CSSE 230 Day 9

Binary Tree Iterators

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

Announcements

- Partner Evaluation done?

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.

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…

Work time

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