**Recursive Functions for Binary Trees/Binary Search Trees**

Typical recursive solutions to binary tree problems often involve a postorder traversal. That is, we formulate the problem as (1) solve the problem on the left subtree, (2) solve the problem on the right subtree (3) combine the two intermediate solutions from the two subtrees and combine them, often with some information from the current node.

**Example:**

Count the nodes in a binary tree:

```
int countNodes(Node curr) {
   if (curr == null)
      return 0;
   else {
      leftSubtreeNodes = countNodes(curr.left);
      rightSubtreeNodes = countNodes(curr.left);
      return leftSubtreeNodes + rightSubtreeNodes + 1;  // 1 for curr itself
   }
}
```

Initial call to code above would be `countNodes(root).`

**Exercise:** Calculate sum of all the nodes in a binary tree (assuming the nodes contain numbers)

```
int sumNodes(Node curr) {
   if (curr == null)

      return ______;
   else {









   }
}
```

**Exercise:** Calculate the height of a binary tree.

```
int height(Node curr) {
   if (curr == null)

      return ______;
   else {









   }
}
```