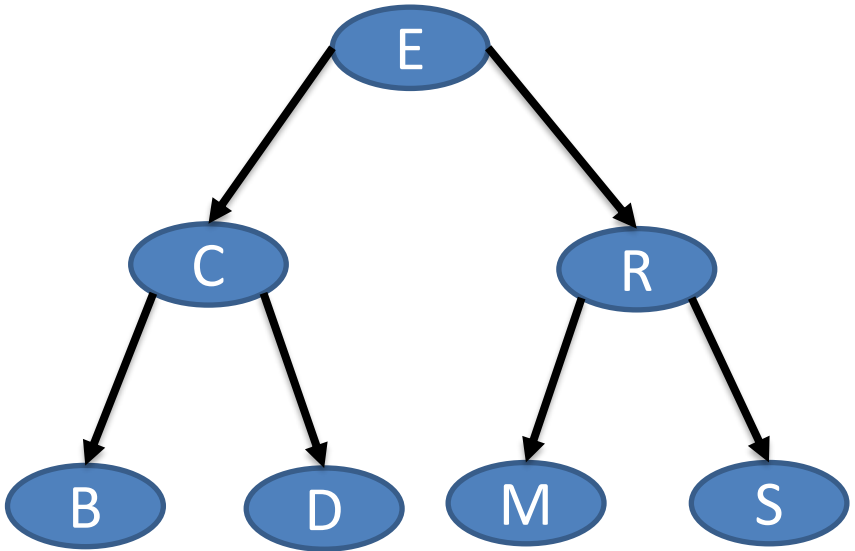
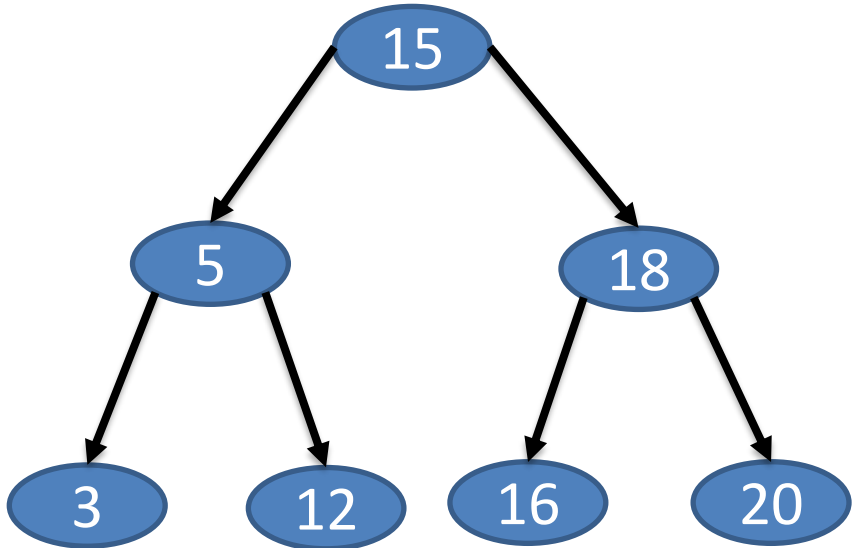
Binary Search Trees

CS223: Data Structures

Binary search trees (BST)

- A tree where each node has 0, 1, or 2 children
- Children are labeled either left or right
  - Nodes have 2 pointers (left and right)
- Each node is larger than all nodes in the left subtree and smaller (or equal in some developments) than all nodes in the right subtree

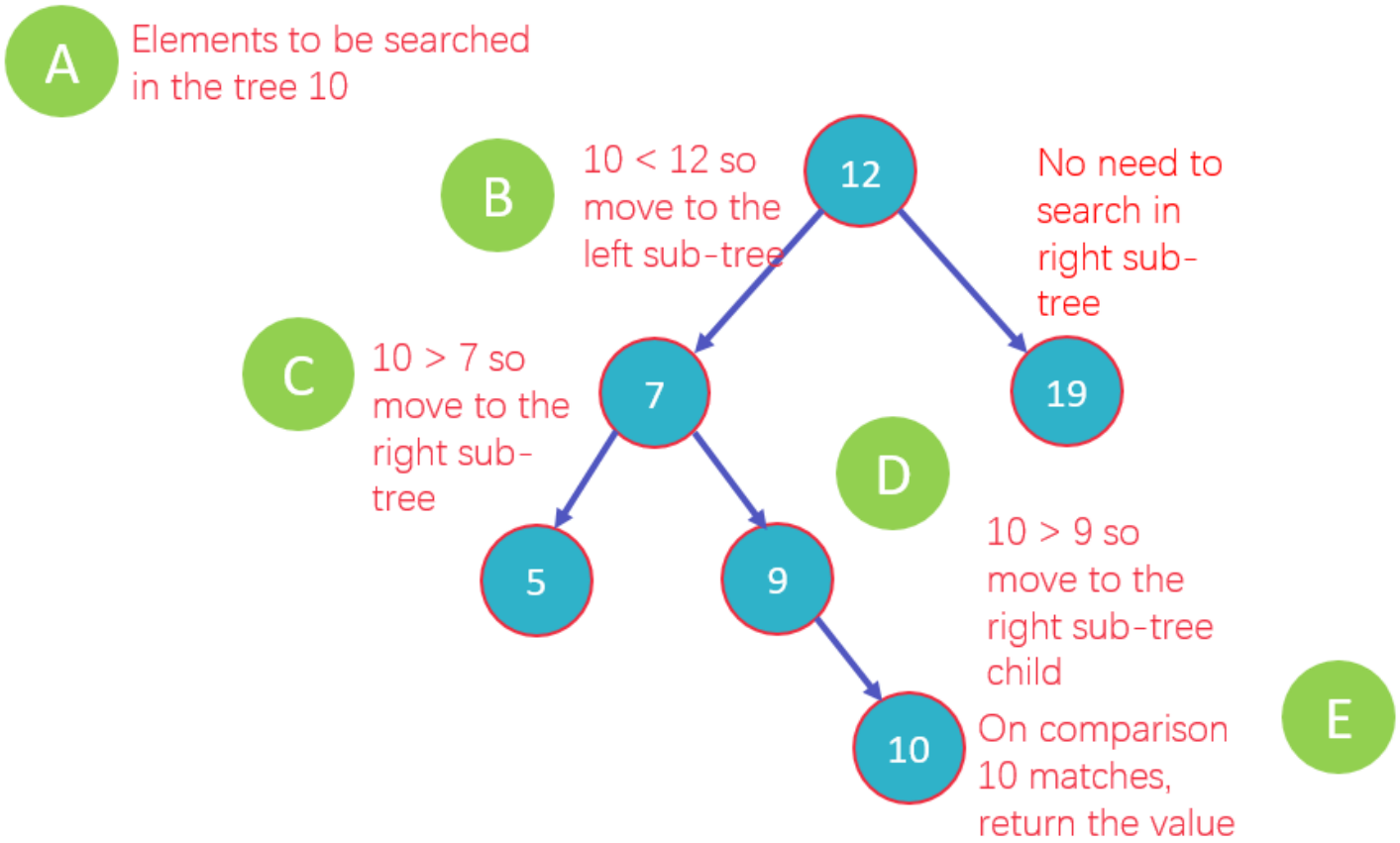




Search BST for a value

- Check, whether value in current node and searched value are equal. If so, **value is found**. Otherwise,
- if searched value is less, than the node's value:
  - if current node has no left child, **searched value doesn't exist in the BST**;
  - otherwise, handle the left child with the same algorithm.
- if a new value is greater, than the node's value:
  - if current node has no right child, **searched value doesn't exist in the BST**;
  - otherwise, handle the right child with the same algorithm.

Search Operation



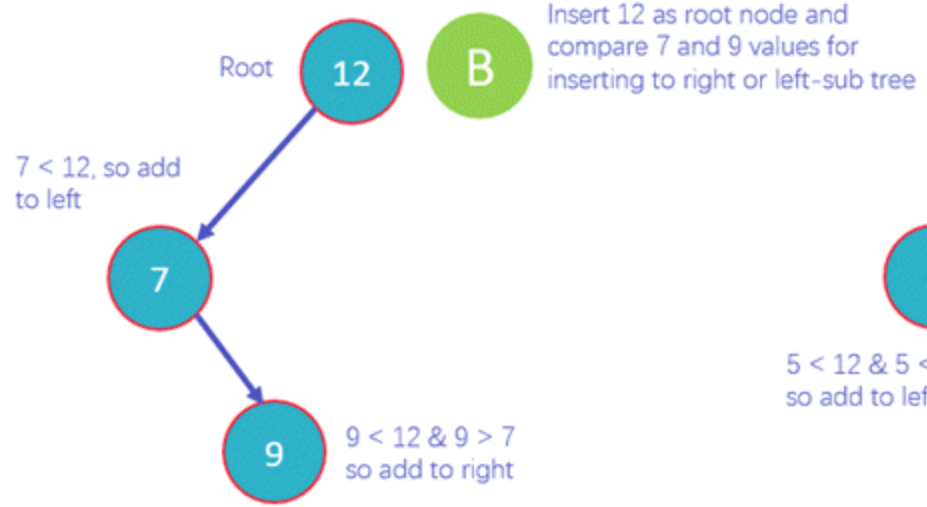
Insert into BST

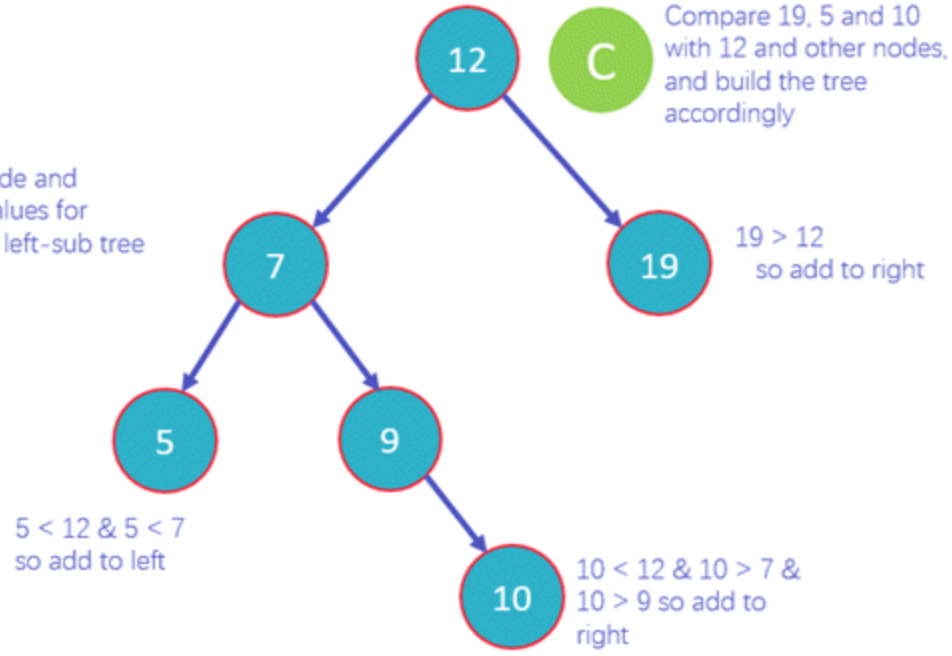
- Check, whether value in current node and a new value are equal. If so, duplicate is found. Otherwise,
- if a new value is less, than the node's value:
  - if a current node has no left child, place for insertion has been found;
  - otherwise, handle the left child with the same algorithm.
- if a new value is greater, than the node's value:
  - if a current node has no right child, place for insertion has been found;
  - otherwise, handle the right child with the same algorithm.

Insert Operation

A

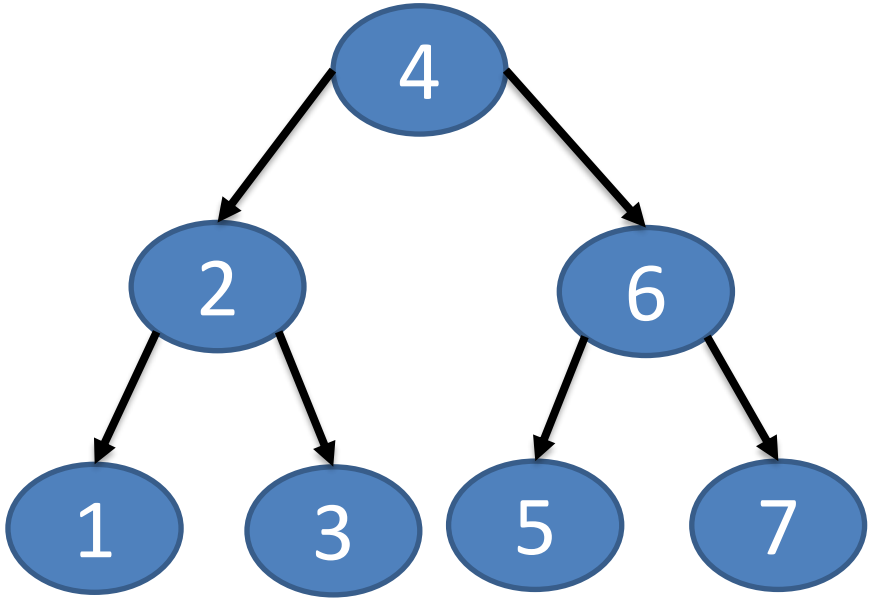
Elements to be inserted in the tree from left to right:
12, 7, 9, 19, 5, 10





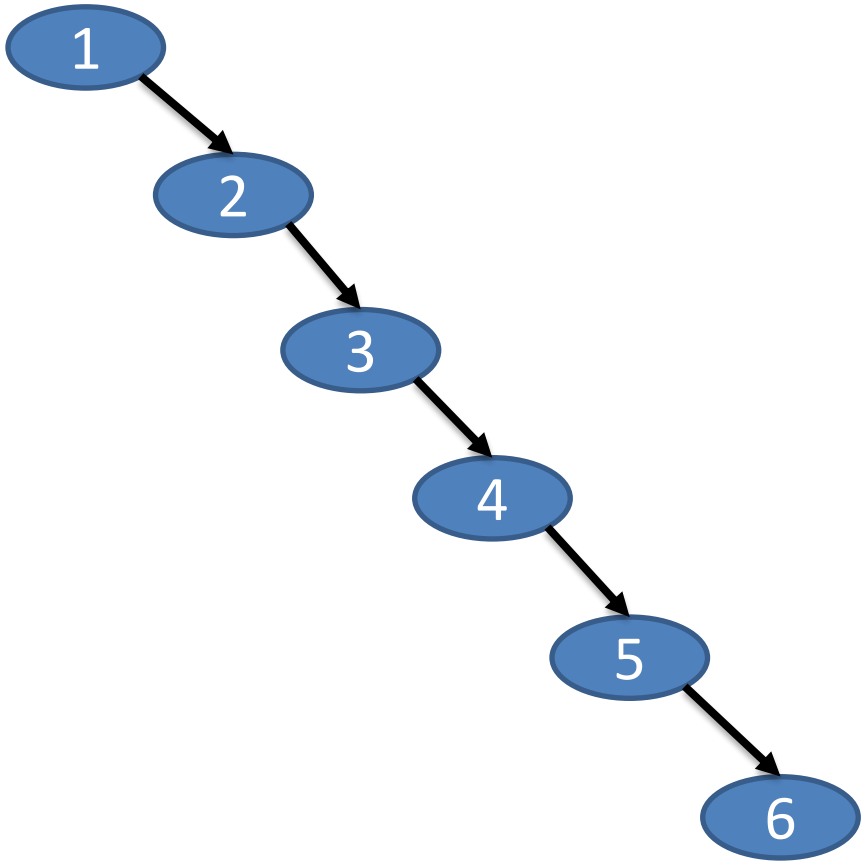
Lets build one

Build a BST for the this sequence of values 4,2,3,6,5,7,1



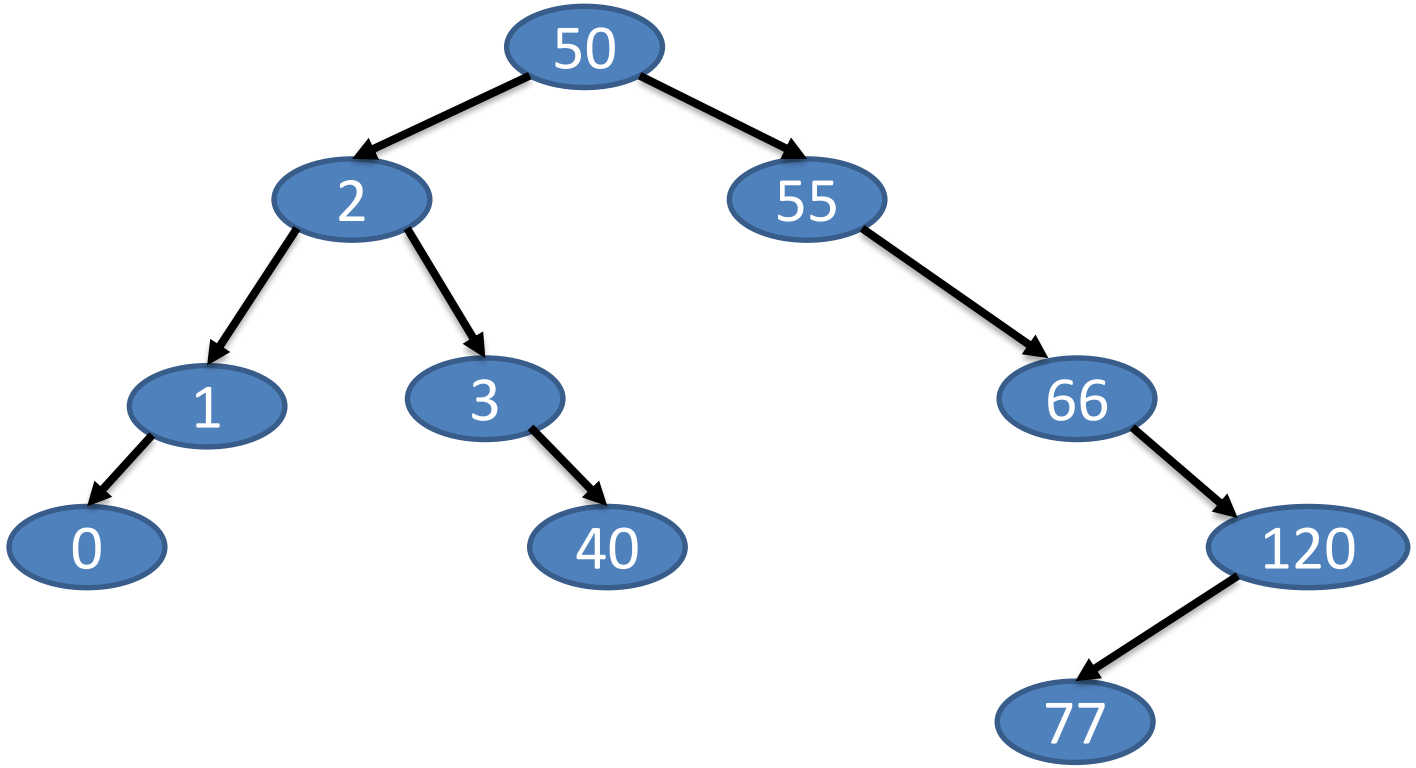
Lets build another

Build a BST for the this sequence of values 1,2,3,4,5,6



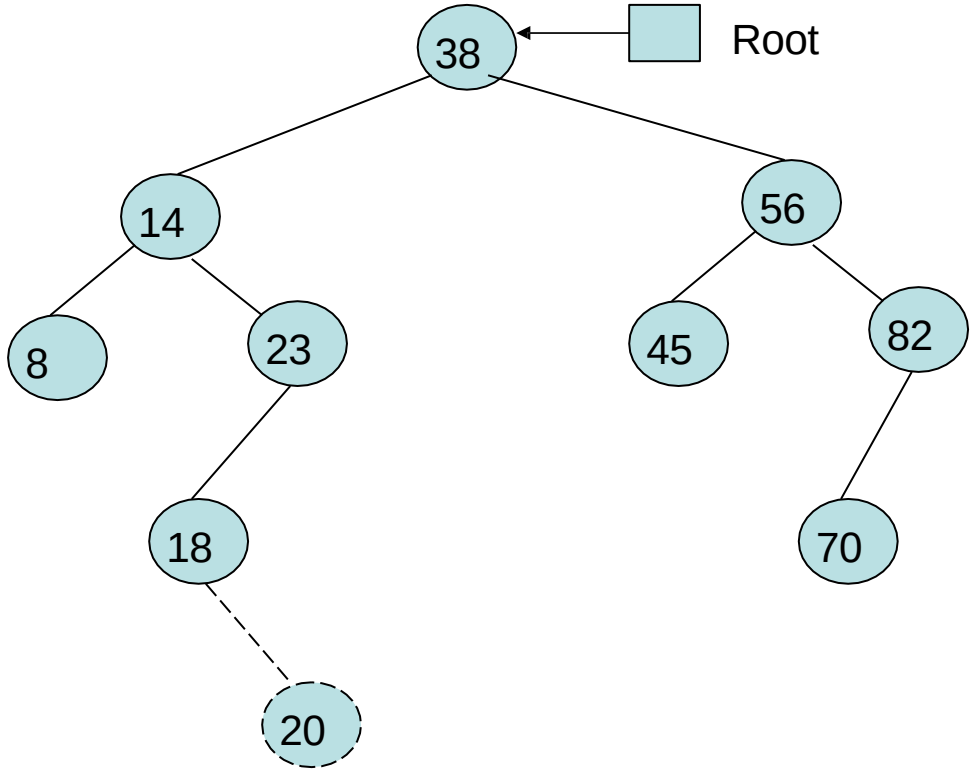
And another one

Build a BST for the this sequence of values 50,2,3,40,55,66,120,1,0,77



Example used for Insert

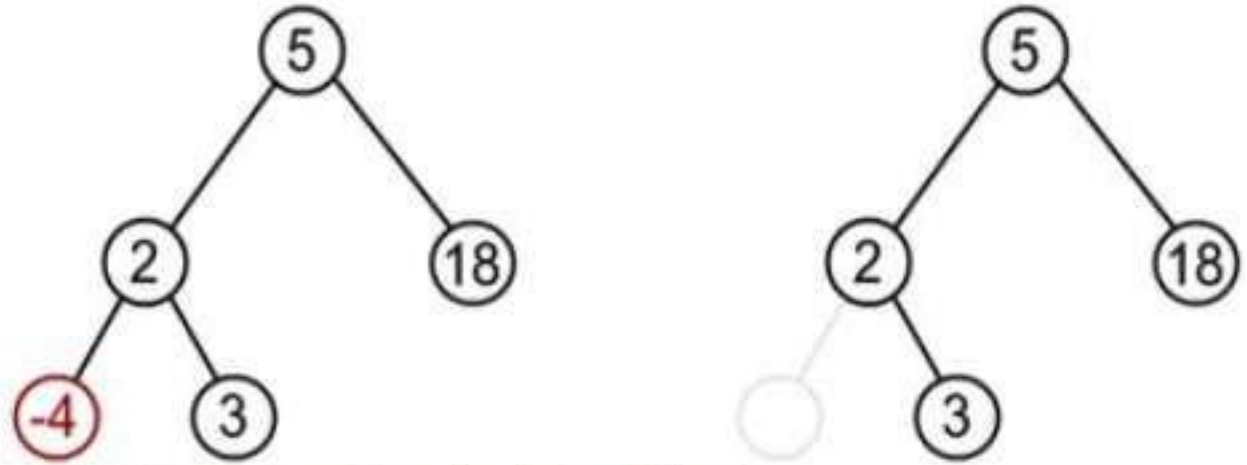
Insert 20



Deleting a node from the BST

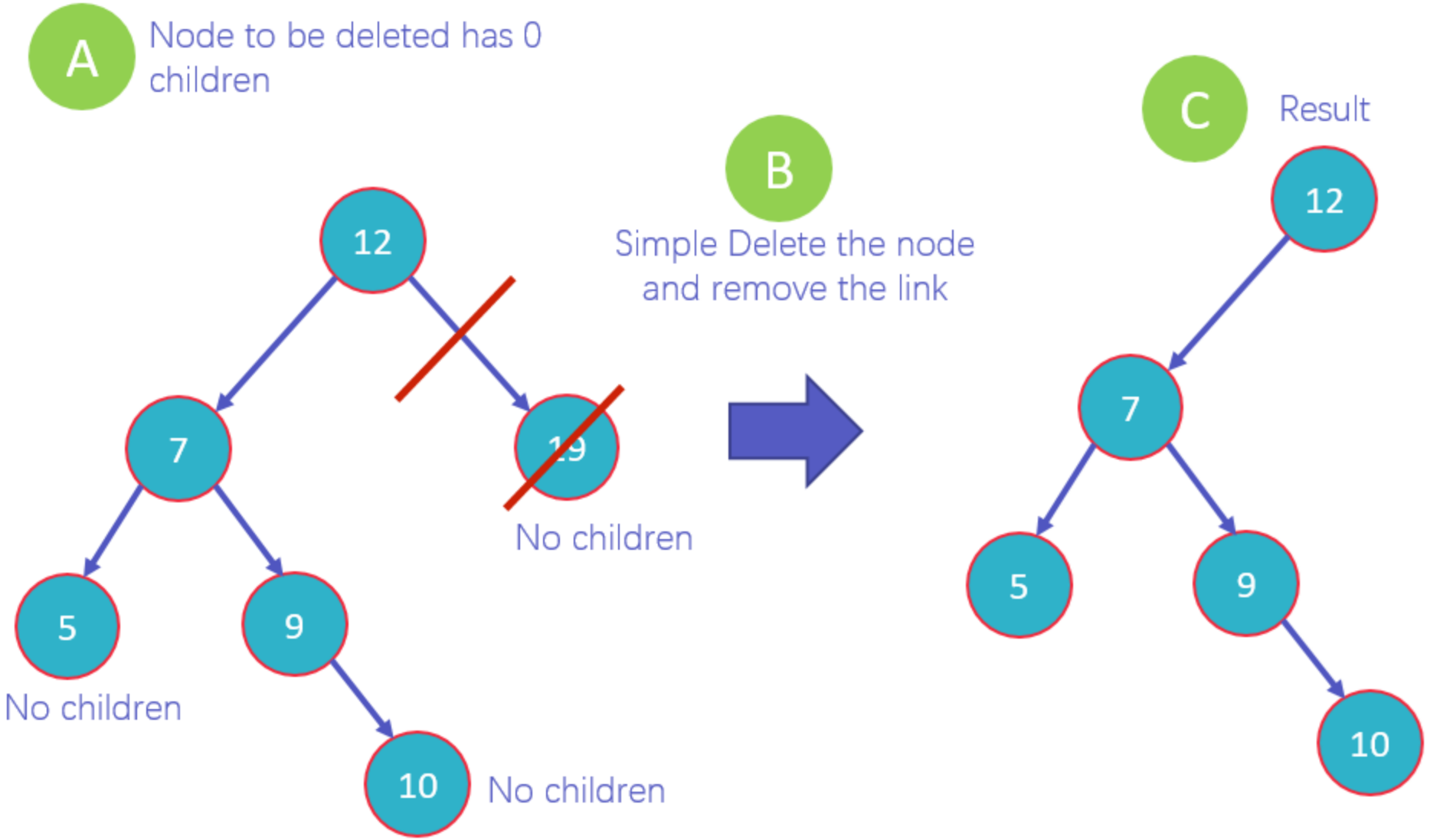
- While deleting a node from BST, there may be three cases:

1. The node to be deleted may be a leaf node:
   - In this case simply delete a node and set null pointer to its parents to the side at which this deleted node exist.



Suppose node to be deleted is -4

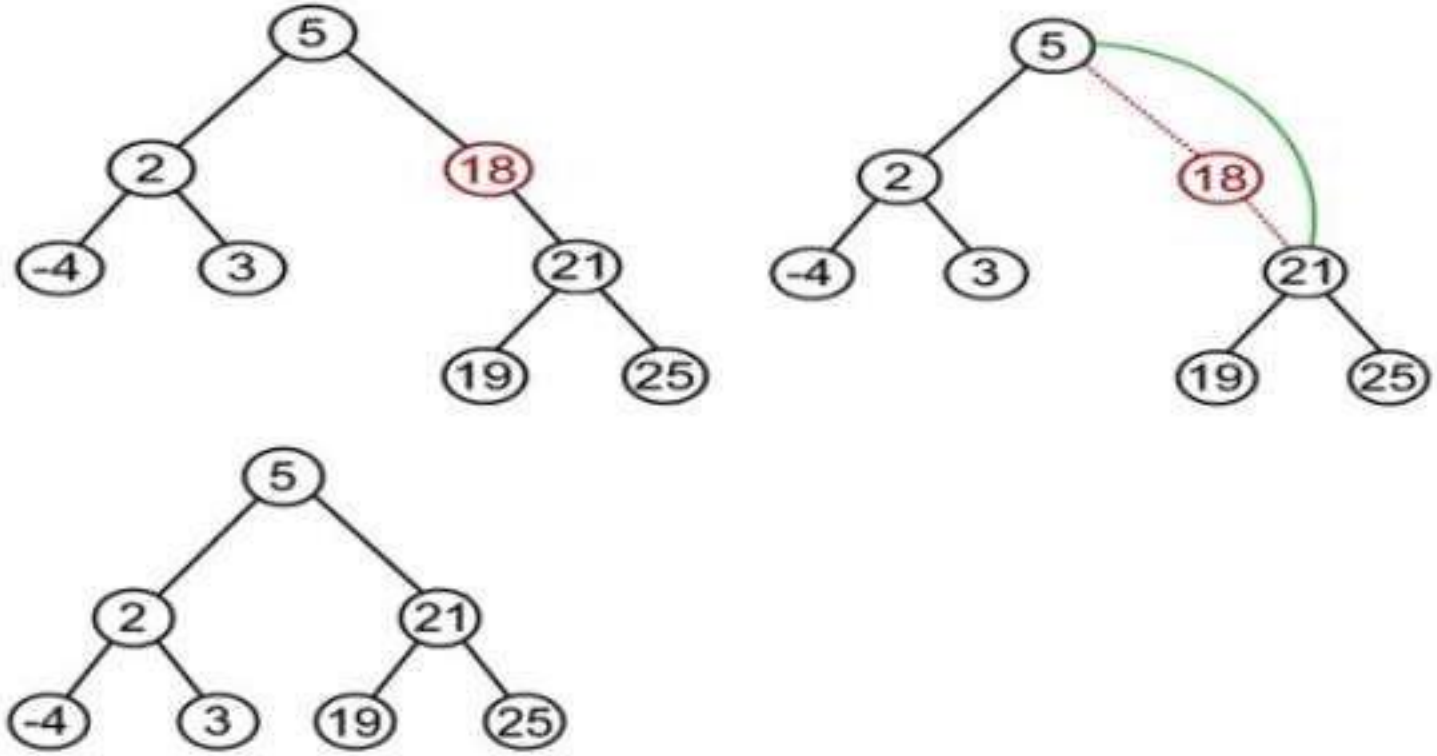
Delete Operation – Case 1



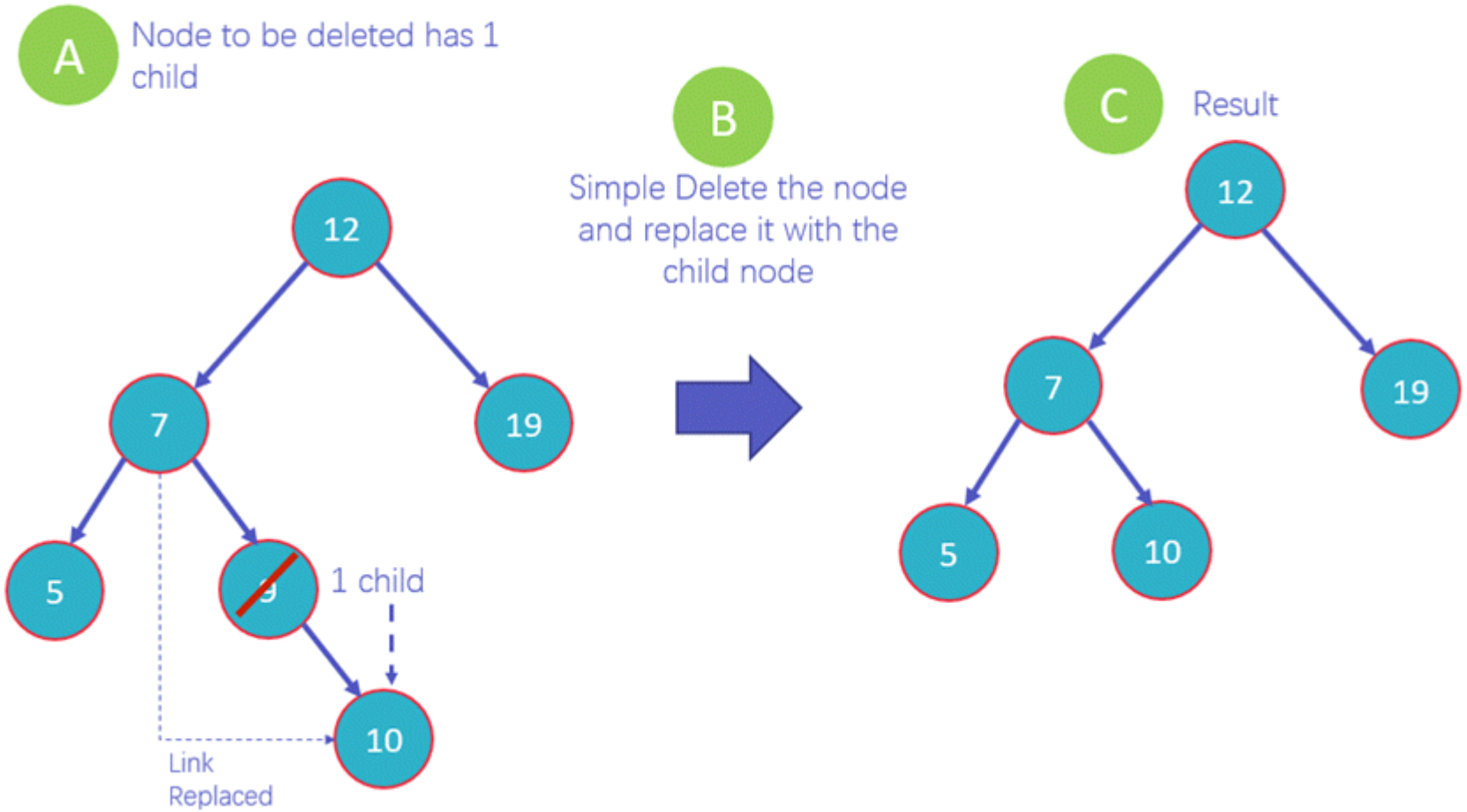
Deleting a node from the BST

2. The node to be deleted has one child
   - In this case the child of the node to be deleted is appended to its parent node.
     - Replace current node with that child.

   Suppose node to be deleted is 18

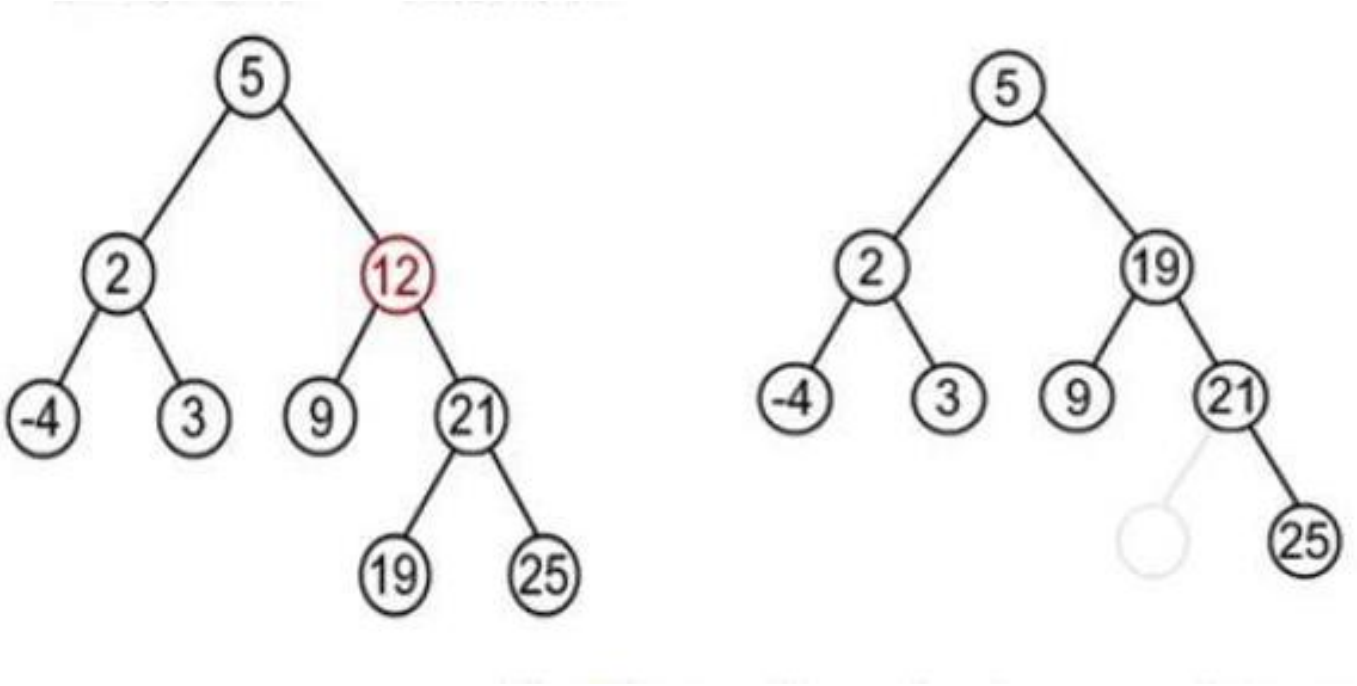


Delete Operation – Case 2



Deleting a node from the BST

3. The node to be deleted has two children
   - Find *largest item* in the *left subtree* or *smallest item* in the *right subtree*.
   - Use it as the parent of the two subtrees



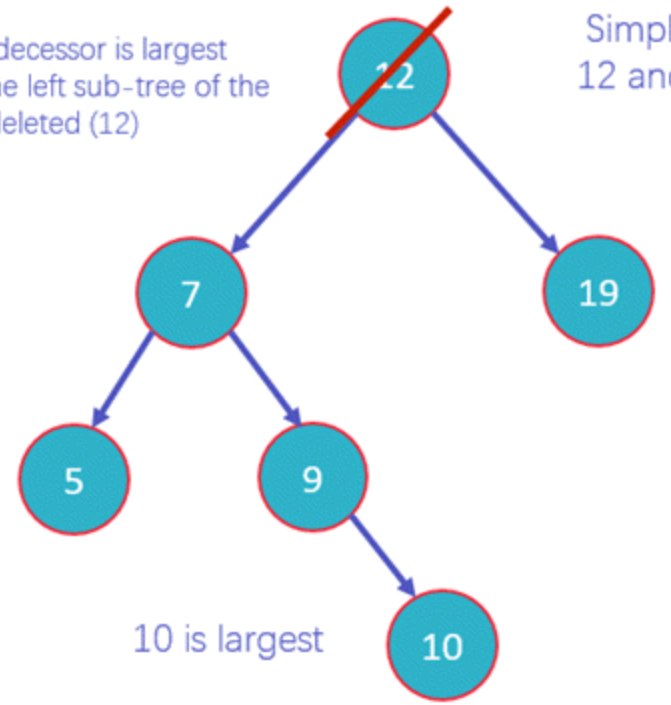
Delete Operation – Case 3

A

Node to be deleted has 2 child
Replace Situation: In Order
Predecessor

In Order predecessor is largest element in the left sub-tree of the node to be deleted (12)

B

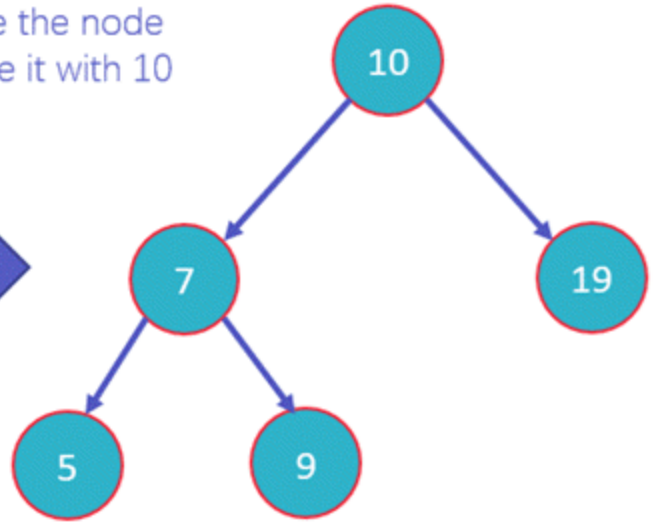


C

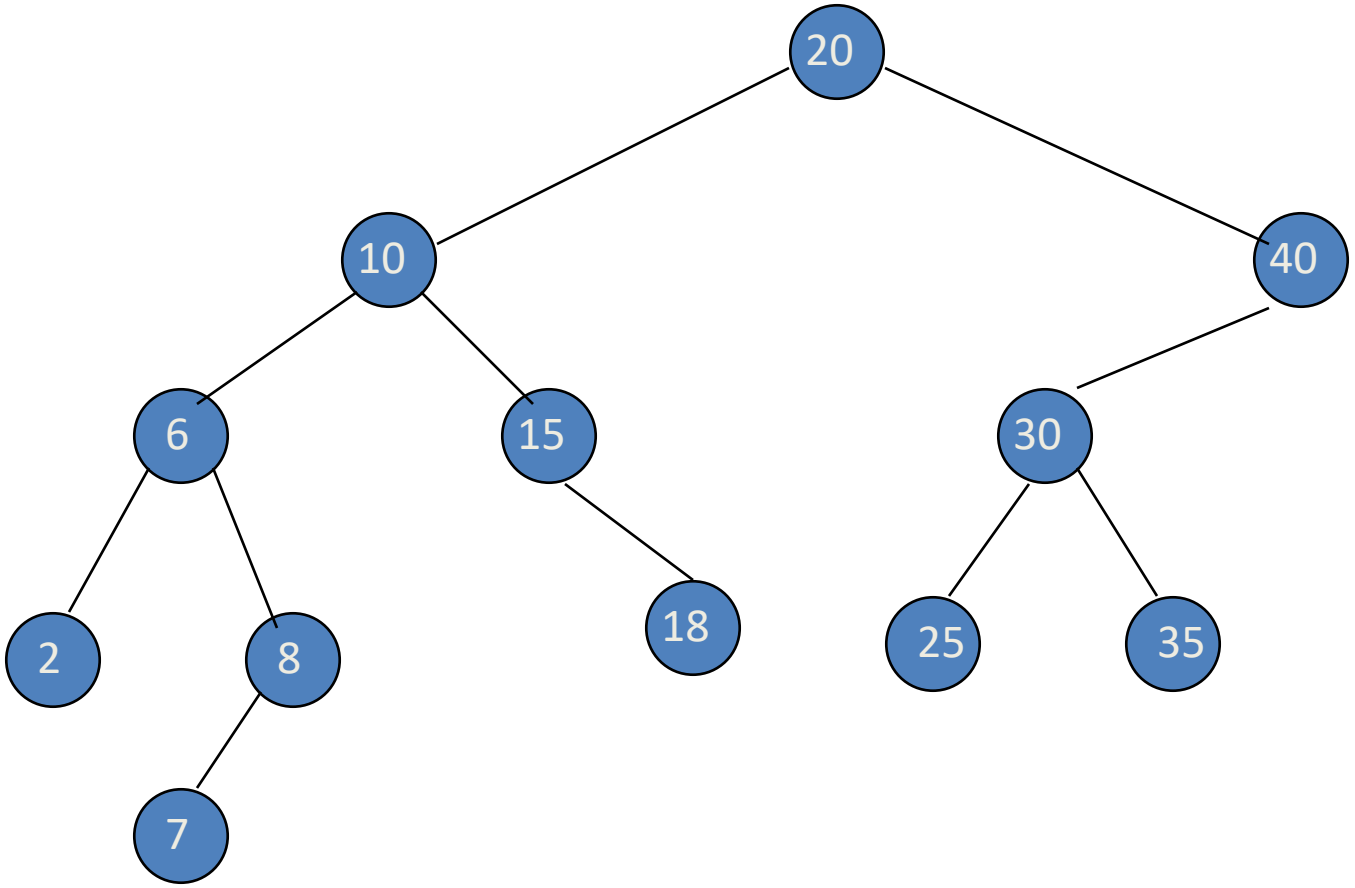
Simple Delete the node
12 and replace it with 10

D

Result

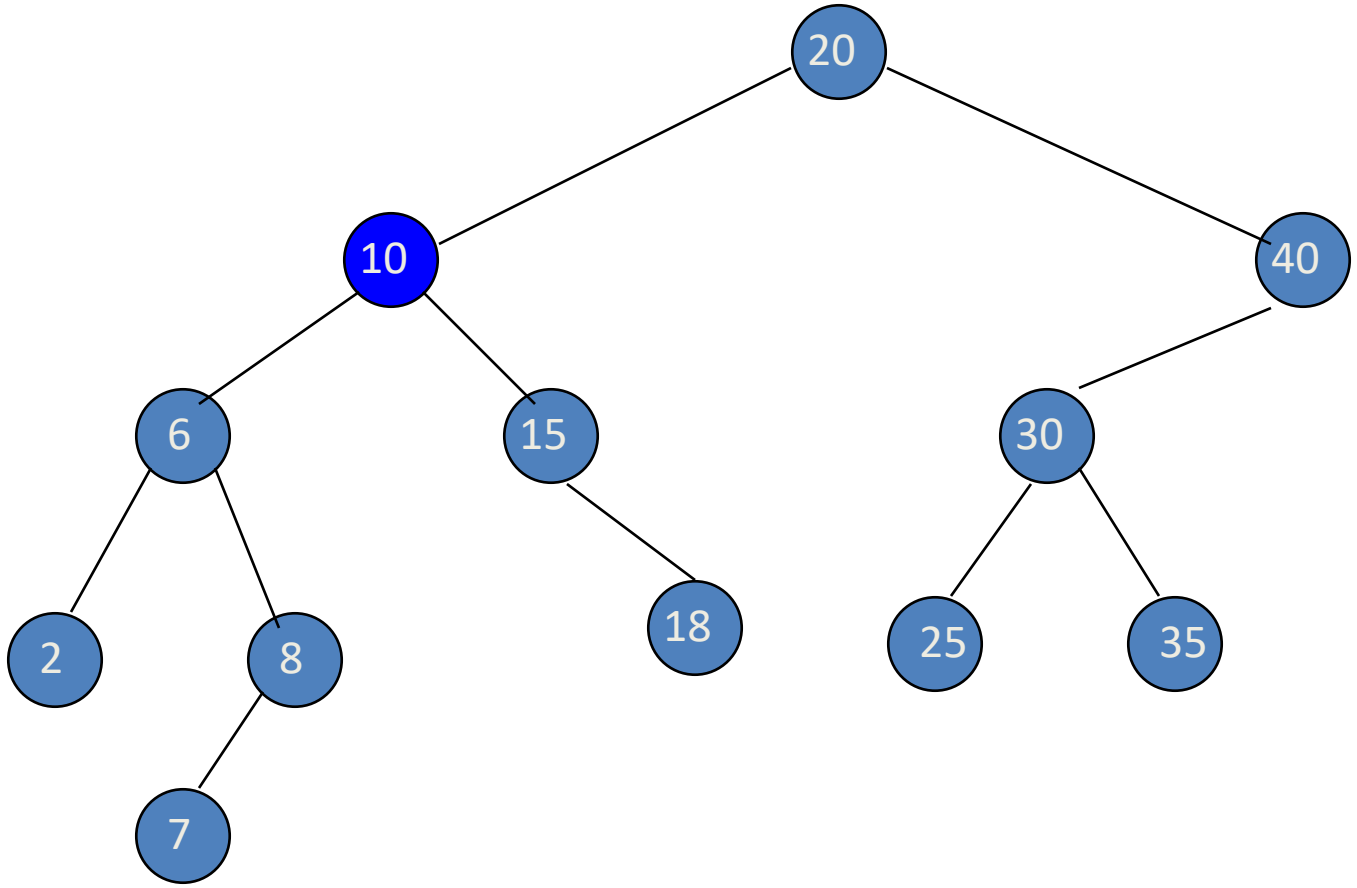


Delete a node that has 2 children



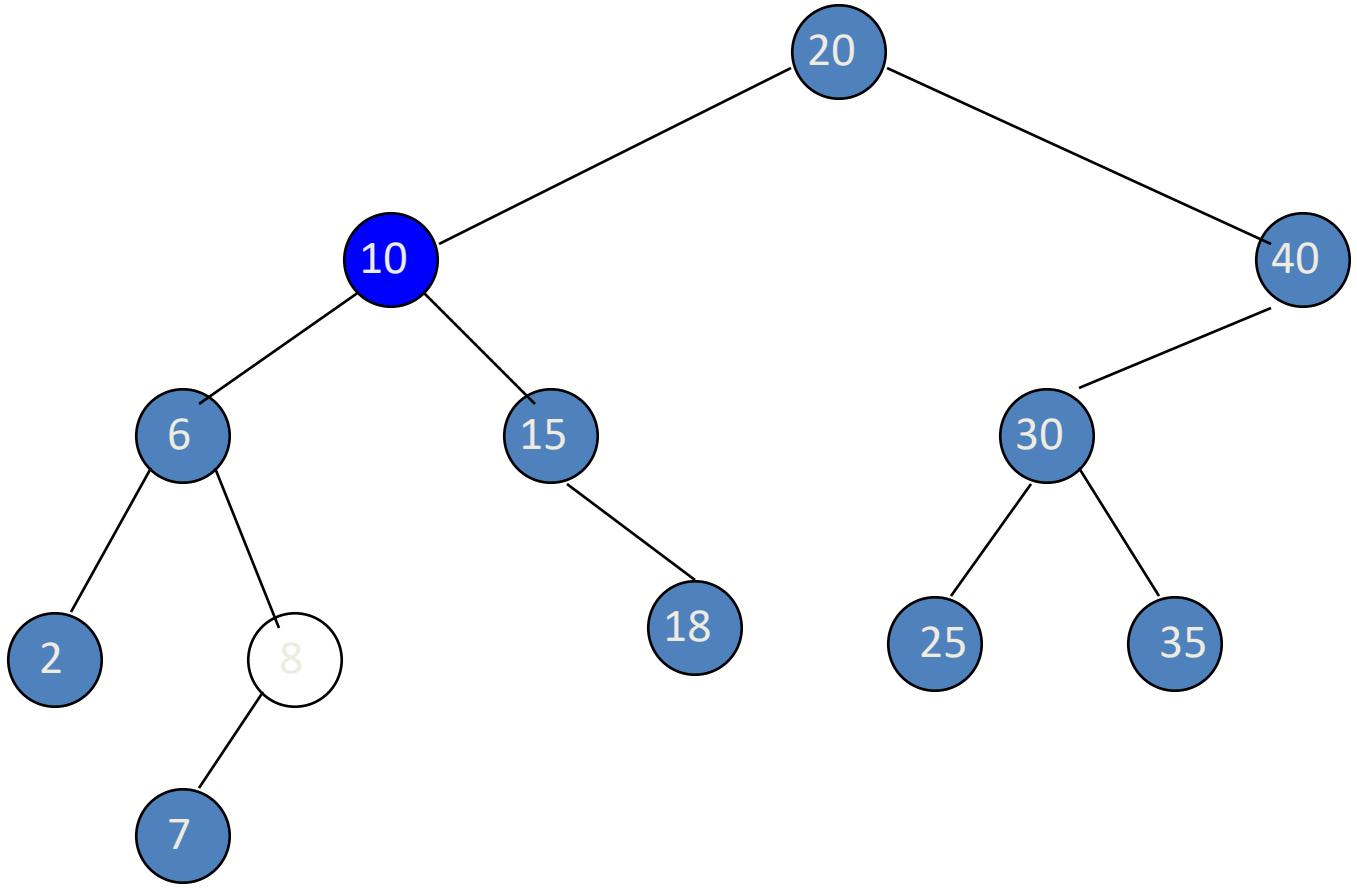
Delete 10

Delete a node that has 2 children



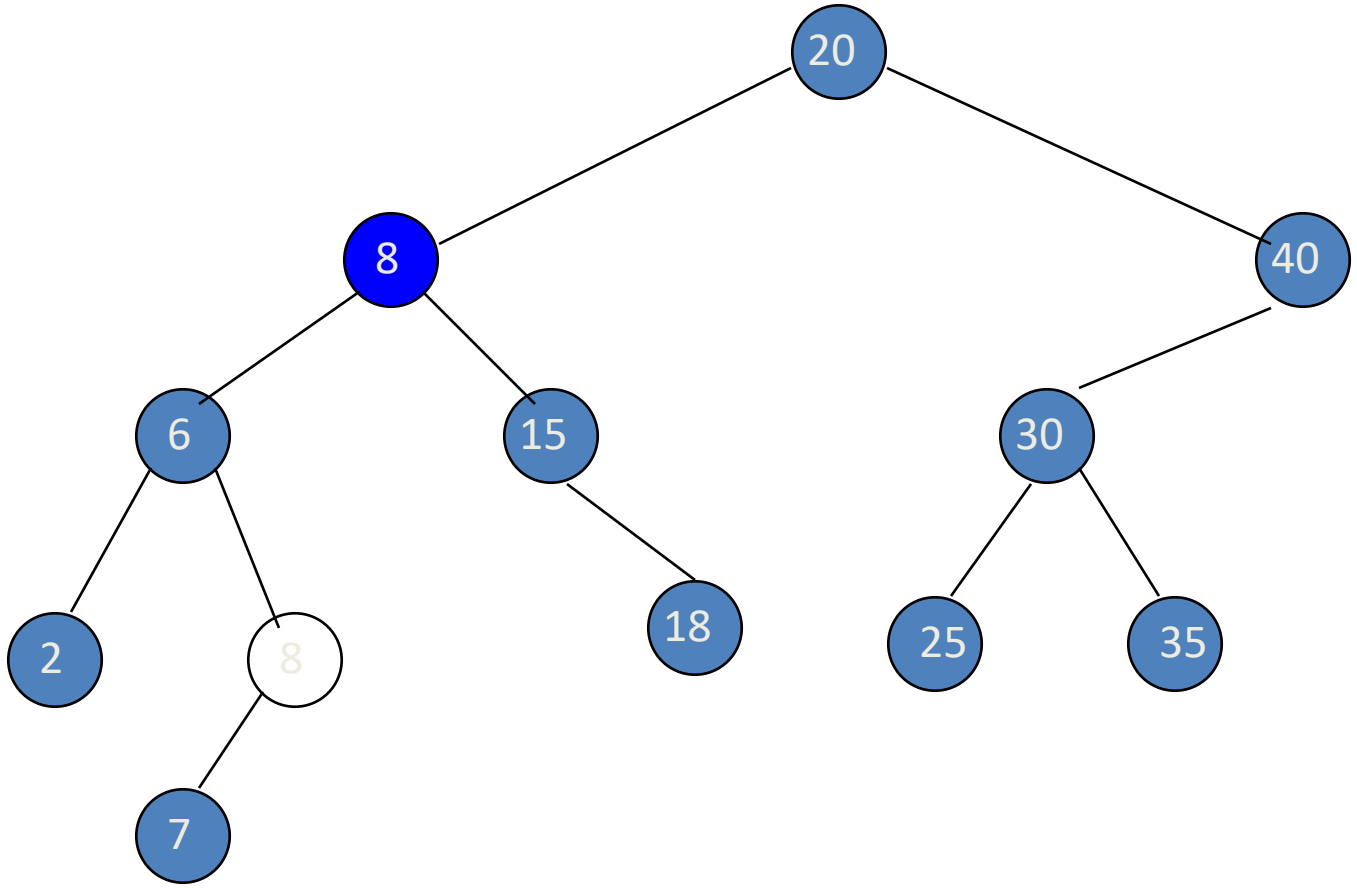
Replace with largest key in left subtree (or smallest in right subtree).

Delete a node that has 2 children



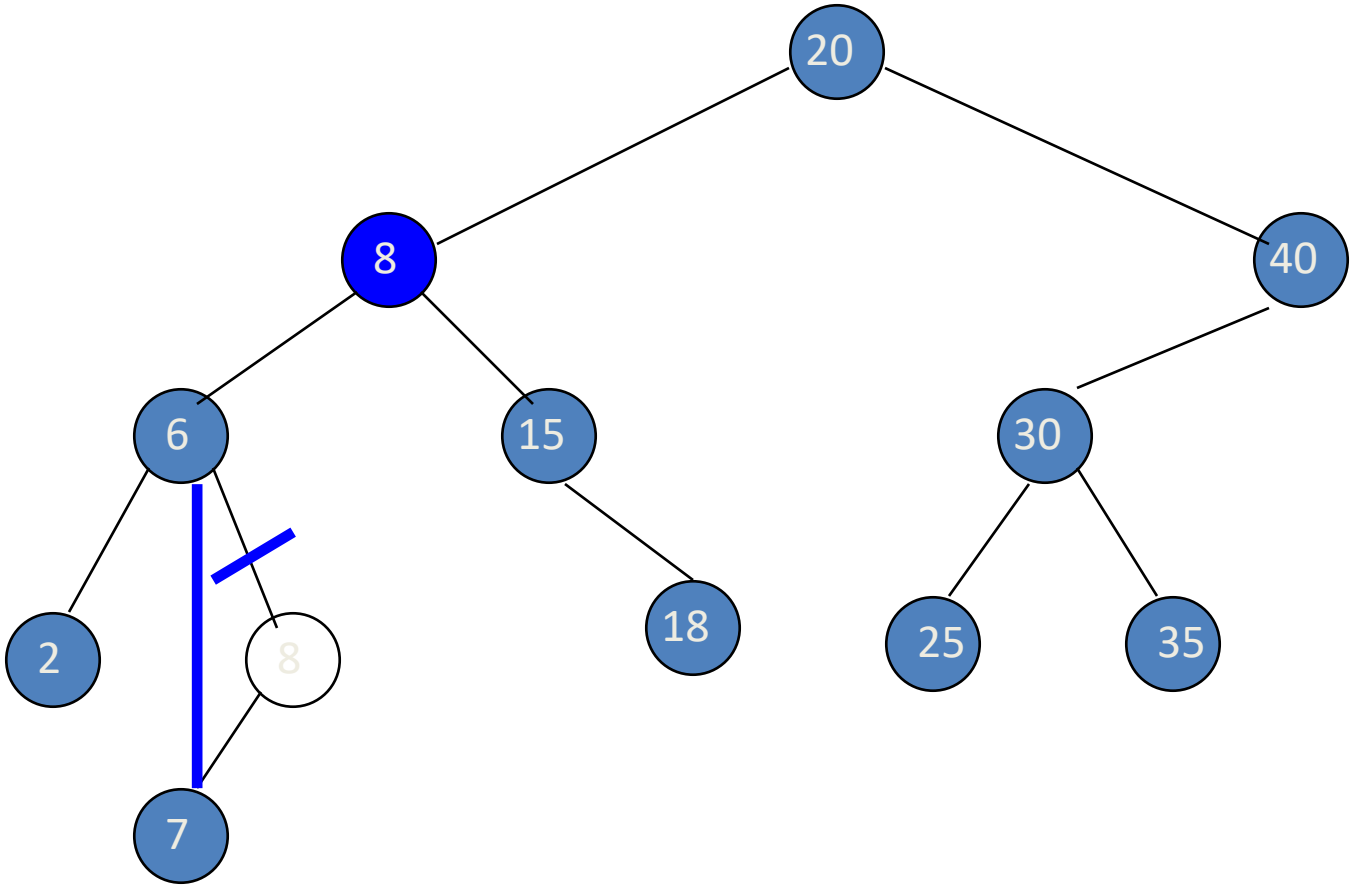
Replace with largest key in left subtree (or smallest in right subtree).

Delete a node that has 2 children



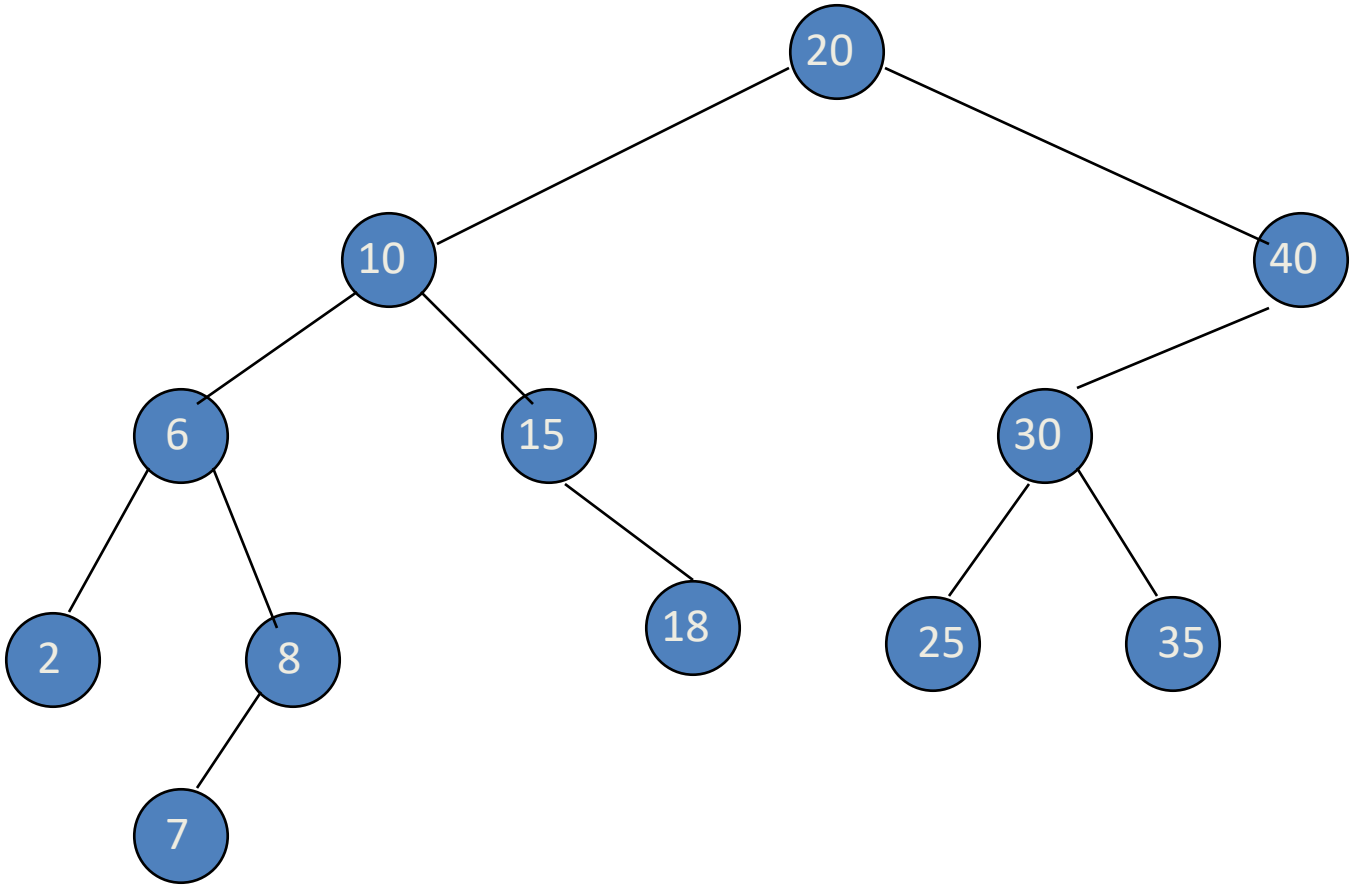
Replace with largest key in left subtree (or smallest in right subtree).

Delete a node that has 2 children



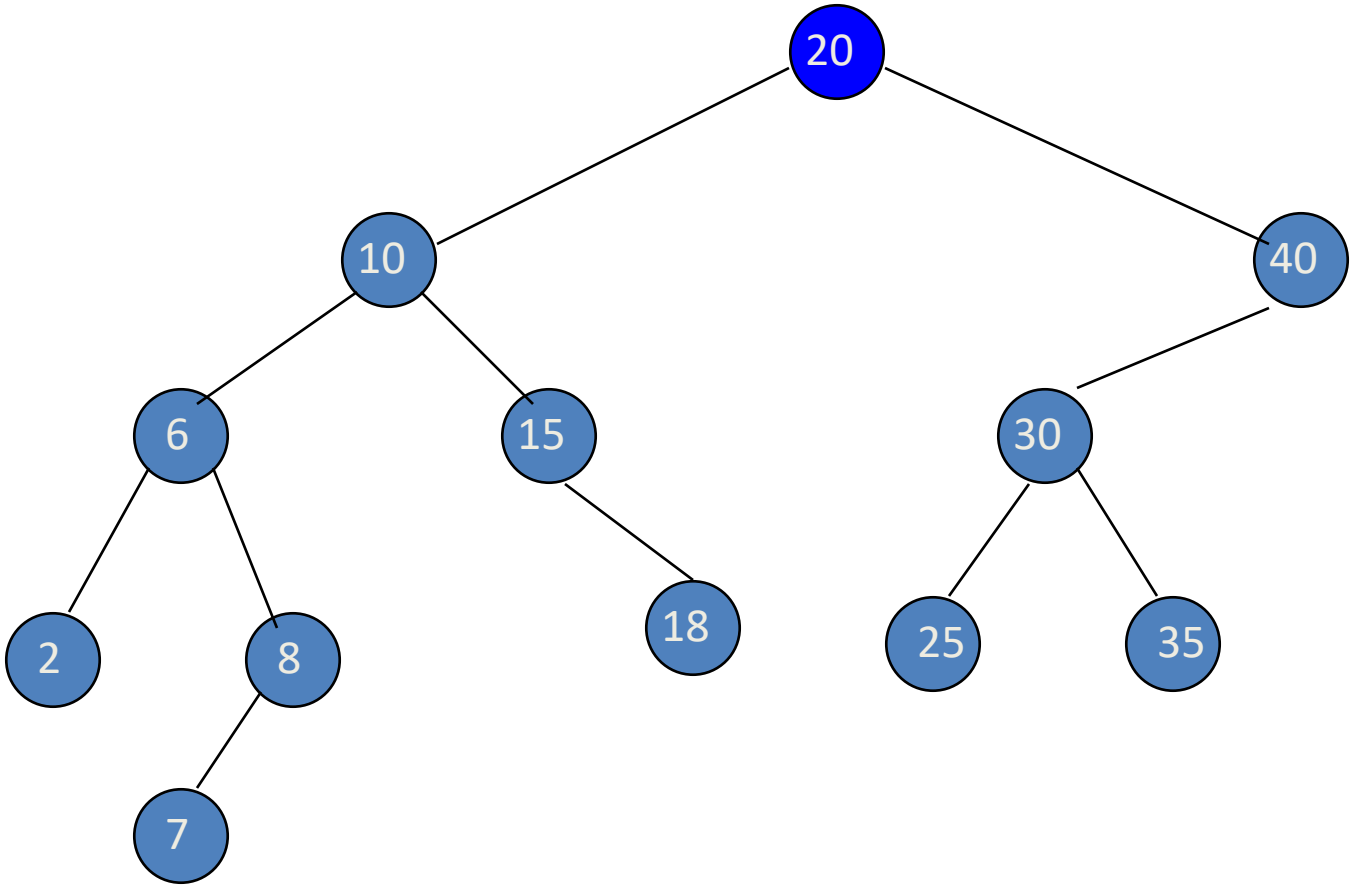
Largest key must be in a leaf or degree 1 node.

Delete a node that has 2 children



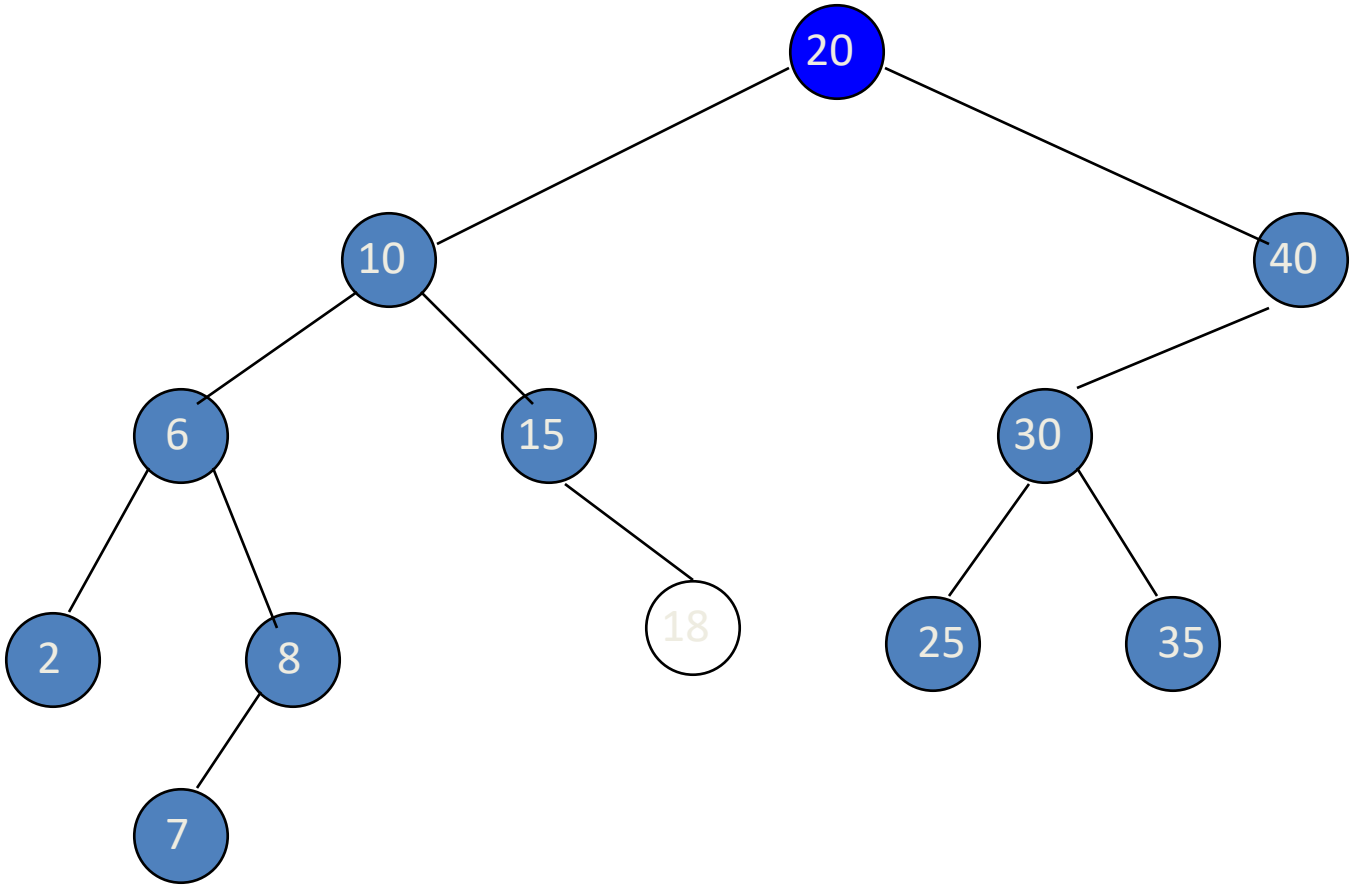
Delete from a degree 2 node. key = 20

Delete a node that has 2 children



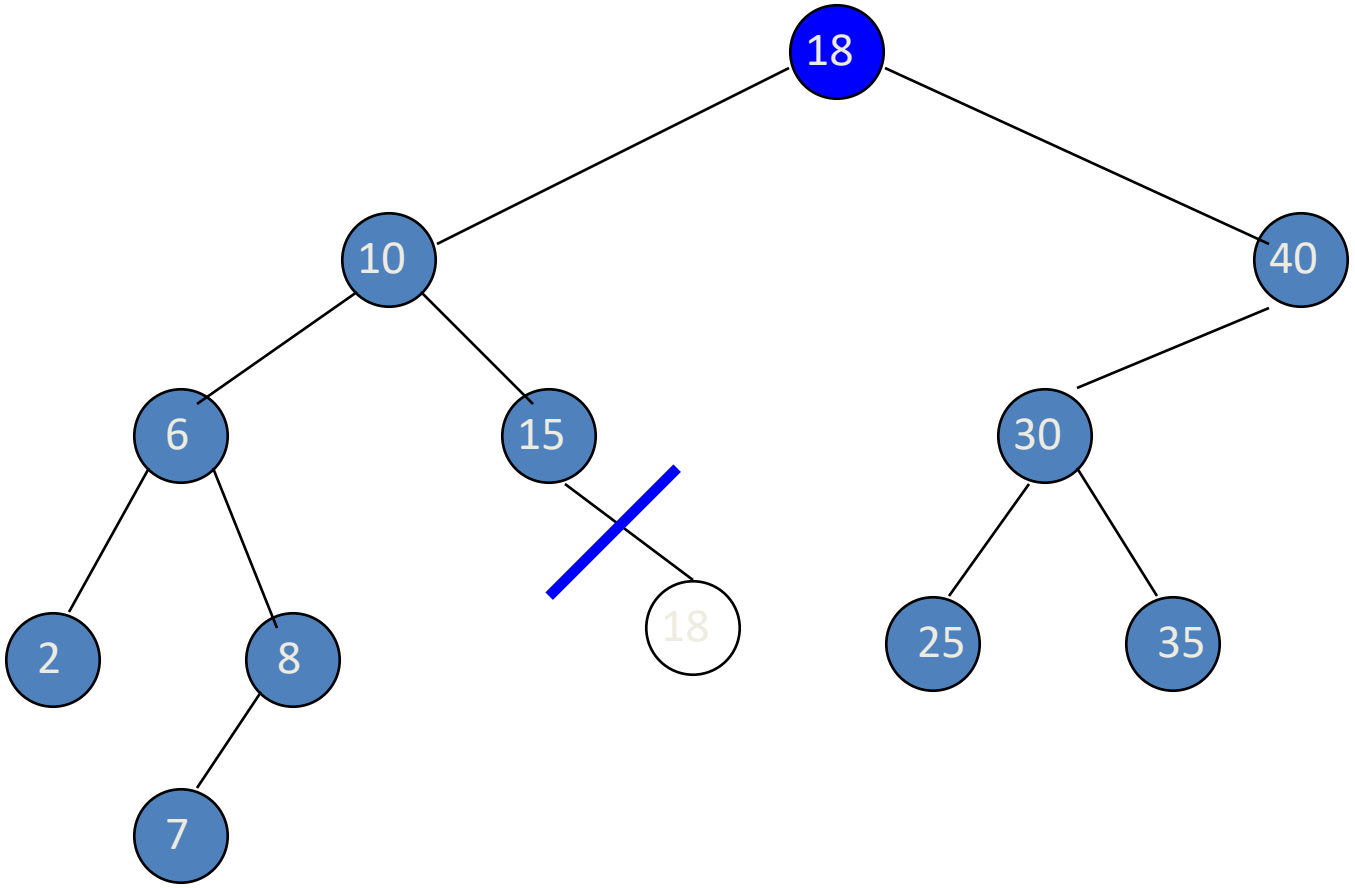
Replace with largest in left subtree.

Delete a node that has 2 children



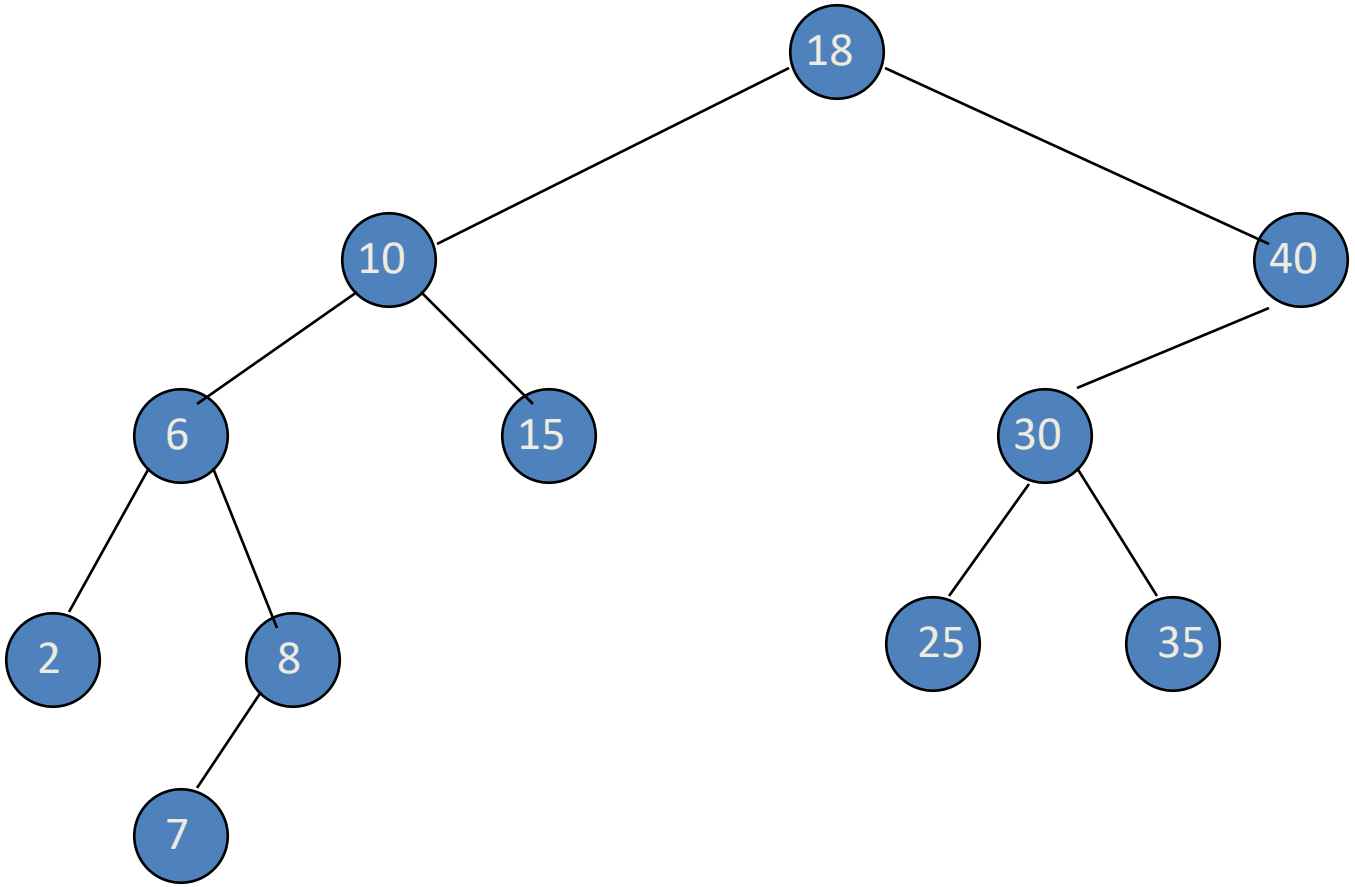
Replace with largest in left subtree.

Delete a node that has 2 children



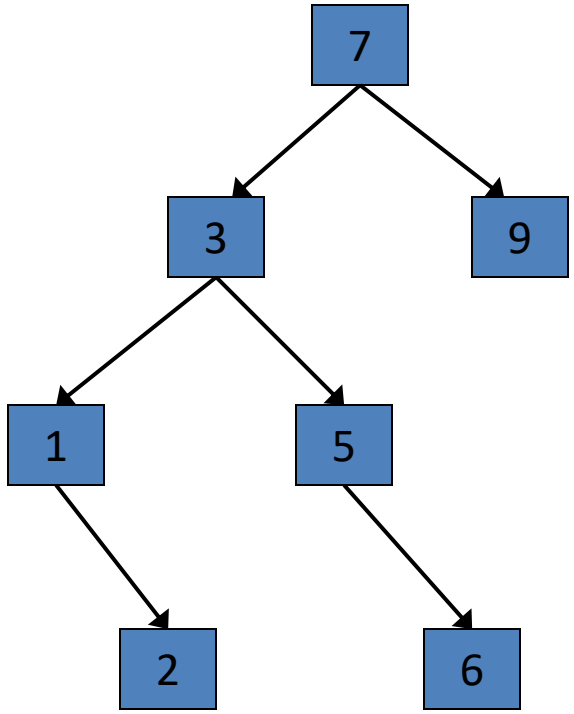
Replace with largest in left subtree.

Delete a node that has 2 children



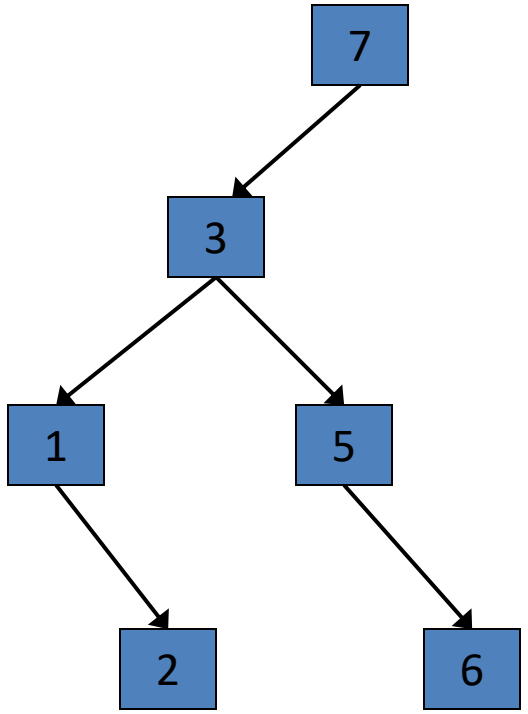
Example Shrinking a Binary Search Tree

delete 9



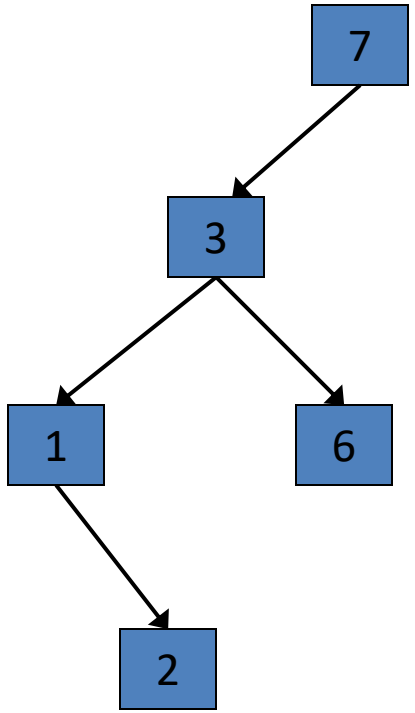
Example Shrinking a Binary Search Tree

delete 5



Example Shrinking a Binary Search Tree

delete 3



Example Shrinking a Binary Search Tree