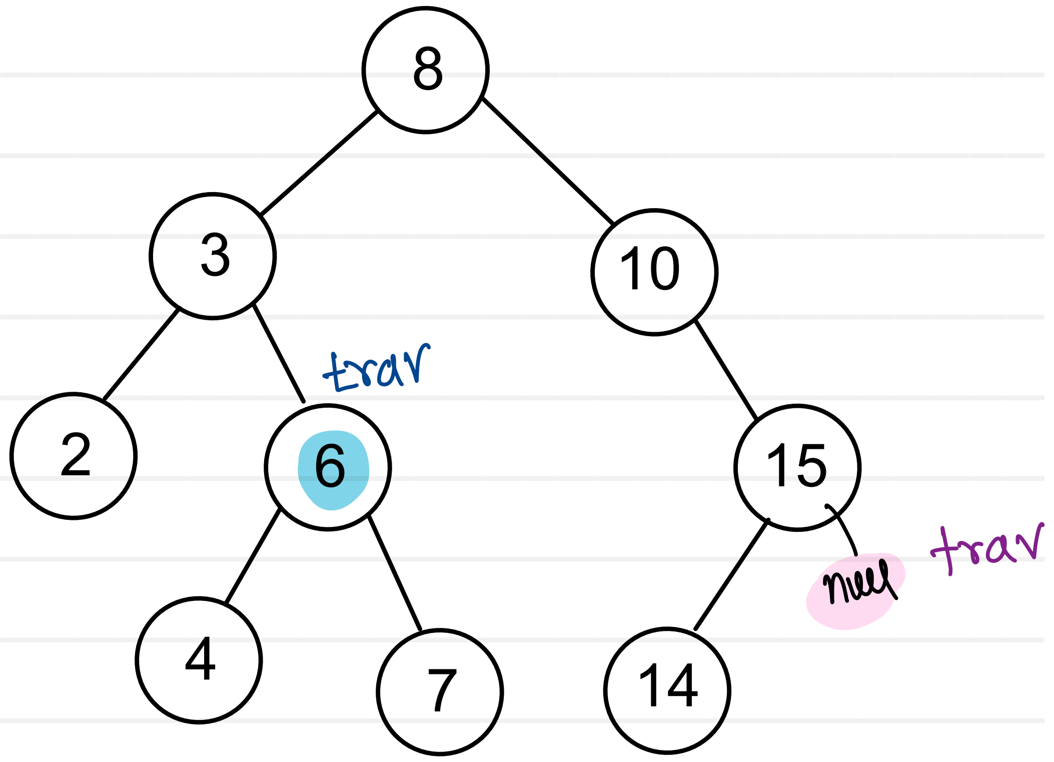
Sunbeam Institute of Information Technology
Pune and Karad

Module – Data Structures and Algorithms

Trainer - Devendra Dhande

Email – devendra.dhande@sunbeaminfo.com

Binary Search Tree - Binary Search

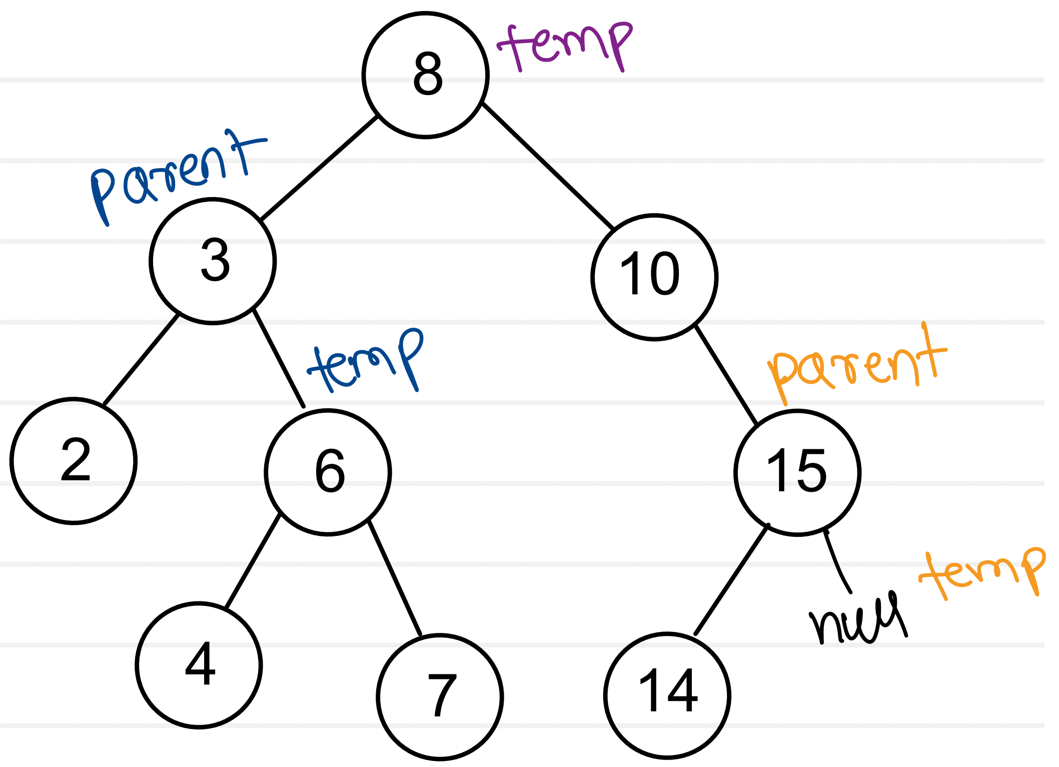


1. Start from root
2. If key is equal to current node data return current node
3. If key is less than current node data search key into left sub tree of current node
4. If key is greater than current node data search key into right sub tree of current node
5. Repeat step 2 to 4 till leaf node

Key = 6

Key = 16

Binary Search Tree - Binary Search with Parent



key = 6

temp	parent
&8	null
&3	&8
&6	&3

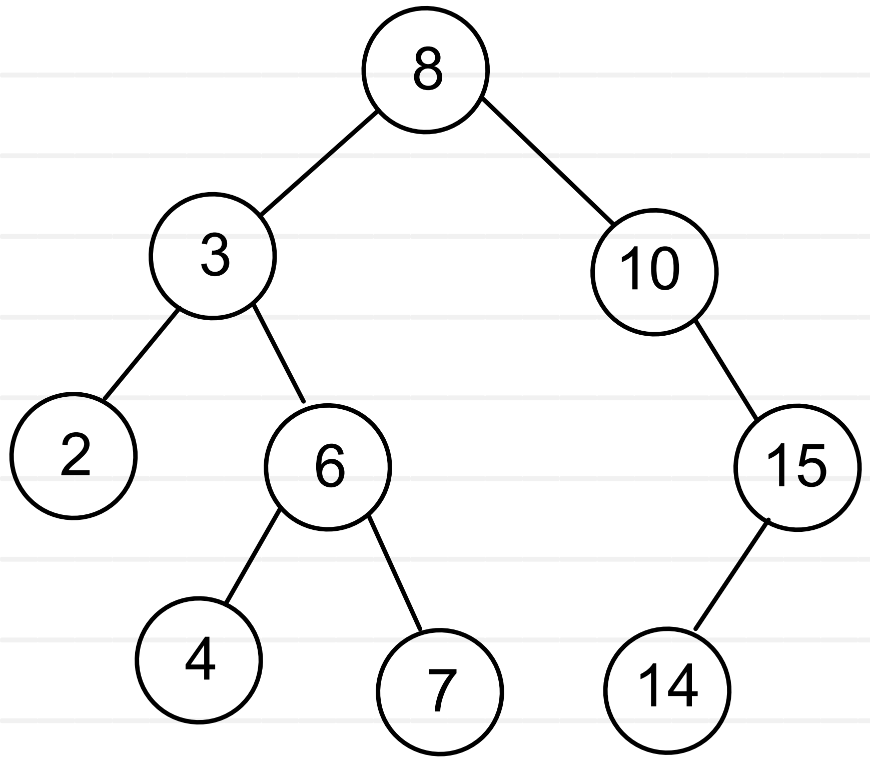
key = 16

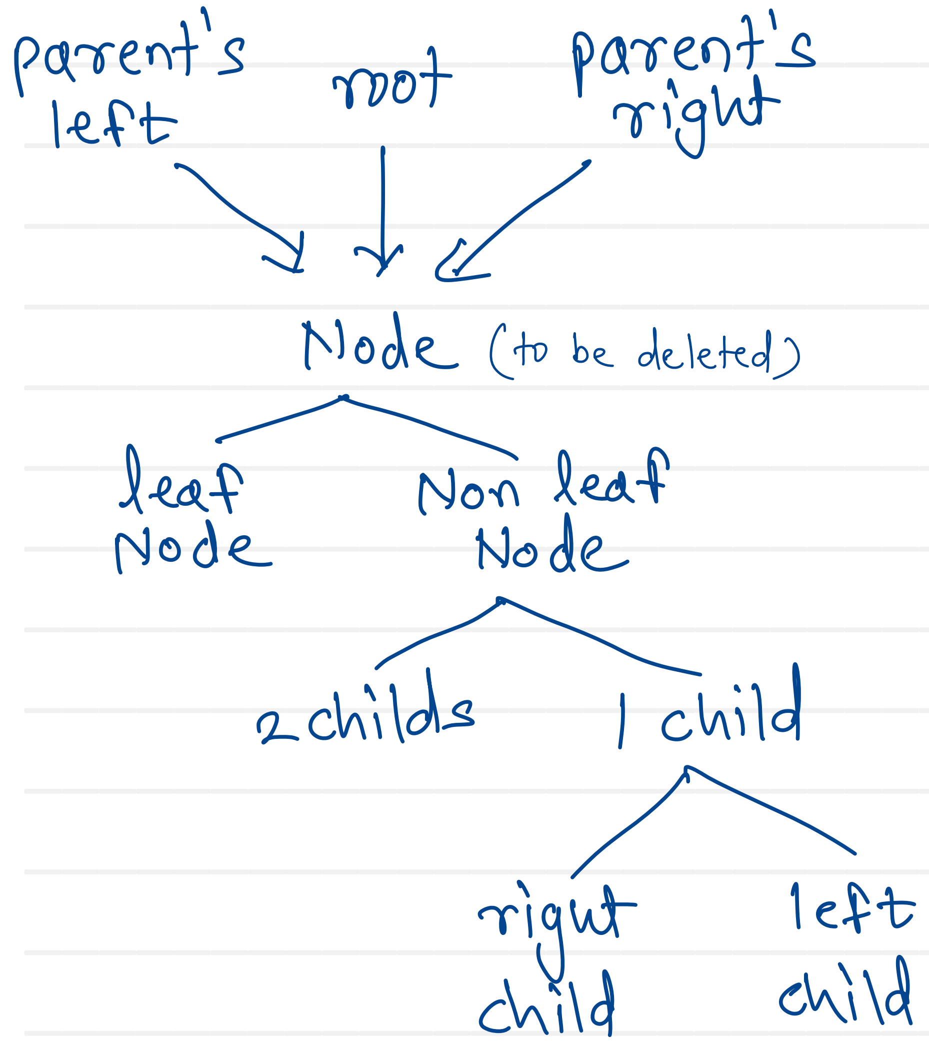
temp	parent
&8	null
&10	&8
&15	&10
null	&15

key = 8

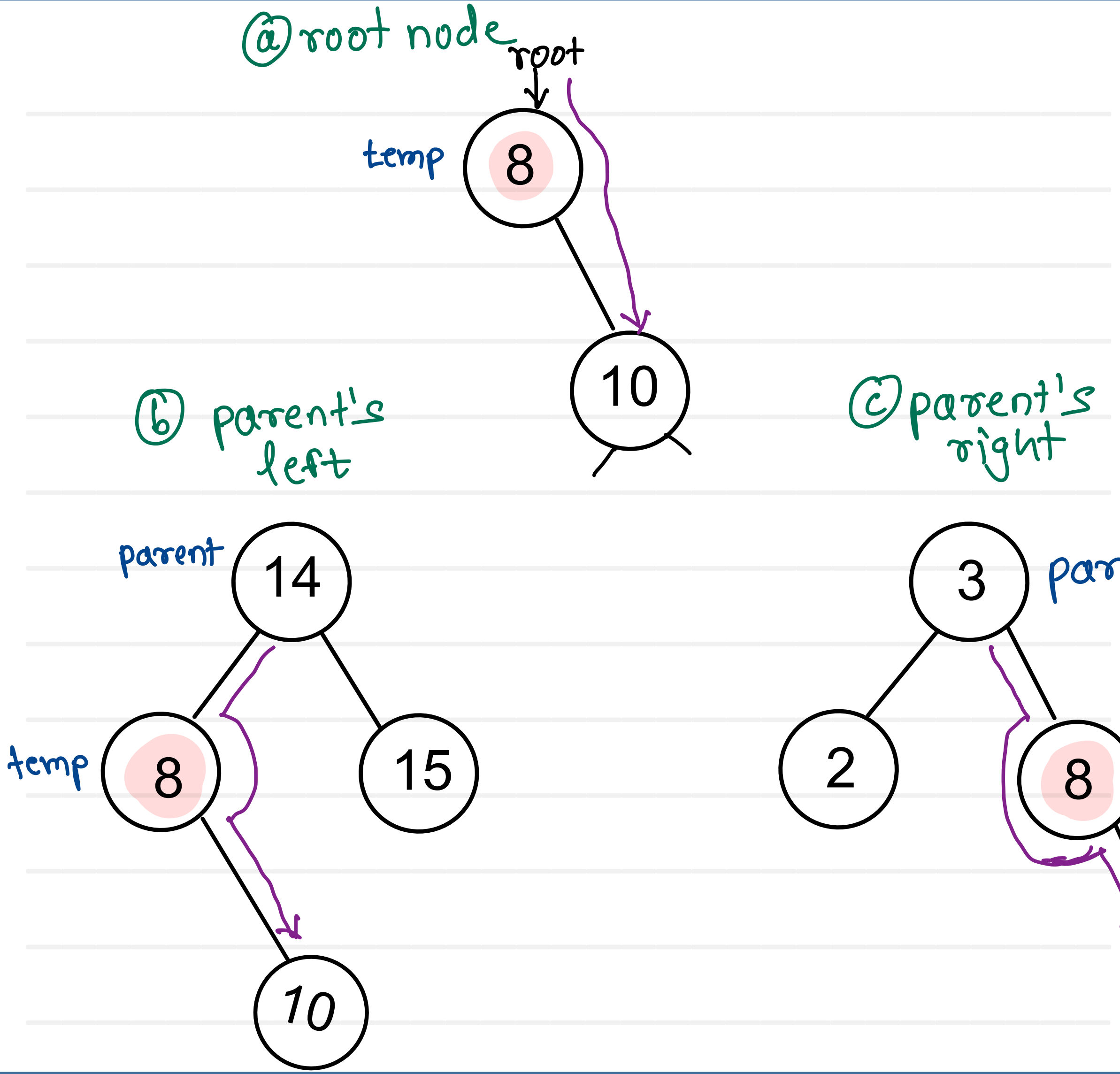
temp	parent
&8	null

Binary Search Tree - Delete Node

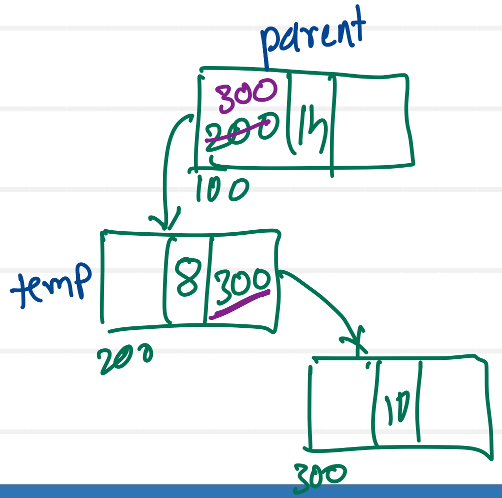




BST - Delete Single child node (Right child)

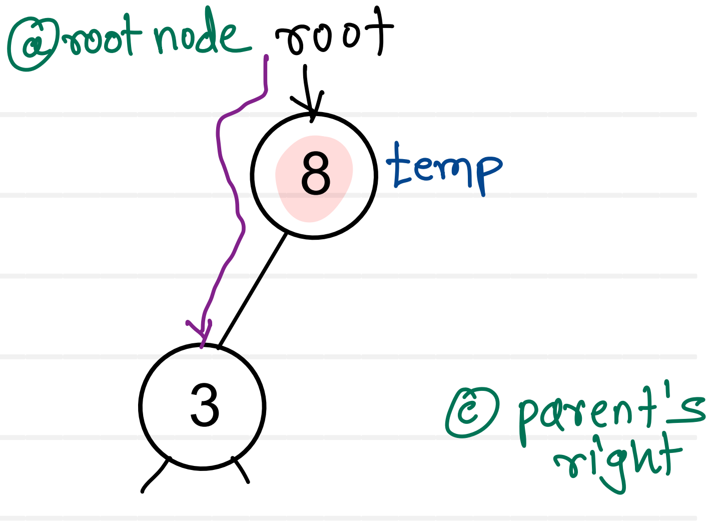


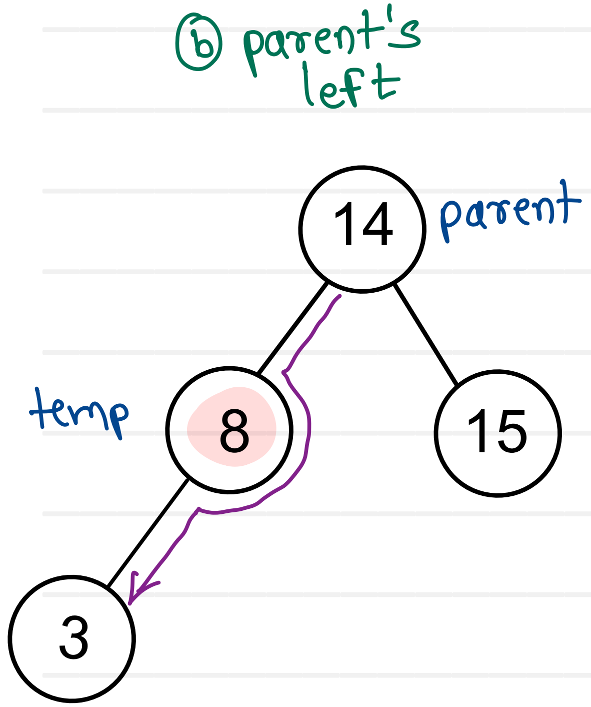
```
if( temp.left == null) {
    if(temp == root)
ⓐ       root = temp.right;
    else if(temp == parent.left)
ⓑ       parent.left = temp.right;
    else if(temp == parent.right)
ⓒ       parent.right = temp.right;
}
```

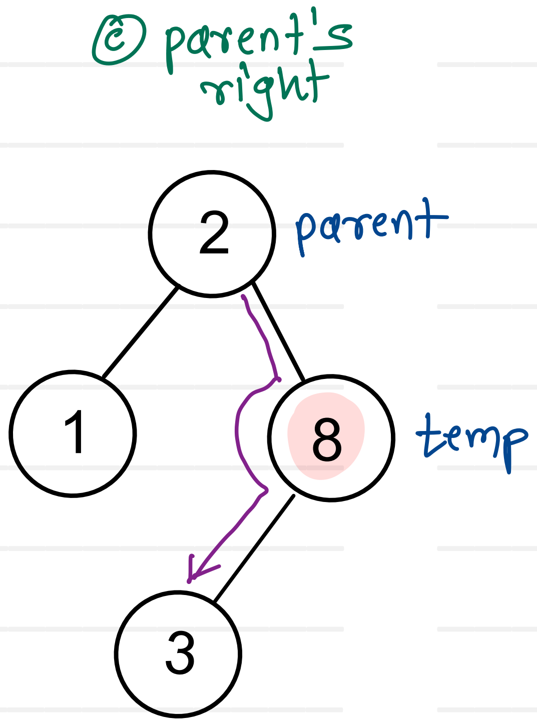


parent.left = 200
temp = 200

BST- Delete Single child node (Left child)

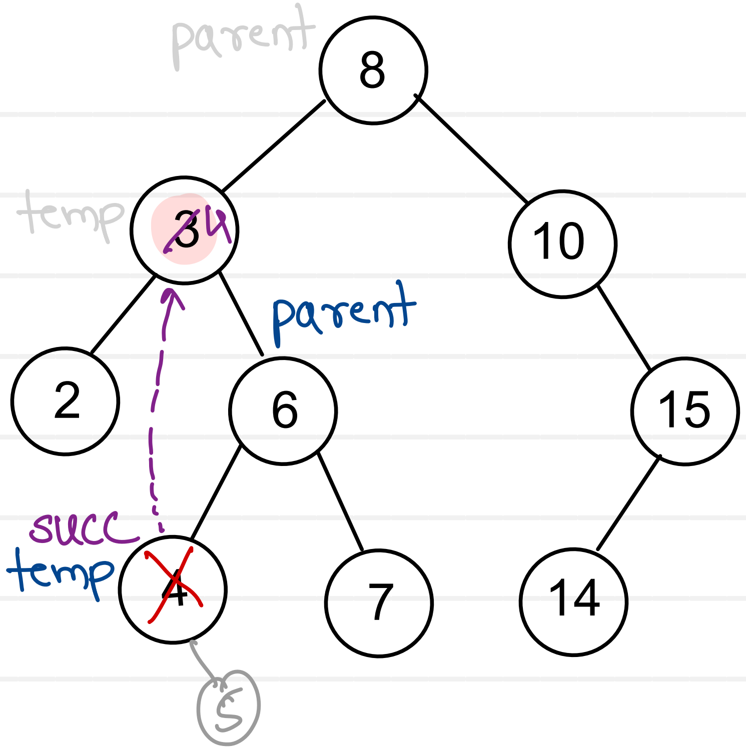






```
if(temp.right == null) {
        if (temp == root)
ⓐ               root = temp.left;
        else if( temp == parent.left)
ⓑ               parent.left = temp.left;
        else if( temp == parent.right)
ⓒ               parent.right = temp.left;
}
```

BST - Delete Two childs node



Inorder : 2 3 4 6 7 8 10 14 15

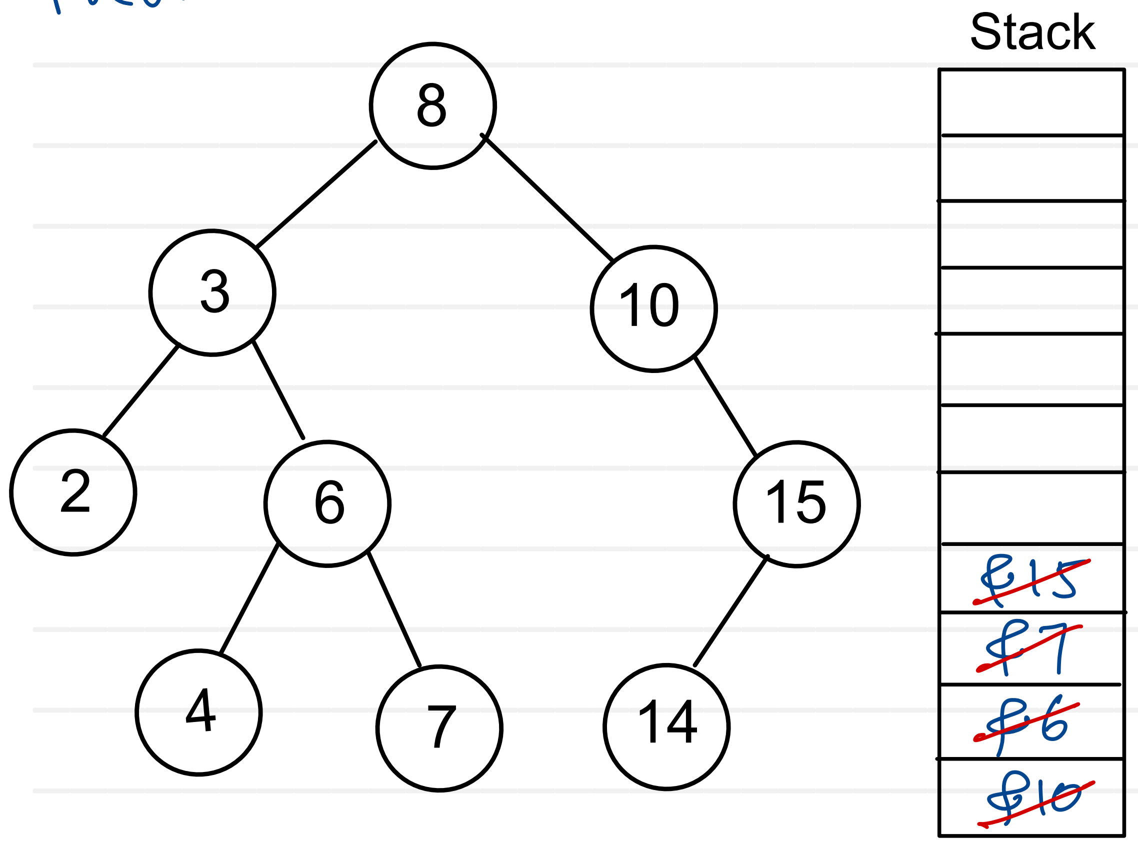
2 → inorder predecessor → left extreme right

4 → inorder successor → right extreme left

```
if(temp.left != null && temp.right != null) {
    //1. find inorder successor of temp
    Node succ = temp.right;
    parent = temp;
    while( succ.left != null) {
        parent = succ;
        succ = succ.left;
    }
    //2. update value of successor to temp
    temp.data = succ.data;
    //3. delete space of successor
    temp = succ;
}
```

BST - Preorder (Non recursive)

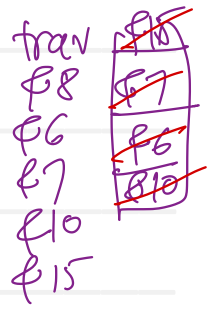
Preorder = VLR



Preorder :

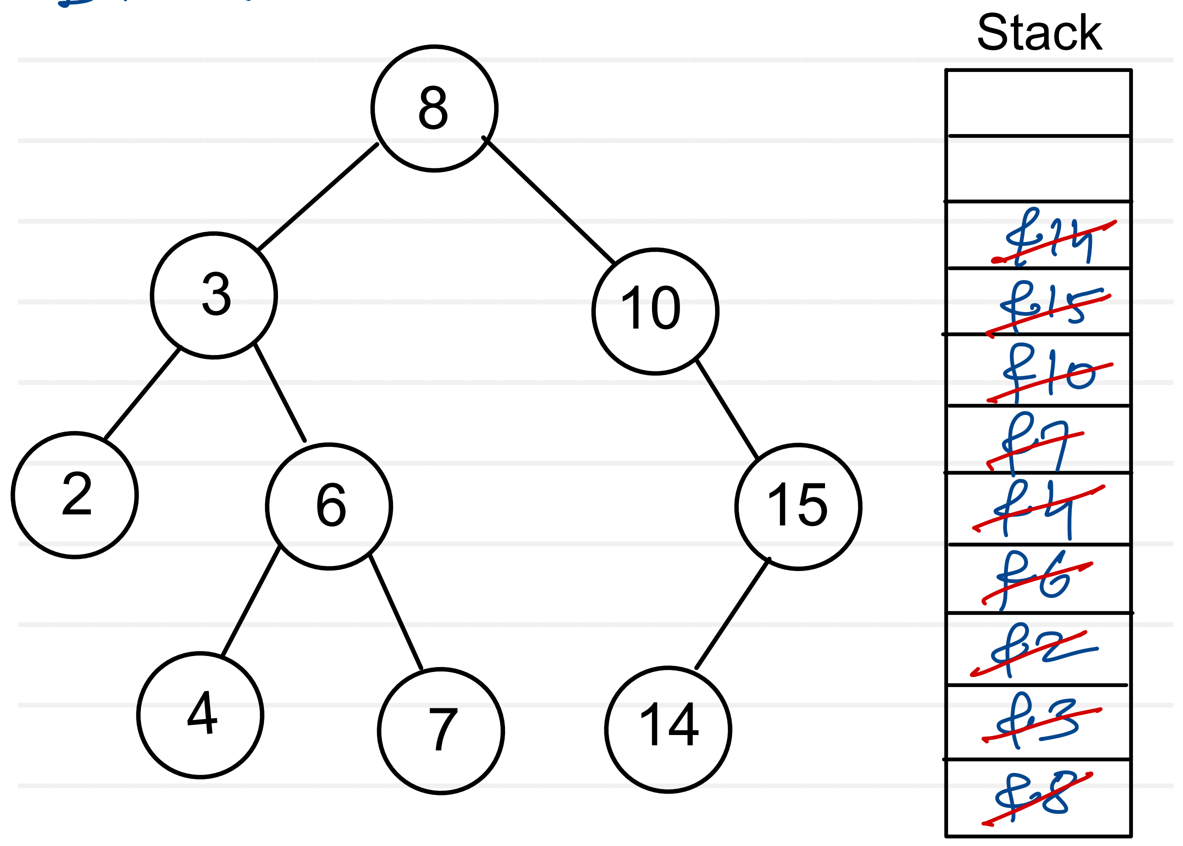
8 , 3 , 2 , 6 , 4 , 7 , 10 , 15 , 14

```
void preorder( ) {
    Stack<Node> st = new stack<>();
    //1. start traversing from root
    Node trav = root;
    while(trav != null || !st.empty()) {
        while(trav != null) {
            //visit current node,
            sysout(trav.data);
            // push right if exists on stack
            if(trav.right != null)
                st.push(trav.right);
            //go on left
            trav = trav.left;
        } // repeat till extreme left
        //2. pop node from stack.
        if( !st.empty())
            trav= st.pop();
    }
}
```

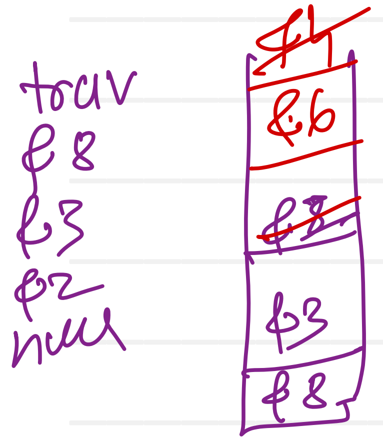


BST - Inorder (Non recursive)

Inorder : LVR

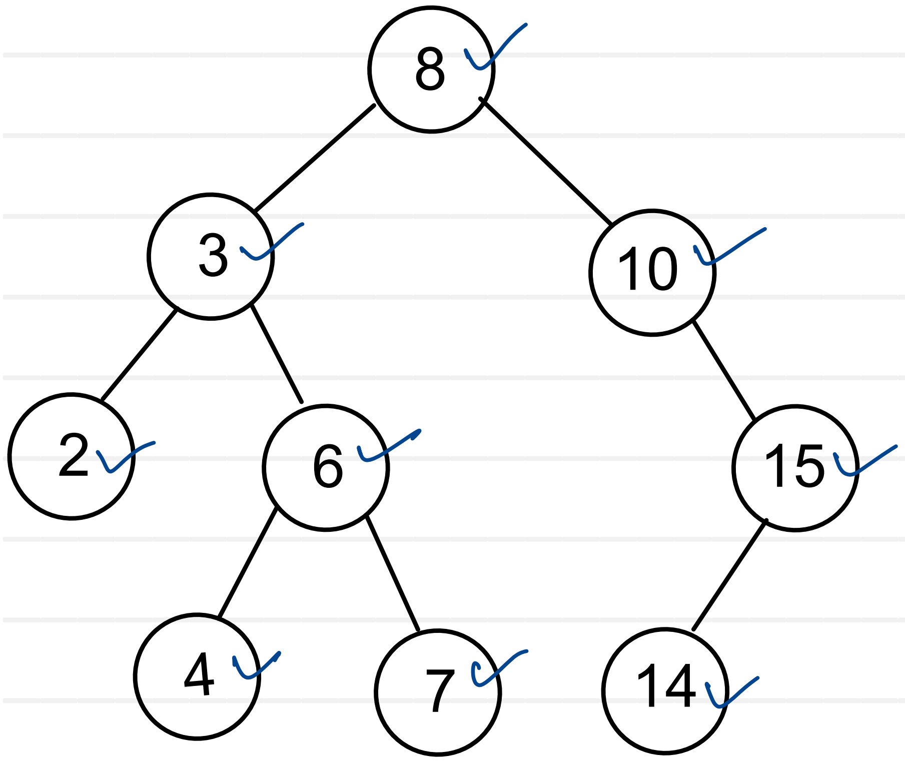


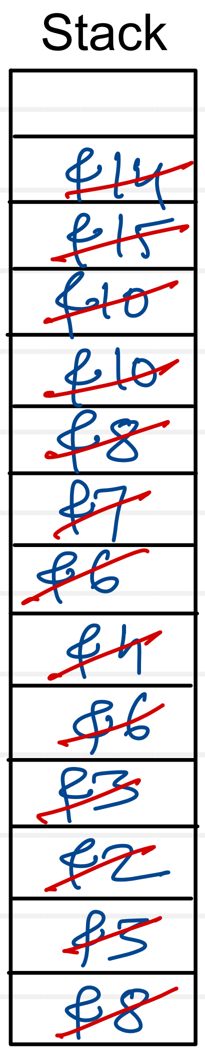
Inorder:
2, 3, 4, 6, 7, 8, 10, 14, 15



```
void inorder( ) {
    Stack<Node> st = new Stack<>();
    //1. start from root
    Node trav = root;
    while(trav != null || !st.isEmpty()){
        // push node on stack & go into
           left, repeat till extreme left
        while(trav != null) {
            st.push(trav);
            trav = trav.left;
        }
        // pop element from stack, visit it
           & go into right.
        if(!st.isEmpty()) {
            trav = st.pop();
            sysout(trav.data);
            trav = trav.right;
        }
    }
}
```

BST - Postorder (Non recursive)





```
void postorder( ) {
    stack<Node> st = new stack<>();
    // start from root
    Node trav = root;
    while(trav != null || !st.isEmpty()) {
        while( trav != null)
            st.push(trav);
            trav = trav.left;
        }
        if( !st.isEmpty()) {
            trav = st.pop();
            if(trav.right == null || trav.right.visited == true){
                sysout(trav.data);
                trav.visited = true;
                trav = null;
            } else {
                s.push(trav);
                trav = trav.right;
        }   }
    }
}
```

Postorder:

2, 4, 7, 6, 3, 14, 15, 10, 8

SUNBEAM

Thank you!!!

Devendra Dhande

devendra.dhande@sunbeaminfo.com