

Solutions

6.1 Tree Terminology

Reference example

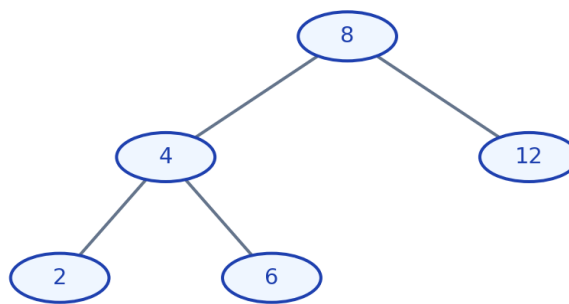
Root: 8

Left child: 4, with children 2 and 6

Right child: 12

Height convention: number of edges on longest root-to-leaf path

A binary search tree



The leaves are 2, 6, and 12. Root-to-leaf height is 2 edges.

6.1.1 - Explain

For the displayed tree, identify the leaves and the depth of node 6.

The leaves are 2, 6, and 12. Node 6 has depth 2, following edges 8 to 4 to 6.

6.1.2 - Explain

What are the tree height and the height of a leaf under the stated edge-count convention?

The tree height is 2. A leaf has height 0.

Solutions

6.2 Traversals

Reference example

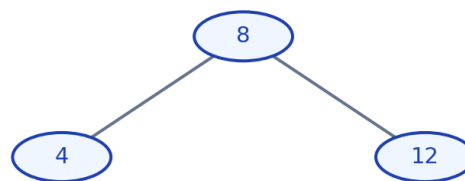
Tree: root 8; left 4; right 12

Preorder: node, left, right

Inorder: left, node, right

Postorder: left, right, node

A three-node tree



Visit order depends on whether the node is processed before, between, or after its subtrees.

6.2.1 - Explain

Give preorder, inorder, and postorder for root 8 with left child 4 and right child 12.

Preorder: 8, 4, 12. Inorder: 4, 8, 12. Postorder: 4, 12, 8.

Solutions

6.2.2 - Write code

Implement inorder traversal for a tree rooted at 8 with children 4 and 12. Print its keys.

Recursively process the left subtree, then the current node, then the right subtree.

```
class Node {
    int key; Node left, right;
    Node(int key) { this.key = key; }
}
class Walker {
    void inorder(Node node) {
        if (node == null) return;
        inorder(node.left);
        System.out.print(node.key + " ");
        inorder(node.right);
    }
}
Node root = new Node(8);
root.left = new Node(4); root.right = new Node(12);
new Walker().inorder(root);
```

Expected output

4 8 12

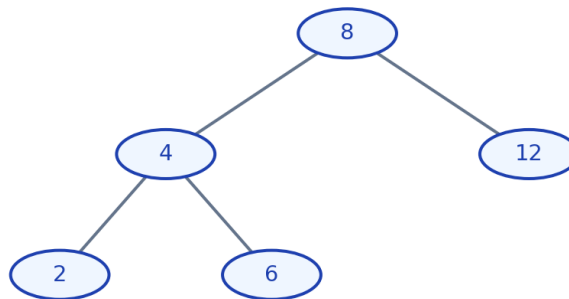
Solutions

6.3 Binary Search Trees

Reference example

```
TreeSet<Integer> keys = new TreeSet<Integer>();  
for (int k : new int[]{8,4,12,2,6}) keys.add(k);  
System.out.println(keys);
```

A binary search tree



The leaves are 2, 6, and 12. Root-to-leaf height is 2 edges.

6.3.1 - Explain

In a BST rooted at 8 with left child 4 and right child 12, where is a new key 10 inserted?

Compare with 8 and go right, then compare with 12 and go left. It becomes the left child of 12 if that position is empty.

6.3.2 - Explain

When removing a BST node with two children, what key can replace it while preserving sorted order?

Use its inorder successor, the smallest key in the right subtree, or its inorder predecessor. Then correctly remove or relink that replacement node.

Solutions

6.4 Balance and Degeneration

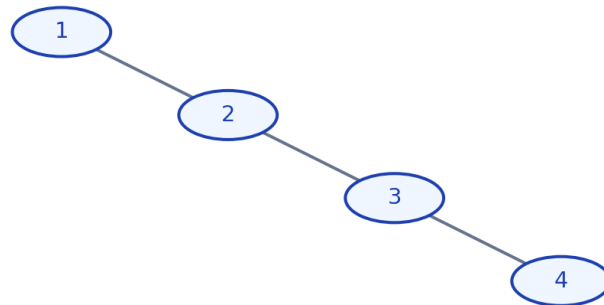
Reference example

Insertion order in an unbalanced BST: 1, 2, 3, 4

Shape: each key becomes the right child of its predecessor

Search for 4: visits every node

A degenerate binary search tree



Sorted insertion without balance can create linear-height search paths.

6.4.1 - Explain

How many nodes may a search for the last key visit in a chain of n nodes?

It may visit all n nodes, giving linear worst-case search time.

6.4.2 - Explain

Why does balance improve worst-case search without changing the BST ordering rule?

Balance limits height while preserving the smaller-left, larger-right ordering. A search therefore makes fewer sequential decisions.