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

-Chinese Proverb
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

The path length from the root to this node is 3.
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

B

C

D

E

F

G

H

I

J

K

L

M

N

O

A

CS314 Binary Trees
Clicker Question 1

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

A. 0  
B. 1  
C. 2  
D. 3  
E. 4
There are many variations on trees but we will start with *binary trees*.

A *binary tree*: each node has at most two children.

- The possible children are usually referred to as the left child and the right child.
Full Binary Tree

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

- What is the maximum height of a full binary tree with 11 nodes?

A. 3
B. 5
C. 7
D. 10
E. 11
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?
Clicker Question 3

- What is the height of a complete binary tree that contains N nodes?

A. O(1)  
B. O(logN)  
C. O(N^{1/2})  
D. O(N)  
E. O(N\log N)
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 the tree
  - height = 0 -> 1 node
  - height = 1 -> 3 nodes
  - height = 2 -> 7 nodes
  - height = 3 -> 15 nodes
public class Bnode <E> {
    private E myData;
    private Bnode<E> myLeft;
    private Bnode<E> myRight;

    public BNode();
    public BNode(E data, Bnode<E> left, Bnode<E> right)
    public E getData()
    public Bnode<E> getLeft()
    public Bnode<E> getRight()

    public void setData(E data)
    public void setLeft(Bnode<E> left)
    public void setRight(Bnode<E> right)
}
Binary Tree Traversals

- Many algorithms require all nodes of a binary tree be visited and the contents of each node processed or examined.

- 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 |
What is the result of a post order traversal of the tree to the left?

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
Breadth First - Depth First

- from NIST - DADS

- **breadth first search:** Any search algorithm that considers neighbors of a *vertex* (node), that is, outgoing *edges* (links) of the vertex's predecessor in the search, before any outgoing edges of the vertex.

- **depth first search:** Any search algorithm that considers outgoing *edges* (links of *children*) of a *vertex* (node) before any of the vertex's (node) *siblings*, that is, outgoing edges of the vertex's predecessor in the search. Extremes are searched first.
Breadth First

- A level order traversal of a tree could be used as a breadth first search
- Search all nodes in a level before going down to the next level
Breadth First Search of Tree
Breadth First Search

Find Node with B

search level 0 first
Breadth First Search

Find Node with B

search level 1 next

CS314 Binary Trees

23
Breadth First Search

Find Node with B

search level 2 next
Breadth First Search

Find Node with B

search level 3 next
Depth First Search

Find Node with B

Diagram of a binary tree with nodes labeled A to Z, demonstrating a depth-first search to find a node with the label B.
BFS - DFS

- Breadth first search typically implemented with a Queue
- Depth first search typically implemented with recursion
- which technique do I use?
  - depends on the problem