# **CSE 214**

# **Recitation 06 – Binary Trees**

1. **(5 minutes)** Consider the following tree:

Shape, circle

Description automatically generated

Pre: Current Node, Left Node, Right Node

In: Left Node, Current Node, Right Node

Post: Left Node, Right Node, Current Node

* 1. What is the in-order traversal of this tree?

5-3-6-1-4-0-2-7-8

* 1. What is the pre-order traversal of this tree?

0-1-3-5-6-4-2-7-

* 1. What is the post-order traversal of this tree?

5-6-3-4-1-7-8-2-0

3. **(5 minutes)** Suppose we have the following results from performing a preorder traversal on a

binary search tree containing numeric keys

**preorder**: 45, 32, 11, 29, 64, 50

Construct a BST that is consistent with the above traversal

3. (**5 minutes**) Given the following traversals, construct the general binary tree and list the post-order traversal.

**Inorder sequence: D B E A F C   
Preorder sequence: A B D E C F**

A

B C

D E F

4.  **(10 minutes)** Complete vs. Full Binary Trees

* 1. A complete binary tree is a binary tree in which every level of the tree has the maximum number of nodes possible except possibly the deepest level. Also, at the deepest level, the nodes are as far left as possible. What are the formulas for the maximum and minimum number of nodes and internal nodes in a complete binary tree of height h.
  2. What is the minimum and maximum number of nodes in a complete binary tree with a height of 4? What is the number of internal nodes?

* 1. A full binary tree is a binary tree in which every non-leaf node has 2 children, and all leaves have the same depth. What is the minimum and maximum number of nodes in a binary tree with a height of 4? What is the number of internal nodes? What is a full binary tree’s relation to a complete binary tree?

5. **(15 minutes)** Assume we have the following declared:

public class TreeNode {

int val;

TreeNode left;

TreeNode right;

TreeNode() {}

TreeNode(int val) { this.val = val; }

TreeNode(int val, TreeNode left, TreeNode right){

this.val = val;

this.left = left;

this.right = right;

}

}

Given a root node, write function(s) that allow us to validate whether the Tree is a valid BST

6. **(15 minutes)** Assume we have the following declared:

public class TreeNode {

int val;

TreeNode left;

TreeNode right;

TreeNode{}

TreeNode(int val) { this.val = val; }

TreeNode(int val, TreeNode left, TreeNode right){

this.val = val;

this.left = left;

this.right = right;

}

}

Given a sorted array, write function(s) to convert that array into a BST (return the root node).