Trees and Traversals

Weiss, from the beginning of chapter 19 through 19.3 (pages 687–706)

I am indebted to my colleagues, John David Stone and Peter-Michael Osera, for assistance with this handout.

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit https://creativecommons.org/licenses/by-sa/4.0/
or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.
Trees

• Tree has a hierarchical structure
  – Child-parent relationship

• A tree consists of nodes that each
  – has an element, a left sub-tree, and a right sub-tree
  – is connected by an edge from another node

• Binary tree

• View trees recursively
Tree Representation

class BinaryNode<AnyType> {
    private AnyType element;
    private BinaryNode<AnyType> left;
    private BinaryNode<AnyType> right;

    public BinaryNode ( AnyType element Node<AnyType> left, Node<AnyType> right ) {
        this.element = element;
        this.left = left;
        this.right = right;
    }
}

public class BinaryTree<AnyType> {
    private BinaryNode<AnyType> root;
    public BinaryTree( ) { root = null; }
    public BinaryTree( AnyType rootElement ) {
        root = new BinaryNode<AnyType>( rootElement, null, null );
    }
}
Recursion in Tree

• Size of a node is the number of descendents of the node (including the node)
• Size of a tree is the size of tree root

```java
public static <AnyType> int size( BinaryNode<AnyType> t ) {
    if ( t == null ) {
        return 0;
    }
    return 1 + size( t.left ) + size( t.right );
}
```
Binary Search Tree

• Order property
  – Elements in the left sub-tree are < the value and elements in the right sub-tree are > the value

• Access an element by following node links according to the order property

• Find minimum and maximum elements
Tree Traversal

• Compute the size of tree by visiting nodes in an order
• Show String representation of elements stored in nodes
  – For an empty tree, print empty string
  – For a non-empty tree, show string representation of node value, followed by string representation of its left and right sub-trees in order
Tree Traversal (cont.)

Pre-order traversal: show node value, process left sub-tree, and process right sub-tree

[5, 2, 1, 3, 8, 7, 6, 9, 10 ]

In-order traversal: process left sub-tree, show node value, and process right sub-tree

Post-order traversal: process left sub-tree, process right sub-tree, and show node value
Running time

- Lookup 4 in the binary search tree
- Running time of look up depends on the length of a path (number of edges) from root node to the target node

\[ O(N) \quad O(\log N) \]