Section Handout 7

Based on handouts by Eric Roberts

Problem One: Scrambled Hashes

Below are three descriptions of hash functions that can be used to produce hash codes for English words. Each of these functions has a problem with it that would make it a poor choice for a hash function. For each of the hash functions, describe what the weakness is.

- Hash function 1: Always return 0.
- Hash function 2: Return a random `int` value.
- Hash function 3: Return the sum of the ASCII values of the letters in the word.

Problem Two: Tracing Binary Tree Insertion (Chapter 16, review question 9, page 711)

In the first example of binary search trees, the text uses the names of the dwarves from Walt Disney's 1937 classic animated film, *Snow White and the Seven Dwarves.* Dwarves, of course, occur in other stories. In J. R. R. Tolkien's *The Hobbit*, for example, 13 dwarves arrive at the house of Bilbo Baggins in the following order:

Dwalin, Balin, Kili, Fili, Dori, Nori, Ori, Oin, Gloin, Bifur, Bofur, Bombur, Thorin

Diagram the binary search tree you get from inserting these names into an empty tree in this order. Once you have finished, answer the following questions about your diagram:

2a. What is the height of the resulting tree?

2b. Which nodes are leaves?

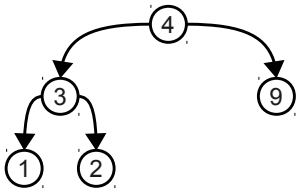
2c. Which key comparisons are required to find the string `"Gloin"` in the tree?

Problem Three: Checking BST Validity

Suppose that you have the following structure representing a node in a binary search tree of integers:

```
struct Node {
      int value;
      Node* left;
      Node* right;
};
```

You are given a pointer to a `Node` that is the root of some type of binary tree. However, you are not sure whether or not it is a binary *search* tree. That is, you might have a tree like this one:



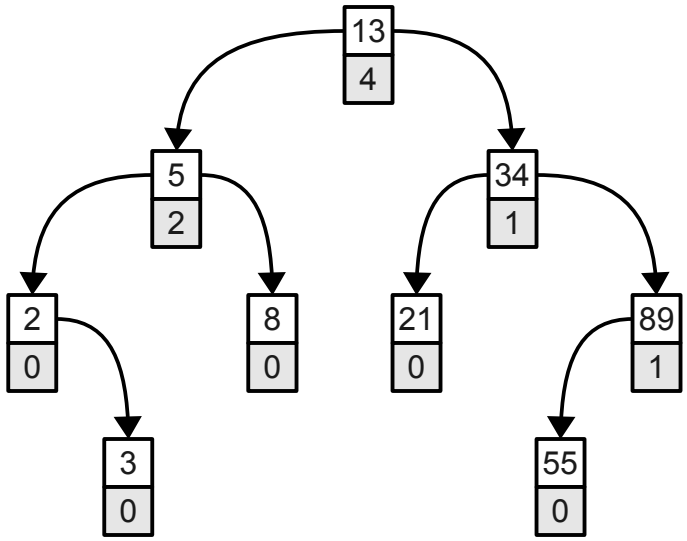
which is not a valid binary search tree. Write a function

```
bool isBST(Node* root);
```

that, given a pointer to the root of a tree, determines whether or not the tree is a legal binary search tree.

Problem Four: Order Statistic Trees

An *order statistic tree* is a binary search tree where each node is augmented with the number of nodes in its left subtree. For example, here is a simple order statistic tree:



Suppose that you have the following struct representing a node in an order statistic tree:

```
struct Node {
      int value;
      int leftSubtreeSize;
      Node* left;
      Node* right;
};
```

Write a function

```
Node* nthNode(Node* root, int n);
```

that accepts as input a pointer to the root of the order statistic tree, along with a number *n*, then returns a pointer to the *n*th smallest node in the tree (zero-indexed). If *n* is negative or at least as large as the number of nodes in the tree, your function should return **`NULL`** as a sentinel.