Delete nodes in BST

3 cases:

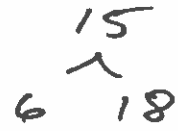
0 child case

Delete 6

1 child case

Delete 15

2 child case

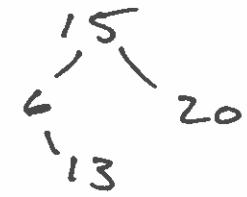


Delete 15

Replacement node in 2 child case

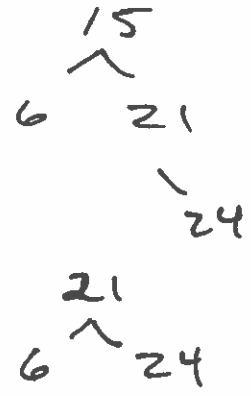
- Max value in left subtree - uncommon

Delete 15
Replace with 13



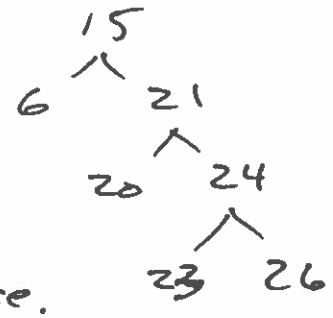
- Convention - Min value in right subtree
- 2 cases
  - Replacement node is deleted node's right child

Delete 15
Replace with 21

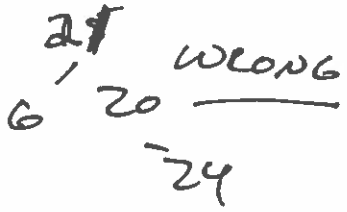


- Replacement is not right child

Delete 15

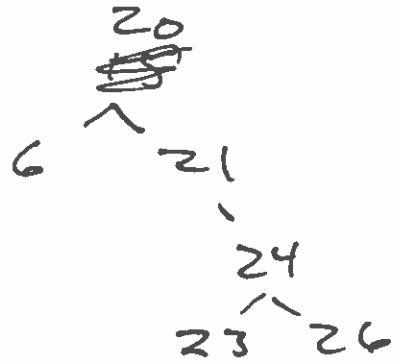


If we replace 15 with 21, 20 is out of place.

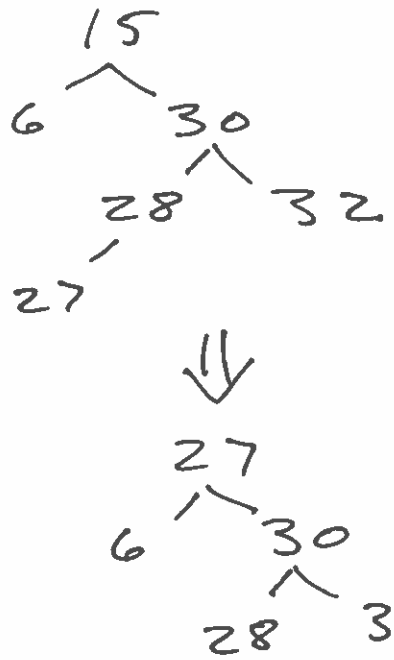


- Replace node with min value in its right subtree.

Delete 15



Ex:



Delete 15.
What replaces it?
27

First, we need to find min in right subtree.

```
Node * min = treeMinimum(n->rightChild)
// input to treeMinimum is right child
// of node to delete.
```

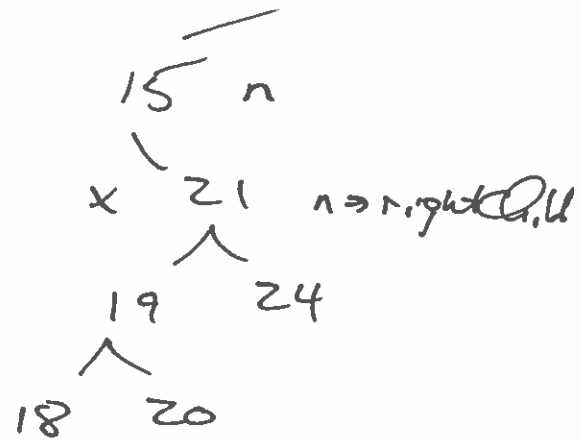
```
treeMinimum(n)   // n is a node pointer.
    Node *x = n;
    while (x->leftChild != NULL)
        x = x->leftChild;

    return x
```

Delete 15

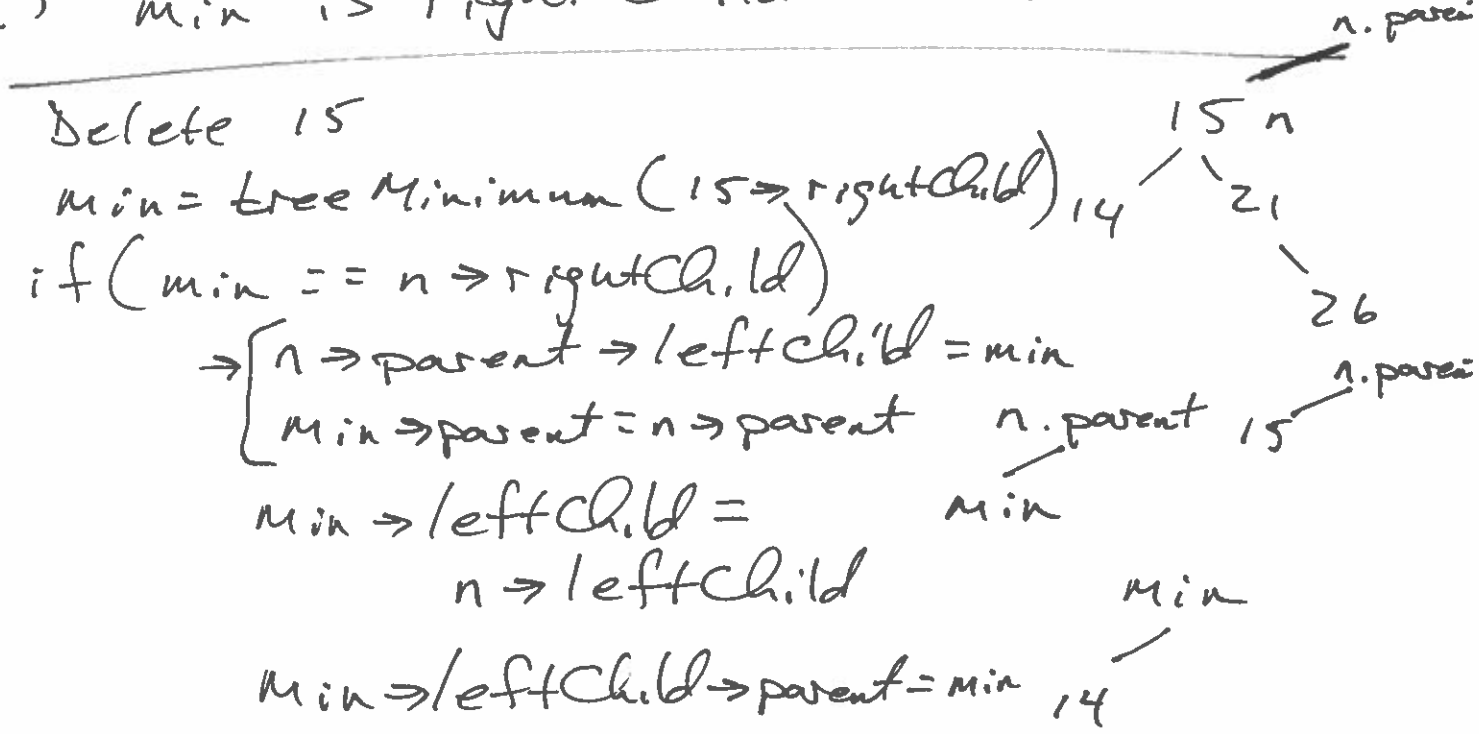
n = search(15)

treeMinimum(n->rightChild)

returns pointer to 18



If min is right child of node delete



Delete 15

min = treeMinimum(15->rightChild)

if (min == n->rightChild)

→ { n->parent->leftChild = min
  { min->parent = n->parent

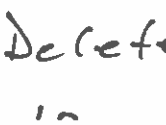
min->leftChild =
    n->leftChild

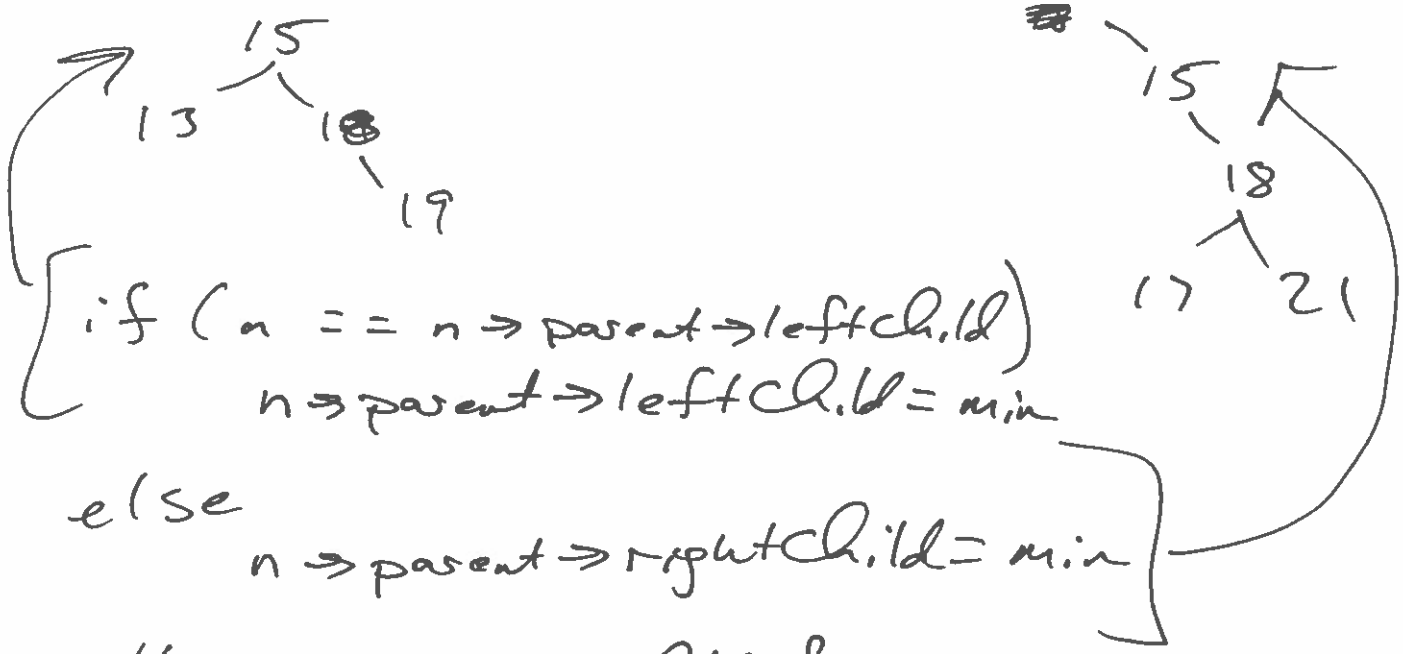
Min->leftChild->parent = min

How does this code change if n is a right child?

Delete 15

Delete 15





// 16 is not right child of 15

else { // condition where min not right child

// nodes involved:

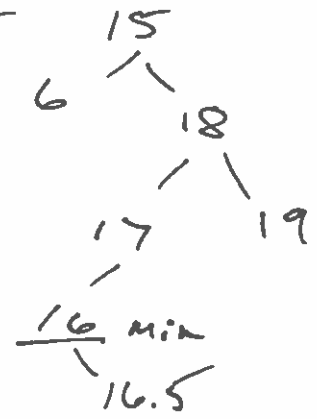
Min right child

Min parent gets new left child

n left child, right child, and parent all have to be updated

Delete 15

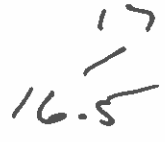
min = 16



Min->parent->leftChild = min->rightChild

17 16.5

17->leftChild = 16.5



Min->rightChild->parent = min->parent

16.5

16.5->parent = 17

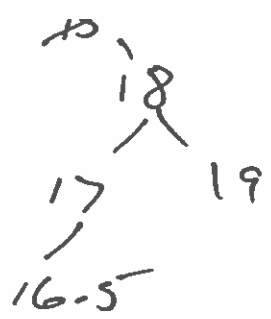
Min->parent = n->parent

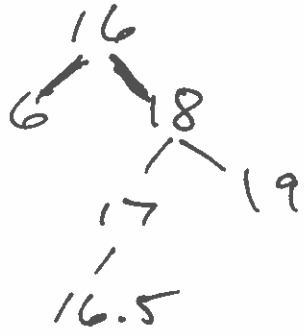
16->parent = 15->parent

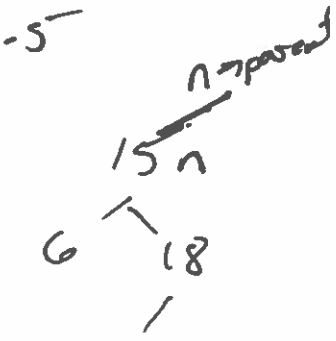
→ n → parent → leftChild = min

Min → leftChild = n → leftChild

16 → leftChild = 15 → leftChild





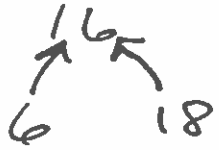


Min → rightChild = n → rightChild

16 → rightChild = 15 → rightChild

n → rightChild → parent = min

n → leftChild → parent = min



delete n

n → key = min → key

delete min