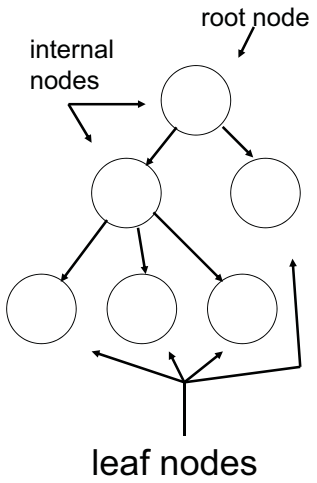
Topic 17
Introduction to Trees

"A tree may grow a thousand feet tall, but its leaves will return to its roots."

-Chinese Proverb

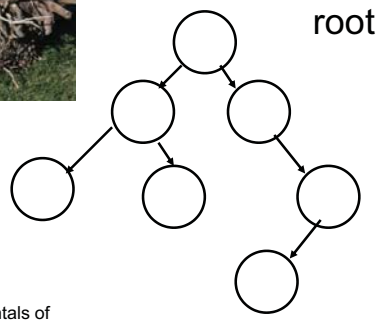
Definitions

- A *tree* is an abstract data type
  - one entry point, the ***root***
  - Each node is either a ***leaf*** or an *internal node*
  - An internal node has 1 or more ***children***, nodes that can be reached directly from that internal node.
  - The internal node is said to be the ***parent*** of its child nodes



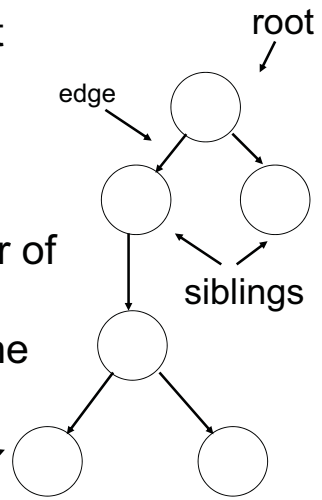
Properties of Trees

- Only access point is the root
- All nodes, except the root, have one parent
  - like the inheritance hierarchy in Java
- Traditionally trees drawn upside down



Properties of Trees and Nodes

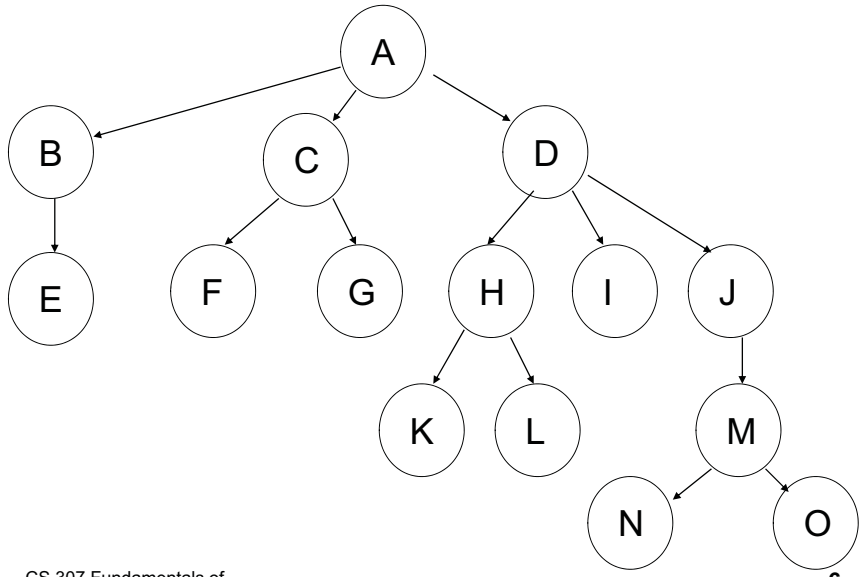
- *siblings:* two nodes that have the same parent
- *edge:* the link from one node to another
- *path length:* the number of edges that must be traversed to get from one node to another



More Properties of Trees

- *depth:* the path length from the root of the tree to this node
- *height of a node:* The maximum distance (path length) of any leaf from this node
  - a leaf has a height of 0
  - the height of a tree is the height of the root of that tree
- *descendants*: any nodes that can be reached via 1 or more edges from this node
- *ancestors:* any nodes for which this node is a descendant

Tree Visualization



Attendance Question 1

- What is the depth of the node that contains M on the previous slide?

A. -1

B. 0

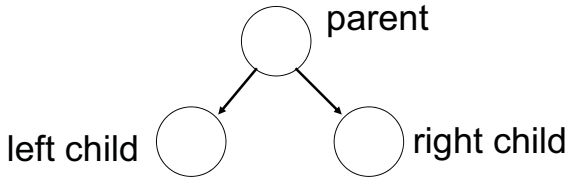
C. 1

D. 2

E. 3

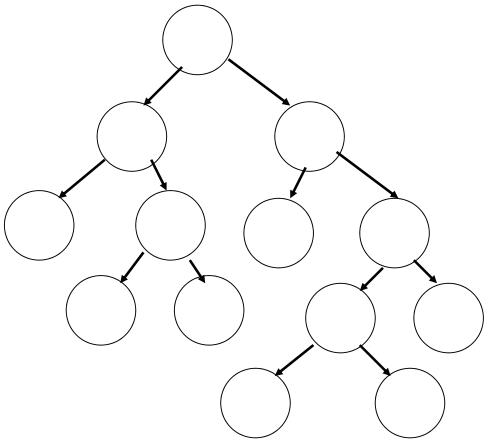
Binary Trees

- There are many variations on trees but we will work with *binary trees*
- *binary tree:* a tree with at most two children for each node
  - the possible children are normally referred to as the left and right child



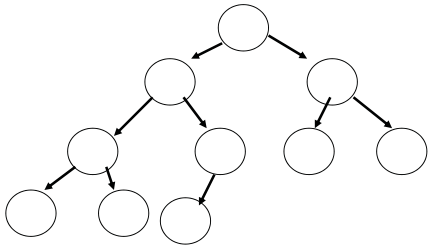
Full Binary Tree

- *full binary tree:* a binary tree is which each node was exactly 2 or 0 children



Complete Binary Tree

- *complete binary tree:* a binary tree in which every level, except possibly the deepest is completely filled. At depth n, the height of the tree, all nodes are as far left as possible



Where would the next node go to maintain a complete tree?

Perfect Binary Tree

- *perfect binary tree:* a binary tree with all leaf nodes at the same depth. All internal nodes have exactly two children.
- a perfect binary tree has the maximum number of nodes for a given height
- a perfect binary tree has $2^{(n+1)} - 1$ nodes where n is the height of a tree
  - height = 0 -> 1 node
  - height = 1 -> 3 nodes
  - height = 2 -> 7 nodes
  - height = 3 -> 15 nodes

A Binary Node class

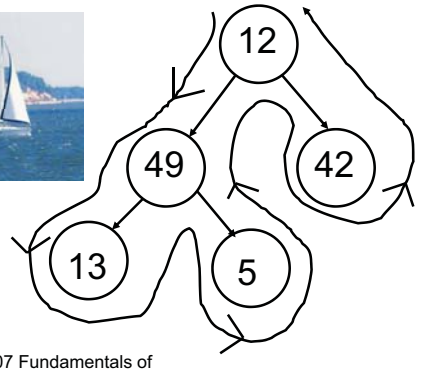
```
public class BNode
{    private Object myData;
     private BNode myLeft;
     private BNode myRight;

     public BNode();
     public BNode(Object data, BNode left,
          BNode right)
     public Object getData()
     public BNode getLeft()
     public BNode getRight()

     public void setData(Object data)
     public void setLeft(BNode left)
     public void setRight(BNode right)
}
```

Binary Tree Traversals

- Many algorithms require all nodes of a binary tree be visited and the contents of each node processed.
- There are 4 traditional types of traversals
  - preorder traversal: process the root, then process all sub trees (left to right)
  - in order traversal: process the left sub tree, process the root, process the right sub tree
  - post order traversal: process the left sub tree, process the right sub tree, then process the root
  - level order traversal: starting from the root of a tree, process all nodes at the same depth from left to right, then proceed to the nodes at the next depth.

Results of Traversals

- To determine the results of a traversal on a given tree draw a path around the tree.
  - start on the left side of the root and trace around the tree. The path should stay close to the tree.

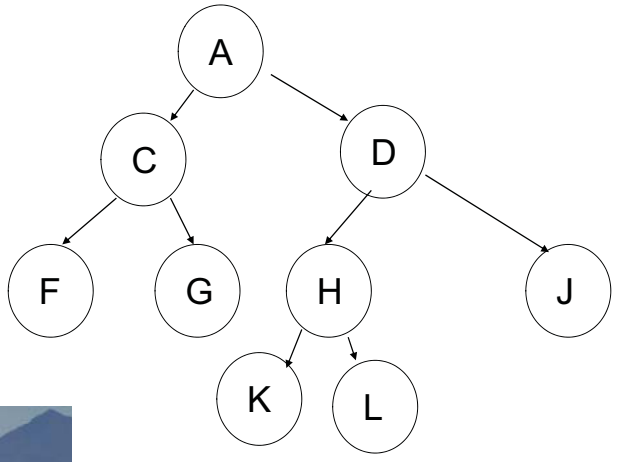


pre order: process when pass down left side of node
12 49 13 5 42

in order: process when pass underneath node
13 49 5 12 42

post order: process when pass up right side of node
13 5 49 42 12

Tree Traversals



Attendance Question 2

- What is a the result of a post order traversal of the tree on the previous slide?

A. F C G A K H L D J
B. F G C K L H J D A
C. A C F G D H K L J
D. A C D F G H J K L
E. L K J H G F D C A

Implement Traversals

- Implement preorder, inorder, and post order traversal
  - Big O time and space?
- Implement a level order traversal using a queue
  - Big O time and space?
- Implement a level order traversal without a queue
  - target depth
- Different kinds of Iterators for traversals?