

CSSE 230 Day 6

Intro to Trees

After today, you should be able to...

- ...use tree terminology
- ...write recursive tree functions

Checkout `BinarySearchTree` from SVN

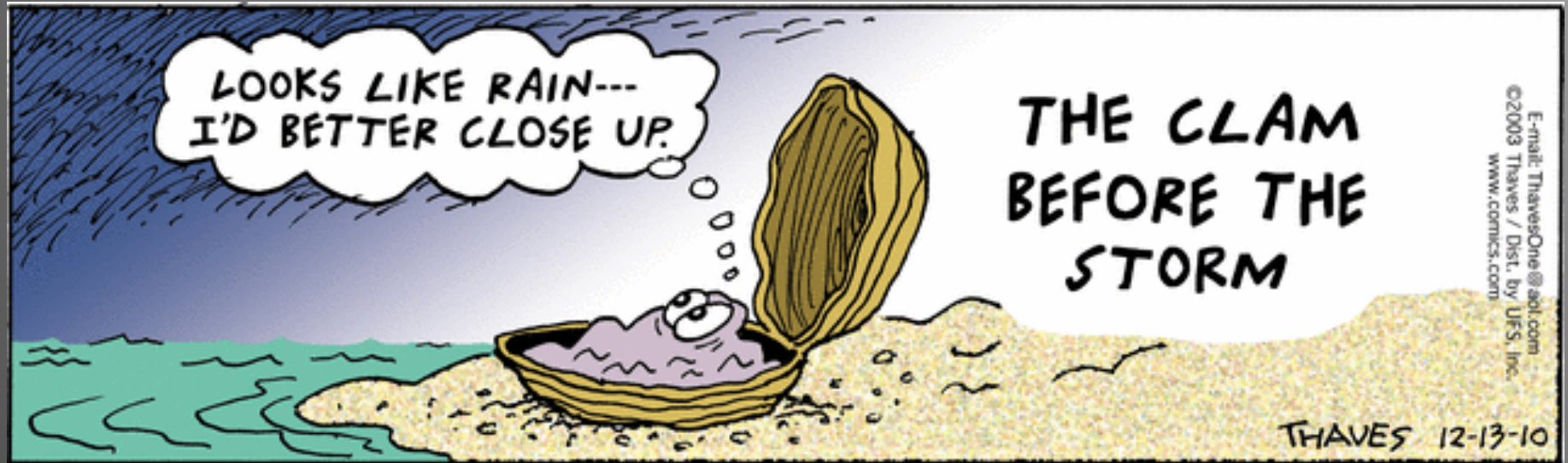
Announcements

- ▶ Review yesterday's quizzes on Java Collections and Data Structures
- ▶ Preview of HW3: includes an essay

Observation about Stacks and Queues Infix \rightarrow Postfix problem

- ▶ It must be $O(n)$, so you can't grow your strings character-by-character:
 - Strings are immutable, so characters must be copied. `s += "*"` is as slow growing an array using the `+1` scheme
- ▶ Solution? Use a `StringBuilder`!
 - They have internal capacity, which doubles when full!
- ▶ See the example at the end of Warmup and Stretching's `ShapeText.java` for an example.

Questions?



Next:

- ▶ an implementation that offers interesting benefits, but is more complex to code than arrays...
- ▶ ... Trees!

Trees

Introduction and terminology
for three types



Reminder: Specifying an ADT in Java

- ▶ Done with an interface, e.g., **java.util.Collection**

java.util

Interface Collection<E>

boolean **add**(E o)

Ensures that this collection contains the specified element (optional operation).

boolean **contains**(Object o)

Returns true if this collection contains the specified element.

boolean **isEmpty**()

Returns true if this collection contains no elements.

boolean **remove**(Object o)

Removes a single instance of the specified element from this collection, if it is present (optional operation).

int **size**()

Returns the number of elements in this collection.

Iterator<E> **iterator**()

Returns an iterator over the elements in this collection.

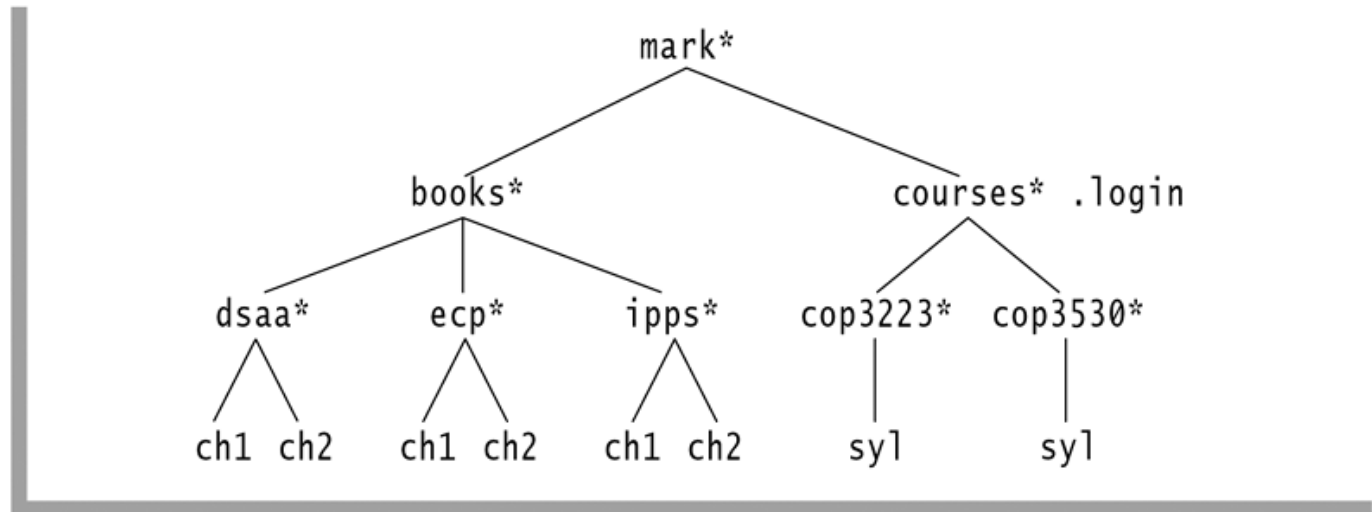
A “factory method”

Trees in everyday life

- ▶ Class hierarchy tree (single inheritance only)
- ▶ Directory tree in a file system

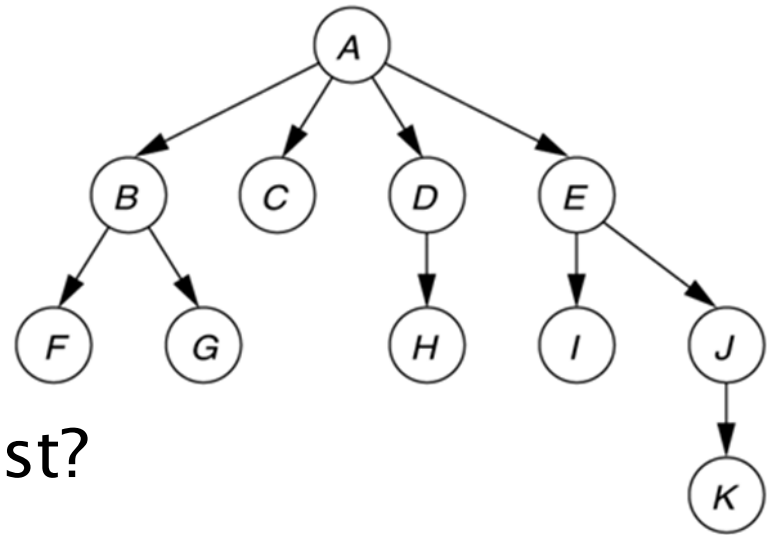
figure 18.4

A Unix directory



A General Tree—Global View

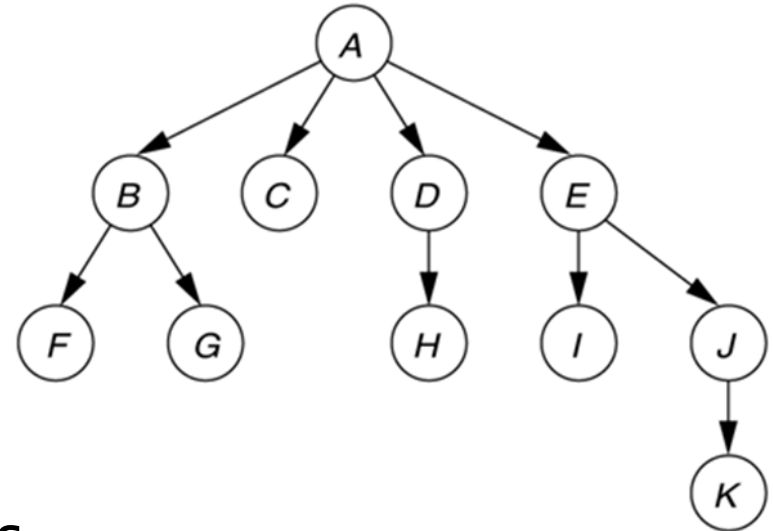
- ▶ A collection of **nodes**
- ▶ Nodes are connected by **directed** edges.
 - One special **root node** has no incoming edges
 - All other nodes have exactly one incoming edge
- ▶ One way that Computer Scientists are odd is that our trees usually have their root at the top!



- ▶ How are trees like a linked list?
- ▶ How are they different?

Tree Terminology

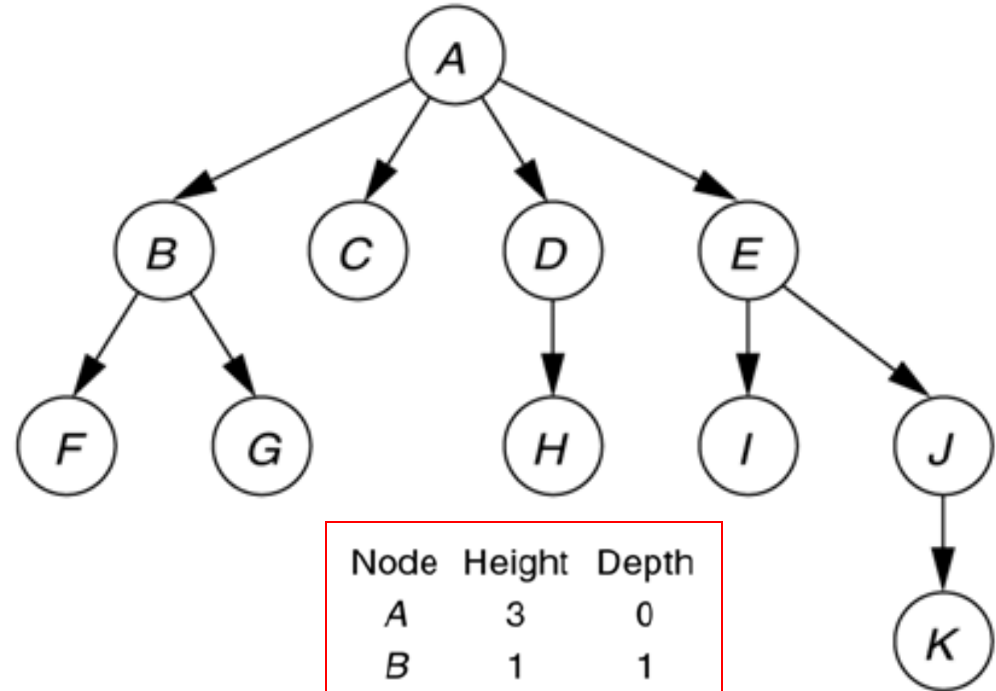
- ▶ Parent
- ▶ Child
- ▶ Grandparent
- ▶ Sibling
- ▶ Ancestors and descendants
- ▶ Proper ancestors, proper descendants
- ▶ Subtree
- ▶ Leaf, interior node
- ▶ Depth and height of a node
- ▶ Height of a tree



Node height and depth examples

figure 18.1

A tree, with height and depth information



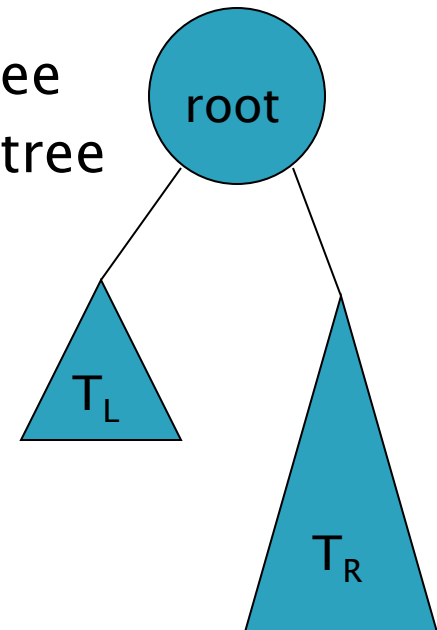
The height of a tree is the height of its root node.

Which is larger, the sum of the heights or the sum of the depths of all nodes in a tree?

| Node | Height | Depth |
|------|--------|-------|
| A | 3 | 0 |
| B | 1 | 1 |
| C | 0 | 1 |
| D | 1 | 1 |
| E | 2 | 1 |
| F | 0 | 2 |
| G | 0 | 2 |
| H | 0 | 2 |
| I | 0 | 2 |
| J | 1 | 2 |
| K | 0 | 3 |

Binary Tree: Recursive definition

- ▶ A **Binary Tree** is either
 - **empty**, or
 - **consists** of:
 - a distinguished node called the root, which contains an element, and
 - A left subtree T_L , which is a binary tree
 - A right subtree T_R , which is a binary tree
- ▶ **Binary trees** contain at most 2 children



Binary Search Trees (BST)

- ▶ Q: What property enables us to search BSTs efficiently?
- ▶ A: Every element in the left subtree is smaller than the root, and every element in the right subtree is larger than the root. And this is true at **every node**, not just the root.

Connections with Linked Lists

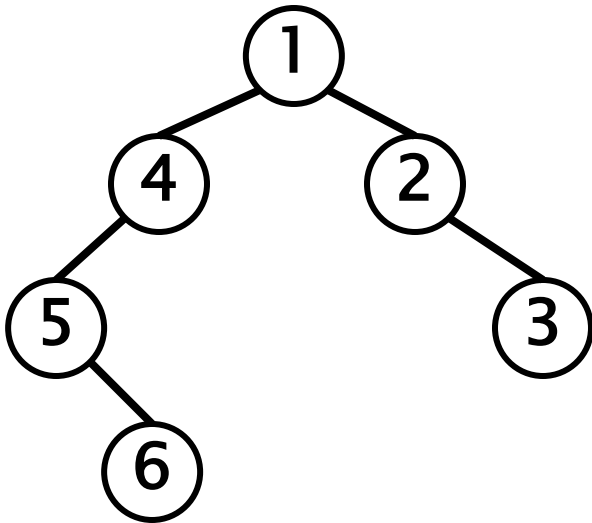
- ▶ Write `size()` for linked list
 - Non-recursively
 - Recursively

- ▶ Write `size()` for a tree
 - Recursively
 - Non-recursively (later)

Growing Trees

- ▶ Let's start the BinarySearchTrees assignment: implement a **BinaryTree<T>** class

Test tree:



A single tiny recursive method for size will touch **every node in the tree**. Let's write, then watch in debugger.