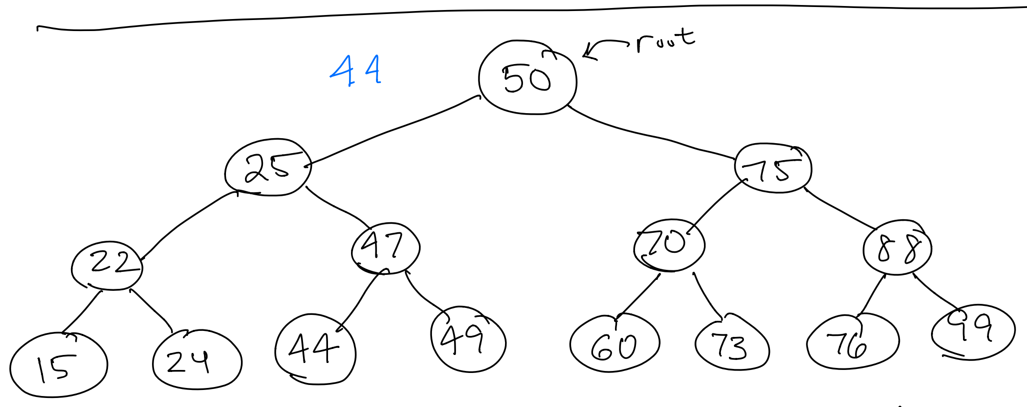
Last day of BSTs (Oct 22)

Searching a BST (pseudocode)

```
boolean search(node, searchKey) {
    if node == null
        return false
    else if searchKey == node.data
        return true;
    else if searchKey < node.data
        return search(node.left, searchKey)
    else    // searchKey > node.data
        return search(node.right, searchKey)
```

$O(\log n)$



Count nodes = 15
height = 4

n = # of nodes in the BST
h = height of the tree

$h \approx \log_2(n)$

$2^h \approx n$

Big-oh - of preorder/inorder/postorder? $O(n)$

Alg for a Binary tree (including BSTs)

Look at # of recursive calls being made

→ If there is only one recursive call (you examine the left or right branch but not both) → O(log n)

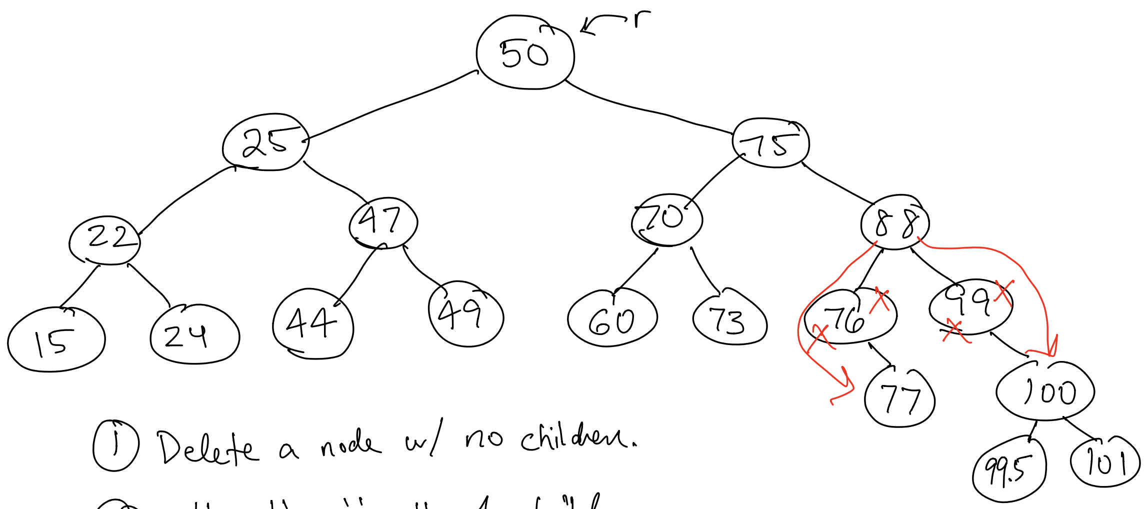
→ If there are 2 recursive calls (you examine both branches) → O(n)

Big-oh of ADD? O(log n)

Suppose we want to add ALL the nodes to the BST?
(Start from an empty BST & Add n total nodes)
(Total) Big-oh → O(n·log n)

SortedList
Add → O(n)
Add n items → O(n^2)

Deletion in a BST



① Delete a node w/ no children.

② " " " " 1 child.

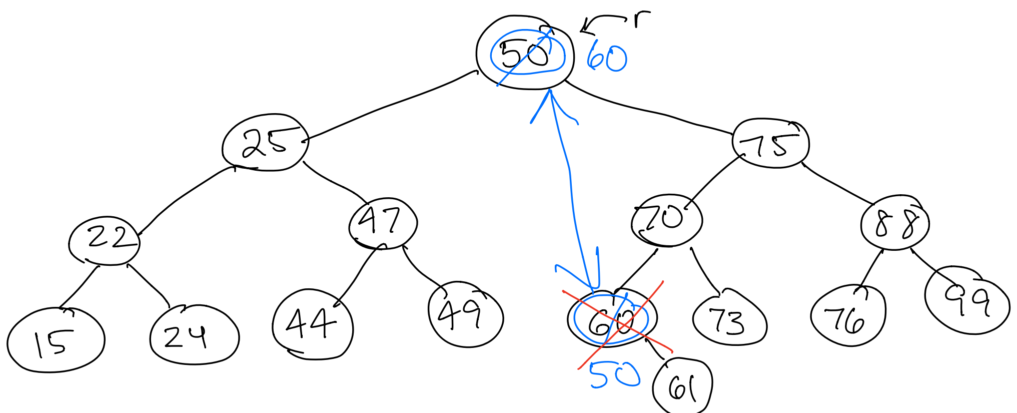
Bring up the child node (+ the entire subtree) to take the place of the parent.

③ Delete a node w/ 2 children. (delete 50)

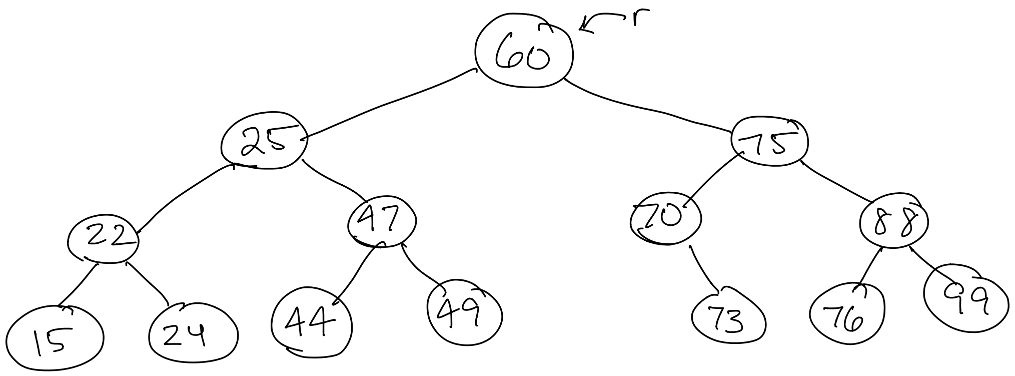
Find the nodes inorder successor. (find 50's inorder successor)

↳ Go right, then as far left as you can.

Exchange the data in those 2 nodes (node to delete & inorder successor)



Delete the node where the inorder successor used to be.



Delete 22

↳ inorder succ = 24