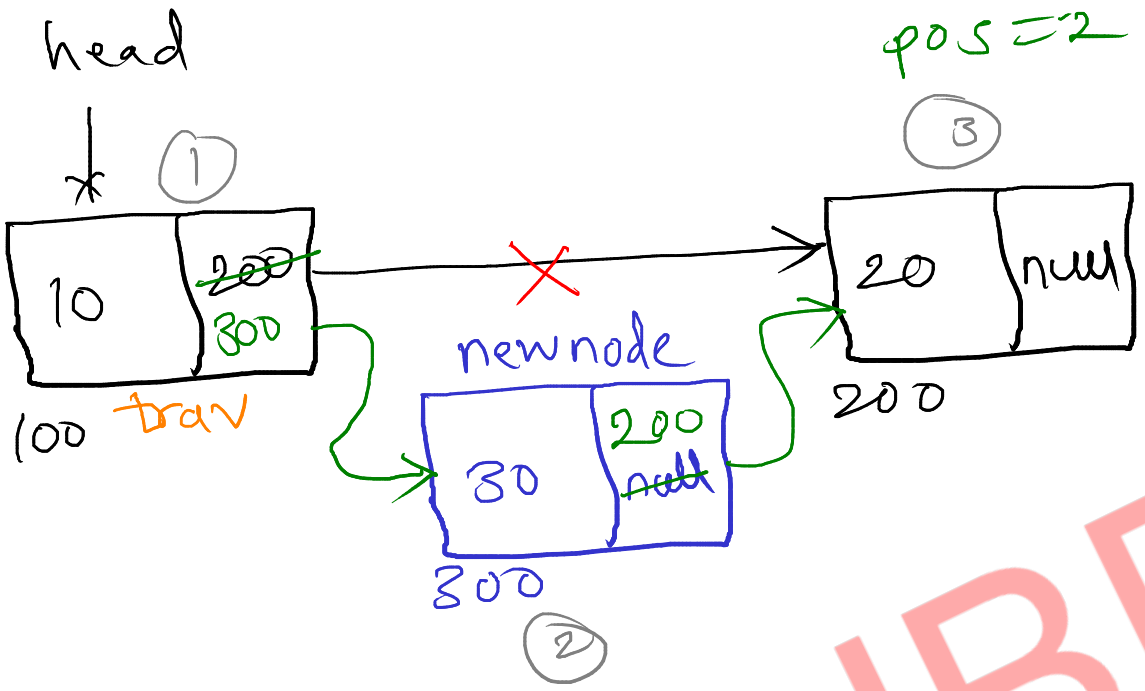
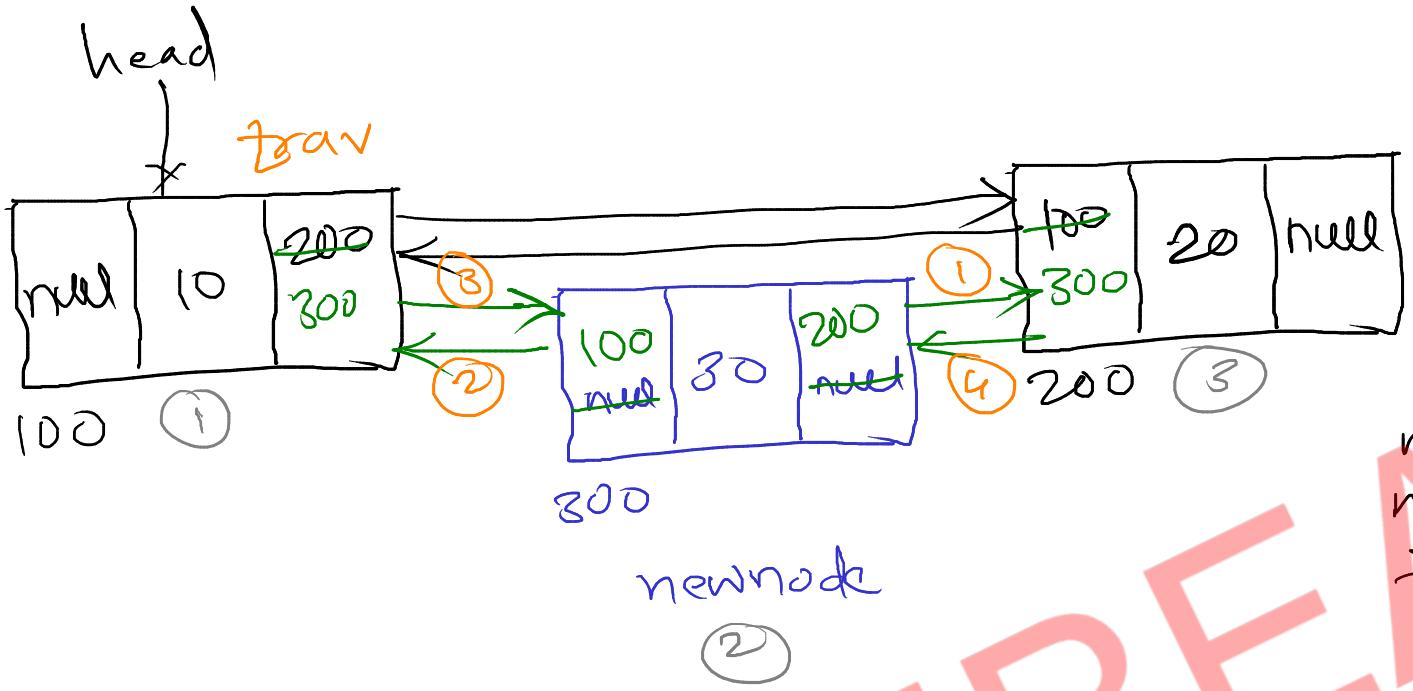


1) traverse till pos-1 node

```
Node trav=head;
for(i=1; i<pos-1 ; i++)
    trav=trav.next;
```

2) create links

```
newnode.next=trav.next;
trav.next=newnode;
```

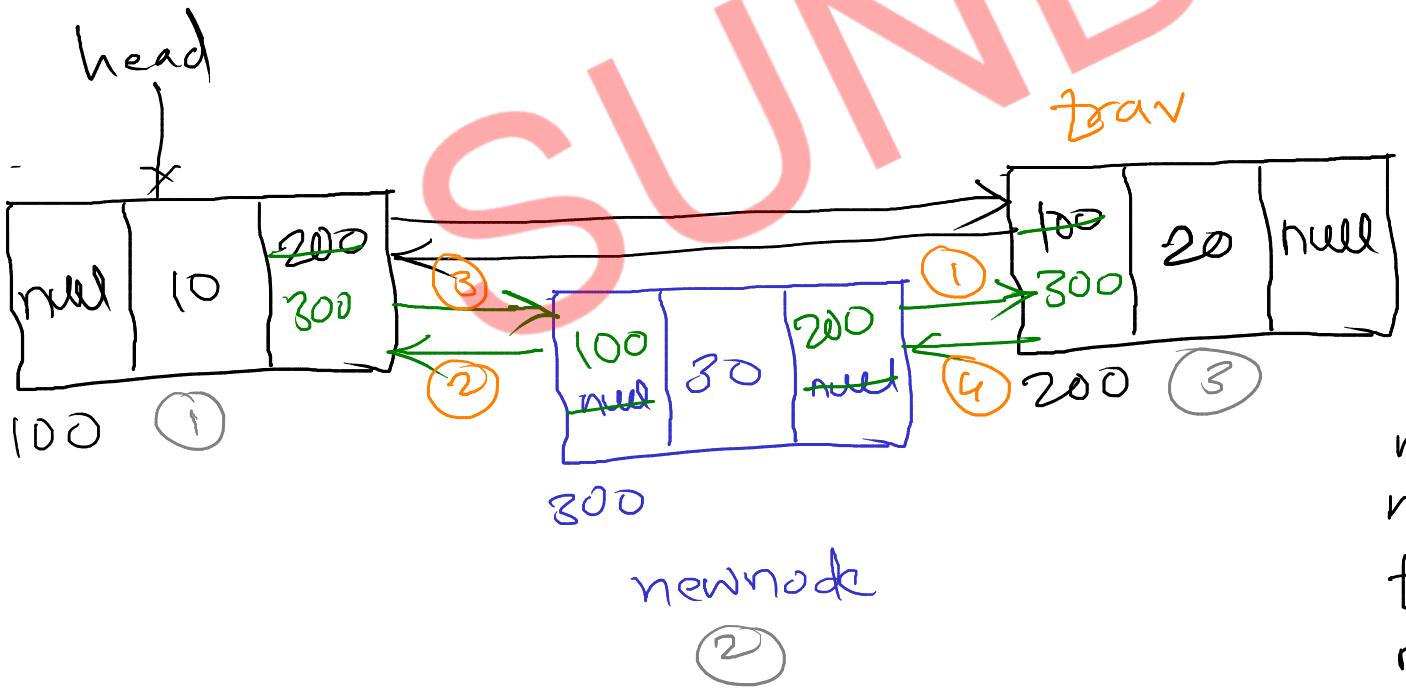


1) traverse till pos-1 node

```
Node trav = head;
for(i=1; i<pos-1; i++)
    trav = trav.next;
```

2) Create links

```
newnode.next = trav.next;
newnode.prev = trav;
trav.next = newnode;
newnode.next.prev = newnode;
```

1) traverse till pos-1 node

```
Node trav = head;
for(i=1; i<pos-1; i++)
    trav = trav.next;
```

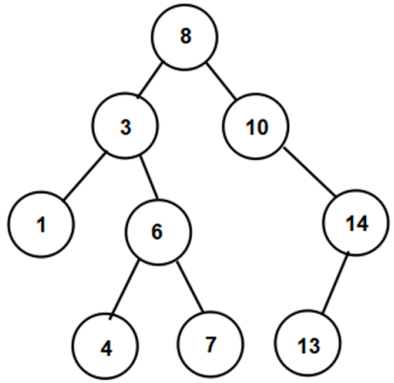
2) Create links

```
newnode.next = trav;
newnode.prev = trav.prev;
trav.prev = newnode;
newnode.prev.next = newnode;
```

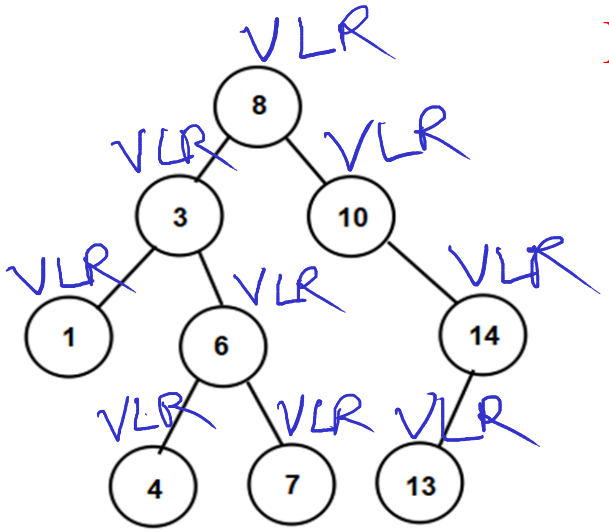
BST - Add Node

```
//1. create node with given value
//2. if BST is empty
        //add newnode into root itself
//3. if BST is not empty
        //3.1  create trav and start at root node
        //3.2 if value is less than current data (trav.data)
                //3.2.1 if left of current node is empty
                        // add newnode into left of current node
                //3.2.2 if left of current node is not empty
                        // go into left of current node
        //3.3 if value is greater or eqaul than current data (trav.data)
                //3.3.1 if right of current node is empty
                        // add newnode into right of current node
                //3.3.2 if right of current node is not empty
                        // go into right of current node
        //3.4 repeat step 3.2 or 3.3 untill node is added into BST
```

BST - Add Node



BST - Traversal

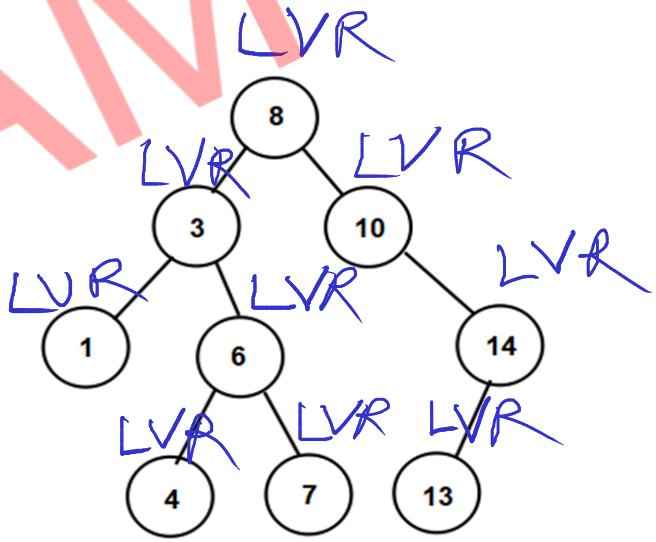


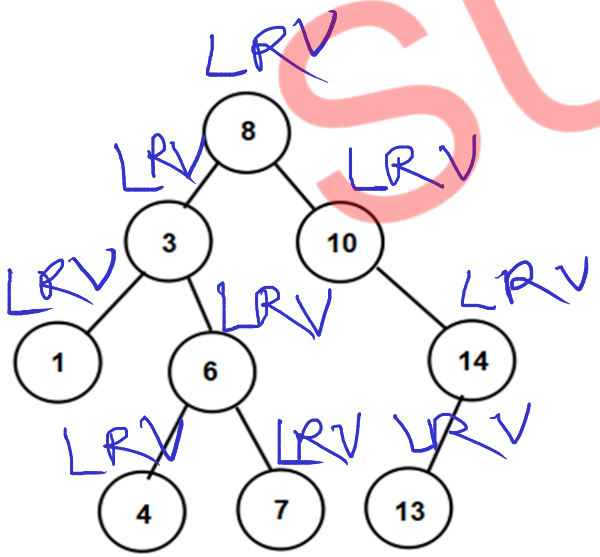
Preorder → VLR

Traversal : 8, 3, 1, 6, 4, 7, 10, 14, 13

Inorder → LVR

Traversal : 1, 3, 4, 6, 7, 8, 10, 13, 14 (sorted)

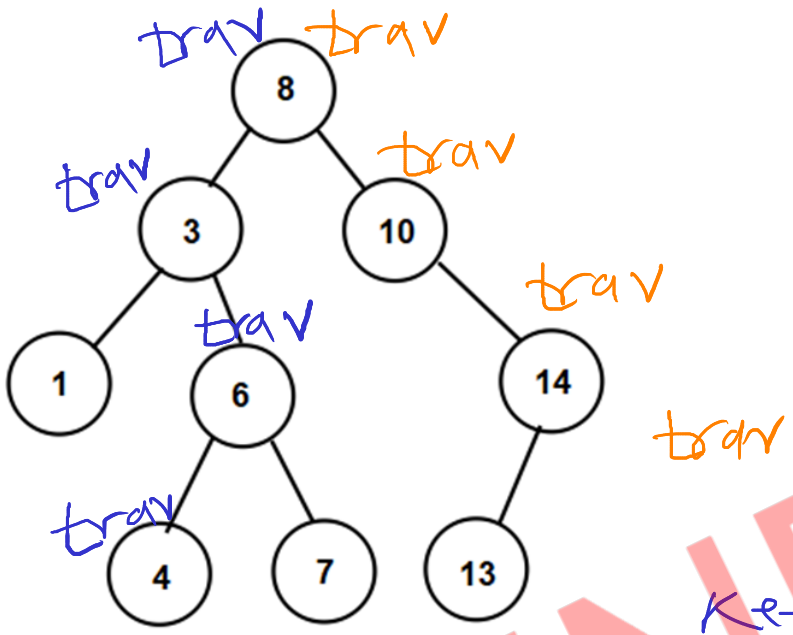




Postorder → LRV

Traversal : 1, 4, 7, 6, 3, 13, 14, 10, 8

BST - Binary Search

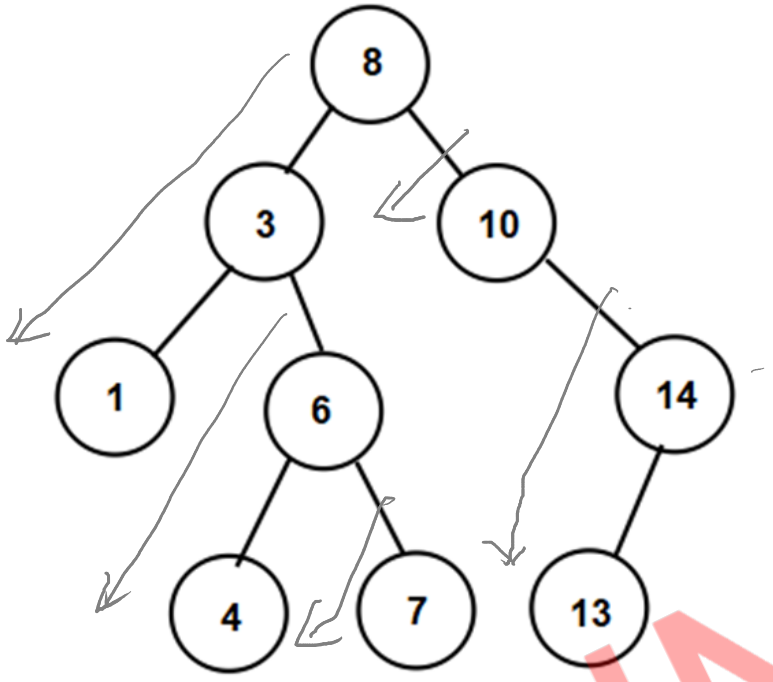


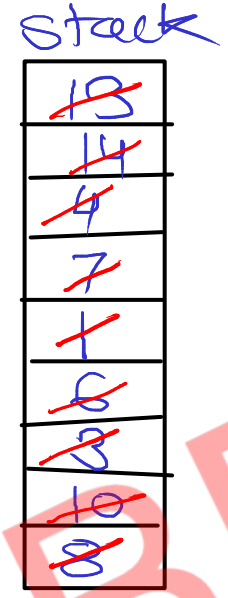
//1. start from root
//2. if key is equal to current data
//return current node
//3. if key is less than current data
// search key into left of current node
//4. if key is greater than current data
// search key into right of current node
//5. repeat step 2 to 4 till leaf nodes

Key = 4, key is found

Key = 15, key is not found

BST - DFS (Depth First search)

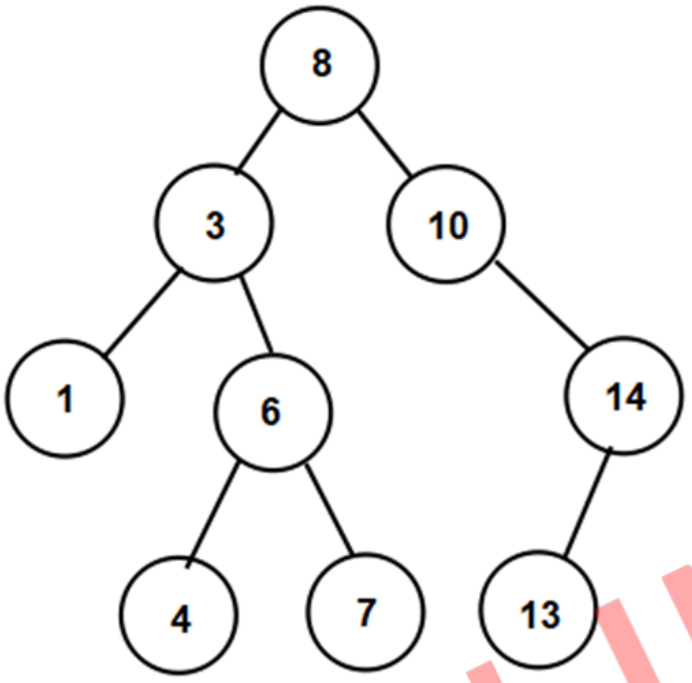


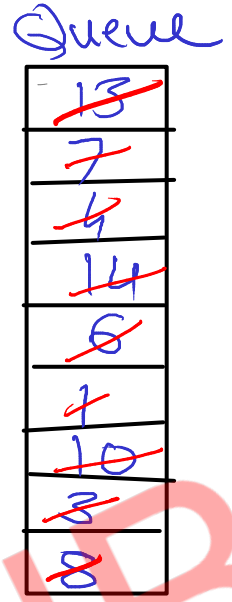


Traversed:
8, 3, 1, 6, 4, 7, 10, 14, 13

//1. push root on stack
//2. pop node from stack
//3. visit poped node
//4. if popped node has right
// push it on stack
//5. if poped node has left
// push it on stack
//6. repeat step 2 to 5 till stack iss not empty

BST - BFS (Bredth First Search)





Traversal:
8, 3, 10, 1, 6, 14, 4, 7, 13

//1. push root on queue
//2. pop node from queue
//3. visit poped node
//5. if poped node has left
// push it on queue
//4. if popped node has right
// push it on queue
//6. repeat step 2 to 5 till stack iss not empty