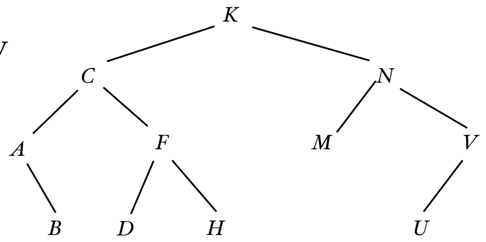
# Lecture 16: Binary Trees

CS 62
Spring 2015
Kim Bruce & America Chambers

# Counting

- Lemma: If T is a binary tree then at level k, T has ≤ $2^k$ nodes.



- Theorem: If T has height h, then # nodes in T ≤ $2^{h+1} - 1$.
- Equivalently, if T has n nodes then

$$n - 1 \geq h \geq \log(n+1) - 1$$

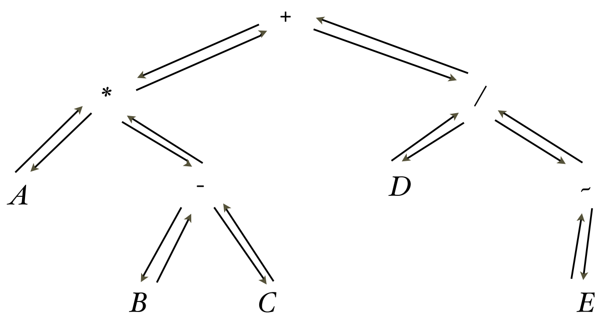
# Assignment

- Postfix calculator
  - State class represents memory of calculator
  - Uses stack to represent partial calculations
  - Separate listener for each digit and operator
    - We provide OpButtonListener — you do DigitButtonListener
    - Accumulate numbers
- Must create JUnit tests for State class
- Do simplified version first that requires "enter" before operation.

# Binary Trees in Java

- No implementation in standard Java libraries
- Structure5 has BinaryTree<E> class, but no interface.
- Like doubly-linked list:
  - value: E
  - parent, left, right: BinaryTree<E>

# Linked Representation



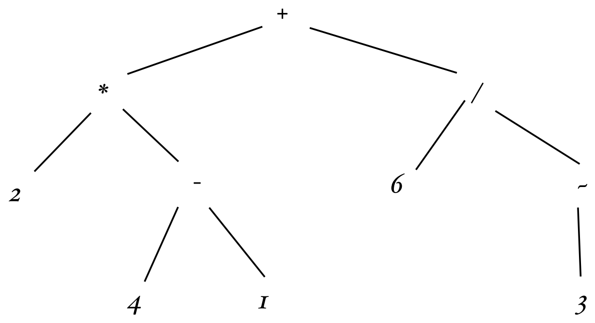
# See BinaryTreeInterface.java

# Tree Traversals

- Traversals:
  - Pre-Order: root, left subtree, right subtree
  - In-Order: left subtree, root, right subtree
  - Post-Order: left subtree, right subtree, root
- Most algorithms have two parts:
  - Build tree
  - Traverse tree, performing operations on nodes

# Evaluate Expression Tree

- Evaluate left subtree, right subtree, perform operation at root.
- Generate stack-based code to evaluate: post-order



# Java Virtual Machine

```
int simple(int m, int n) {
   return (m + n - 1)
}
```

*translates to:*

```
method int simple(int, int)
0 iload_1
1 iload_2
2 iadd
3 iconst_1
4 isub
5 ireturn
```

# Animals Game

- Guess animal using only true-false questions.
- See demo program

# 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.