COMP 587
Spring 2020

Automated Testing with Data Structures

1.) Consider the Java code below, which implements a binary tree:

```
public class Node {
  final int value;
  final Node left;
  final Node right;
  public Node(int value, Node left, Node right) {
    this.value = value; this.left = left; this.right = right;
  }

  // gets a random integer in the range [start, end)
  public static int randomInt(int start, int end) { ... }

  // null represents a leaf
  public static int getSum(final Node root) { ... }
}
```

1.a.) The following Java signature is intended to produce a random tree, suitable as an input to `getSum` above. Complete the code, which will generate nodes. You may use `randomInt`. For this part, don't worry about the behavior of `getSum`; only worry about generating `Node` values. As a hint, it's likely easier to write this in a recursive fashion.

```
// makes a tree which is, at most, maxDepth levels deep
public static Node makeTree(int maxDepth) {
  if (maxDepth <= 0 || randomInt(0, 2) == 0) {
    return null;
  } else {
    return new Node(randomInt(0, 10),
                    makeTree(maxDepth - 1),
                    makeTree(maxDepth - 1));
  }
}
```

1.b.) Describe how you could use `makeTree` to test `getSum`. If you need to make any modifications to `makeTree` for this testing, explain them, possibly with pseudocode.

- Pass outputs from makeTree into getSum
- Modify makeTree to record the number of Nodes created, and store this number somewhere
- The expected from from getSum should be in the range [0, 9*N], where N is the number of nodes in the tree. This follows from the fact that every node can contain a value in the range [0, 10). If the sum from getSum is outside of this range, the test fails.
- Alternatively, record the specific values used in the nodes, and check that the sum from getSum equals the sum of these specific values. The tests would be more specific (good), but the generator would be a lot more complicated and specialized to getSum (bad).

1.c.) Now consider the following method, which is intended to get the depth of a given tree:

```
public static int getDepth(final Node root) { ... }
```

Describe how you could use makeTree to test `getDepth`. If you need to make any modifications to `makeTree` for this testing, explain them, possibly with pseudocode.

- Pass the output of makeTree to getDepth
- The returned depth from getDepth should be in the range 0 <= returnedDepth <= maxDepth, where maxDepth is the parameter initially passed to makeTree. If returnedDepth is outside of this range, the test fails.
- No modifications to makeTree are necessary