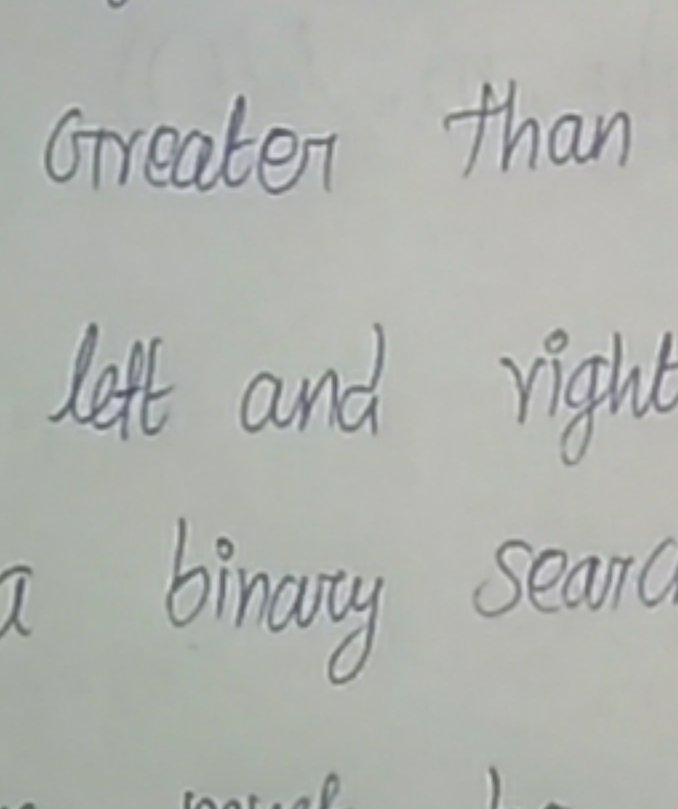
Day 33

Binary Search tree :

Binary Search tree, is a node Based binary tree data structure which has following properties.

* The left subtree of node Contains only nodes with keys legger than the nodes key
* The right subtree of a node Contains only with keys Greater than ~~than~~ the nodes key
* The left and right subtree each must also be a binary search tree.
* There must be no duplicate nodes.



Implementation for searching :-

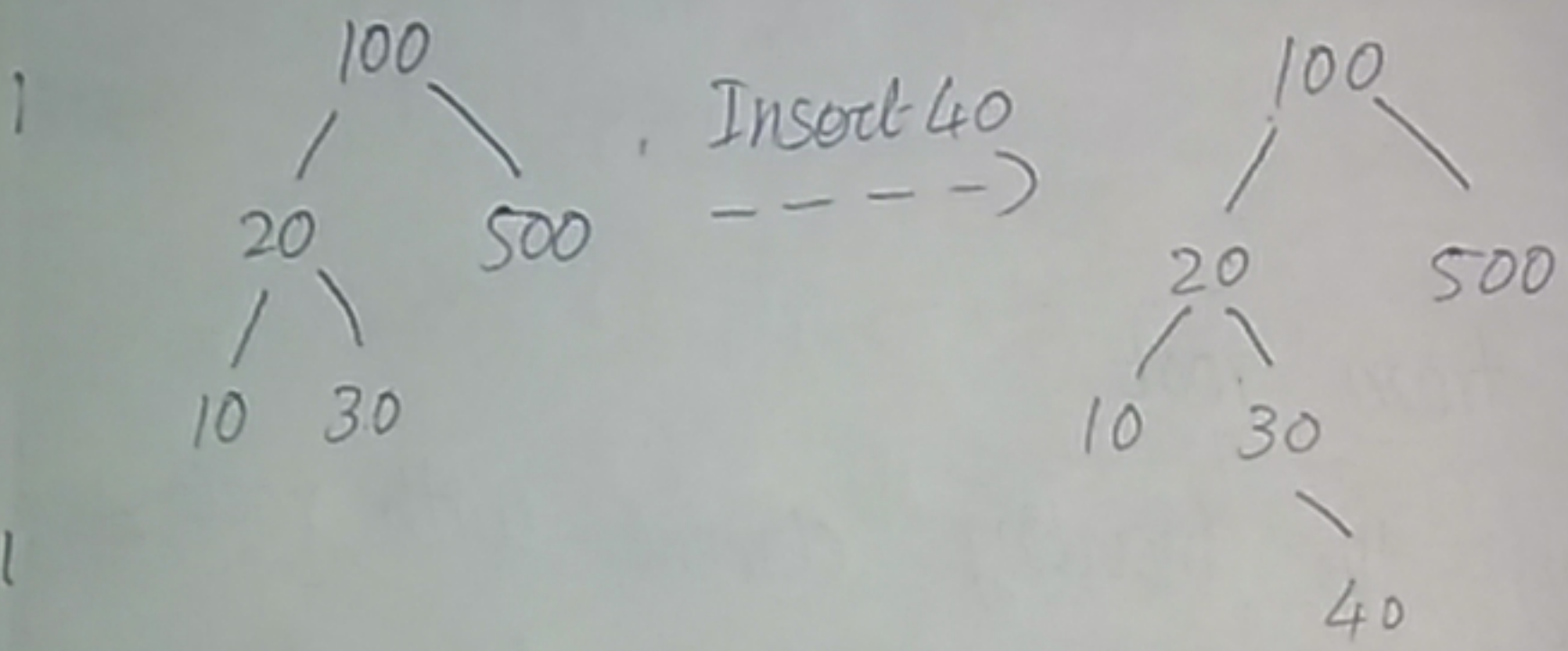
```
def search(root, key):
    if root is None or root.val == key:
        return root
    if root.val < key:
        return search(root.right, key)
    return search(root.left, key)
```

Illustration :

* start from root
* Comparing the inserting element with root, if less than root, then recurse for left, else recurse for right
* If element to search is found anywhere return true else false.

Insertion of a key :

A new key is always inserted at leaf. We started searching a key from root till we hit a leaf node. Once a leaf node is found, the new node is added as a child of the leaf node.



Implementation :

Class Node :

```
def __init__(self, key):
    self.left = None
    self.right = None
    self.val = key.
```

```
def insert (root, key):
    if root is None:
        return Node(key)
    else:
        if root.val == key
            return root
        elif root.val < key:
            root.right = insert(root.right, key)
        else:
            root.left = insert(root.left, key)
    return root

def inorder(root):
    if root:
        inorder(root.left)
        print(root.val)
        inorder(root.right)
```