Tree Iteration

CS136

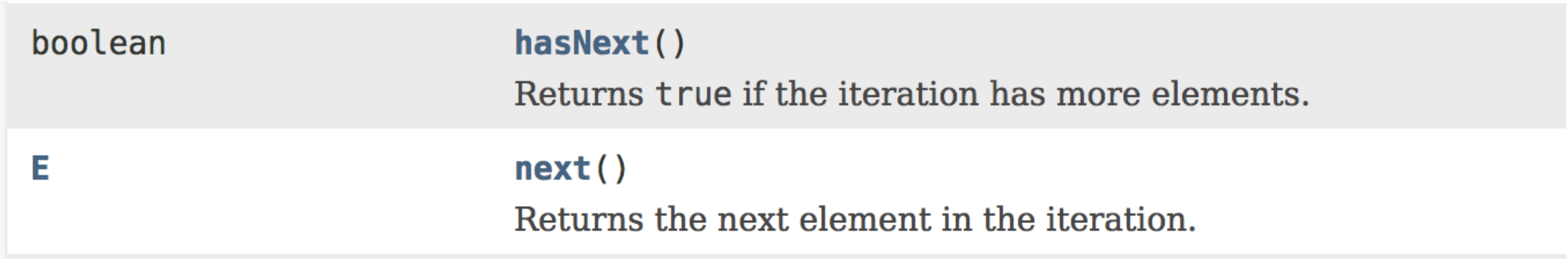
Review

- Trees contain nodes connected by branches (edges)
- Root
- Leaves
- Degree
- Path
- Parent, child, sibling, grandparent, etc.
- Height/depth
- Rule of thumb: most operations should be O(lg *n*) for *n* nodes on a balanced tree

Iteration

- Node Visiting Order
  - Depth First
  - Breadth First
  - Best First
- Binary Tree Processing Order
  - In order
  - Preorder
  - Postorder
  - Level order

Java Iterator Interface



`boolean`	`hasNext()` Returns `true` if the iteration has more elements.
`E`	`next()` Returns the next element in the iteration.

Depth First

- Visit all descendants of one child before moving on to the next child
- Like wandering a maze
- Uses a **stack**

Breadth First

- Visit all children before other descendants
- Like playing a game naively
- Uses a **queue**

Best First

- Given a heuristic for how “important” a subtree is, visit them in that order
- Like playing a game well
- Uses a **priority queue** [a queue that reorders]

Binary Tree Iteration

- When do we print/process a node’s **value**? (not when do we visit the node)
  - Preorder
  - Postorder
  - In order
  - Level order
- (see Bailey)