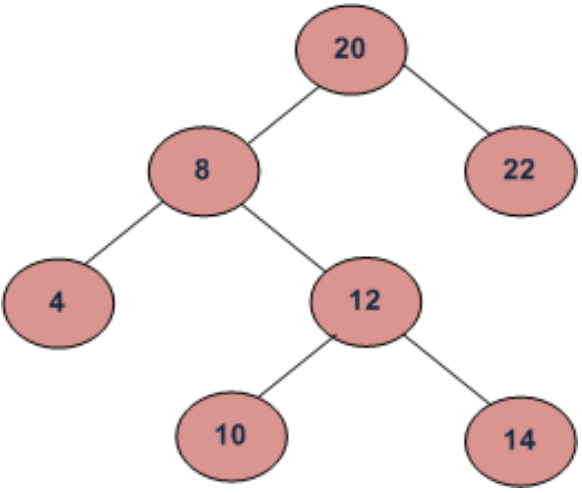


CSSE 230 Day 12

Binary Search Tree intro
BST with order properties

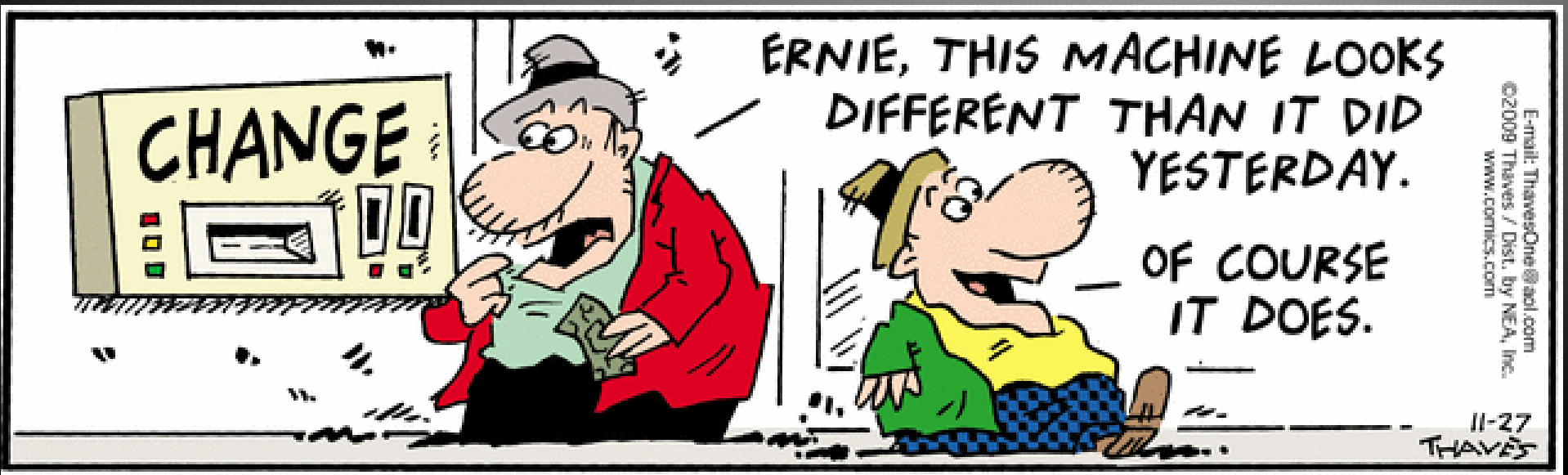
Check out BSTNullNode project from SVN

Announcements

- Hardy/Colorize Partner Evaluation
- Doublets Partner Preference survey

Questions?

Displayable, WA4, …



Size and Height of Binary Trees

- Notation:
  - Let T be a tree
  - Write h(T) for the height of the tree, and
  - N(T) for the size (i.e., number of nodes) of the tree
- Given h(T), what are the bounds on N(T)?
- Given N(T), what are the bounds on h(T)?

Extreme Trees

- A tree with the maximum number of nodes for its height is a **full** tree.
  - Its height is O(log N)
- A tree with the minimum number of nodes for its height is essentially a _____ _____
  - Its height is O(N)
- Height matters!
  - We will see that the algorithms for search, insertion, and deletion in a Binary search tree are O(h(T))

Time out for math!

- Want to prove some properties about trees
- Weak induction isn't enough
- Need strong induction instead:

The former governor of California

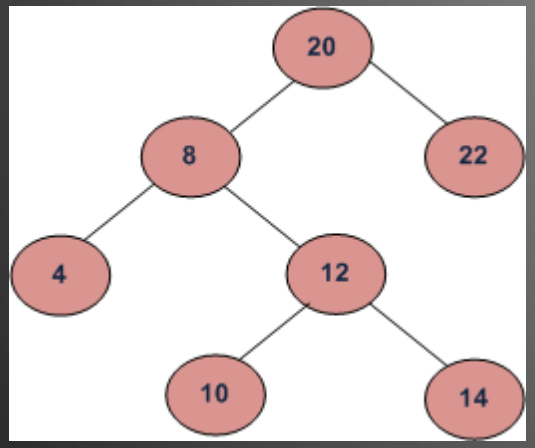
Q9-11, hand in

Strong Induction

- To prove that p(n) is true for all $n >= n_0$:
  - Prove that $p(n_0)$ is true, and
  - For all $k > n_0$, prove that if we assume $p(j)$ is true for $n_0 \leq j < k$, then p(k) is also true

- Weak induction uses the previous domino to knock down the next
- Strong induction uses a whole box of dominoes to knock down the rest!

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

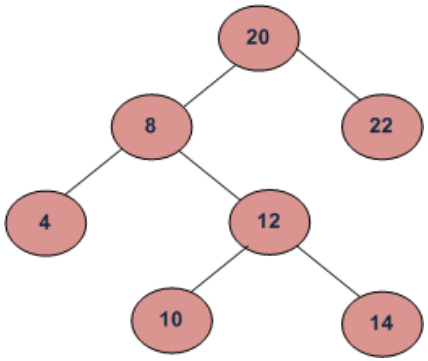
Q1

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

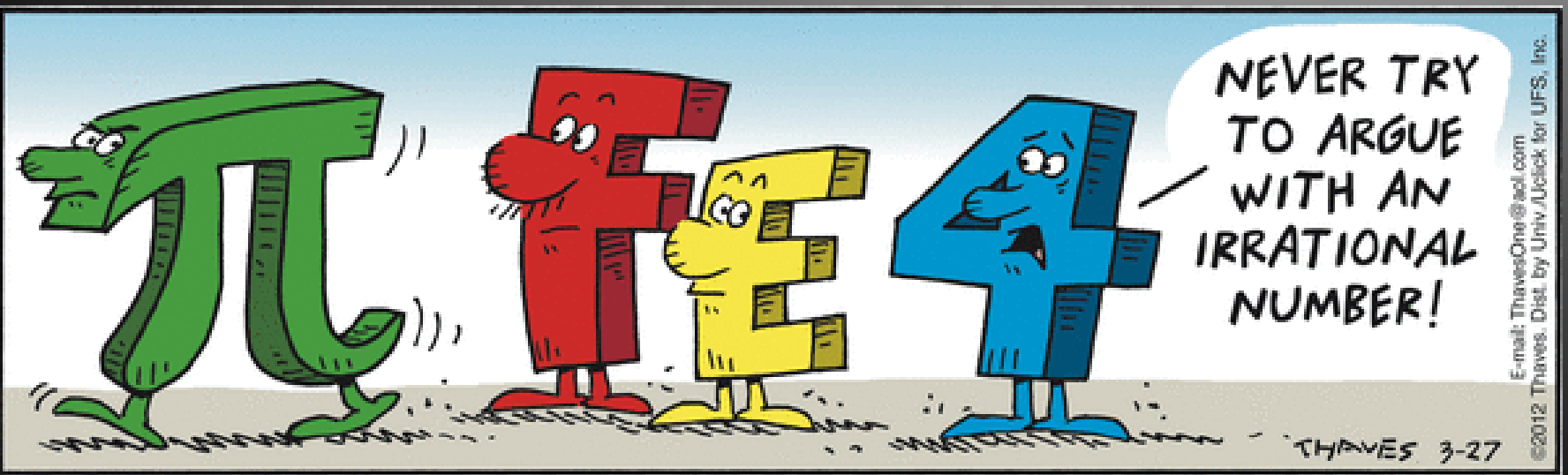

```
public class BinarySearchTree<T extends Comparable<T>> {

 private BinaryNode<T> root;

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

 // insert obj, if not already there
 public void insert(T obj)

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

 // delete obj, if it's there
 public void delete(T obj)
```



BST with Rank

Explore the concept

How do Find and Insert work?



BSTs are an efficient way to represent ordered lists

- What's the performance of
  - insertion?
  - deletion?
  - find?
  - iteration?
- What about finding the k^{th} smallest element?

We can find the kth smallest element easily if we add a *rank* field to BinaryNode

- Gives the in-order position of this node within its own subtree
  - i.e., the size of its left subtree

0-based indexing

- How would we do *findK*$_{th}$?
- *Insert* and *delete* start similarly