

# Twenty Questions

*(or: How your next homework works)*

*(or: Trees!)*



I'm thinking of a noun.



One more time!



Is it bigger than a breadbox?

No

Yes

Is it something you  
find indoors?

Is it alive?

Is it bigger than a breadbox?

No

Yes

Is it something you  
find indoors?

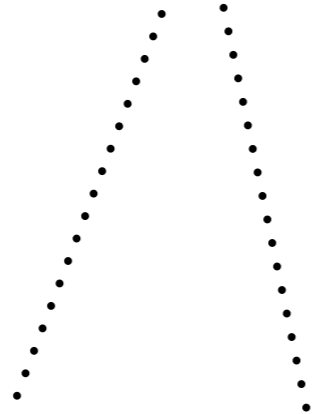
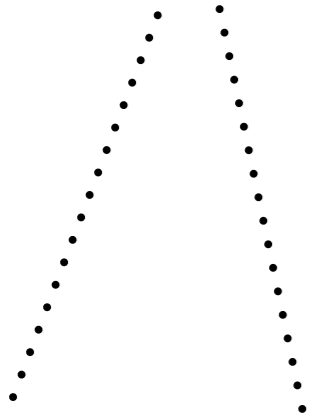
Is it alive?

No

Yes

Is it alive?

Is it something  
you move often?



Is it bigger than a breadbox?

No

Yes

Is it something you find indoors?

No

Yes

Is it alive?

No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

Yes

Can you eat it?

Is it blue?

No

Yes



Trees are more than  
just for Sets & Maps!



Is it bigger than a breadbox?



No

Yes

Is it something you find indoors?

Is it alive?

No

Yes

No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

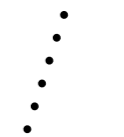
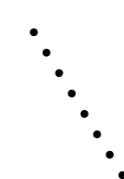
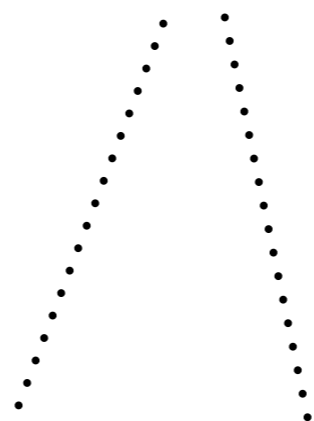
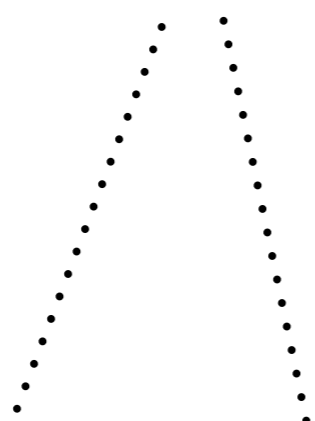
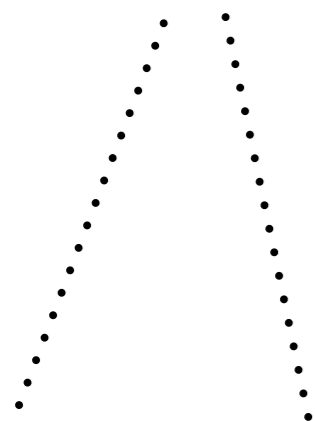
Yes

Can you eat it?

Is it blue?

No

Yes





Is it bigger than a breadbox?

No

Yes

Is it something you find indoors?

No

Yes

Is it alive?



No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

Yes

Can you eat it?

Is it blue?

No

Yes



Is it bigger than a breadbox?

No

Yes

Is it something you find indoors?

No

Yes

Is it alive?

No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

Yes

Can you eat it?

Is it blue?

No

Yes



Is it bigger than a breadbox?

No

Yes

Is it something you find indoors?

No

Yes

Is it alive?

No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

Yes

Can you eat it?

Is it blue?

No

Yes



Is it bigger than a breadbox?

No

Yes

Is it something you find indoors?

No

Yes

Is it alive?

No

Yes

Is it alive?

Is it something you move often?

Is it a solid?

Is it an animal?

No

Yes

Can you eat it?

Is it blue?

No

Yes





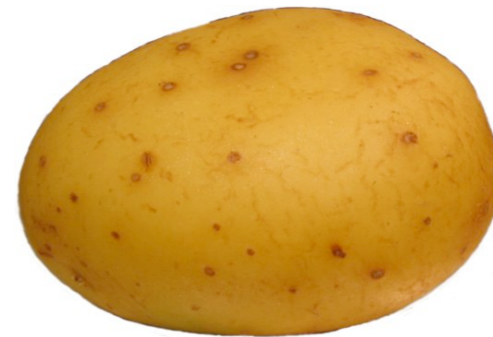
Grows below ground?

No

Yes

Goes in sandwiches?

Often in salads?



Grows below ground?

What do we need to store here?

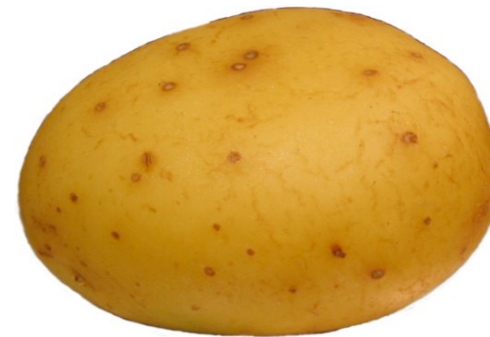
No

Yes

Goes in sandwiches?

Often in salads?

Or here?



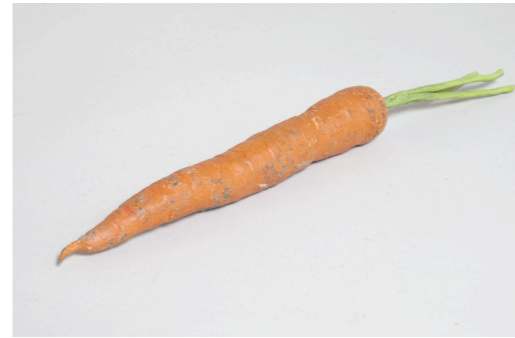
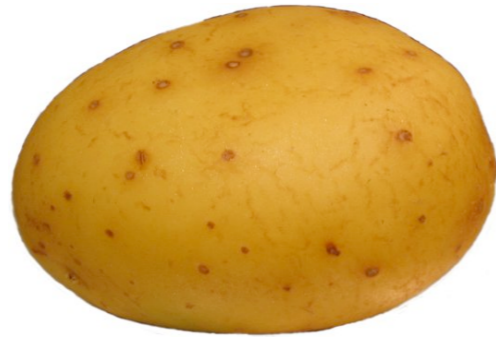
# Learning Twenty Questions

(below ground)

Often in salads?

No

Yes



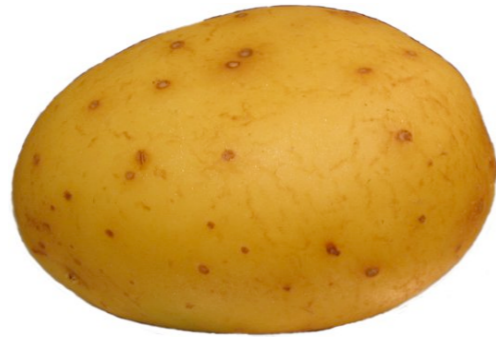
# Learning Twenty Questions

(below ground)

Often in salads?

No

Yes





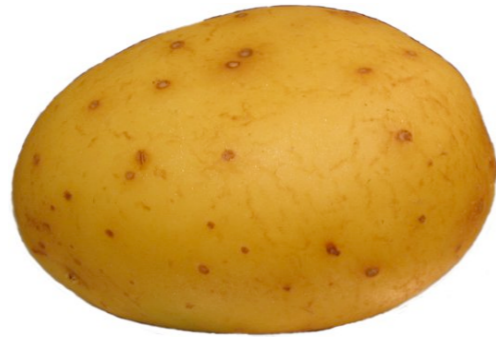
# Learning Twenty Questions

(below ground)

Often in salads?

No

Yes



*This is a ginger root.*

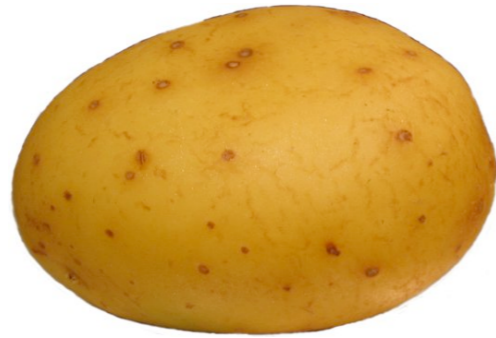
# Learning Twenty Questions

(below ground)

Often in salads?

No

Yes



Is it commonly mashed?

# Learning Twenty Questions

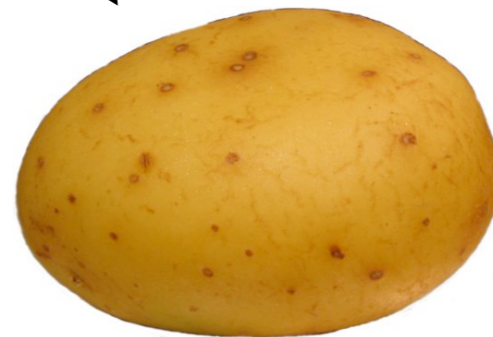
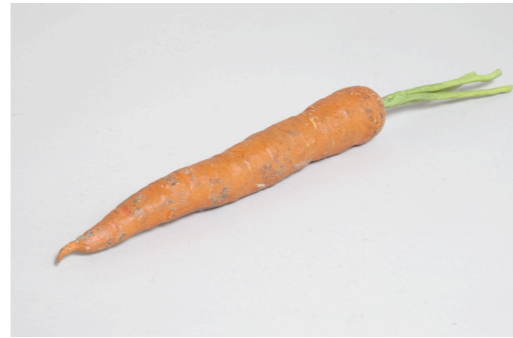
(below ground)

Often in salads?

No

Yes

Is it commonly mashed?



# *Saving* Twenty Questions

# Saving Twenty Questions

```
// Suppose we have a ListNode class...
public void saveList(ListNode n) {
    if (n == null) {
        return;           Base case
    }
    System.out.println(n.getValue()); Deal with this node
    saveList(n.getNext()); Recurse
}
```

# Saving Twenty Questions

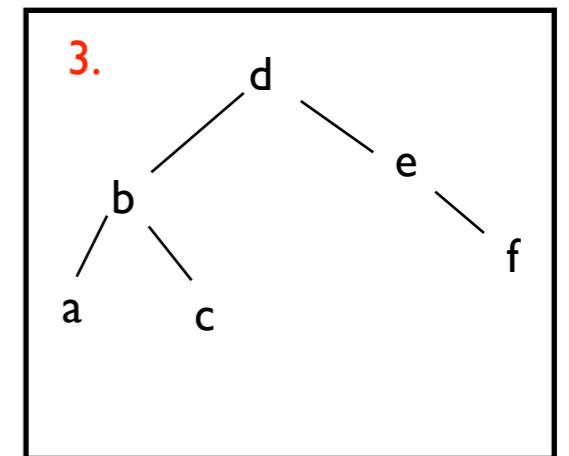
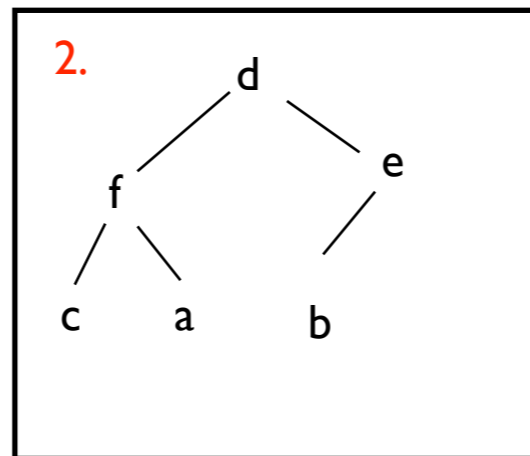
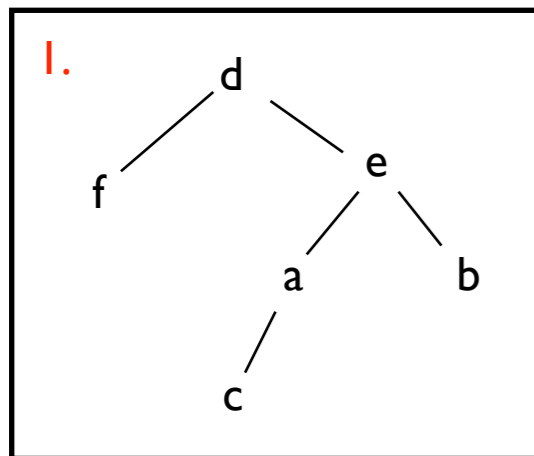
```
// Suppose we have a ListNode class...
public void saveList(ListNode n) {
    if (n == null) {
        return;           Base case
    }
    System.out.println(n.getValue()); Deal with this node
    saveList(n.getNext()); Recurse
}
```

```
// Suppose we have a TreeNode class...
public void saveTree(TreeNode n) {
    if (n == null) {
        return;
    }
    System.out.println(n.getValue());
    saveTree(n.getLeftChild());
    saveTree(n.getRightChild());
}
```

# Saving Twenty Questions

// Suppose we have a `TreeNode` class...

```
public void saveTree(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    System.out.println(n.getValue());  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
}
```

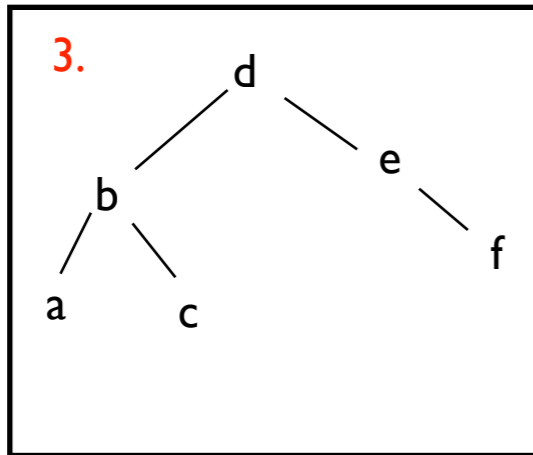


<http://goo.gl/4Txf4>

# Saving Twenty Questions

// Suppose we have a `TreeNode` class...

```
public void saveTree(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    System.out.println(n.getValue());  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
}
```



```
saveTree("d");  
    print("d");  
    saveTree("b");  
        print("b");  
        saveTree("a");  
            print("a");  
                saveTree(null);  
                saveTree(null);  
        saveTree("c");  
            print("c");  
            saveTree(null);  
            saveTree(null);  
    saveTree("e");  
        print("e");  
        saveTree(null);  
        saveTree("f");  
            print("f");  
            saveTree(null);  
            saveTree(null);
```


“d b a c e f”



# Saving Twenty Questions


// Suppose we have a `TreeNode` class...

```
public void saveTree2(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    System.out.println(n.getValue());  
    saveTree(n.getRightChild());  
}
```



// Suppose we have a `TreeNode` class...

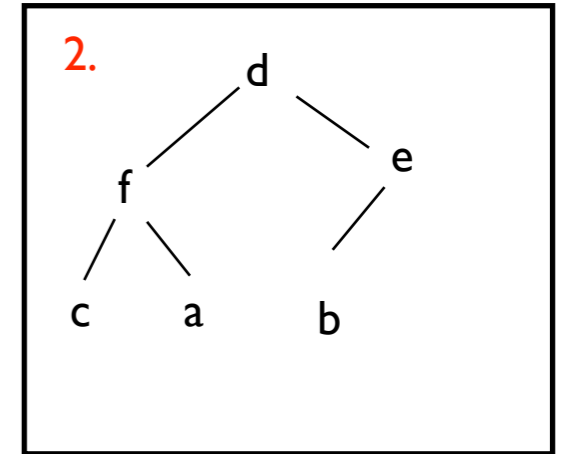
```
public void saveTree3(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
    System.out.println(n.getValue());  
}
```



# Saving Twenty Questions

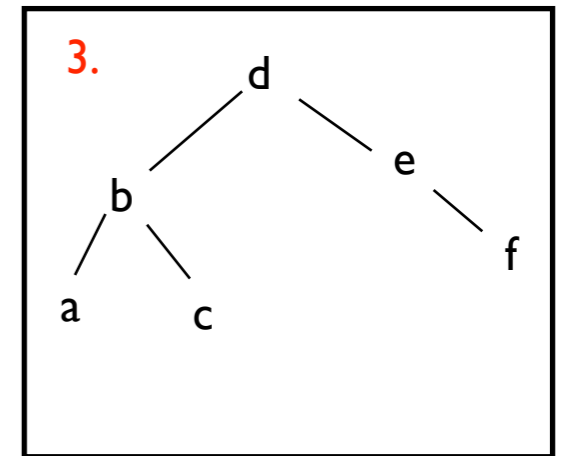
// Suppose we have a `TreeNode` class...

```
public void saveTree2(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    System.out.println(n.getValue());  
    saveTree(n.getRightChild());  
}
```



// Suppose we have a `TreeNode` class...

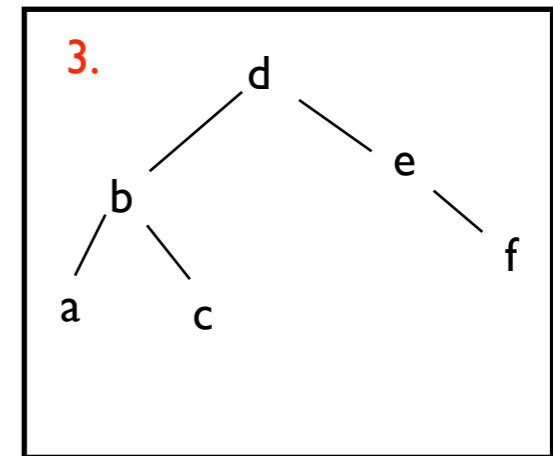
```
public void saveTree3(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
    System.out.println(n.getValue());  
}
```



<http://goo.gl/P4o5R>

# Notice anything?

```
// Suppose we have a TreeNode class...
public void saveTree2(TreeNode n) {
    if (n == null) {
        return;
    }
    saveTree(n.getLeftChild()); Less-than-mes
    System.out.println(n.getValue()); Me
    saveTree(n.getRightChild()); Greater-than-mes
}
```

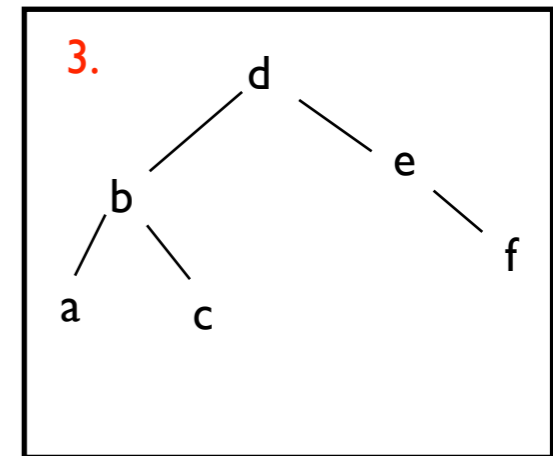


**Search tree!**

*(Everything in my left subtree is less than (or equal to) me; everything in my right subtree is greater than me)*

# Notice anything?

```
// Suppose we have a TreeNode class...
public void saveTree2(TreeNode n) {
    if (n == null) {
        return;
    }
    saveTree(n.getLeftChild()); Less-than-mes
    System.out.println(n.getValue()); Me
    saveTree(n.getRightChild()); Greater-than-mes
}
```



*Search tree!*

*(Everything in my left subtree is less than (or equal to) me; everything in my right subtree is greater than me)*

<http://goo.gl/hh9NL>

# The Three Tree Traversals

```
public void saveTree(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    System.out.println(n.getValue());  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
}
```

Pre-order traversal

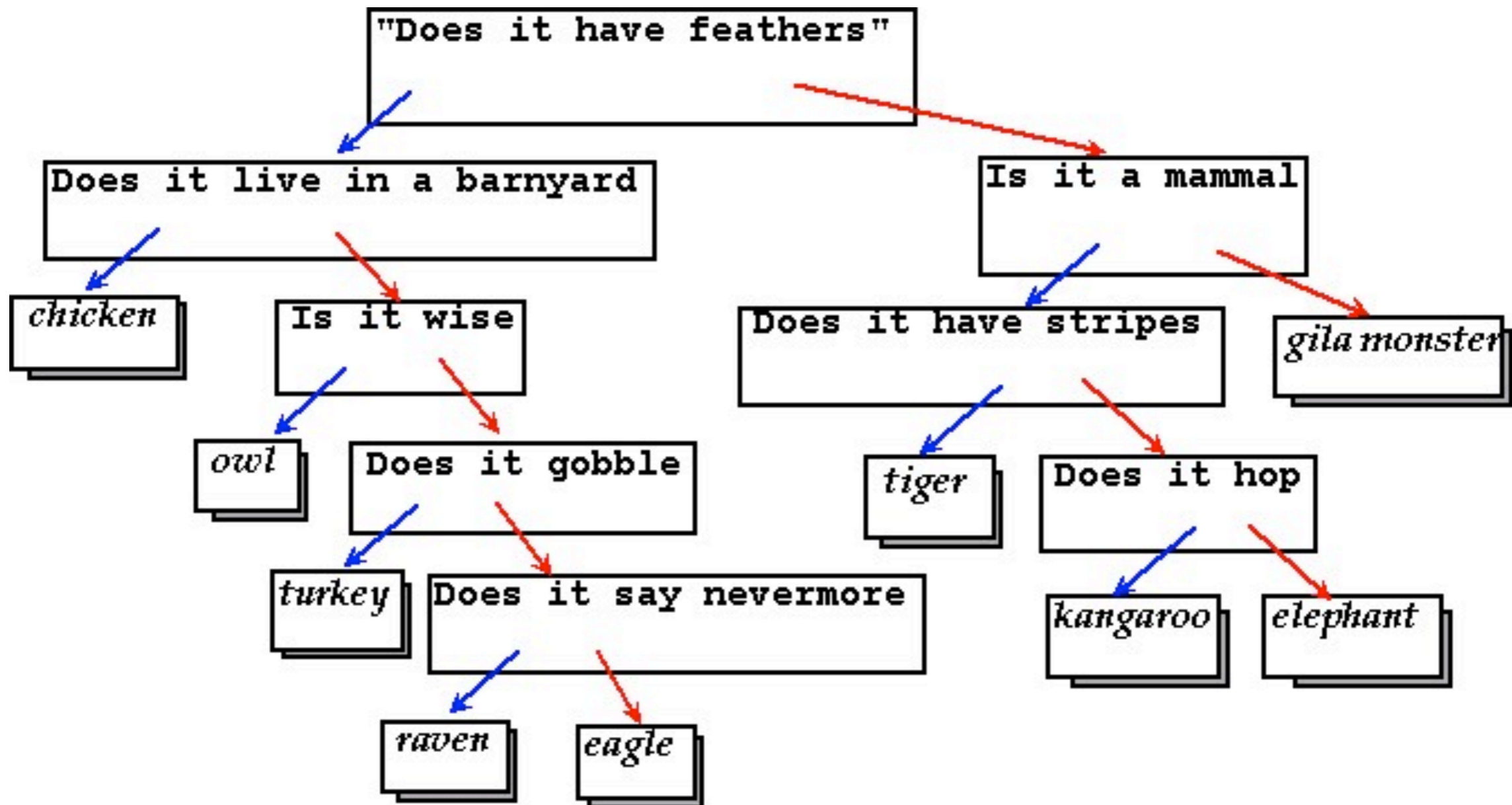
```
public void saveTree2(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    System.out.println(n.getValue());  
    saveTree(n.getRightChild());  
}
```

In-order traversal

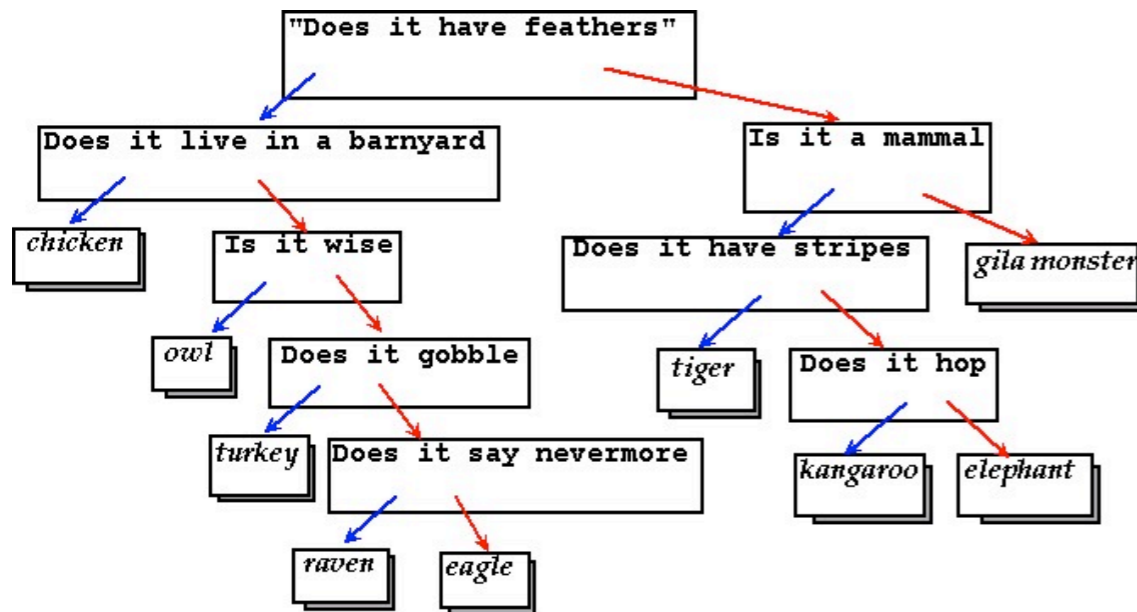
```
public void saveTree3(TreeNode n) {  
    if (n == null) {  
        return;  
    }  
    saveTree(n.getLeftChild());  
    saveTree(n.getRightChild());  
    System.out.println(n.getValue());  
}
```

Post-order traversal

# Saving the tree (again)



# Saving the tree (again)



## Pre-order traversal

- Interior nodes marked!
- #Q: Does it have feathers?
  - #Q: Does it live in a barnyard?
  - Is it a chicken?
  - #Q: Is it wise?
  - Is it an owl?
  - #Q: Does it gobble?
  - Is it a turkey?
  - #Q: Does it say "Nevermore"?
  - Is it a raven?
  - Is it an eagle?
  - #Q: Is it a mammal?
  - #Q: Does it have stripes?
  - Is it a tiger?
  - #Q: Does it hop?
  - Is it a kangaroo?
  - Is it an elephant?
  - Is it a gila monster?

# Reading the tree

#Q: Does it have feathers?  
#Q: Does it live in a barnyard?  
Is it a chicken?  
#Q: Is it wise?  
Is it an owl?  
#Q: Does it gobble?  
Is it a turkey?  
#Q: Does it say “Nevermore”?  
Is it a raven?  
Is it an eagle?  
#Q: Is it a mammal?  
#Q: Does it have stripes?  
Is it a tiger?  
#Q: Does it hop?  
Is it a kangaroo?  
Is it an elephant?  
Is it a gila monster?



# Reading the tree

#Q: Does it have feathers?  
#Q: Does it live in a barnyard?  
Is it a chicken?  
#Q: Is it wise?  
Is it an owl?  
#Q: Does it gobble?  
Is it a turkey?  
#Q: Does it say “Nevermore”?  
Is it a raven?  
Is it an eagle?  
#Q: Is it a mammal?  
#Q: Does it have stripes?  
Is it a tiger?  
#Q: Does it hop?  
Is it a kangaroo?  
Is it an elephant?  
Is it a gila monster?

*“Recursion: the cause, and solution,  
of all of life’s problems.”*

*- Homer*



# Reading the tree

To save:

- Check base case
- Write out the current node
- recurse left
- recurse right

We want to produce a `TreeNode`

Homer is right!

#Q: Does it have feathers?  
#Q: Does it live in a barnyard?  
Is it a chicken?  
#Q: Is it wise?  
Is it an owl?  
#Q: Does it gobble?  
Is it a turkey?  
#Q: Does it say “Nevermore”?  
Is it a raven?  
Is it an eagle?  
#Q: Is it a mammal?  
#Q: Does it have stripes?  
Is it a tiger?  
#Q: Does it hop?  
Is it a kangaroo?  
Is it an elephant?  
Is it a gila monster?

*“Recursion: the cause, and solution,  
of all of life’s problems.”*

- Homer



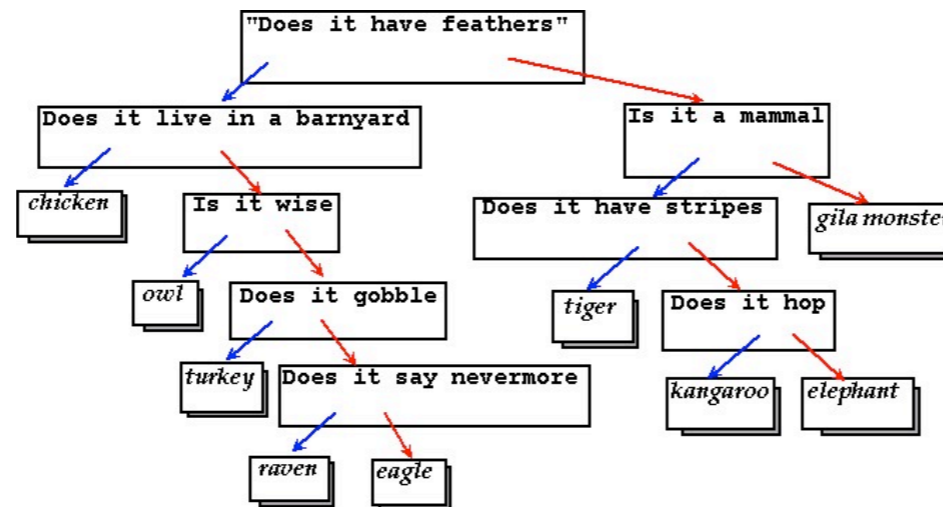
To save:

- Check base case
- Write out the current node
- recurse left
- recurse right

# Reading the tree

```
public TreeNode readTree() {  
    // Assume you have a method readLine() that returns the next  
    // line of the file, as a String:  
    String line = readLine();  
}
```

- #Q: Does it have feathers?
- #Q: Does it live in a barnyard?
- Is it a chicken?
- #Q: Is it wise?
- Is it an owl?
- #Q: Does it gobble?
- Is it a turkey?
- #Q: Does it say "Nevermore"?
- Is it a raven?
- Is it an eagle?
- #Q: Is it a mammal?
- #Q: Does it have stripes?
- Is it a tiger?
- #Q: Does it hop?
- Is it a kangaroo?
- Is it an elephant?
- Is it a gila monster?



*"Recursion: the cause, and solution, of all of life's problems."*

- Homer

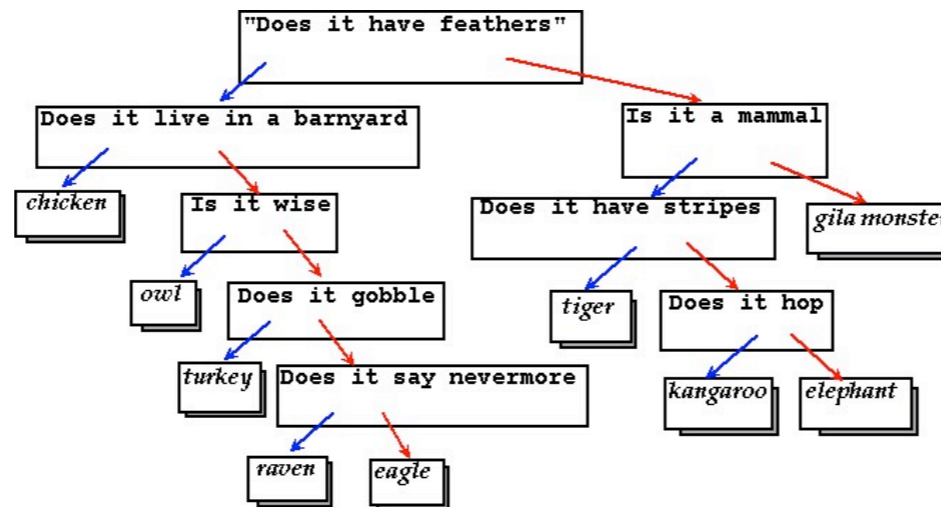
- To save:
- Check base case
  - Write out the current node
  - recurse left
  - recurse right

# Reading the tree

```
public TreeNode readTree() {
    // Assume you have a method readLine() that returns the next
    // line of the file, as a String:
    String line = readLine();
}
```

<http://goo.gl/i1nt9>

- #Q: Does it have feathers?
- #Q: Does it live in a barnyard?
- Is it a chicken?
- #Q: Is it wise?
- Is it an owl?
- #Q: Does it gobble?
- Is it a turkey?
- #Q: Does it say "Nevermore"?
- Is it a raven?
- Is it an eagle?
- #Q: Is it a mammal?
- #Q: Does it have stripes?
- Is it a tiger?
- #Q: Does it hop?
- Is it a kangaroo?
- Is it an elephant?
- Is it a gila monster?



*"Recursion: the cause, and solution, of all of life's problems."*

- Homer