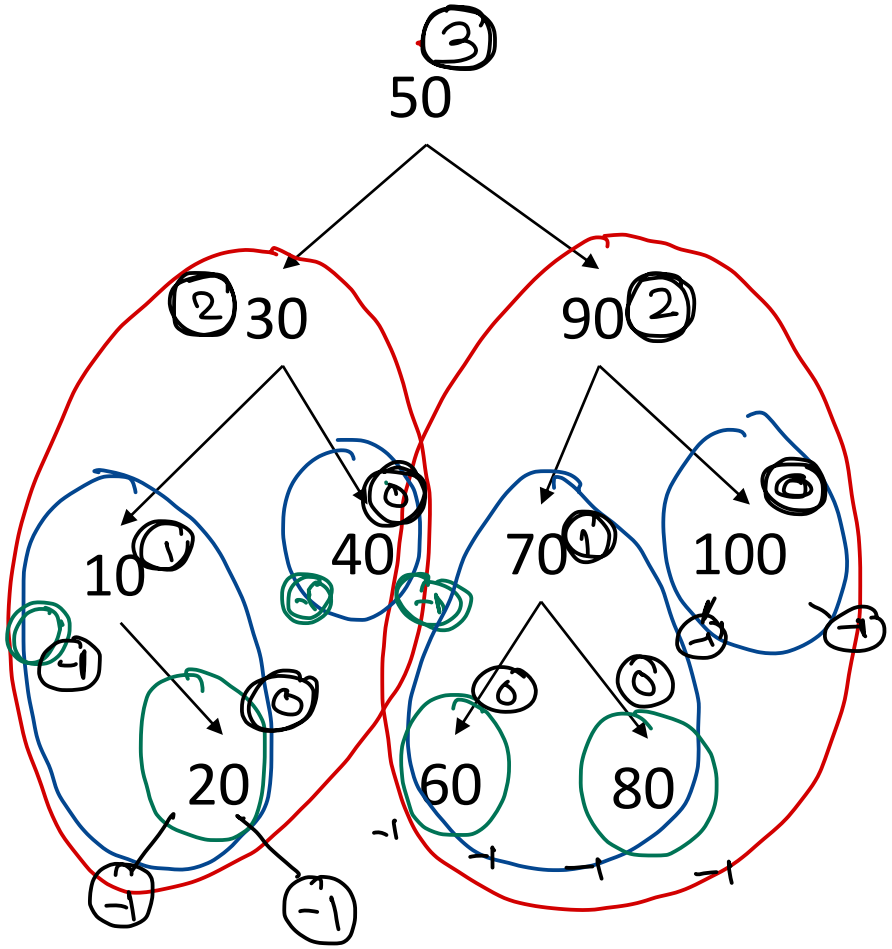
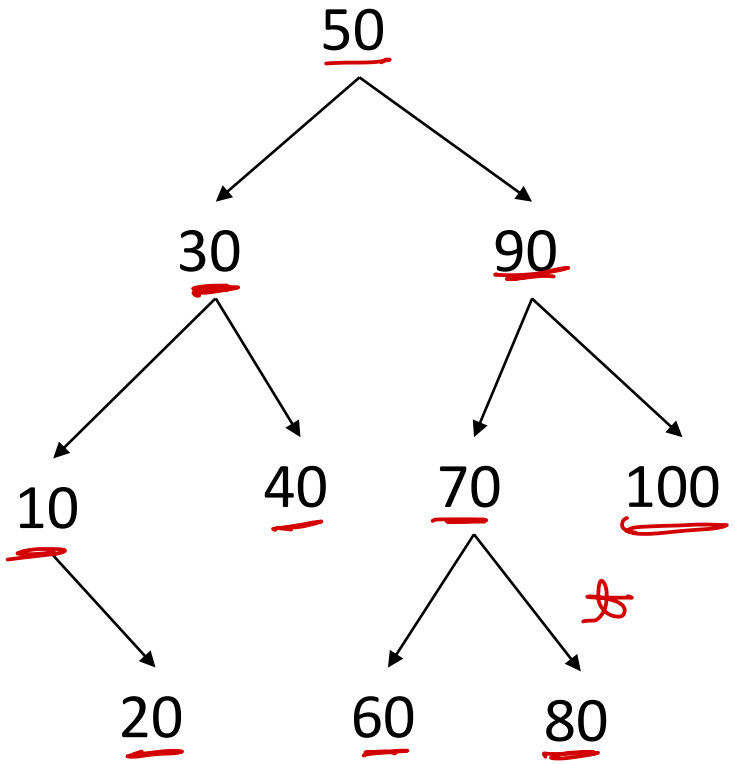
Data Structure & Algorithms

Sunbeam Infotech

BST - height()

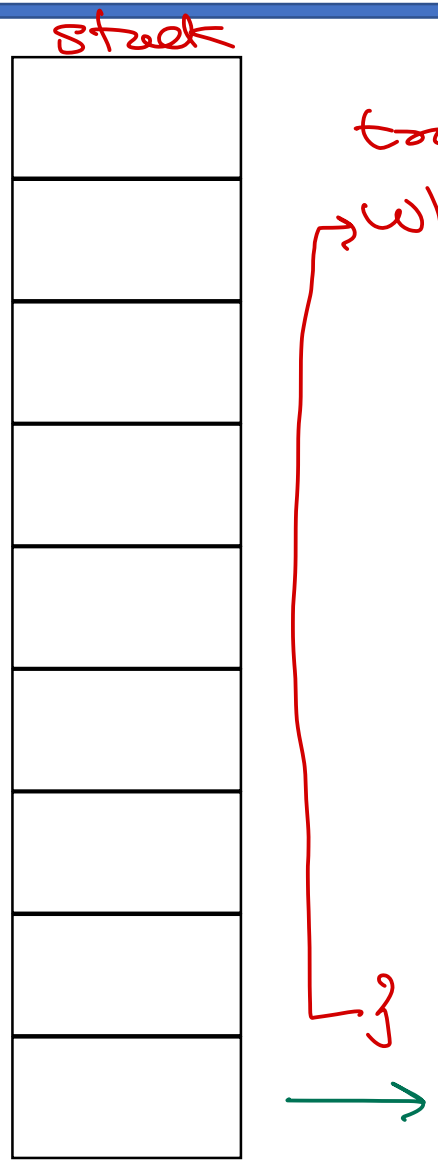


BST - preorder



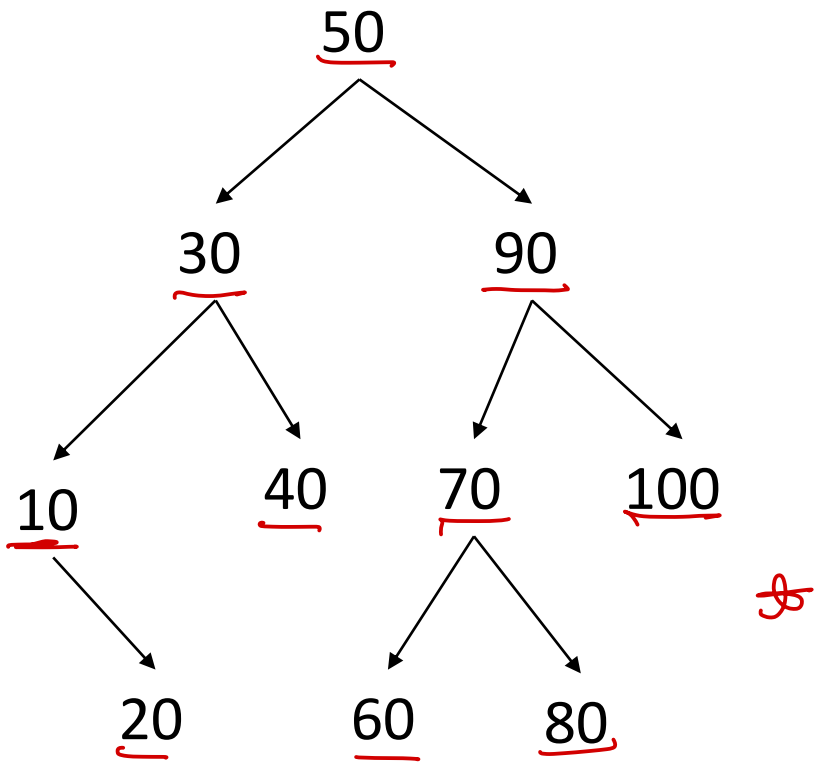
50 30 10 20 40

90 70 60 80 100

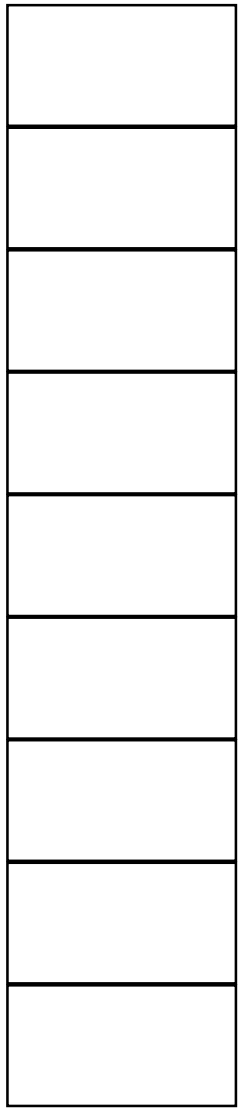



```
trav = root;
while(trav != null || !s.isEmpty()) {
    while(trav != null) {
        print(trav.data);
        if(trav.right != null)
            s.push(trav.right);
        trav = trav.left;
    }
    if(!s.isEmpty())
        trav = s.pop();
}
```

BST - inorder

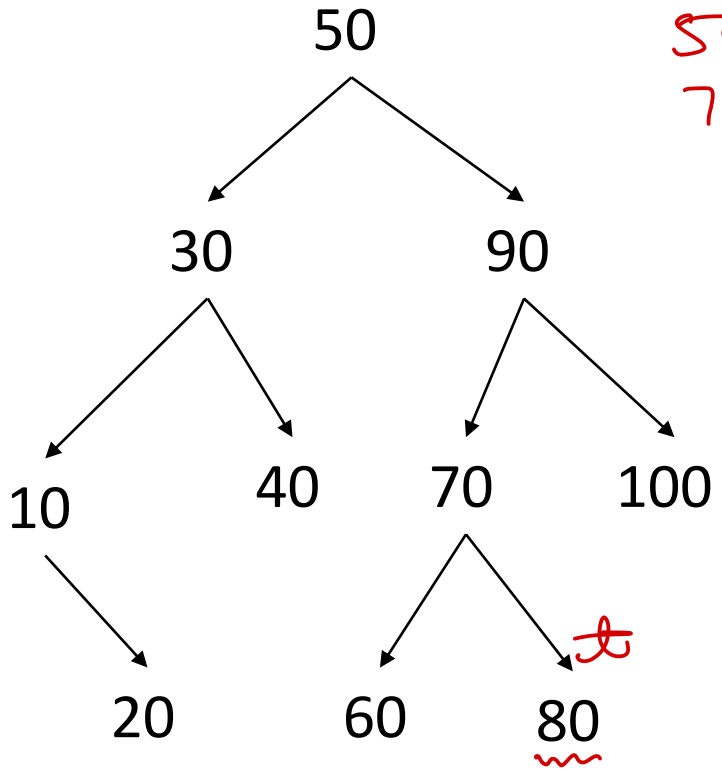


10 20 30 40 50
60 70 80 90 100



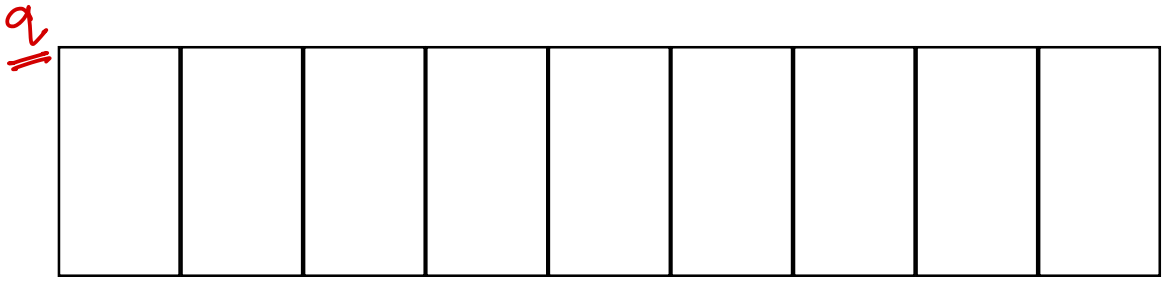
```
trav = root;
while(trav!=null || !s.isEmpty()) {

    while(trav!=null) {
        s.push(trav)
        trav = trav.left;
    }
    if(!s.isEmpty()) {
        trav = s.pop();
        print(trav.data);
        trav = trav.right;
    }
}
```

BST - BFS - Breadth First Search - Level wise Search - Non Rec



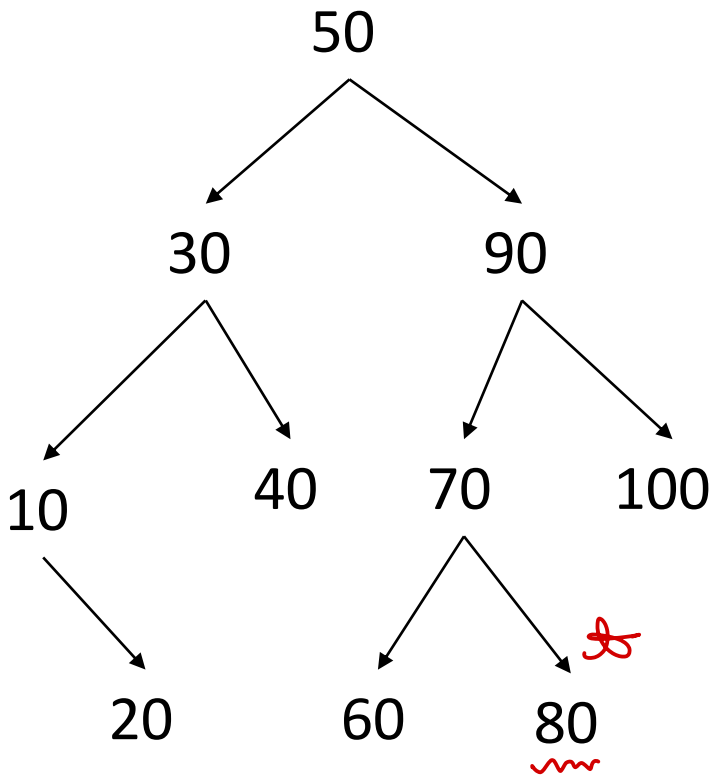
50 30 90 10 40
70 100 20 60 80

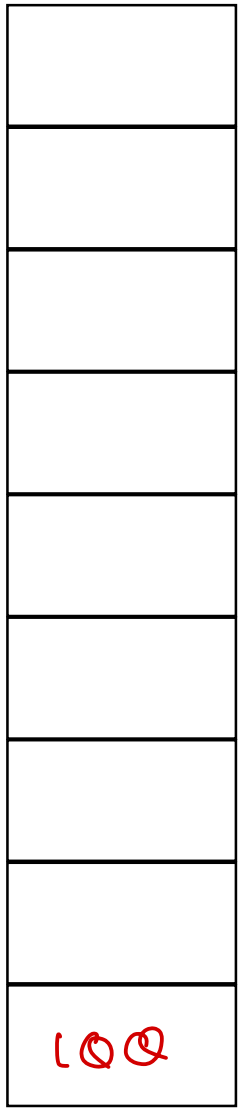
O(n)




```
q.push(root);
while(!q.isEmpty()) {
    trav = q.pop()
    if(key == trav.data)
        return trav;
    if(trav.left != null)
        q.push(trav.left);
    if(trav.right != null)
        q.push(trav.right);
}
return null;
```

BST - DFS - Depth First Search - O(n)

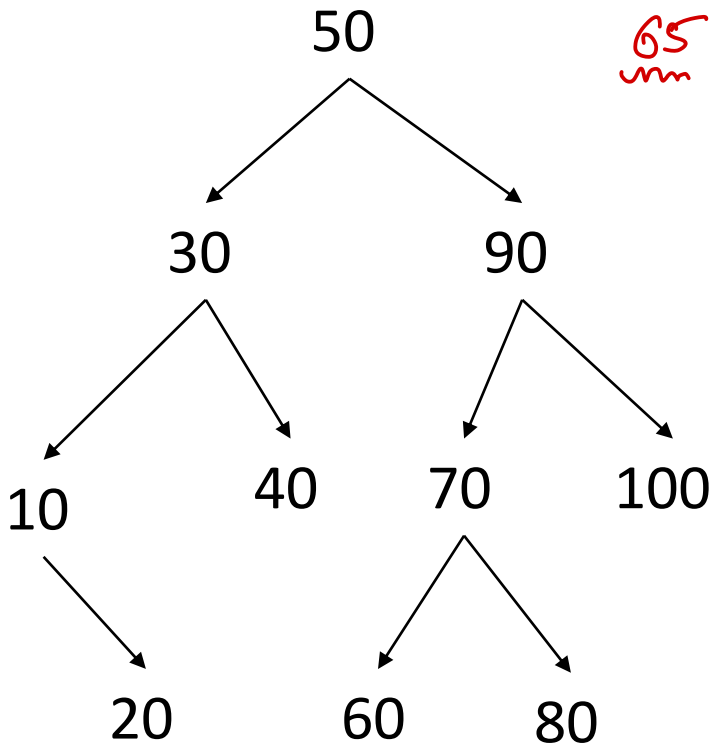




50 30 10 20 40 90 70 60 80

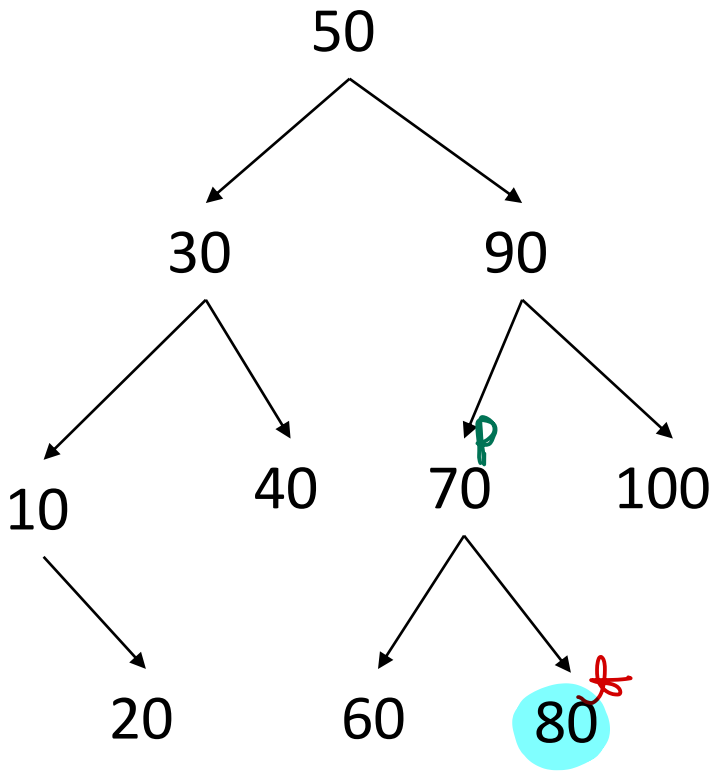
```
s.push(root);
while(!s.isEmpty()) {
    trav = s.pop();
    if(key == trav.data)
        return trav;
    if(trav.right != null)
        s.push(trav.right);
    if(trav.left != null)
        s.push(trav.left);
}
return null;
```

BST - Binary Search - O(h) → height



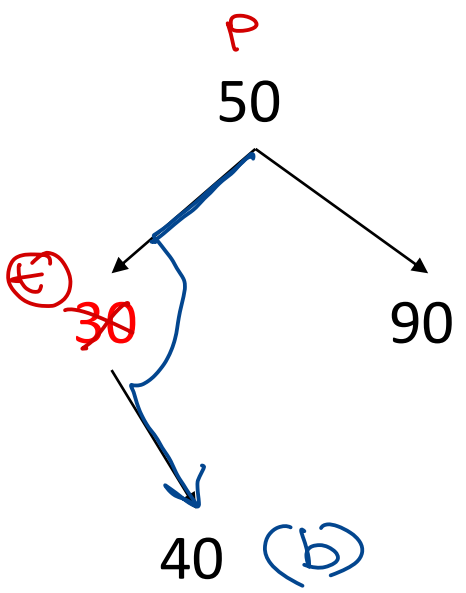

```
trav = root;
while(trav != null) {
    if(key == trav.data)
        return trav;
    if(key < trav.data)
        trav = trav.left;
    else
        trav = trav.right;
}
return null;
```

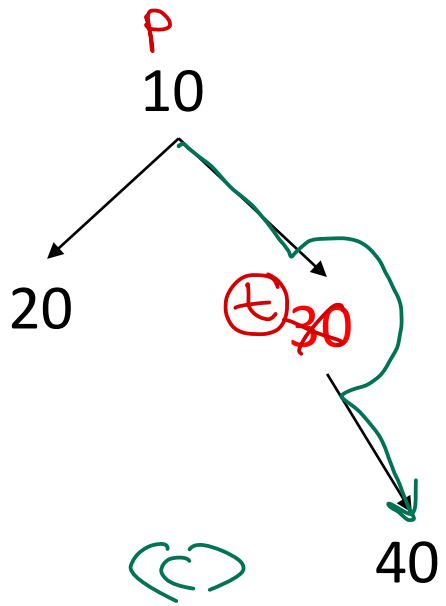
BST - Binary Search - O(h) → height

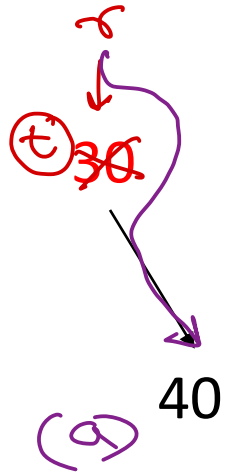



```
Parent = null;
trav = root;

while(trav != null) {
    if(key == trav.data)
        return {trav, parent};
    Parent = trav;
    if(key < trav.data)
        trav = trav.left;
    else
        trav = trav.right;
}

return {null, null};
```

BST – Delete Node → trav.left == null



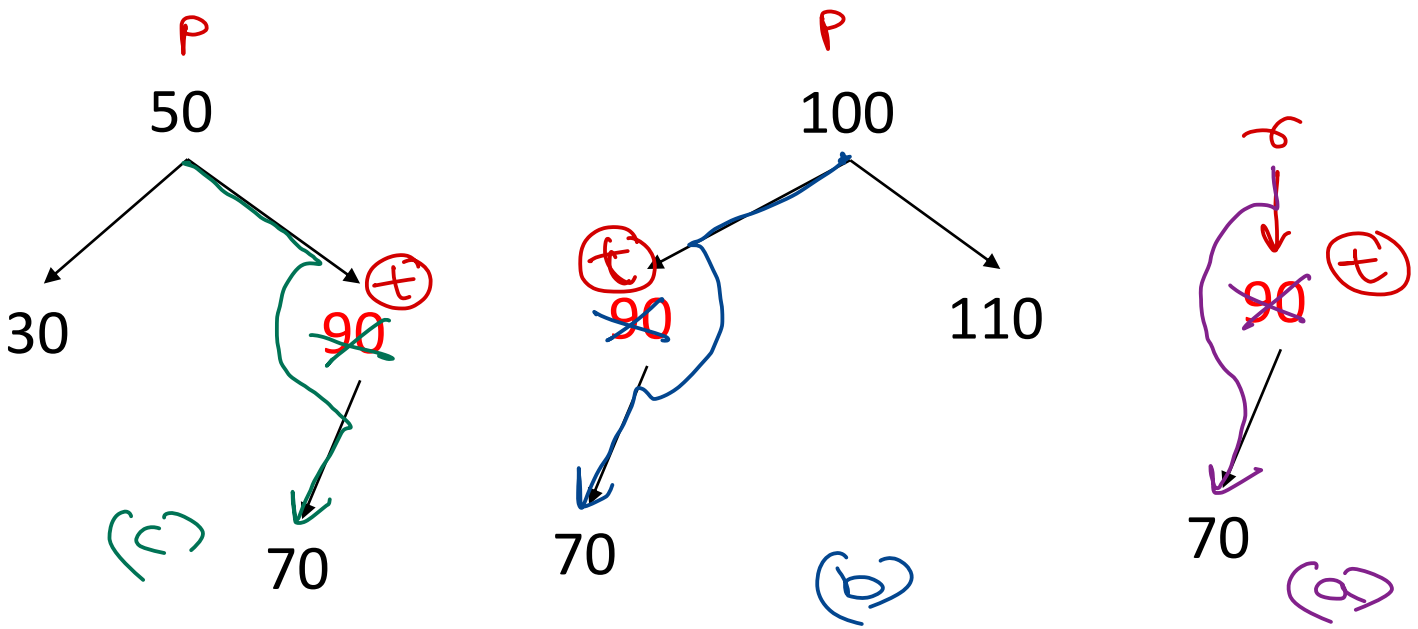





```
if(trav == root)
    root = trav.right;
else if(trav == p.left)
    p.left = trav.right;
else
    p.right = trav.right;
```

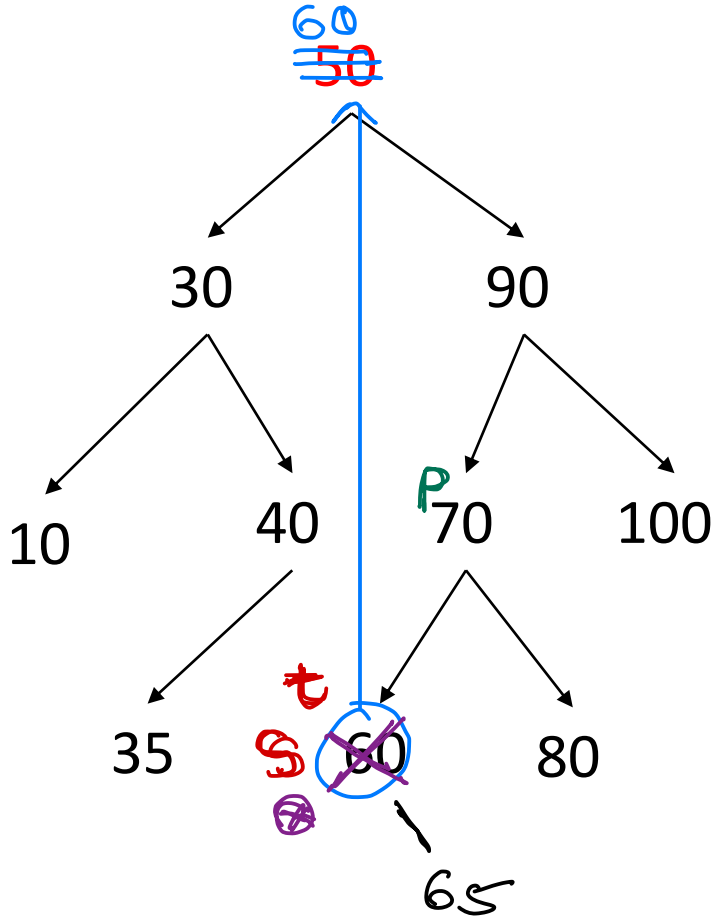
BST – Delete Node

trav->right == null



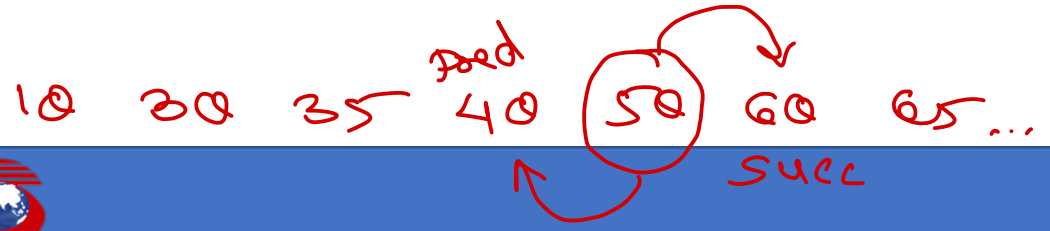

```
if(trav == root)
    root = trav->left;
else if(trav == p->left)
    p->left = trav->left;
else
    p->right = trav->left;
```

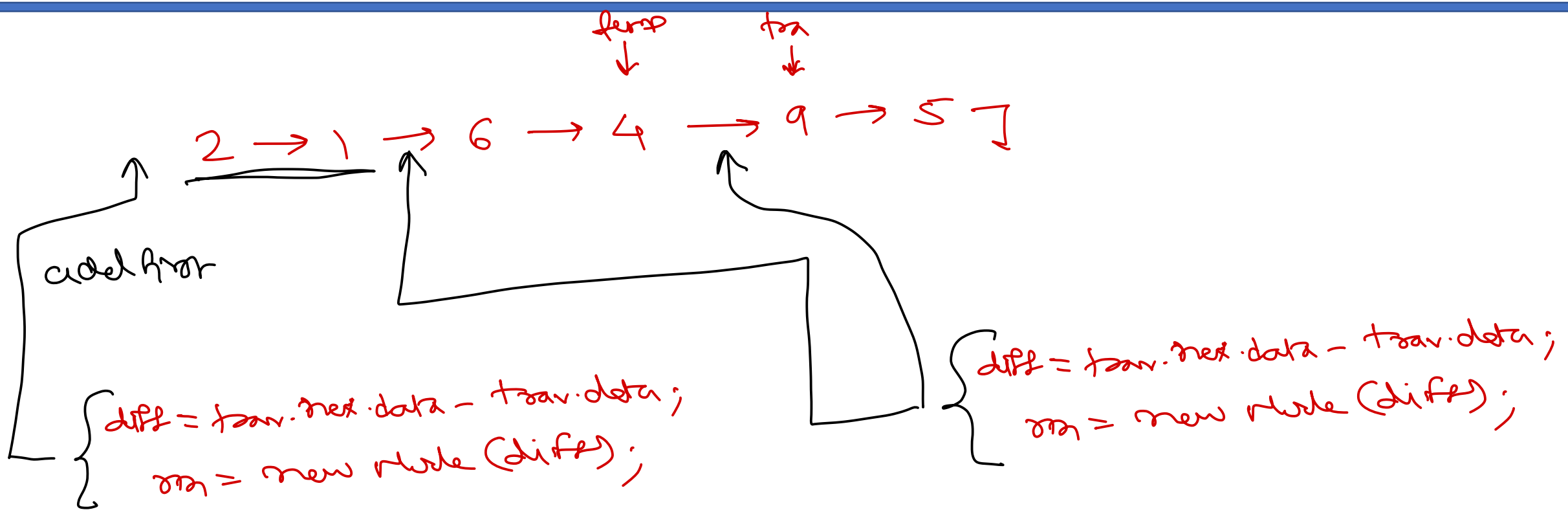
BST – Delete Node → trav.left != null && trav.right != null




```
parent = trav;
succ = trav.right;
while(succ.left != null){
    parent = succ;
    succ = succ.left;
}
trav.data = succ.data;
trav = succ;
```

```
parent.left = trav.right;
trav = null;
```




temp
trav
2 → 1 → 6 → 4 → 9 → 5
addNode
diff = trav.next.data - trav.data;
nn = new node(diff);
diff = trav.next.data - trav.data;
nn = new node(diff);

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>