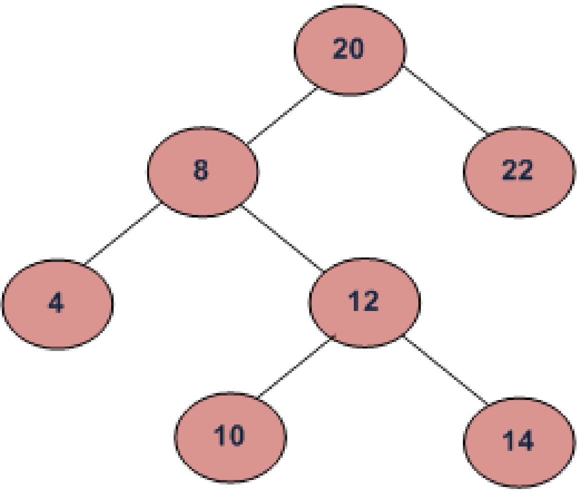


CSSE 230 Day 9

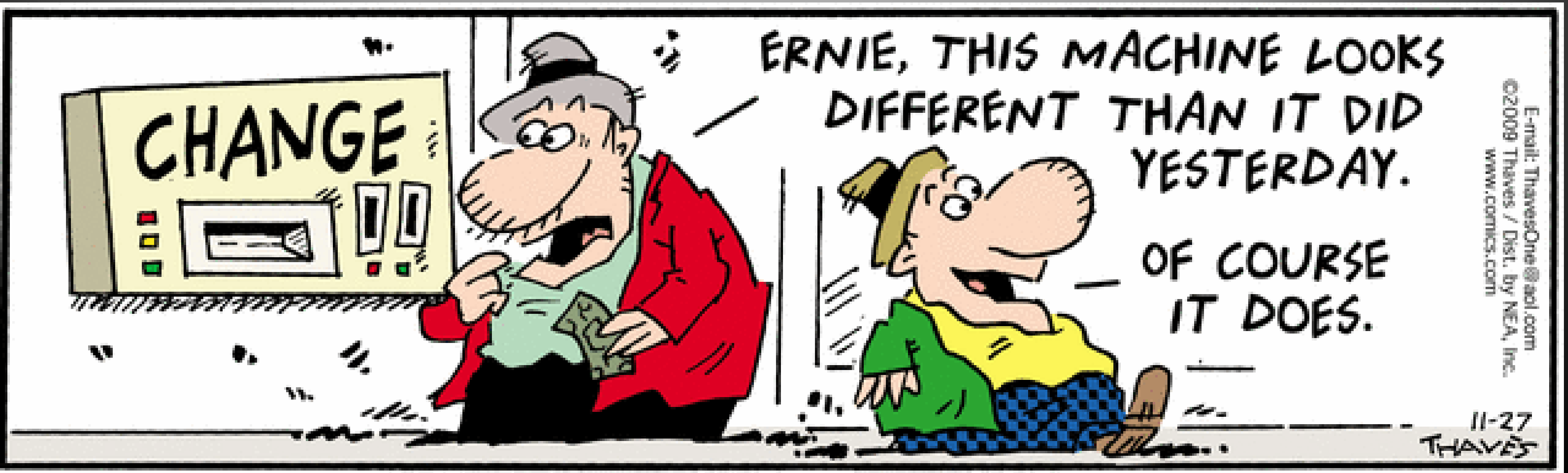
Binary Search Tree intro
BST with order properties

After today, you should be able to…
… implement _lazy_ iterators for trees
… implement deletion from a BST

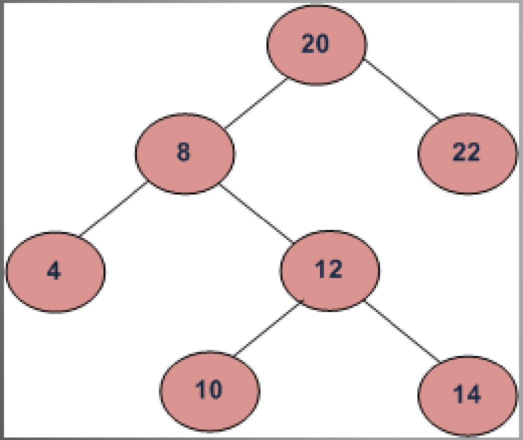
Announcements

- Partner Evaluation done?
- Questions about upcoming test?

Questions?


CHANGE
ERNIE, THIS MACHINE LOOKS DIFFERENT THAN IT DID YESTERDAY.
OF COURSE IT DOES.
E-mail: ThavesOne@aol.com
©2009 Thaves / Dist. by NEA, Inc.
www.comics.com
11-27
THAVES

Binary Search Trees



Binary Trees that store elements in increasing order

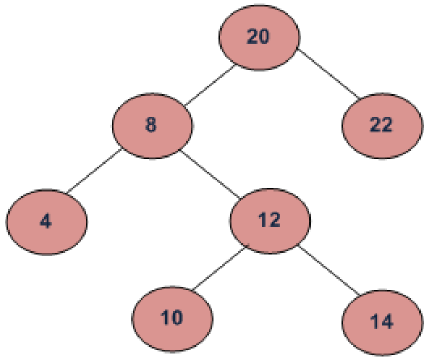
A Binary Search Tree (BST) allows easy and fast lookup of its items because it keeps them ordered

Draw a "birthday BST"

- A BST is a Binary Tree T with these properties:
  1. Elements are Comparable, and non-null
  2. No duplicate elements
  3. All elements in T's left subtree are less than the root element
  4. All elements in T's right subtree are greater than the root element
  5. Both subtrees are BSTs
- **Advantage:** Lookup of items is O(height(T))
- What does the inorder traversal of a BST yield?

BST insert, contains, and delete are different than in a regular binary tree

```
public class BinarySearchTree<T extends Comparable<T>> {

 private BinaryNode root;

 public BinarySearchTree() {
   this.root = null; // or NULL_NODE;
 }

 // insert obj. If already there, return false
 public boolean insert(T obj)

// delete obj. If not there, return false
 public boolean delete(T obj)
             // 3 cases (see text)

// Does this tree contain obj?
 public boolean contains(T obj)
```



Implementation issues, part 1

- The **recursive BinaryNode** insert() and delete() in the text return BinaryNodes. So how do the BinarySearchTree methods return Booleans?

- Can you return 2 things?
  - Create a simple composite class to hold both a boolean and a BinaryNode?
- Can you pass and mutate a parameter?
  - Parameters are call-by-value, so primitives can be mutated.
  - Pass a simple BooleanContainer object so you can mutate the Boolean inside?

Implementation issues, part 2

- Modifying (inserting/deleting) from a tree should cause any current iterators to fail (throw a ConcurrentModificationException).
  - How do you detect this?
- How do you remove from an iterator?
  - Just call BST remove().
  - But throw exceptions if next() hasn’t been called, or if remove is called twice in a row. (Javadoc for TreeSet iterator has details.)