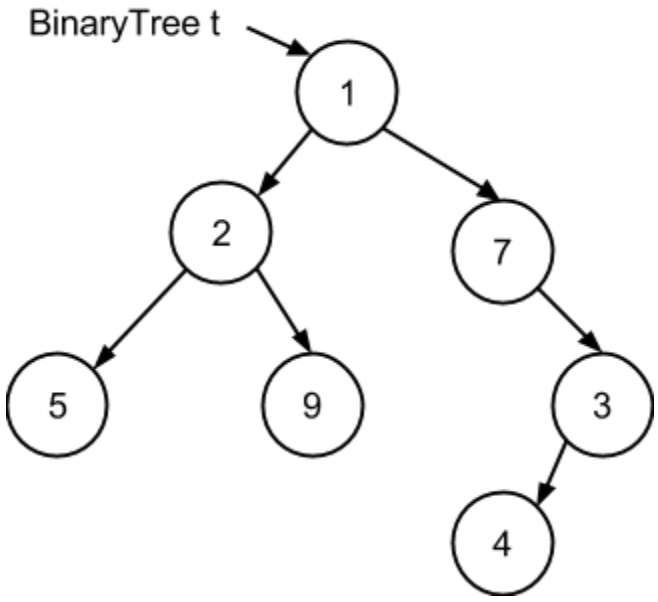
**Consider this these classes/methods and the following BinaryTree *t*:**

```
class BinaryTree<V> {
   BinaryTree<V> right;
   BinaryTree<V> left;
   V val;
}

interface Fringe<V> {
   public void add(V item);
   public V removeNext();
   public boolean isEmpty();
}
```




```
public static <V> void printTraversal(BinaryTree<V> t,
                                      Fringe<BinaryTree<V>> f) {
   f.add(t);
   while (!f.isEmpty()) {
       BinaryTree curr = f.removeNext();
       if (curr.left != null) f.add(curr.left);
       if (curr.right != null) f.add(curr.right);
       System.out.println(curr.val);
   }
}

class Queue<V> implements Fringe<V> {
   private LinkedList<V> data = new LinkedList<V>();
   public void add(V item) {
       data.addFirst(item);
   }
   public V removeNext() {
       return data.removeLast();
   }
   public boolean isEmpty() {
       return data.isEmpty();
   }
}
```

```
class Stack<V> implements Fringe<V> {
   private LinkedList<V> data = new LinkedList<V>();
   public void add(V item) {
       data.addLast(item);
   }
   public V removeNext() {
       return data.removeLast();
   }
   public boolean isEmpty() {
       return data.isEmpty();
   }
}
```

**1. What will Java output?**

```
printTraversal(t, new Queue<BinaryTree>());
```

```
printTraversal(t, new Stack<BinaryTree>());
```

## 2. Height

Write *height*, which takes in a BinaryTree and outputs the height of the tree. Assume that a tree with just the root node is of height 1.

*height(t) => 4*
*height(t.left) => 2*
*height(t.right) => 3*

```
public static int height(BinaryTree node) {









}
```

**What's the runtime of height?**

**3. Is it balanced?**

Given the above, write *isBalanced*, which takes a BinaryTree and outputs whether or not the tree is balanced. A Tree is balanced if the left and right branches differ in height by at most one, and are themselves balanced.

*isBalanced(t) => false*
*isBalanced(t.left) => true*
*isBalanced(t.right) => false*
*isBalanced(t.right.right) => true*

```
public static boolean isBalanced(BinaryTree tree) {









}
```

**How long does your method take:**
**in general?**
**on balanced trees?**

**[Extra Exercises] Can you do better? How much better? Outline the algorithm, come up with a runtime, *then* write the code.**
*Hint: What part of the previous algorithm was expensive?*
*How can you make it less expensive?*