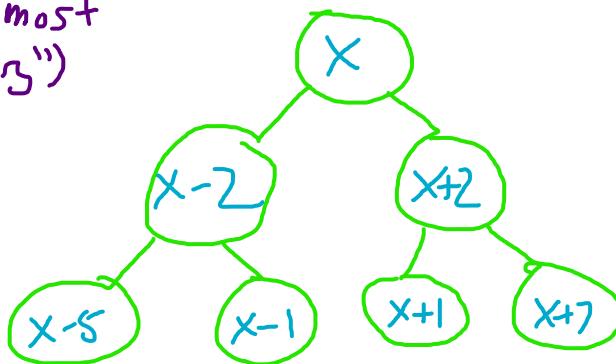


# Binary Search Trees (BSTs)

## BST Properties

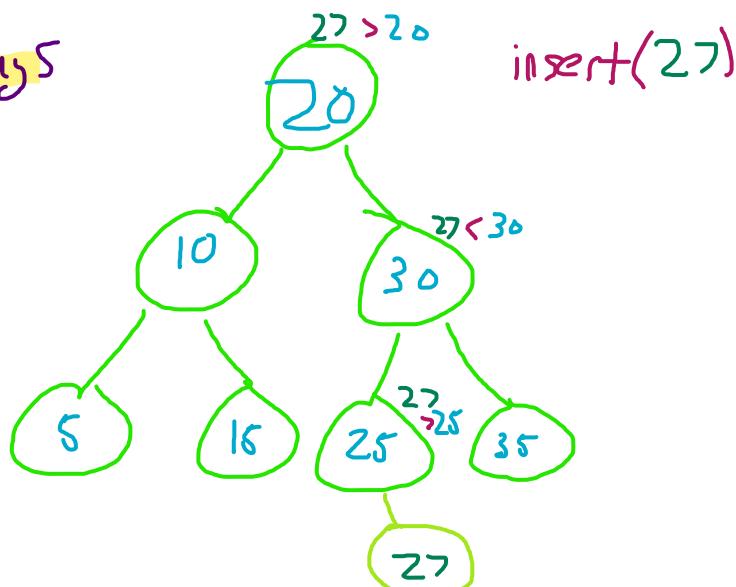
- Every key in the left subtree of a node  $X$  is always less than  $X$ .
- Every key in the right subtree of a node  $X$  is always greater than  $X$ .
- Each node has at most 2 children ("binary")
- NO DUPLICATES



## BST.insert(X)

- ① Start at curr-node = root
  - ② if  $X > \text{curr-node.value}$ :  
 $\text{curr-node} = \text{curr-node.right}$   
else:  
 $\text{curr-node} = \text{curr-node.left}$
  - ③ insert new Node(X) at current location
- repeat until curr-node is NULL

\* New nodes always added as leaf

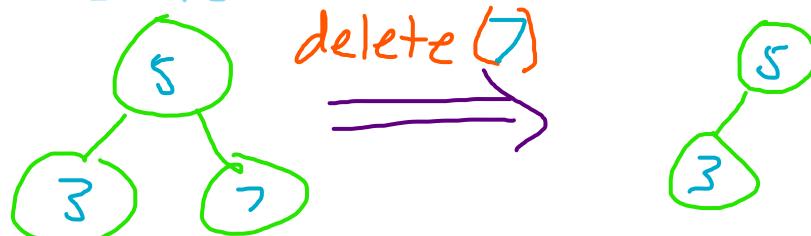


Challenge: how do we code this?

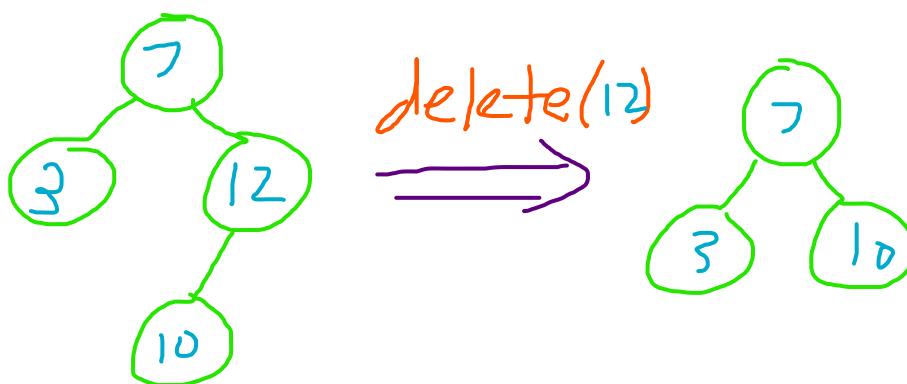
## BST.delete(X)

3 cases:

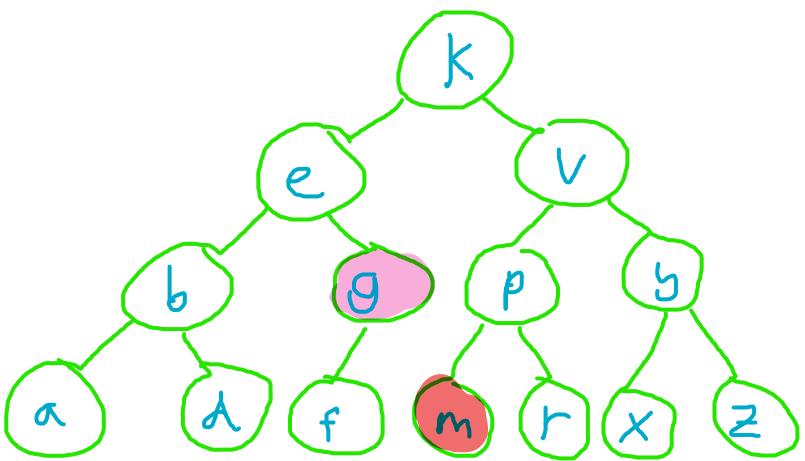
- ① If node to be deleted has 0 children  
just delete!



- ② If node to be deleted has 1 child  
link parent to child node to be deleted.



- ③ If node to be deleted has 2 children  
to be deleted node is replaced by predecessor  
(largest node of left subtree of node) or successor  
(smallest node of right subtree of node)



$\text{delete}(k)$

or

$\text{delete}(k)$

