Data Structures

CSCI-UA-102-011-Spring-2025

Recitation - 6

- Rasmika Billa
  (02/28/2025)

Today’s Agenda

- Q8.5 with a recursive algorithm
- Problem Statements
- Quiz (Last 20 mins)

8.5

```
public class LeftLeafCounter {

public static int countLeftLeaves(TreeNode root, boolean isLeft) {
if (root == null) {
return 0;

if (isLeft && root.left == null && root.right == null) {
return 1;
}
return countLeftLeaves(root.left,true)+ countLeftLeaves(root.right,false ;

public static int countLeftLeaves(TreeNode root) {
return countLeftLeaves(root, false);
}
public static TreeNode buildTree(Integer[] values, int index) {
if (index >= values.length || values[index] == null) {
return null;
}
TreeNode node = new TreeNode(values[index]);
node.left = buildTree(values, 2 * index + 1);
node.right = buildTree(values, 2 * index + 2);

return node;

public static void main(String[] args) {
// Example tree:
// 1
// / \
// 2 3
// / \ \
// 4 5 6
// /
// 7
Integer[] treeValues = {1, 2, 3, 4, 5, null, 6, 7};
TreeNode root = buildTree(treeValues, 0);

// Count left leaves
int result = countLeftLeaves(root);
System.out.println("Number of left leaves: " + result);
```

8.42

```
public class treeheight {

  public static int computeHeight(TreeNode node) {
    if (node == null) {
      return -1; // Base case: Null node has height -1
    }
    if (node.children.isEmpty()) {
      System.out.println("Node " + node.value + ", Height: 0");
      return 0; // If it's a leaf node, height is 0
    }

    // Compute height as 1 + max height among children
    int maxHeight = 0;
    for (TreeNode child : node.children) {
      maxHeight = Math.max(maxHeight, computeHeight(child));
    }

    int height = 1 + maxHeight;
    System.out.println("Node " + node.value + ", Height: " + height);
    return height;
  }

  public static void main(String[] args) {

    TreeNode root = new TreeNode(1);
    TreeNode child1 = new TreeNode(2);
    TreeNode child2 = new TreeNode(3);
    TreeNode child3 = new TreeNode(4);
    TreeNode child4 = new TreeNode(5);

    root.children.add(child1);
    root.children.add(child2);
    child1.children.add(child3);
    child1.children.add(child4);

    computeHeight(root);
  }
}
```

Problem Statements

- Q8.18 – Group 8
- Q8.19 – Group 7
- Q8.20 – Group 6
- Q8.21 – Group 5
- Q8.22 – Group 4
- Q8.23 – Group 3
- Q8.42 – Group 2
- Q8.45 – Group 1