CSSE 230 Day 8

Binary Tree Iterators

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

Reminders

- Exam 1 – Day 11: but when and where?
  - Coverage:
    - Everything from reading and lectures, Sessions 1-10
    - Programs through BinaryTrees
    - Homeworks 1-3
  - Allowed resources:
    - Written part: ½ of one side of 8.5 x 11 paper
      - Goal: to force you to summarize.
    - Programming part:
      - Textbook
      - Eclipse (including programs you wrote in your repos)
      - Course web pages and materials on Moodle
      - Java API documentation
    - A previous 230 Exam 1 is available in Moodle

Agenda

- Binary Tree Iterators
  - Especially (yawn) *lazy* ones
- BinarySearchTree (BST) insertion

Binary Tree Iterators

What if we want to iterate over the elements in the nodes of the tree one-at-a-time instead of just printing all of them?

Why is the ArrayListIterator an inefficient iterator?

- Consider a tree with 1 million elements.
- What is the runtime of iterating over only the first 100 elements?

- (example on board)

- To improve efficiency, the iterator should only get as few elements as possible
  - The one time where being lazy has a reward!

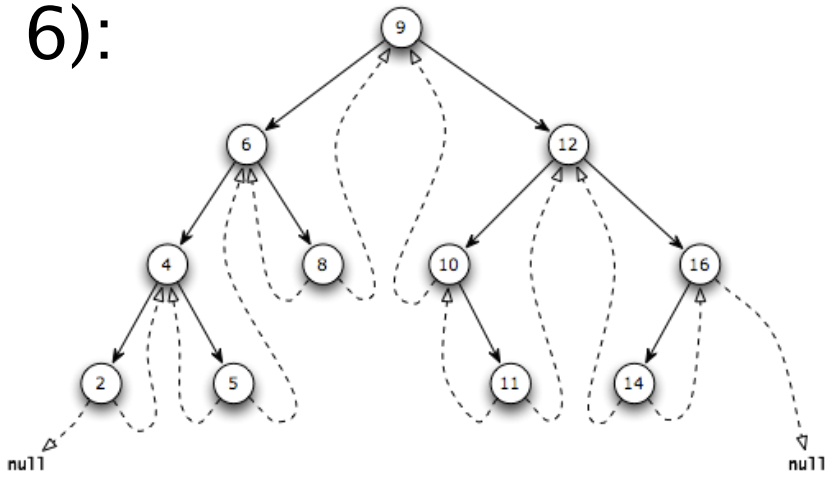
Recall the four types of traversals

- What are they?
- How would you make a lazy **pre-order** iterator? (brainstorm an algorithm now)
- What do you need to add to create the other recursive iterators?
- What about the last iterator?
  - A quick change. Magic? Not really…

Summary: use recursion when you want to process the whole tree at once

Otherwise, you'll use a loop. Examples:

- Lazy iterators (today):
  - use a stack too.
- AVL trees (week 4-5):
  - use pointer to parents to move up tree and "rebalance"
- Threaded trees (HW5 and 6):
  - use pointer to next and previous in-order nodes



Work time

Aim to complete at least Milestone 1 of BinarySearchTrees by next class

We'll start next topic during last 20 min of class

Brainstorm

- How does one insert into a BST?
- Rules:
  - Assume you have a BST
  - All elements are Comparable
  - There is only one place to insert the element while keeping the tree a BST
  - Duplicate elements not allowed (we are implementing TreeSet)
- More on BSTs next class