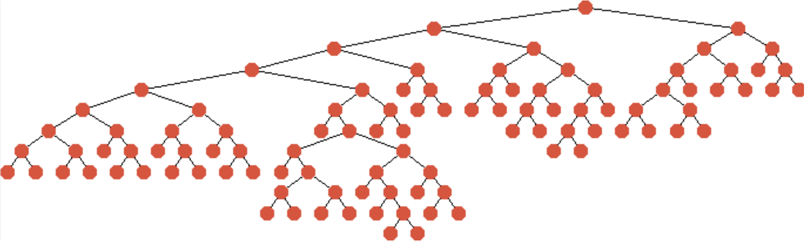


CSSE 230 Day 8

Binary Tree Iterators

After today, you should be able to…
… implement a simple iterator for trees
… implement _lazy_ iterators for trees

Announcements

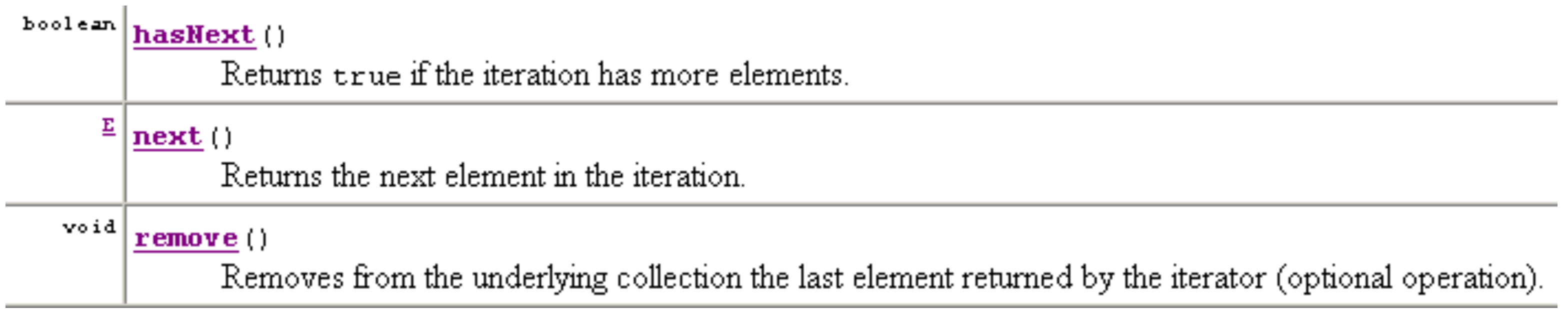
- Stacks & Queues Partner Evaluation done?
- Doublets progress?
  - Overview of workflow
  - Questions?

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?

What's an iterator?

- In Java, specified by `java.util.Iterator<E>`



`boolean`	`hasNext()` Returns `true` if the iteration has more elements.
`E`	`next()` Returns the next element in the iteration.
`void`	`remove()` Removes from the underlying collection the last element returned by the iterator (optional operation).

Implement an iterator using our toArrayList().

- Pros: easy to write.
- So let’s recall or write toArrayList() now and use it.
- Cons? We’ll see shortly!

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?

- 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)
- How could the design be extended to create lazy in-order and post-order iterators?

Work time

A good goal would be to complete Milestone 1 of BinarySearchTrees by next class