BinaryTree 2
Tree Terminology

A tree is a collection of nodes. A tree is either empty or it contains a node called the root that is linked to zero or more subtrees. The subtrees of a node T are called the children of T and T is called the parent of the subtrees.

A node with zero children is called a leaf.

A path in a tree is a sequence of nodes, \( n_0, n_1, \ldots, n_j \), where \( n_i \) is the parent of \( n_{i+1} \) for \( 0 \leq i < j \).

The length of a path is one less than the number of nodes in the path (this is equivalent to saying the length is the number of edges in the path).

The height of a tree is the length of the longest path from the root to a leaf. The height of an empty tree is -1. The height of a tree with only one node (the root) is 0.

The depth of a node, \( n \), is the length of the path from the root to \( n \).
**Tree Terminology**

A **preorder** traversal of a tree rooted at node \( n \), is a traversal where node \( n \) is visited (the meaning of visited will vary based on the purpose of the traversal) followed by a preorder traversal of each child \( T \).

A **postorder** traversal of a tree rooted at node \( n \), is a traversal where a postorder traversal of each child of \( n \) is completed followed by a visit to \( n \).

A **level** order traversal of a tree rooted at node \( n \), is a traversal where the nodes in the tree are visited based on the depth of the node: \( n \) is visited, followed by all nodes at depth 1, followed by all nodes at depth 2, ... until all the nodes at a depth equal to the height of the tree have been visited. A level order traversal is sometimes called a **breadth first** traversal.

A **binary tree** is a tree where each node has at most 2 children.

An **inorder** traversal of a binary tree rooted at node \( n \), is a traversal where an inorder traversal of the left child is completed followed by a visit to \( n \) followed by an inorder traversal of the right child.
A **full binary tree** is a binary tree with the maximum number of nodes at each level (or depth).

A **complete binary tree** is a binary tree with the maximum number of nodes at each level except the deepest level. The deepest level either has the maximum number of nodes or the nodes are pushed as far left as possible.
Binary Tree Questions

What is the maximum number of nodes in level (or depth) N of a binary tree?

What is the maximum number of nodes in a height N binary tree?

What is the minimum number of nodes in a height N binary tree?

What is the minimum number of nodes in a height N complete binary tree?
Binary Tree Questions

What is the minimum height of a binary tree with N nodes?

What is the maximum height of a binary tree with N nodes?

What is the height of a complete binary tree with N nodes?
public class BinaryTree {
    private class Node {
        private Node left;
        private int data;
        private Node right;

        private Node(Node L, int d, Node r) {
            left = L;
            data = d;
            right = r;
        }
    }
    private Node root;
}
Binary Tree Constructors

```java
public BinaryTree() {
}

public BinaryTree(int d) {
}

public BinaryTree(BinaryTree b1, int d, BinaryTree b2) {
}
```
Inorder Traversal
Print node values in an in order Pattern

public void inorder() {
}

private void inorder(Node r) {
}
String Representation of a Binary Tree

Empty Binary Tree

Non-empty Binary Tree (nodeValue BinaryTree BinaryTree)

Examples

( 10 ! ! )

( 10 ( 20 ( 30 ! ! )! )! )

( 10 ! ( 5 ! ( 1 ! ! ) ) )

( 10 ( 2 ! ! ) ( 4 ! ! ) )

( 10 ( 5 ( 33 ! ( 99 ! ! ) ) ! ) ( 12 ! ( 77 ! ! ) ) )
public BinaryTree(String t) {
}