Binary Search Trees

- Dictionary Operations:
  - IsEmpty()
  - Get(key)
  - Insert(key, value)
  - Delete(key)

Complexity Of Dictionary Operations Get(), Insert() and Delete()

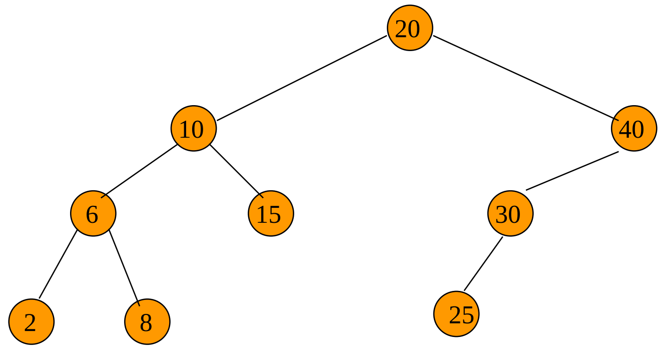
Data Structure	Worst Case	Expected
Hash Table	O(n)	O(1)
Binary Search Tree	O(n)	O(log n)
Balanced Binary Search Tree	O(log n)	O(log n)

n is number of elements in dictionary

Definition Of Binary Search Tree

- A binary tree.
- Each node has a (key, value) pair.
- For every node x, all keys in the left subtree of x are smaller than that in x.
- For every node x, all keys in the right subtree of x are greater than that in x.

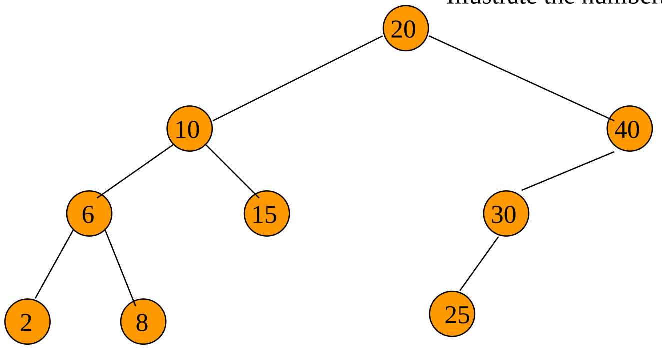
Example Binary Search Tree



Only keys are shown.

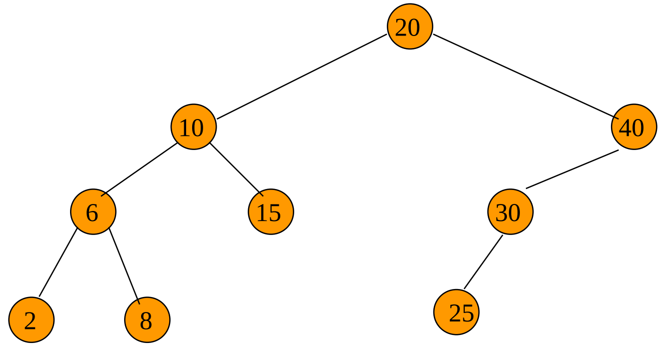
The Operation Ascend()

Illustrate the numbers in increasing order.



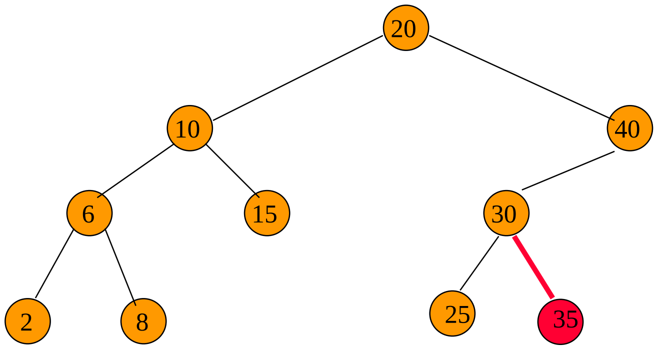
Do an inorder traversal. O(n) time.

The Operation Get()



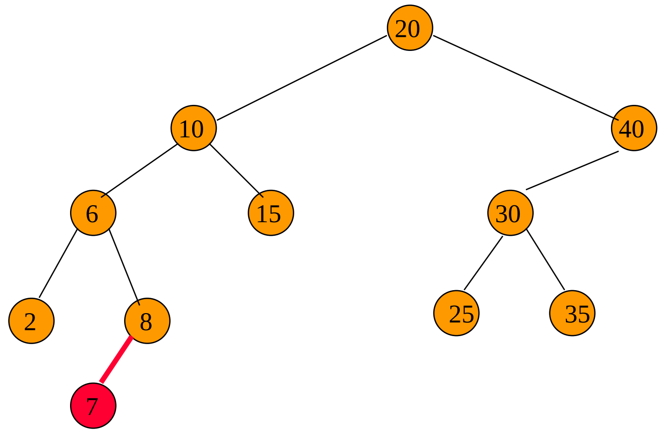
Complexity is O(height) = O(n) (in worst case), where n is number of nodes/elements.

The Operation Insert()



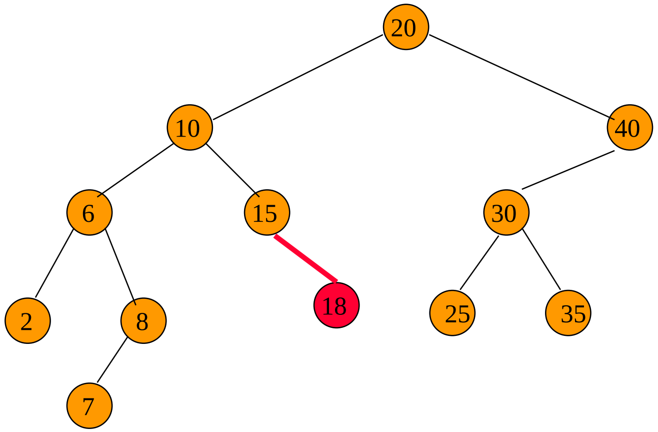
Insert a pair whose key is 35.

The Operation Insert()



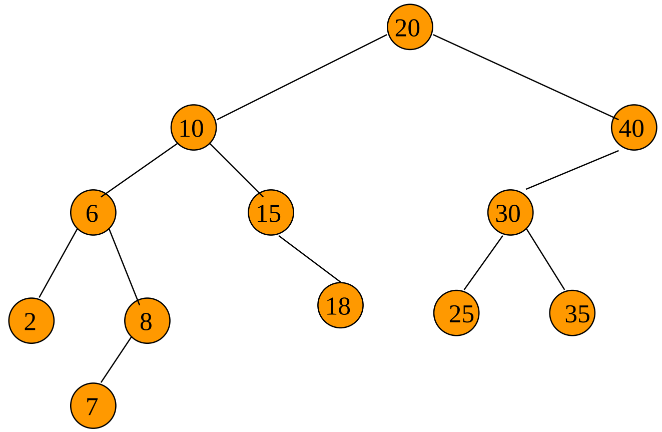
Insert a pair whose key is 7.

The Operation Insert()



Insert a pair whose key is 18.

The Operation Insert()



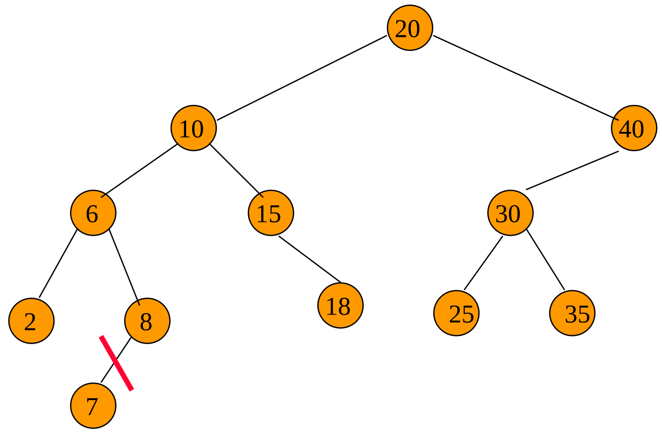
Complexity of Insert() is O(height).

The Operation Delete()

Four cases:

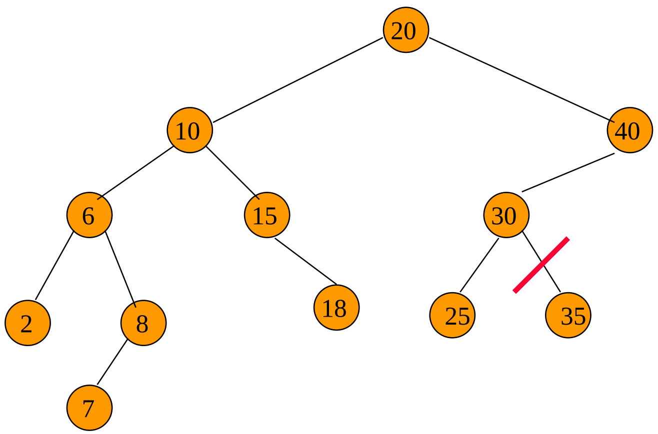
- No element with delete key.
- Element is in a leaf.
- Element is in a degree 1 node. (with one child node)
- Element is in a degree 2 node. (with two child nodes)

Delete From A Leaf



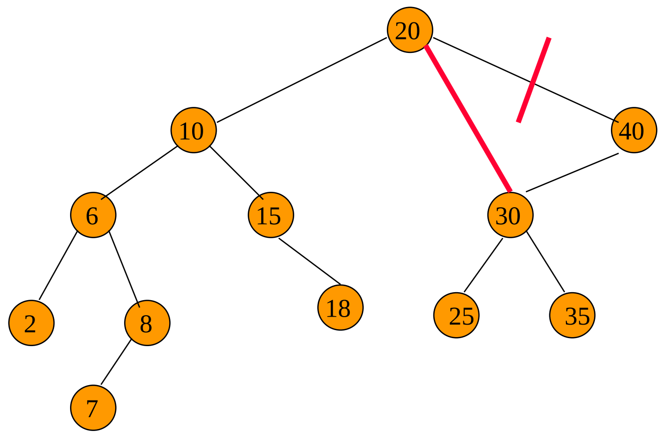
Delete a leaf element. key = 7

Delete From A Leaf (contd.)



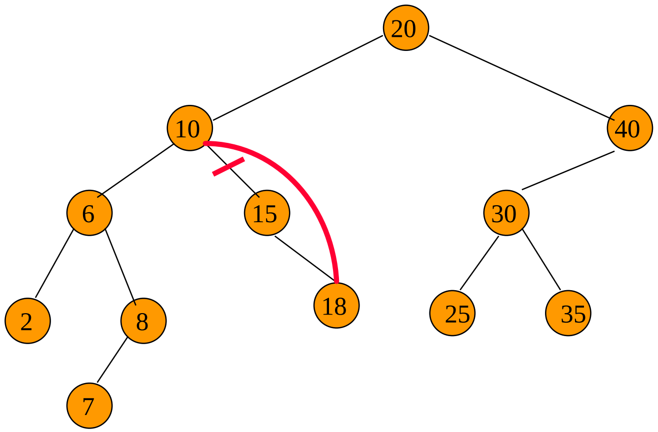
Delete a leaf element. key = 35

Delete From A Degree 1 Node



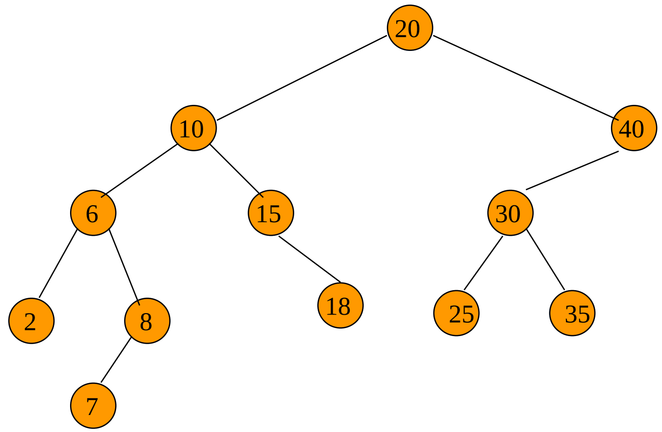
Delete from a degree 1 node. key = 40

Delete From A Degree 1 Node (contd.)



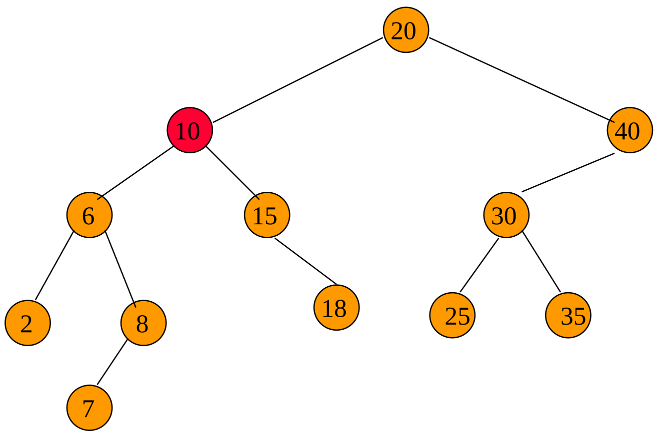
Delete from a degree 1 node. key = 15

Delete From A Degree 2 Node



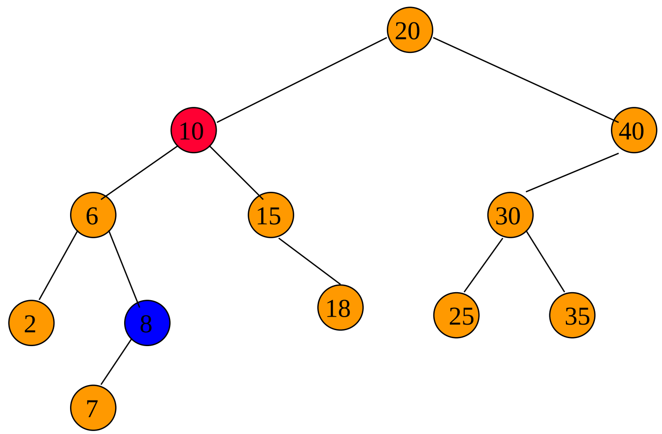
Delete from a degree 2 node. key = 10

Delete From A Degree 2 Node



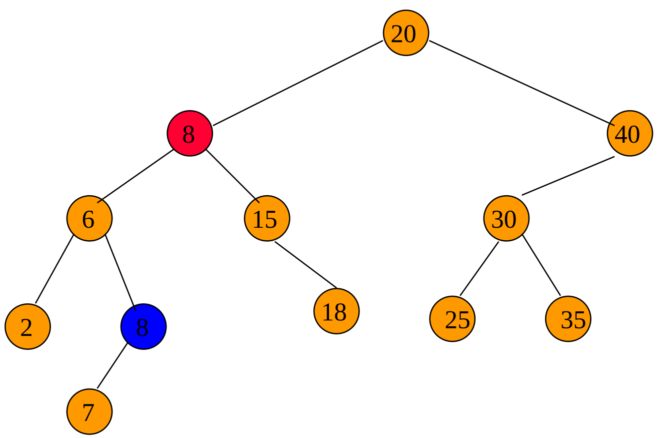
Replace with largest key in left subtree (or smallest in right subtree).

Delete From A Degree 2 Node



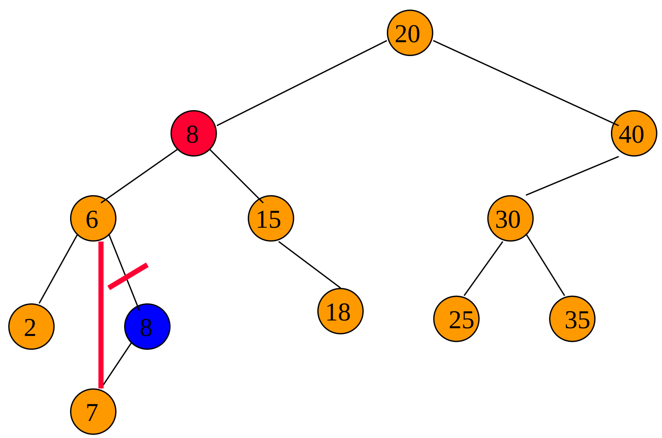
Replace with largest key in left subtree (or smallest in right subtree).

Delete From A Degree 2 Node



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

Delete From A Degree 2 Node

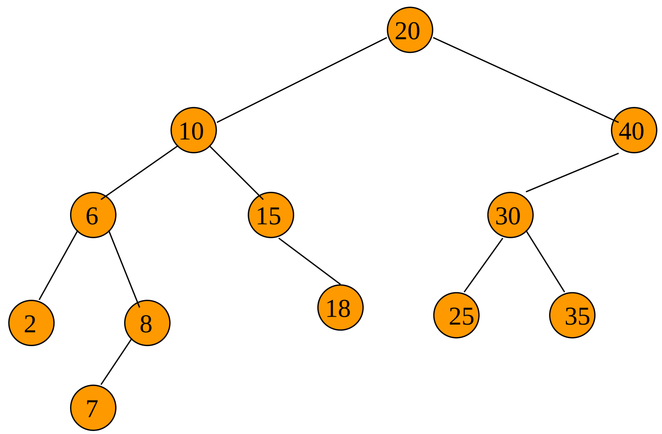


Largest key must be in a leaf or degree 1 node.

In Class Exercise

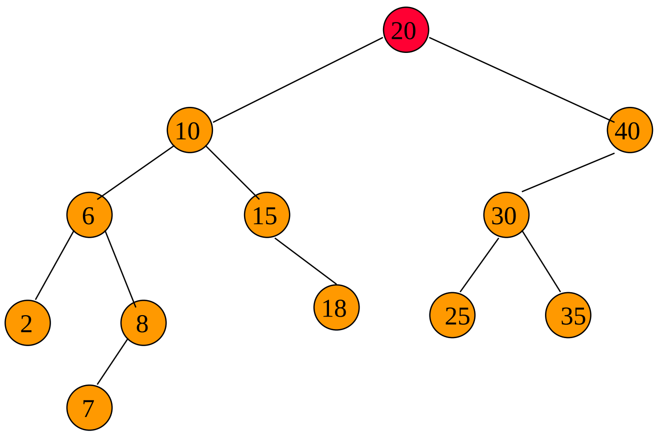
- Prove the statement: “Largest key must be in a leaf or degree 1 node”.

Another Delete From A Degree 2 Node



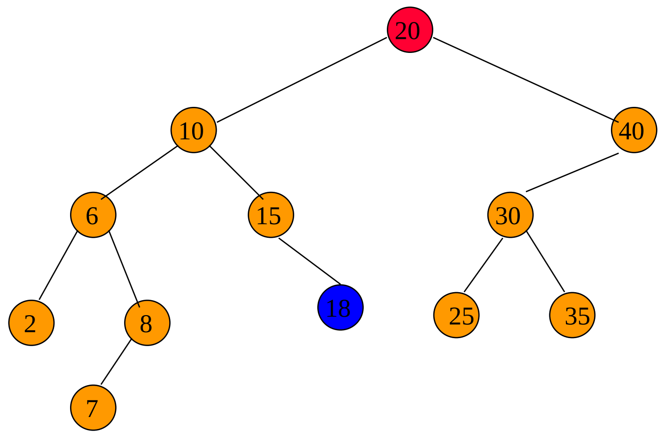
Delete from a degree 2 node. key = 20

Delete From A Degree 2 Node



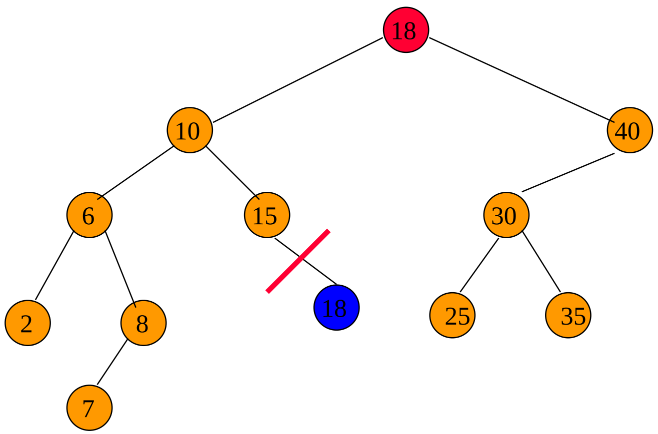
Replace with largest in left subtree.

Delete From A Degree 2 Node



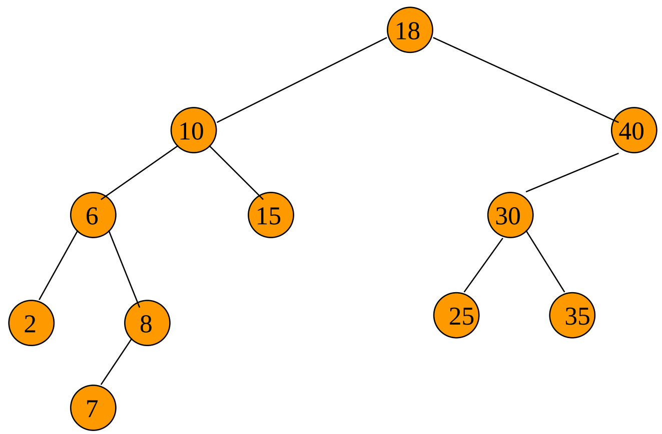
Replace with largest in left subtree.

Delete From A Degree 2 Node



Replace with largest in left subtree.

Delete From A Degree 2 Node



Complexity is O(height).