B, C, and D are the children of node A. Nodes E and F are the children of node B.

Parent:

❑ A node that has subtree(s) is called the parent node. Here, node A is the parent of nodes B, C, and D. Similarly, node B is the parent of nodes E and F.

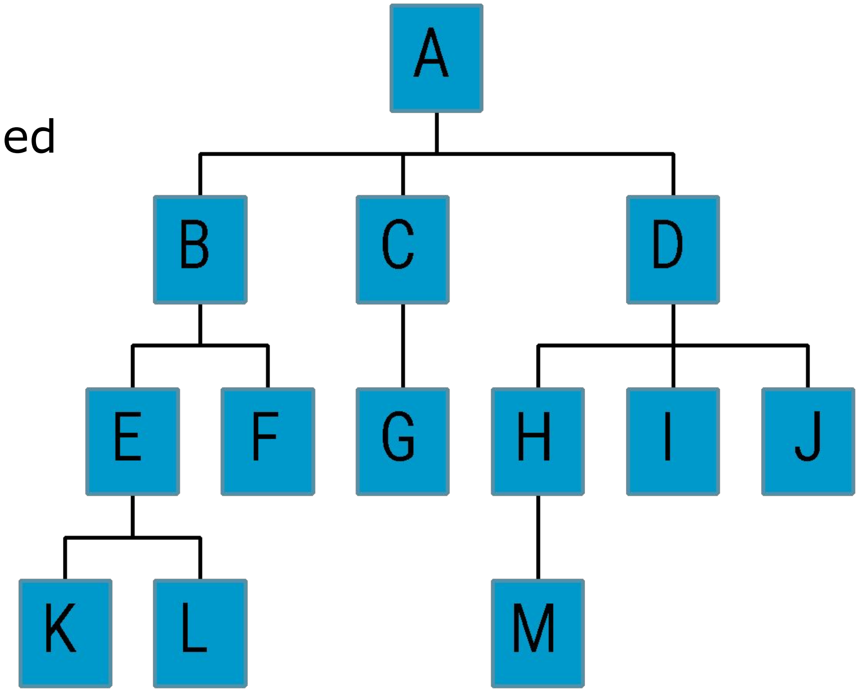
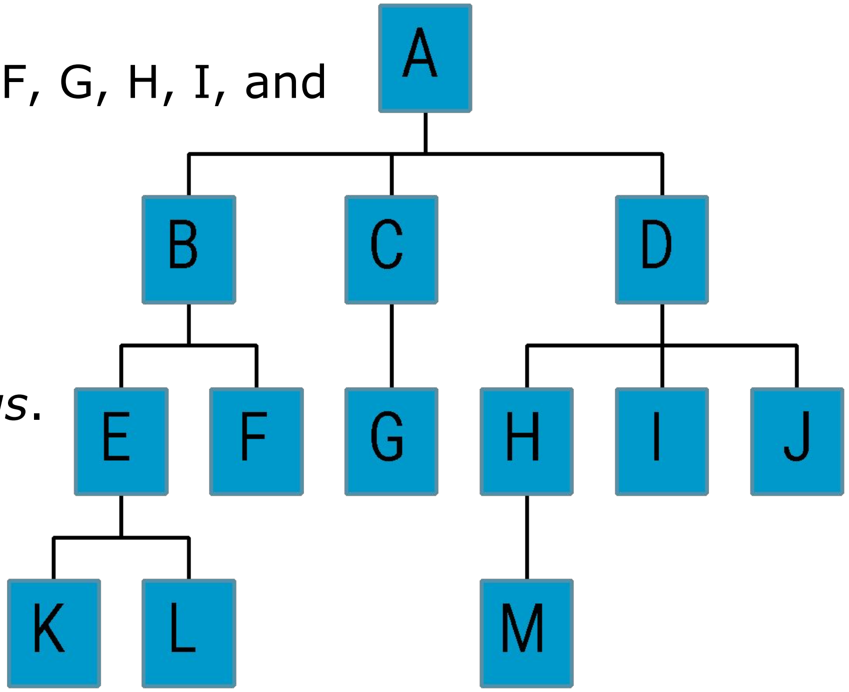


Figure: Tree

Tree Terminology

Grandparents, Grandchildren:

- ❑ Node A is the grandparent of nodes E, F, G, H, I, and J.
- ❑ Nodes E, F, G, H, I, and J are the grandchildren of node A.



Siblings:

- ❑ Children of the same parent are *siblings*.
  -- E and F are siblings.

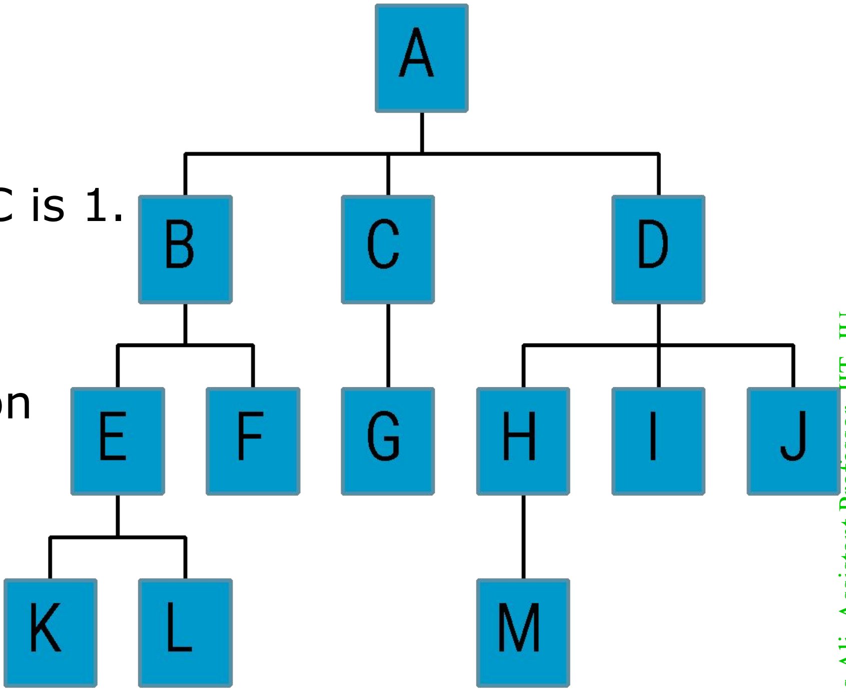
Root / Root Node:

- ❑ The distinguished node of a tree which have no predecessor is called root. In other words, a node without a parent is a root.
  – Here, node A is the root of the tree.

Leaf or Terminal node:

- ❑ The node with degree 0 is called a leaf or terminal node. In other words, the nodes at the lowest levels of the tree (the ones with no sub-trees) are called leaves. Terminal nodes have no successors. The nodes at the lowest levels of the tree (the ones with no sub-trees) are called **leaves**.

Tree Terminology



Degree of a node:

- The degree of a node is the number of subtrees of the node
  - The degree of A is 3; the degree of C is 1.

Descendant of a node:

- A descendant ***n*** of a node is any node on the path from ***n*** to a leaf.
  - Nodes E, F, K, and L are the descendants of node B.

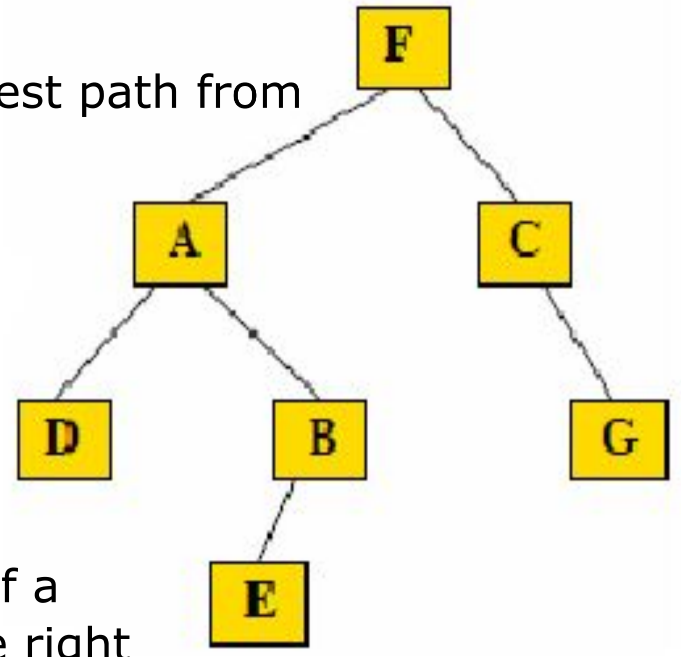
Ancestor of a node:

- An ancestor of a node ***n*** is a node on the path from ***n*** to the root.
  - Here, nodes E, B, and A are the ancestors of node L.

Predecessor of a node:

- Every node N in a tree , except the root, has a unique parent, called the *predecessor* of N.

Tree Terminology



Height of a node:

- ❑ The height of a node is the length of the longest path from the node to a leaf below it.

Depth of the Tree:

- ❑ The depth of a tree is the maximum number of nodes in a branch of the tree. This is 1 more than the largest level number of the tree.

Left Successors and Right Successors:

- ❑ In a binary tree, all the children on the left of a node are called its left successors and on the right are called right successors.

Left Child and Right Child:

- ❑ In a binary tree, a node N may have two children. One is left and another is right child.
  - D is left and B is right child of A.

Level of a node:

- ❑ In a binary tree, every node is assigned a level number, as follows:
- ❖ The root R of the tree is assigned the level number 0, and every other node is assigned a level number which is one more than the level number of its parent.
- ❖ Furthermore, those nodes with the same level number are said to belong the to the same generation.

Binary Tree

- The simplest form of tree is a **binary tree** in which each node has at most two descendants.
- A binary tree consists of

1) a *node* (called the **root** node) and
2) left and right sub-trees. Both the sub-trees are themselves binary trees.

- A binary tree T is defined as a finite set of elements, called nodes, such that:

- ❑ T is empty (called the null tree or empty tree), or
- ❑ T contains a distinguished node R, called the root of T, and the remaining nodes of T form an ordered pair of disjoint binary trees T_1 and T_2.

- If T does contain a root R, then the two trees T_1 and T_2 are called, respectively, the left and right subtrees of R.
- If T_1 is nonempty, then its root is called the left successor of R; similarly, if T_2 is nonempty, then its root is called the right successor of R.

Representing Binary Tree

- A binary tree T is frequently presented by means of a diagram as shown below.

- ❖ The tree consists of 11 nodes (represented by A, B, C etc.).
- ❖ A is the root node of the tree.
- ❖ B is a left successor and C is a right successor of the node A.
- ❖ The left subtree of A consists of the nodes B, D, E and F.
- ❖ The right subtree of A consists of the nodes C, G, H, J, K and L.

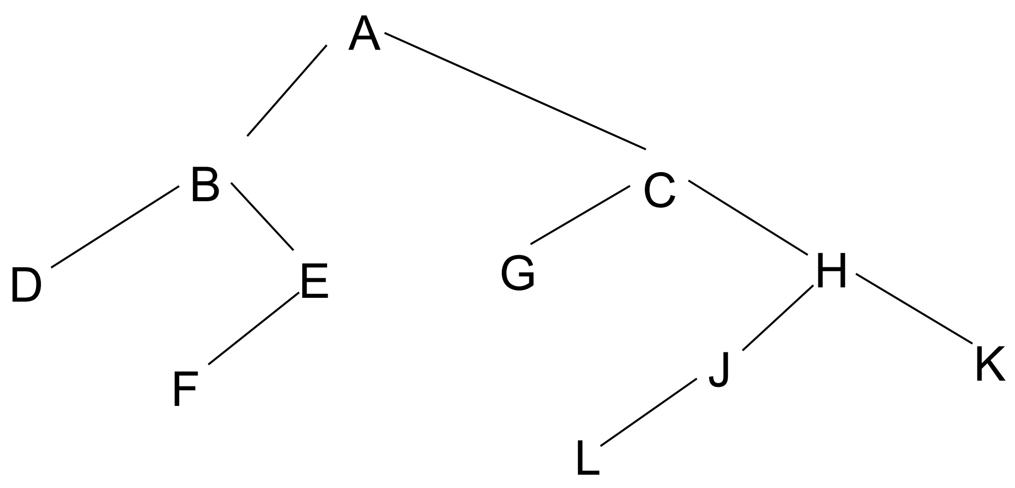


Figure: A binary tree

Properties of a Binary Tree

- Any node in a binary tree has either 0, 1 or 2 successor.
- ❖ the nodes A, B, C and H have two successors,
- ❖ the nodes E and J have only one successor, and
- ❖ the nodes D, F, G, L and K have no successors
- The nodes with no successors are called *terminal nodes.*
- ❖ the nodes D, F, G, L and K are terminal nodes.
- Every node N of a binary tree T contains a left and a right subtree. If N is a terminal node, then both its left and right subtree are empty.

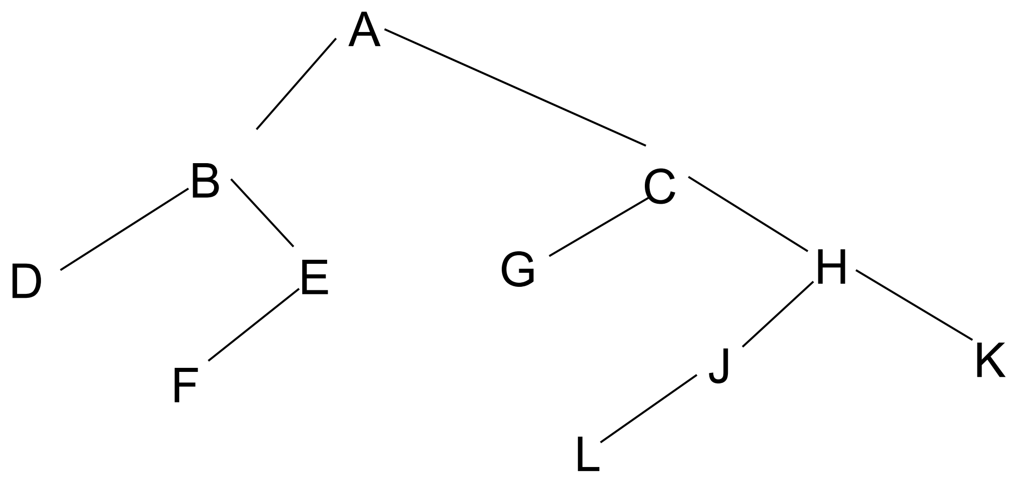


Figure: A binary tree

Properties of a Binary Tree

- Binary tree T and T′ are said to be similar if they have the same structure or, in other words, if they have the same shape.
- Binary tree T and T′ are said to be copies if they are similar and if they have the same contents at corresponding nodes.

Example:

- Consider the four binary trees shown in figure below.
- The three trees (a), (c) and (d) are similar.
- In particular, the trees (a) and (c) are copies since they also have the same data at corresponding nodes.
- The tree (b) is neither similar nor a copy of the tree.

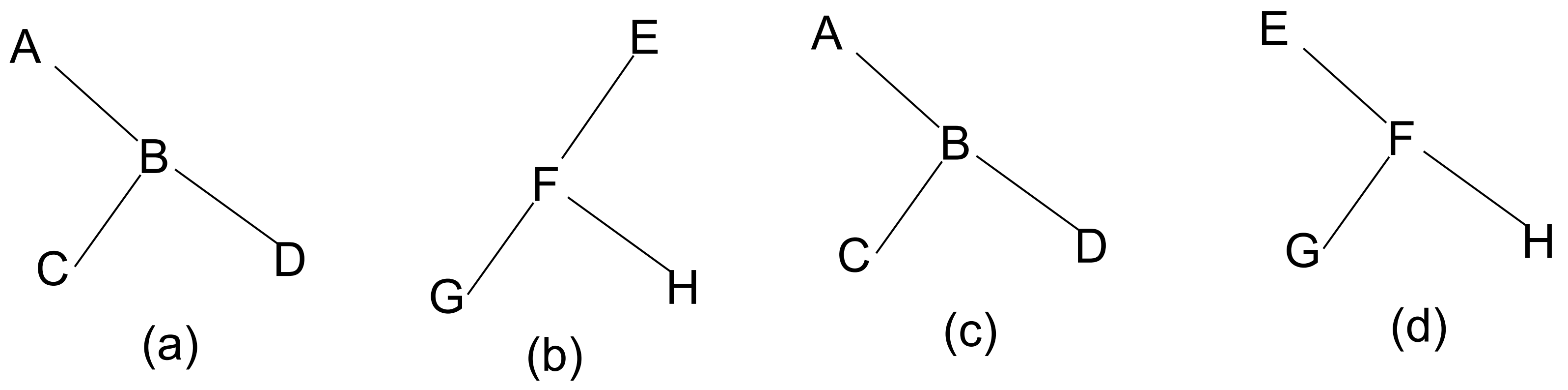


Figure: Binary tree

General Tree

- A general tree (tree) is defined to be a nonempty finite set of elements, called nodes, such that:

- ❖ T contains a distinguished element R, called the root of T
- ❖ The remaining elements of T form an ordered collection of zero or more disjoint trees T_1, T_2, …….., T_m.
- ❖ Trees T_1, T_2, …….., T_m are called subtrees of the root R, and the roots of T_1, T_2, …….., T_m are called successors of R.

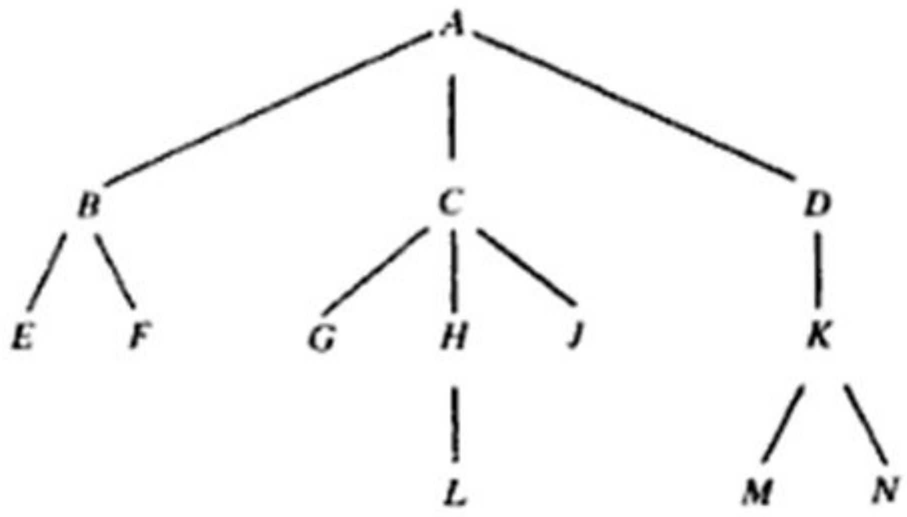


Figure: A general tree with 13 nodes

General Tree

- In a general tree, every node has a "parent" node and 0 or more "child" nodes, except for the root node which has no parent.
- General trees are those in which the number of subtrees for any node is not required to be 0, 1, or 2.
- The tree may be highly structured and therefore may have 3 subtrees per node in which case it is called a ternary tree.
- However, it is often the case that the number of subtrees for any node may be variable. Some nodes may have 1 or no subtrees, others may have 3, some 4, or any other combination.

General Tree Vs. Binary Tree

- In a general tree, every node has a "parent" node and 0 or more "child" nodes, except for the root node which has no parent.
- On the other hand, a binary tree is a tree with one more restriction: no node may have more than 2 children.
- In case of a general tree, it is obvious that most of the algorithms for searching, traversing, adding and deleting nodes become much more complex than those of binary tree in that they must now cope with situations where there are not just two possibilities for any node but multiple possibilities.
- Fortunately, a general tree can be converted to a binary tree. When converted, all nodes of a general tree will be same as the nodes of the binary tree.
- A binary tree can be empty , whereas the general tree cannot be empty.

Conversion of a General Tree into a Binary Tree

A general tree can be converted to a binary tree.

Process of converting a general tree to a binary tree is as follows:

1. Use the root of the general tree as the root of the binary tree.

2. Determine the left child of the root of the binary tree. This will be the leftmost node in the general tree at the next level. We don't need to determine the right child of the root of the binary tree.

3. Except the root node of the binary tree, continue finding the left child (if any) of each parent node. The leftmost child of a parent node in the general tree will be the left child of the parent node in the binary tree.

4. Except the root node of the binary tree, continue finding the right child (if any) of each parent node. The right child of each node in binary tree will be the next sibling of that node in the general tree.

Conversion of a General Tree into a Binary Tree

Example-1:

Given the following general tree.
Convert it to a binary tree.

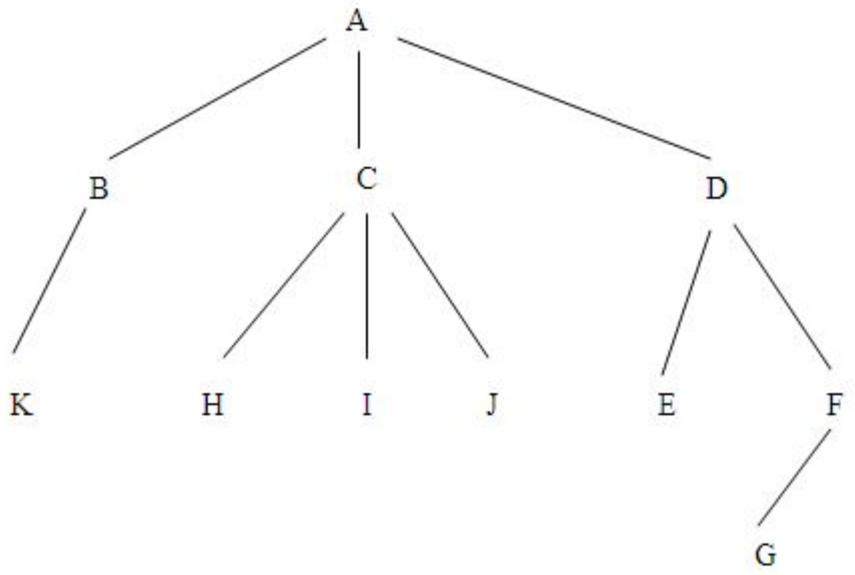


Figure: General tree

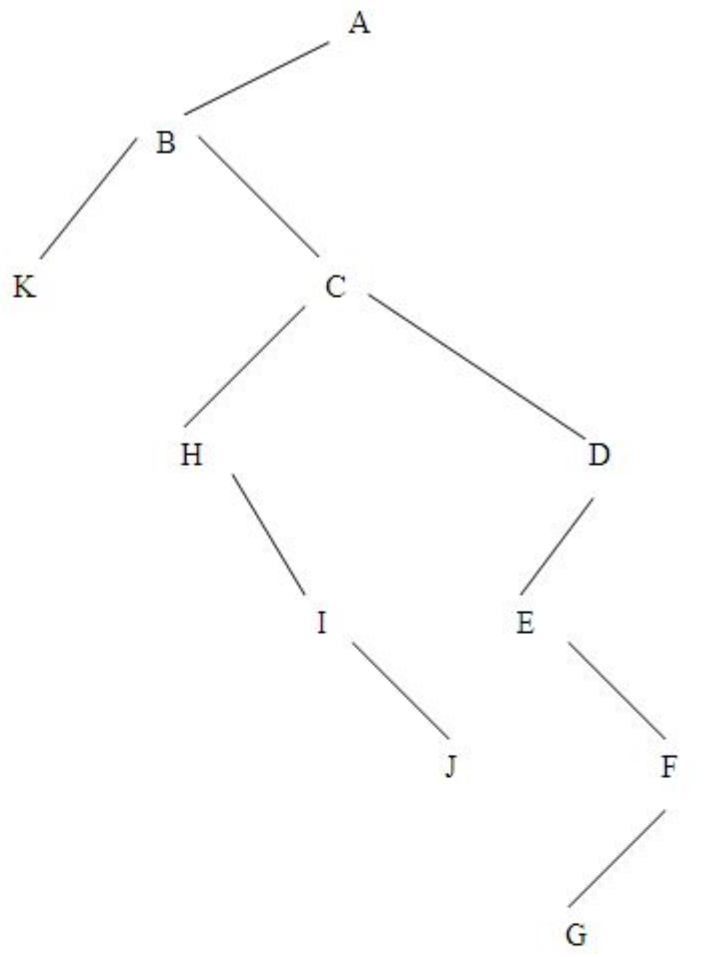


Figure: Binaryl tree

Conversion of a General Tree into a Binary Tree

Example-2:

Given the following general tree.
Convert it to a binary tree.

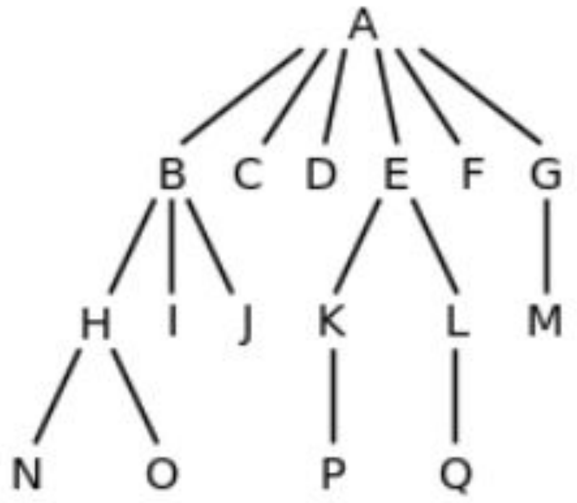


Figure: General tree

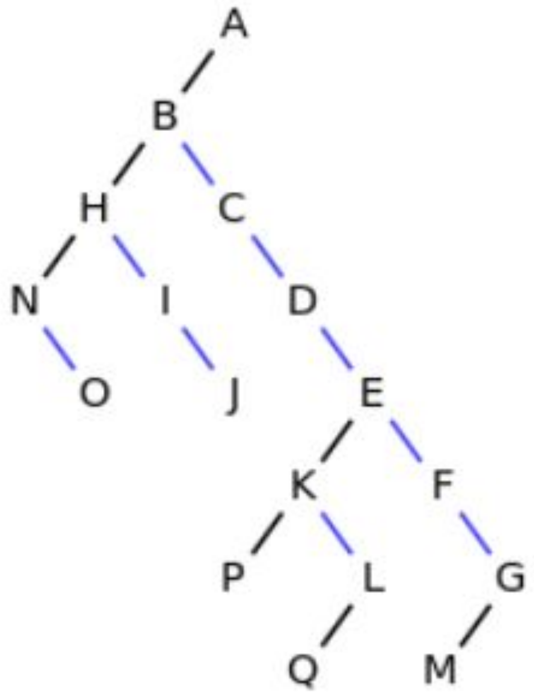
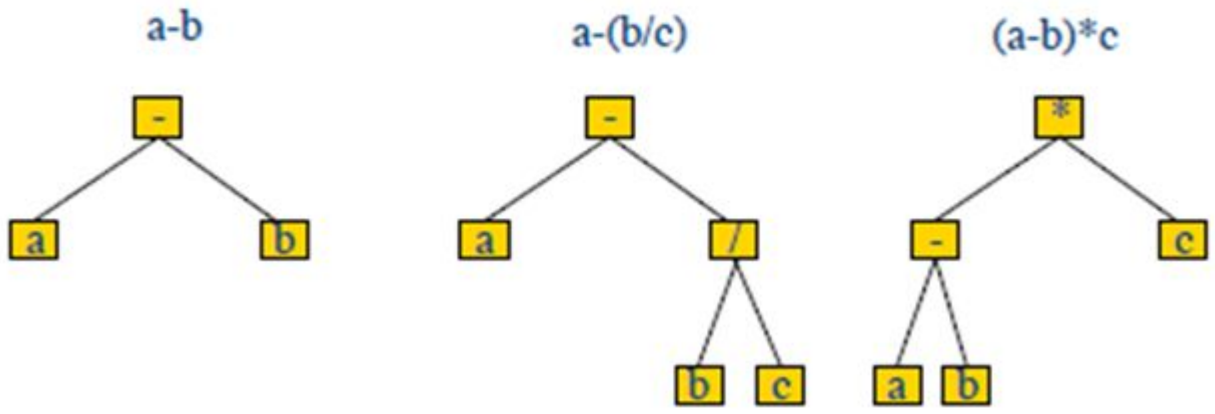


Figure: Binaryl tree

Note that black left edges represents the *first child* and the blue right edges represents *next sibling*.

Arithmetic Expression as Binary Tree

- Arithmetic expressions are often represented as binary trees.
- Every algebraic expression will correspond to a unique tree, and vice versa.
- ❖ Operators in the arithmetic expression are represented as internal nodes and operands are represented as external nodes or leaves.
- ❖ Precedence of the operator is enforced by the tree shape.



Arithmetic Expression as Binary Tree

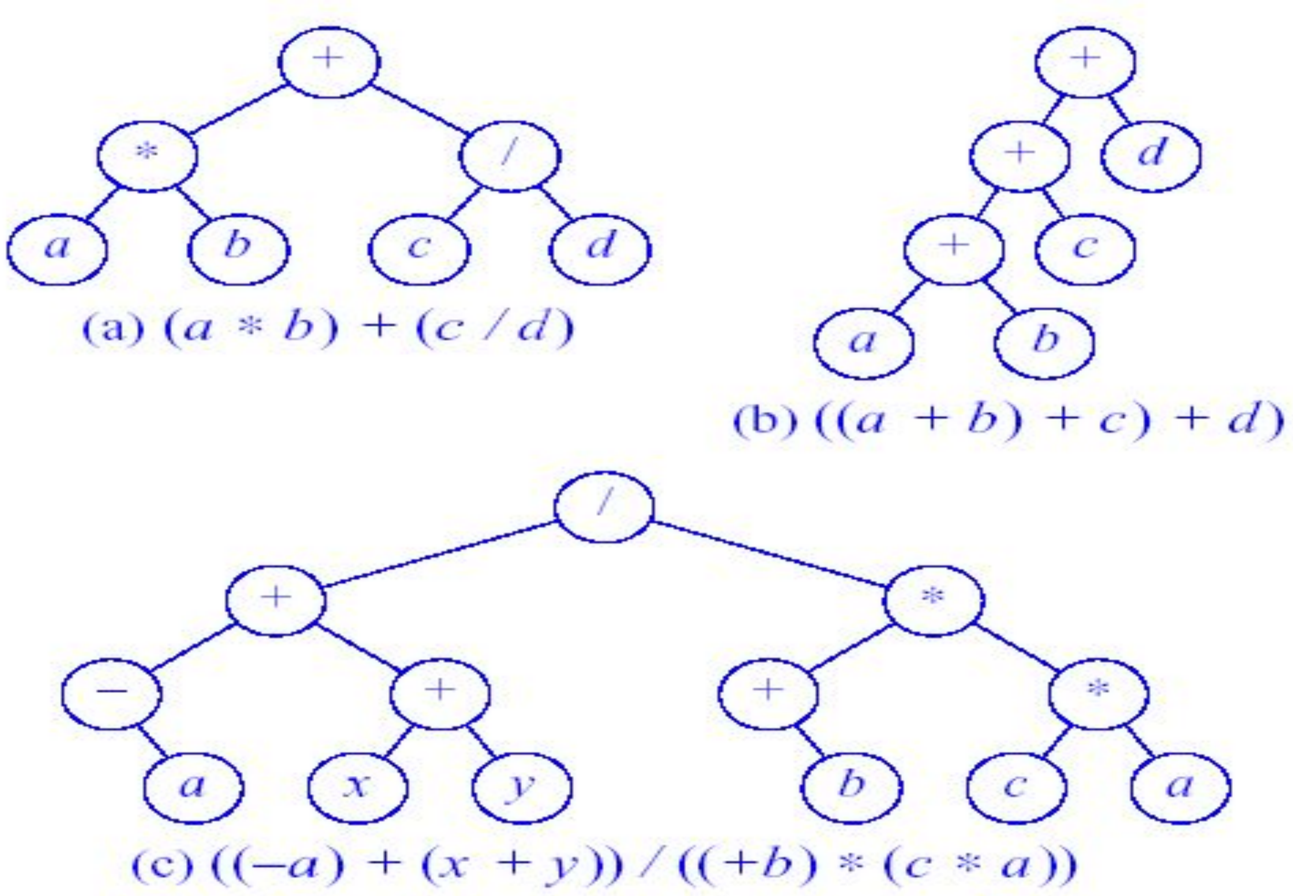


Figure: Binary trees representing arithmetic expressions

Arithmetic Expression as Binary Tree

Example: Algebraic Expressions

- Consider any algebraic expression E involving only binary operations, such as

  E = (a-b) / ((c * d) + e)

- E can be represented by means of the binary tree T pictured in figure below.
  - ❖ Operators in the arithmetic expression are represented as internal nodes and operands are represented as external nodes or leaves.

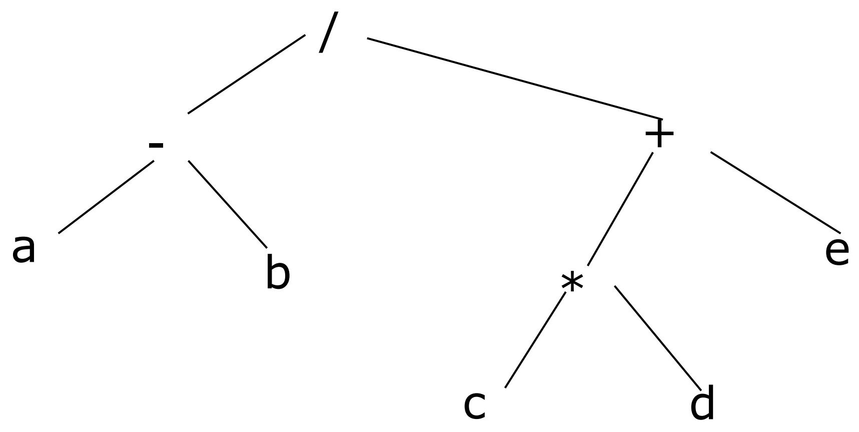


Fig: Binary tree representing E = (a – b) / ((c * d) + e)

Representing Binary Tree

- A binary tree T is frequently presented by means of a diagram as shown below.
- A binary tree can be easily maintained in the memory of a computer.
- The importance of a binary tree is that it can create a data structure that mimics a "yes/no" decision making process.
- ❖ For example, if you construct a binary tree to store numeric values such that each left sub-tree contains larger values and each right sub-tree contains smaller values then it is easy to search the tree for any particular value.

Complete Binary Tree

- The binary tree T is said to be complete if all its levels, except possibly the last, have the maximum number of possible nodes, and if all the nodes at the last level appear as far left as possible.
- Note that level r of T can have at most 2^r nodes.
- Figure below shows a complete binary tree T_{26} with 26 nodes.

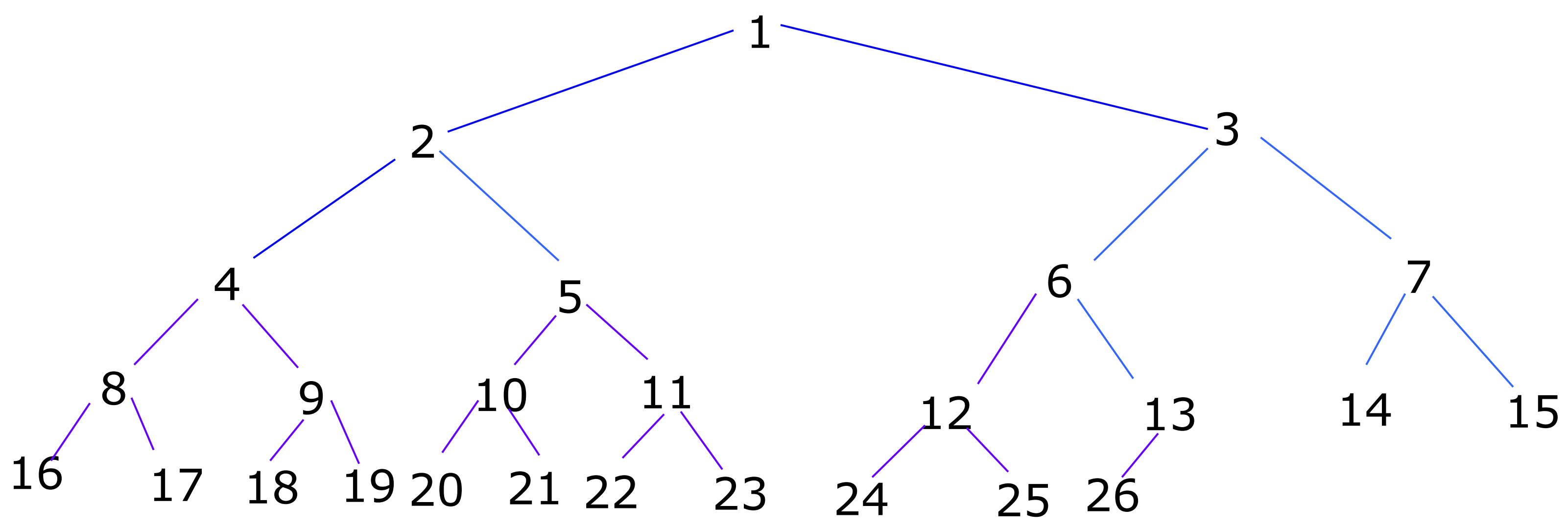


Figure: Complete tree T_{26}

Complete Binary Tree

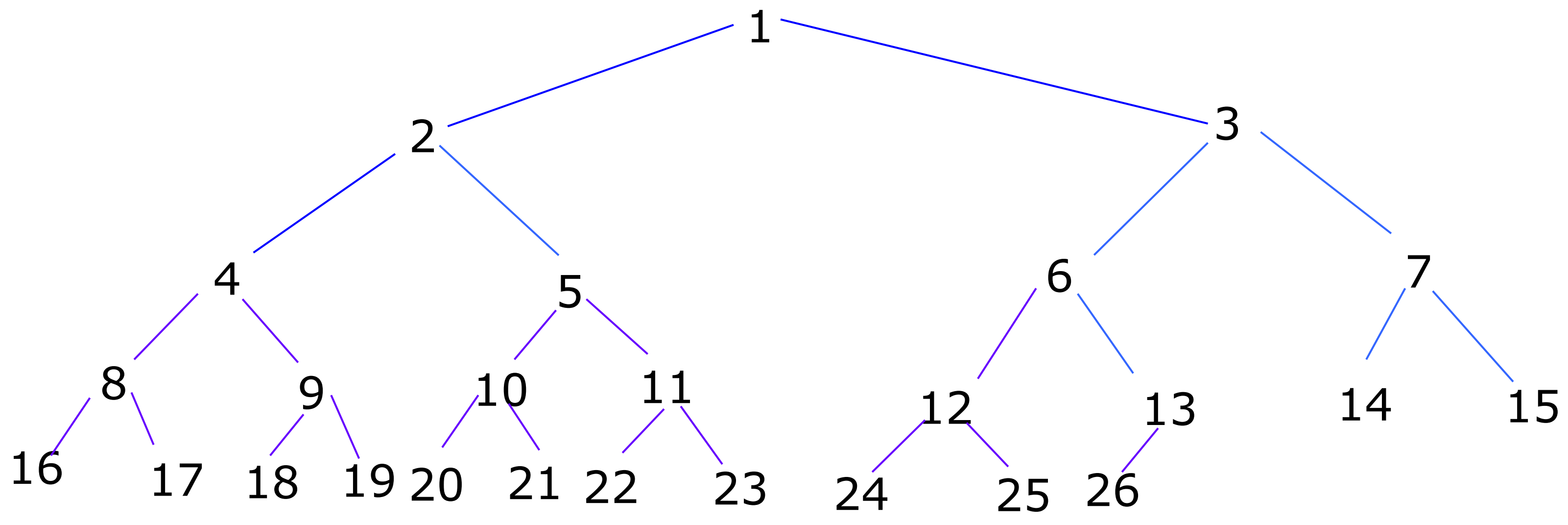


Figure: Complete tree T_{26}

- In a complete binary tree, the left and right children of the node K are 2 * K and 2 * K + 1 respectively, and the parent of K is the node [K/2].
- The children of node 9 are the nodes (2*9) = 18, (2*9+1) = 19, and its parent is the node [9/2] =4.
- The depth d_n of the complete tree T_n with n nodes is given by

$$D_n = [\log_2 n + 1]$$

- This is relatively small number. For example if the complete tree T_n has n =1000000 nodes, then its depth D_n =21. (Note that, the depth or height of a tree is the maximum number of nodes in a branch of the tree. This is 1 more than the largest level number of the tree.)

Complete Binary Tree

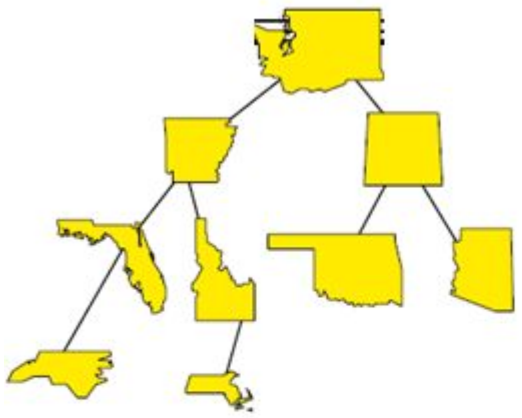
- When a complete binary tree is built, its nodes are generally added one at a time. As with any tree, its first node must be the root.
  - ❑ With a complete binary tree, the second node must be the left child of the root.
  - ❑ The next node (third node) must be the right child of the root.
- Following this rule, where would the fourth node have to be placed? The next nodes must always fill the next level from left to right.
- And so we continue, adding nodes.

- Therefore, in a complete binary tree, the way that we add nodes is restricted:
  - ❑ Nodes must completely fill each level from left-to-right before proceeding to the next level.

Complete Binary Tree

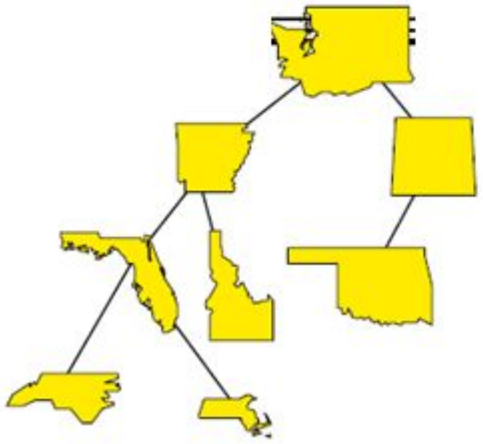
Just to check your understanding, is this binary tree a complete binary
tree?

No, it isn't since the nodes of the bottom level have not been added
from left to right.



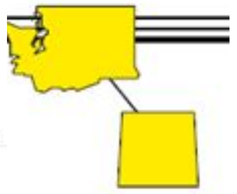
Is this complete?

No, since the third level was not completed before starting the fourth level.

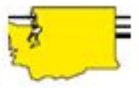


Is This Complete?

This is also not complete since has a right child but no left child.



Is This Complete?



But this binary tree is complete. It has only one node, the root

Extended Binary Tree or 2-Tree

- A binary tree T is said to be a 2–tree or an extended binary tree if each node N has either 0 or 2 children.
  - the nodes with two children are called internal nodes
  - the nodes with 0 children are called external nodes
- For distinguished purpose,
  - Circles are used for internal nodes.
  - Squares are used for external nodes.

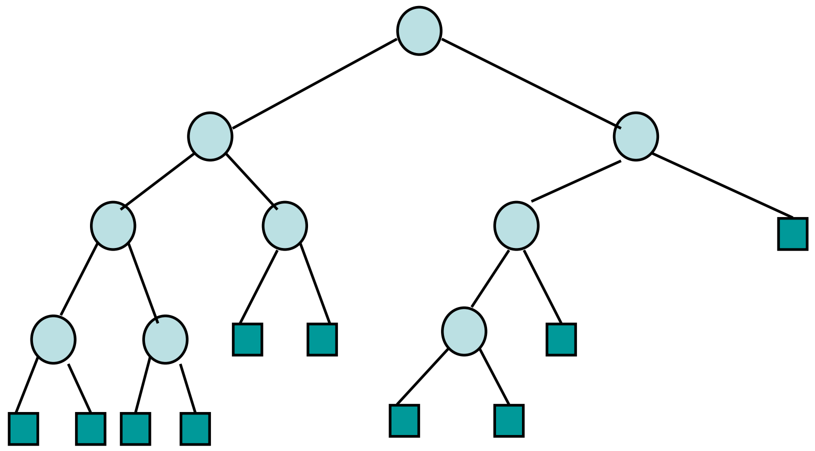


Figure: Extended binary tree

Extended Binary Tree or 2-Tree

Converting a Binary Tree into an Extended Binary Tree or 2- Tree

- Consider a binary tree shown in figure-A on the left.
- This binary tree can be converted into a 2-tree by adding an external node wherever there is an empty subtree. The resulting extended binary tree is shown in the figure-B on the right.

❖ Note that the nodes in the binary tree on the left are now the internal nodes in the extended binary tree, and added new nodes are the external nodes in the extended tree.

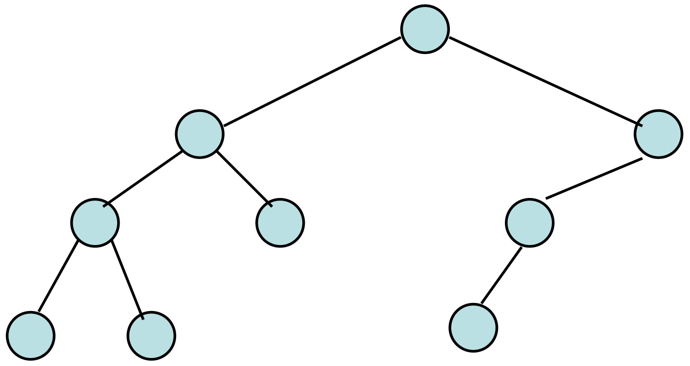


Figure-A: Binary Tree

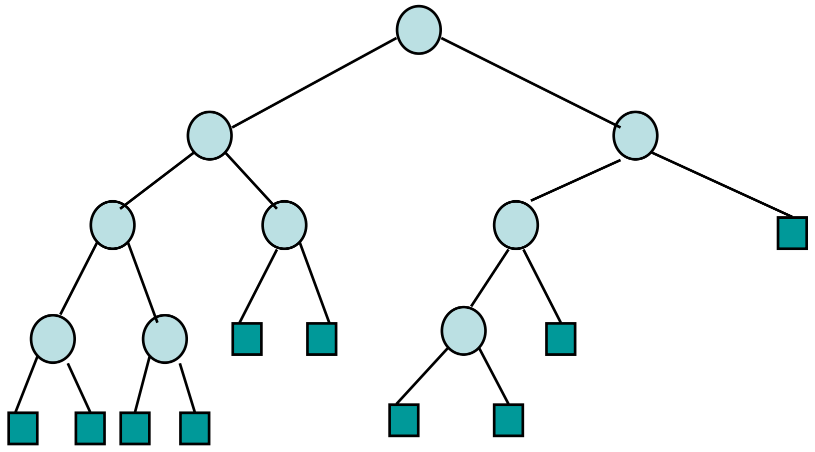


Figure-B: Extended Binary Tree

Representing Binary Tree in Memory

There are two ways of representing a binary tree in memory:

(1) Sequential or array representation

(2) Linked representation

Representing Binary Tree in Memory

Array Representation of Binary Trees:

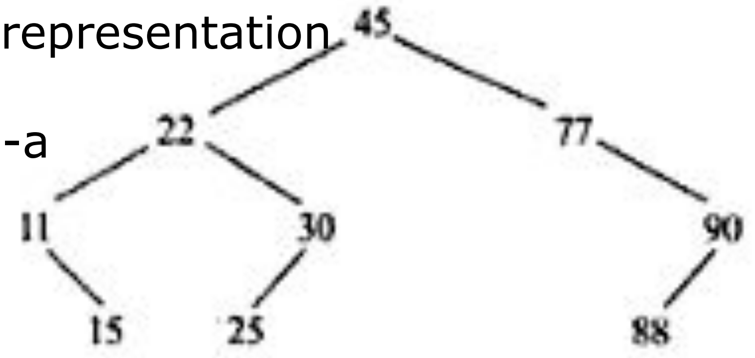
- **Sequential Representation** of a binary tree T uses only a single linear array TREE together with a pointer variable END as follows:

  (a) The root R of T is stored in TREE[1].

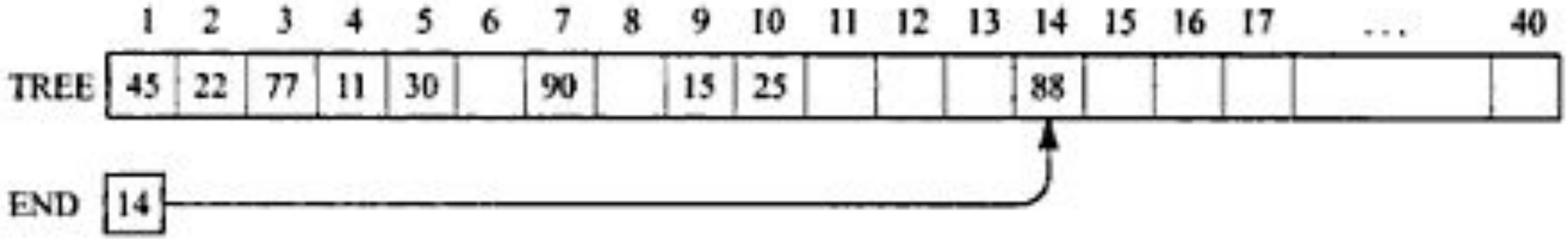
  (b) If a node occupies TREE[k], then its left child is stored in TREE[2 * K] and its right child is stored in TREE[2*k+1]

  (c) END contains the location of the last node of T.

  - The sequential representation of binary tree shown in figure-a is appeared in figure-b.



(a)



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	...	40
TREE	45	22	77	11	30		90		15	25				88					

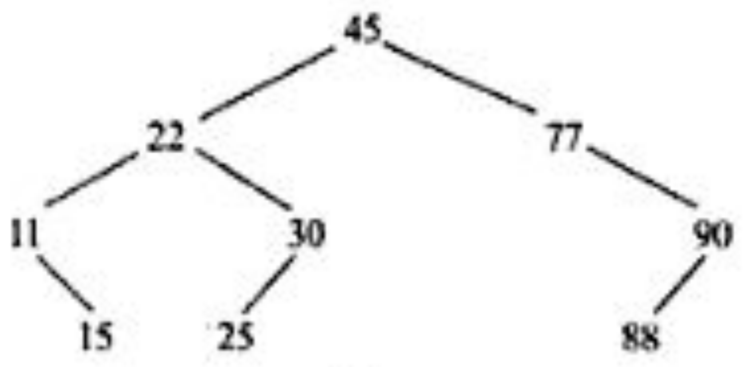
Representing Binary Tree in Memory

Array Representation of Binary Trees:

- For a binary tree with depth d, how many elements will require in the array when sequential representation is used?

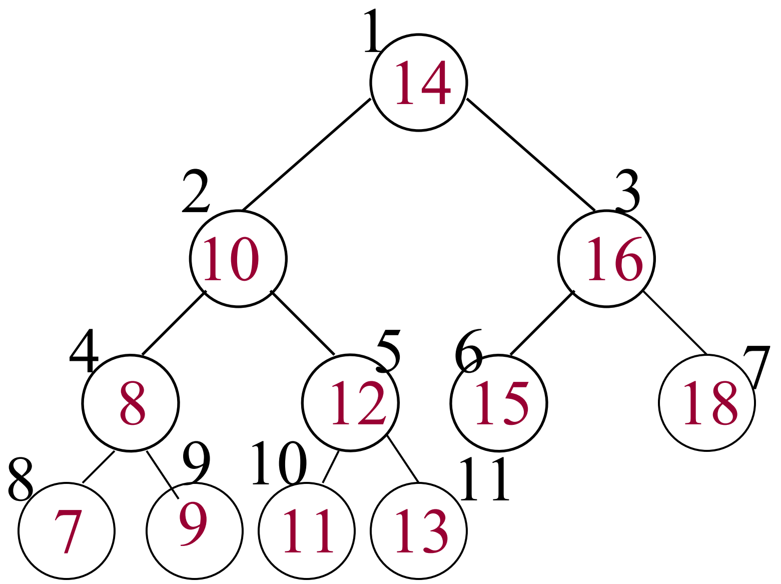
Ans. 2^{d+1}

- For example, the following tree has 9 nodes and depth 4, which means it would require an array TREE with approximately $2^{4+1} = 32$ locations if we represent the tree using sequential representation.
- So, this representation is inefficient unless the binary tree is complete or nearly complete.



Representing Binary Tree in Memory

Binary Trees: Array Representation



	1	2	3	4	5	6	7	8	9	10	11
Array A:	14	10	16	8	12	15	18	7	9	11	13

Representing Binary Tree in Memory

Link Representation of Binary Tree:

- This is the most popular way to present a binary tree in the memory of a computer.
- This representation uses three parallel arrays, INFO, LEFT and RIGHT, and a pointer variable ROOT.
- First of all, each node N of T will correspond to a location K such that:

1. INFO[K] contains the data at the node N
2. LEFT[K] contains the location of the left child of node N
3. RIGHT[K] contains the location of the right child of node N.

- The pointer variable ROOT will contain the location of the root R of T.
- If any subtree is empty, then the corresponding pointer will contain the null value.
- If the tree T itself is empty, then ROOT will contain the null value.

Linked Representation of Binary Tree in Memory

Example:

- Consider the binary tree shown right on the figure-A.
- A schematic diagram of the tree of the linked representation is shown in the figure-B below.
- Note that each node is pictured with three fields, and the empty subtrees are pictured by using X for the null entries.

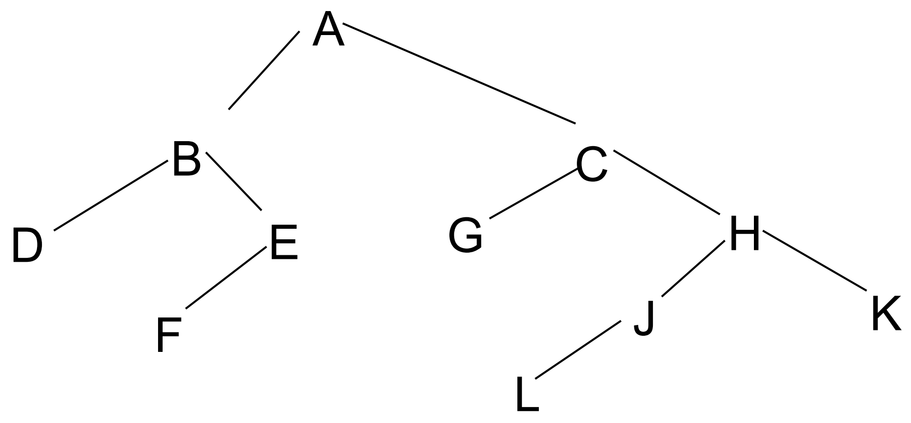


Figure-A: Binary tree

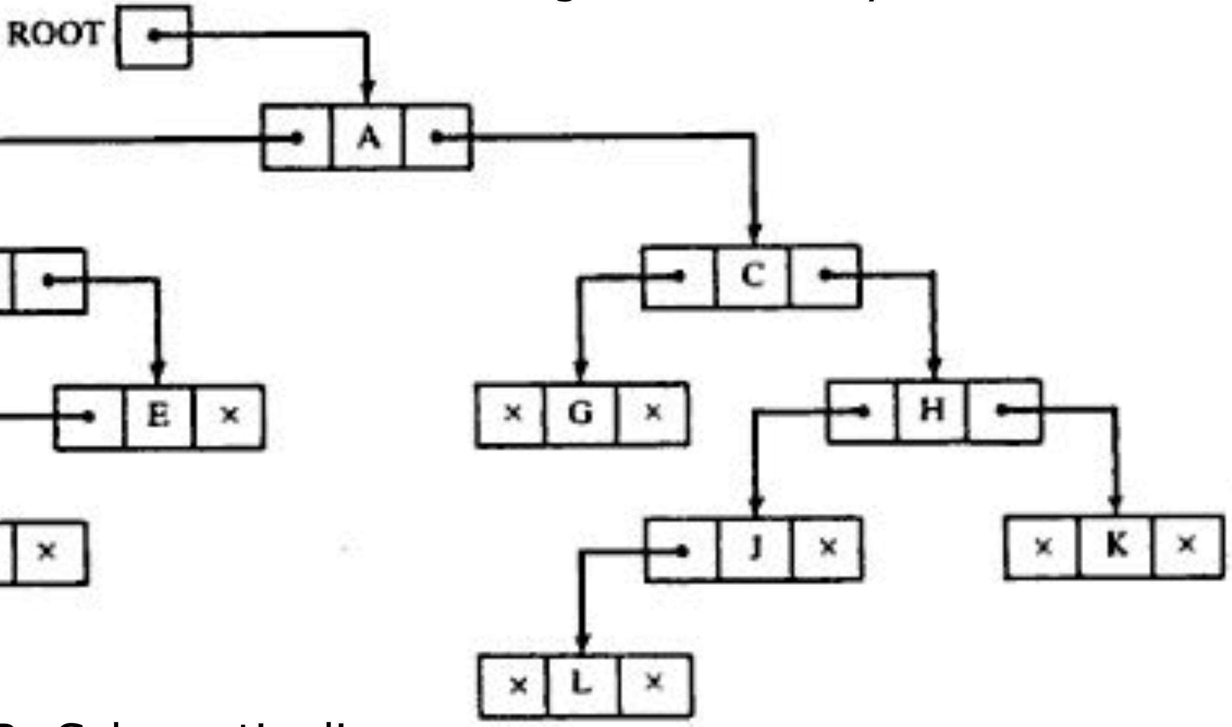
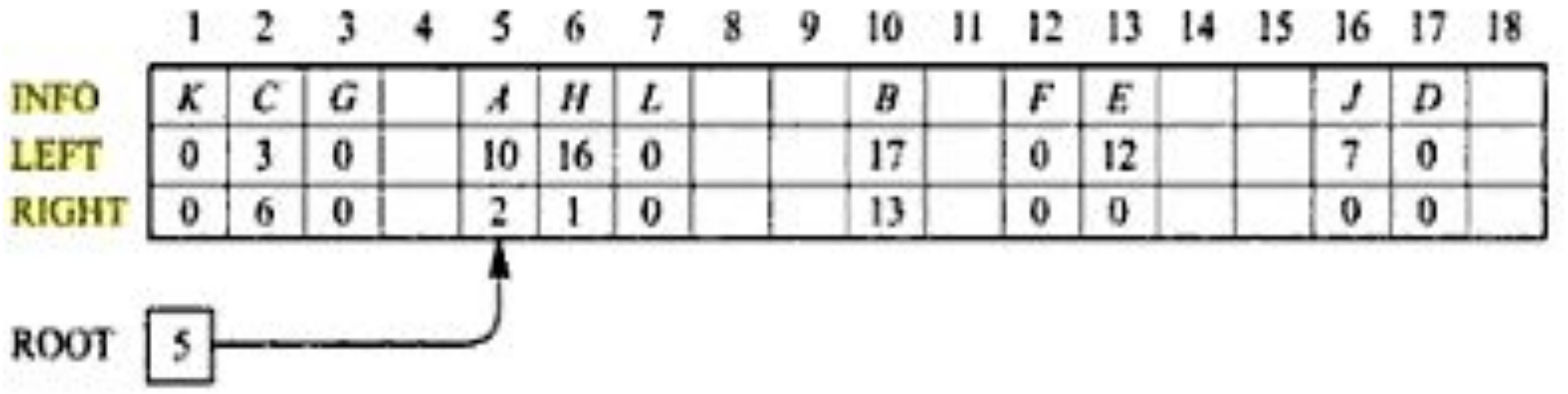


Figure-B: Schematic diagram

Linked Representation of Binary Tree in Memory

- The following figure shows how the above linked representation may appear in memory.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
INFO	*K*	*C*	*G*		*A*	*H*	*L*			*B*		*F*	*E*			*J*	*D*	
LEFT	0	3	0		10	16	0			17		0	12			7	0	
RIGHT	0	6	0		2	1	0			13		0	0			0	0	

ROOT = 5 points to INFO(5) = A since A is the root of T.

LEFT[5] = 10 points to INFO[10] = B since B is the left child of A

RIGHT[5] = 2 points to INFO[2] = C since C is the right child of A.

The choice of 18 elements for the array is arbitrary.

Representing Binary Tree in Memory

Merits and Demerits of Sequential Representation of Binary Tree

Advantages of linear representation:

1. Simplicity.
2. Given the location of the child (say, k), the location of the parent is easy to determine (k / 2).

Disadvantages of linear representation:

1. Additions and deletions of nodes are inefficient, because of the data movements in the array.
2. Space is wasted if the binary tree is not complete. That is, the linear representation is useful if the number of missing nodes is small.

- Linear representation of a binary tree can be implemented by means of a linked list instead of an array.
- This way the above mentioned disadvantages of the linear representation is resolved.