Lecture 17: Iterators, Expression Trees & Array Representation of Trees

CS 62
Spring 2015
Kim Bruce & America Chambers

Look at BinaryTree.java

Notice leaves are nodes w/null values

Iterators

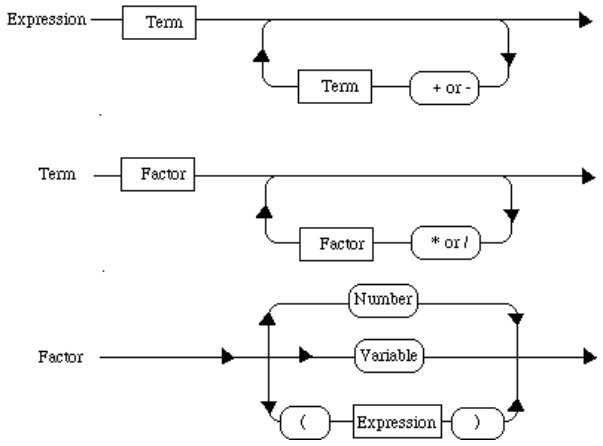
- Pre-order: root, left subtree, right subtree
- Post-order: left subtree, right subtree, root
- In-order: left subtree, root, right subtree.

in-order

```
if (!isEmpty()){
  left.inOrder()
  doSomething to this.value()
  right.inOrder()
}
```

Iterative Iterators -- see code

Parsing Expressions



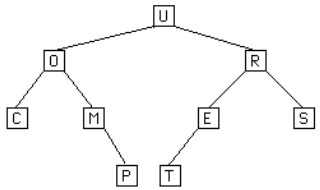
Representing Expressions

- Represent 3 * 7 + 6 / 2 - (3 + 7) as tree
  - Parser builds tree
  - Send message to tree to print or evaluate
- Mutual recursion in parser
- Different classes for different kinds of nodes.
- See Parser code

Array Representations of Trees

Array Representation

- data[0..n-1] can hold values in trees
  - left subtree of node i in 2*i+1, right in 2*i+2,
  - parent in (i-1)/2



Indices:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
data[]:	U	O	R	C	M	E	S	-	-	-	P	T	-	-	-

Array Representation: Efficiency

- Tree of height h, takes $2^{h+1}-1$ slots, even if only has O(h) elements
  - Bad for long, skinny trees
  - Good for full or complete trees.
- Recall complete tree is full except possibly bottom level and has all leaves at that level in leftmost positions.

Min-Heap

- Min-Heap H is complete binary tree s.t.
  - H is empty, or
  - Both of the following hold:
    - The value in root position is smallest value in H
    - The left and right subtrees of H are also heaps.
      Equivalent to saying parent ≤ both left and right children
- Excellent implementation for priority queue
  - Dequeue elements w/lowest priority values before higher