Trees

A rooted tree $T$ is a tree in which some node is distinguished as a root. It consists of one or more nodes. The remaining set of zero or more nodes denoted $T_1, T_2, \ldots, T_k$ are themselves trees.

- Consider node $x$ in a rooted tree $T$ with root $r$. Any node $y$ on the unique path from $r$ to $x$ is called an ancestor of $x$. If $y$ is an ancestor of $x$, then $x$ is a descendant of $y$. The subtree rooted at $x$ is the tree induced by descendants of $x$.

- Parent, child, siblings are easily understood.

- A leaf node has no child. A nonleaf node is an internal node.

- The length of the path from $r$ to $x$ is the depth of $x$.

- The height of $x$ is the number of edges on the longest path to a leaf.

Subtle points: Non empty set of nodes, ordered tree $T$ (order among $T_i$)
Linked Tree Implementation

Degree of a node in a tree is the number of children

Trees (of unbounded degree) may be implemented using the `firstChild` and `nextSibling` representation

```java
class TreeNode {
    Object element;
    TreeNode left;
    TreeNode right;
}
```
Binary Trees

A *binary tree* $T$ is a structure defined on a finite set of nodes that either

- contains no nodes
- is composed of three disjoint sets of nodes: a root node, a binary tree called its left subtree, and a binary tree called its right subtree

Left child: Root of the nonempty left subtree
NIL: The binary tree that contains no nodes If a subtree is NIL, the child is *absent*
If nodes at every depth are present, $T$ is called a complete binary tree. (Exception at the last level, present from left to right.)
Implementation: An array (as opposed to a linked structure).
More on binary trees

- A binary tree can be empty
- A binary tree is not just an ordered tree in which every node has at most two children (Distinction between a binary tree with one left child versus a binary tree with one right child)
- Replace nodes with one missing child by a new node. Resulting tree is a *proper* binary tree
- Replace every NIL node with a new node. Resulting tree is an external or full binary tree.

Now number of leaves is one more than the number of internal nodes
On exploration, we can tell where we fell off the tree
Tree Traversals

- A *traversal* of a tree is a process that *visits* the nodes in the tree a fixed number of times (for example, once).
- Common tree traversals: preorder and postorder
- For binary trees, we have, in addition, inorder and the Euler tour
Recursion Aids Tree Traversal

preorder(v) {
  if (v != null) {
    visit(v);
    for child w of v do
      preorder(w)
  }
}

postorder(v) {
  if (v != null) {
    for child w of v do
      postorder(w);
    visit(v);
  }
}

Reading a document from beginning to end

Collecting information from children
Euler Tour Traversal

• Generic traversal of a binary tree in the sense that the preorder, inorder, and postorder traversals are special cases

• Walk around the tree and visit each node three times. While looking at the node, the walk is either

  • – To the left
  • – Below
  • – To the right