Below is a class for nodes in a binary tree of integers, and a method that uses it:

```java
public static int mystery(BinaryIntNode root) {
    if (root.left == null && root.right == null)
        return 1;
    else if (root.left == null)
        return 1 + mystery(root.right);
    else if (root.right == null)
        return 1 + mystery(root.left);
    else
        return 1 + mystery(root.left) + mystery(root.right);
}
```

class BinaryIntNode {
    int key;
    BinaryIntNode left;
    BinaryIntNode right;
}

1. Assume the `mystery()` method is defined in a class called `X` and that inside its `main()`, we create this tree and store its root in a variable called `r`:

```
  8
 / \
9   1
 /   \
4     5
```

Trace the call `mystery(r)` on a scrap piece of paper. Use a simplification of the full memory model: only trace the run-time stack, and when showing the value for a reference to a `BinaryIntNode`, just write the value of its `key` (instead of the address of the object). Since all of the nodes have different `key` values, this is enough to distinguish them. Below, draw what the stack looks like at its deepest.
2. What value does this method call return? ____________

3. Write a good external comment for this method.

4. What would be a better name for this method? ____________

5. On this copy of the code, write suitable internal comments:

   public static int mystery(BinaryIntNode root) {

       if (root.left == null && root.right == null)

           return 1;

       else if (root.left == null)

           return 1 + mystery(root.right);

       else if (root.right == null)

           return 1 + mystery(root.left);

       else

           return 1 + mystery(root.left) + mystery(root.right);
   }