BST Worksheet

1 Definition

%. A **binary search tree** has the following invariant: for any node k in the tree, all nodes with smaller values are in the left subtree of k, and all nodes with larger values are in the right subtree of k.

2 Construction

Takeaway: order of construction affects runtime.
Consider a tree of elements $1, 2, 3, 4, 5, 6, 7$

Draw the BST resulting from insertion in the order $1, 2, 3, 4, 5, 6, 7$

Draw the BST resulting from insertion in the order $4, 3, 5, 2, 6, 1, 7$

Now, consider the runtime of `contains` for any arbitrary BST. What is the worst case and best case?

3 BST Traversal

Consider the 'bushy' tree from above. What is its...

Pre-order Traversal:

In-order Traversal:

Post-order Traversal:

Note: be familiar with the recursion order of each of these traversals.

4 BST Deletion

Process:

1. Let N be the node to be deleted.
2. If N has no children, simply remove N from the tree
3. If N has one child, remove N and replace it with its child
4. Else, let us define S as the in-order successor of N. Copy the value of S into N, and remove S.

5 Practice Q1

The following code is meant to check if a given binary tree is a binary search tree. However, for some binary trees it is returning the wrong answer. Explain why, and give an example of a binary tree for which the method fails.

```
public static boolean isBST (TreeNode T) {
   if (T == null) {
      return true;
    } else if (T.left != null && T.left.val > T.val) {
      return false;
    } else if (T.right != null && T.right.val < T.val) {
      return false;
    } else {
      return isBST(T.left) && isBST(T.right);
    }
}
```

6 Practice Q2

Define a root-to-leaf path as a sequence of nodes from the root of a tree to one of its leaves. Write a method `printSumPaths(TreeNode T, int k)` that prints out all root-to-leaf paths whose values sum to k. For example, if `RootNode` is the binary tree rooted in 10 in the diagram below and k is 13, then the program will print out `10 2 1` on one line and `10 4 -1` on a another. Provide your solution by filling in the code below:

```
public static void printSumpaths(TreeNode T, int k) {
   if (T != null) {
      sumPaths(                                );
    }
}

public static void sumPaths(                   ) {










}
```

Solutions to practice questions are available on the Fall 2016 CS61B website, under Discussion 8 Solutions https://inst.eecs.berkeley.edu/ cs61b/fa16/materials/disc/discussion8sol.pdf