Binary Search Tree Runtime Analysis

Version 1

Draw the binary search tree that results from the following sequence of adds:
20, 12, 16, 32, 27, 7, 37

How many steps does `lookup(7)` take (i.e., how many nodes are accessed)?

How many steps does `lookup(16)` take?

Generalize: In terms of n (the number of items in the tree), how long does `lookup` take in the worst case for this tree?

Binary Search Tree Runtime Analysis

Version 2

Draw the binary search tree that results from the following sequence of adds:
37, 32, 27, 20, 16, 12, 7

How many steps does `lookup(7)` take (i.e., how many nodes are accessed)?

How many steps does `lookup(16)` take?

Generalize: In terms of n (the number of items in the tree), how long does `lookup` take in the worst case for this tree?

Binary Search Tree Runtime Analysis

Putting it Together

1. What's the worst-case runtime for `lookup` that *any* tree and input could have?

2. What's the best-case runtime for `lookup` that *any* tree and input could have?

3. Suppose you're given a particular tree. What can you say about the worst-case runtime for `lookup` for *that particular* tree? Can you give a better bound than your result from part (1) above?

4. What's the shortest possible height, h, for a tree with n nodes? What would you expect the height of the tree to be, on average? What does this tell you about the *average* runtime for `lookup`?