Java 2 Trees

- Tree:
  - A collection in which the elements are arranged in a hierarchy.
  - A **list** is a one-dimensional structure because it defines linear relationships between elements: **predecessor, successor**
  - A **Tree** is a two-dimensional structure because it defines hierarchical relationships among elements: **parent, child**
  - A tree is composed of nodes and branches (also known as edges)
    - nodes are places in the tree where the elements are stored
    - branches are connections between nodes, from parent to child.
    - **The terms "nodes" and "branches" are abstract and do not imply a particular implementation**
    - A **parent node** has one or more **children**.
    - A **leaf node** has no **children**
    - A **child node** has exactly one **parent**
    - The **root node** has no **parent**
    - The **order** of a tree is an integer ≥ 2 that represents the upper limit on the number of children that any node can have.
      - Order = 2 is a Binary Tree
      - Order = 3 Ternary Tree
      - General Tree = a tree with no specified order
    - **Path** is a sequence of node from one to another node, going from parent to child.
      - *Path from A to J = A-B-E-J*
      - Has to be unique because every node has exactly one parent.
    - **Branches** are "one way streets"
    - **Path Length -** The number of nodes on the path
      - *Path from A to J is 4*
      - This length is inclusive
    - **Ancestor -** Node X is an ancestor of Node Y iff there is a path from X to Y
    - **Descendant -** Node X is a descendant of node Y iff there is a path from Y to X.
    - A Path is sometimes defines as a sequence of edges (branches) instead of nodes. In this class we define them as a sequence of nodes and path length is counted by nodes.
    - **Subtree -** A tree within a larger tree, rooted at a given Node X. The subtree consists of X and all descendants of X.
      - There as many subtrees as there are nodes in the tree.
      - This will be handled recursively
      - The tree itself is a subtree
    - **Height** is a metric that is defined in terms of a given node, but is typically used to describe a tree or subtree.
      - When height is applied to a tree or subtree, it refer to the height of its root.
      - Height measures the distance of a given node from the "bottom" of the tree.
      - Height = length of the longest path from a given node to a descendant leaf.
      - Height depends on how path and path lengths are defined.
      - You will be off from one from the textbook.
    - **Depth** measures the distance of a given node from the "top" of the tree
      - Depth is the same concept of "level" in the textbook.
      - Depth = length of the path from the root of the tree to a given node.
      - Depth depends on how path and path are defined.
    - **Full -** A tree is full if all leaves have the same depth and every parent node has the maximum number of children.
    - **Complete -** A tree is complete if it is full to the next-to-last level, and the leaves on the lowest level are "left-justified".
    - A full or complete tree is the shortest possible that could store N nodes.
    - **Balanced -** A tree is balanced if for each node, its subtrees have similar heights. The term "similar" is intentionally vague since different balancing schemes exist.

- A balanced tree will have near optimal height