Binary Search Tree

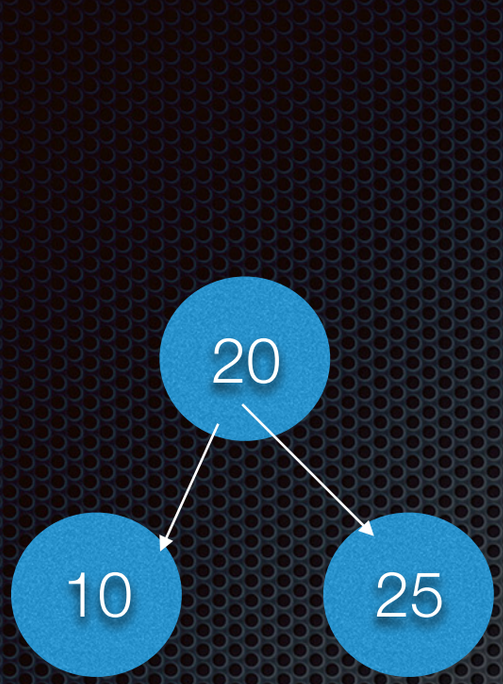
Binary Search Tree

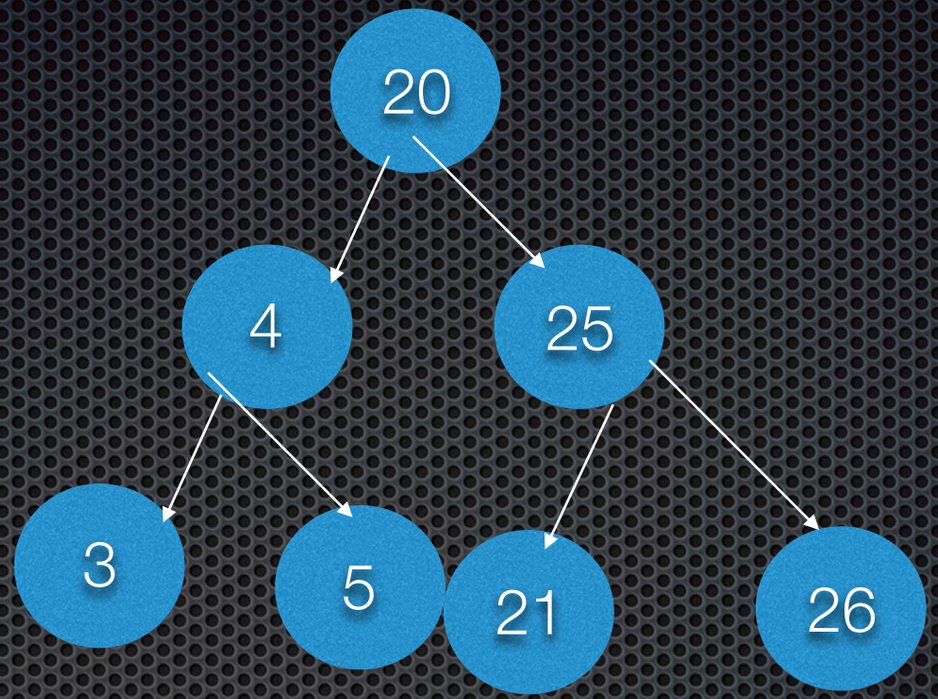
Binary Search Tree is a binary tree in which every node contains only smaller values in its left subtree and only larger values in its right subtree.

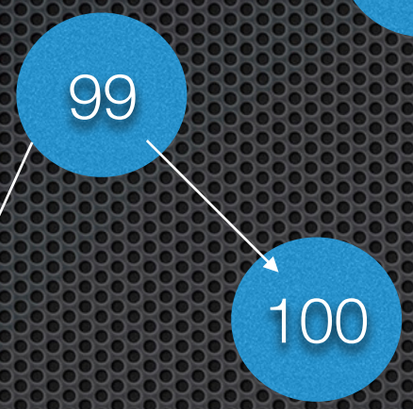
1. Everything in a node's left subtree is smaller than the node's value

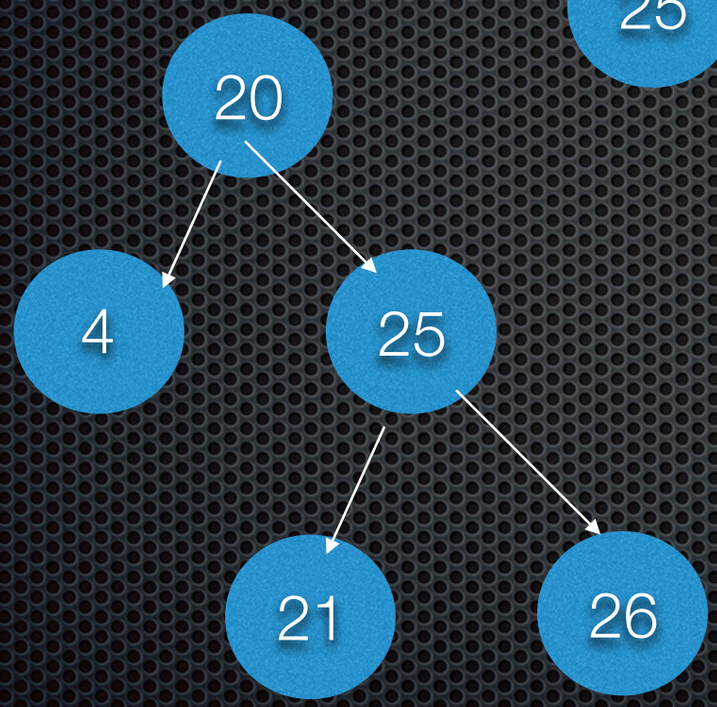
2. Everything in a node's right subtree is bigger than the node's value

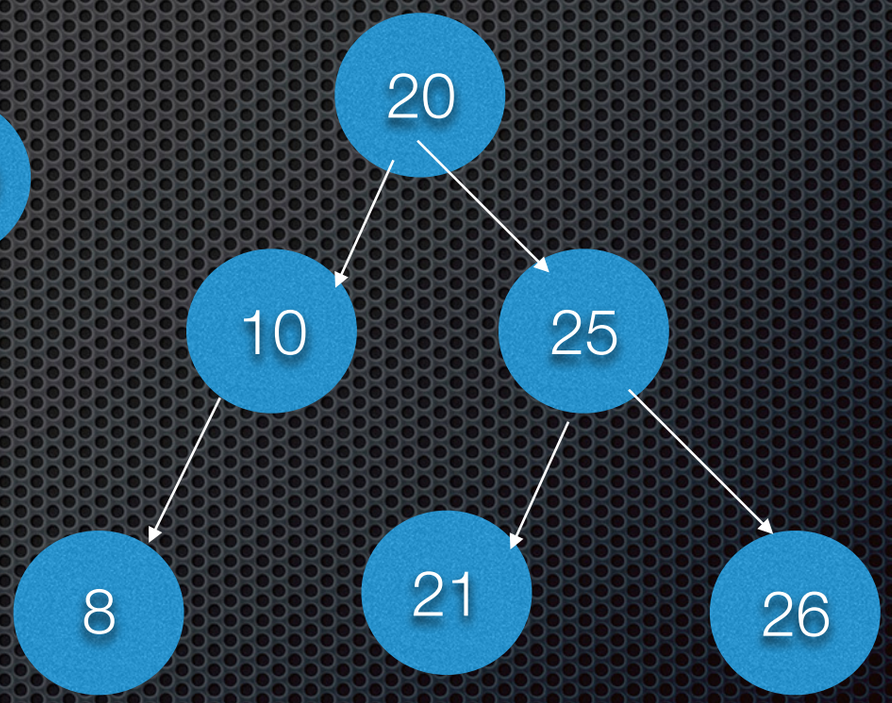
3 The left and right subtree each must also be a binary search tree


20
10
25


20
4
25
3
5
21
26


99
25
100


20
4
25
21
26


20
10
25
8
21
26

Operations on a Binary Search Tree

1. Insertion
2. Search
3. Deletion
4. Pre-order Traversal
5. In-order Traversal
6. Post-order Traversal

Binary Search Tree Creation

Binary Search Tree is a binary tree in which every node contains only smaller values in its left subtree and only larger values in its right subtree.

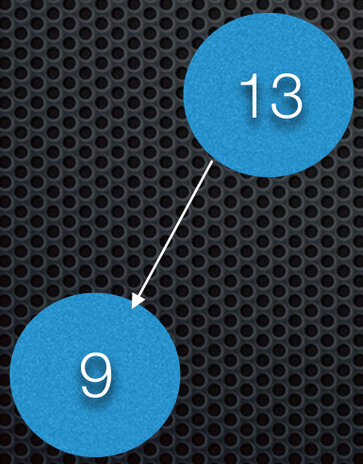
Construct a Binary Search Tree (BST) for the following sequence of numbers-

13, 9, 5, 12, 16, 19

Step 1:

Root

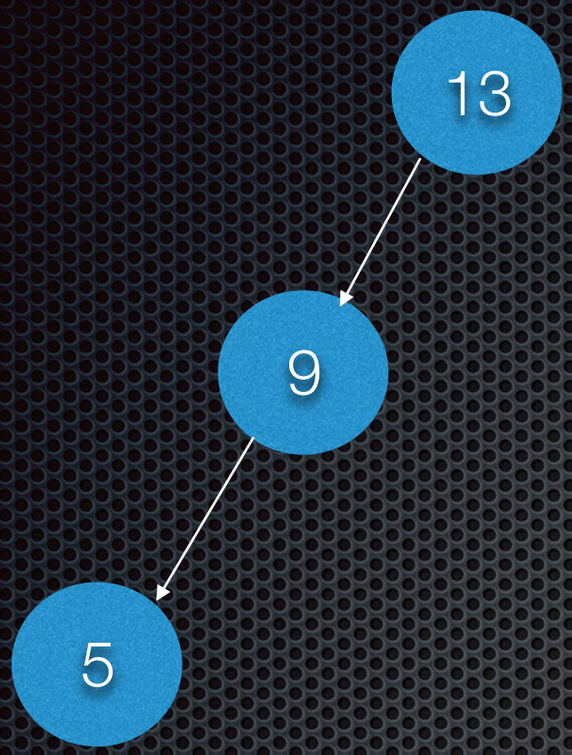
Step 2: Insert or add 9 : 9 < 13, so insert 9 to the left of 13.



Root

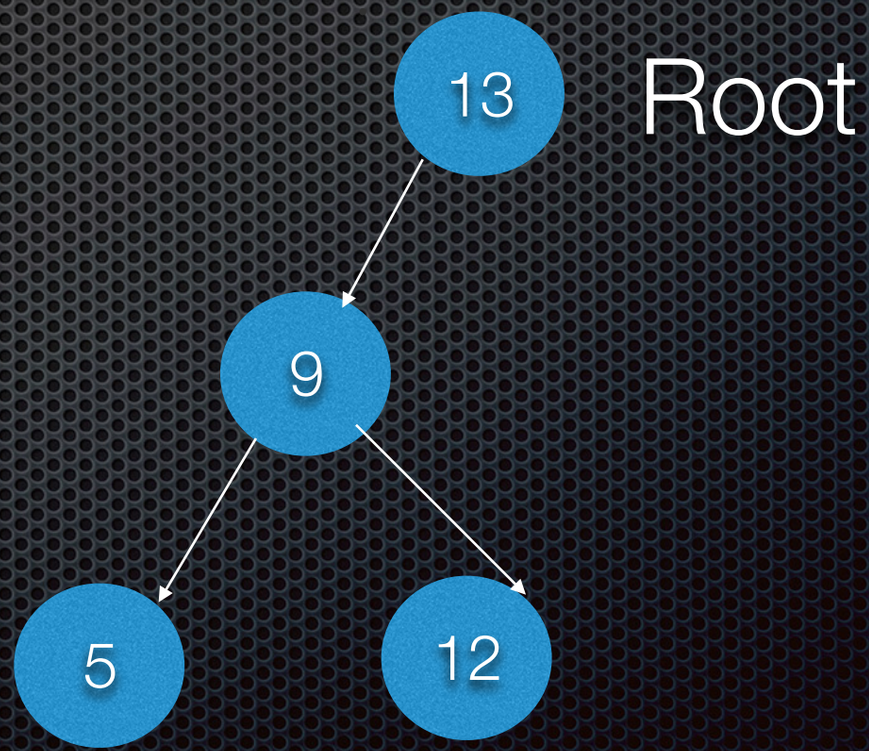
13, 9, 5, 12, 16, 19

Step 3: Insert or add 5 : As 5 < 13, so insert 5 to the left of 13.
As 5 < 9, so insert 5 to the left of 9.



13, 9, 5, 12, 16, 19

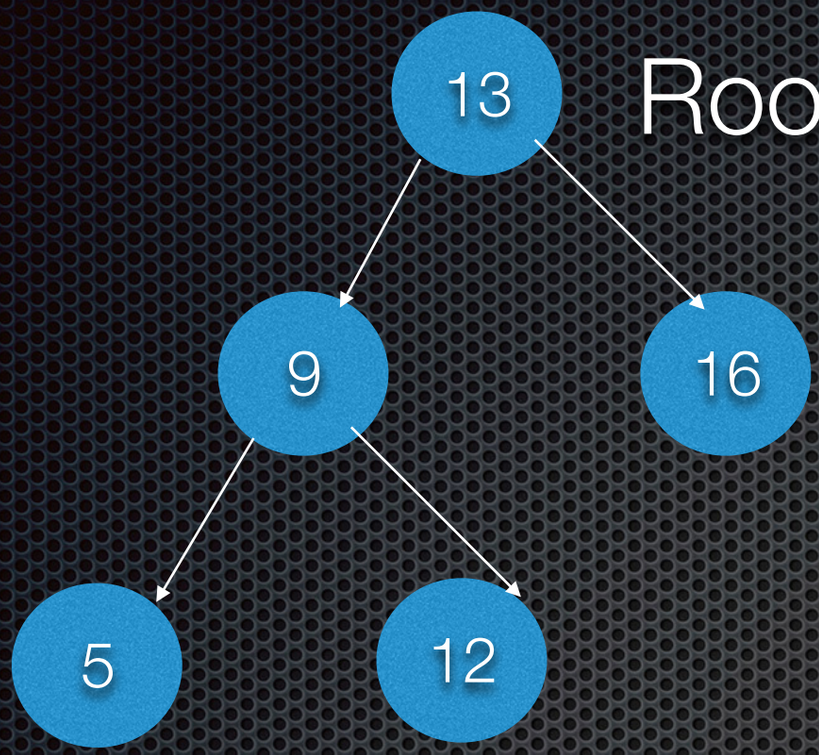
Step 4: Insert or add 12 :
As 12 < 13, so insert 12 to the left of 13.
As 12 >9, so insert 12 to the right of 9.



13, 9, 5, 12, 16, 19

Step 5:

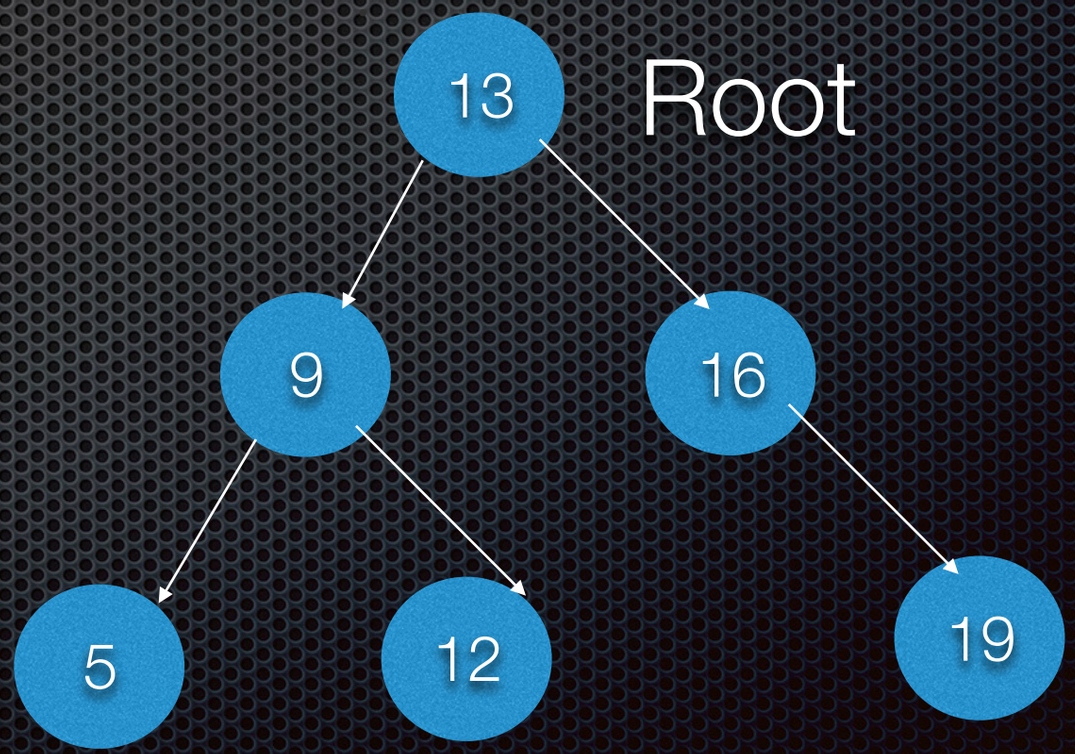
Insert or add 16 :
As 16> 13, so insert 16 to the right of 13.



13, 9, 5, 12, 16, 19

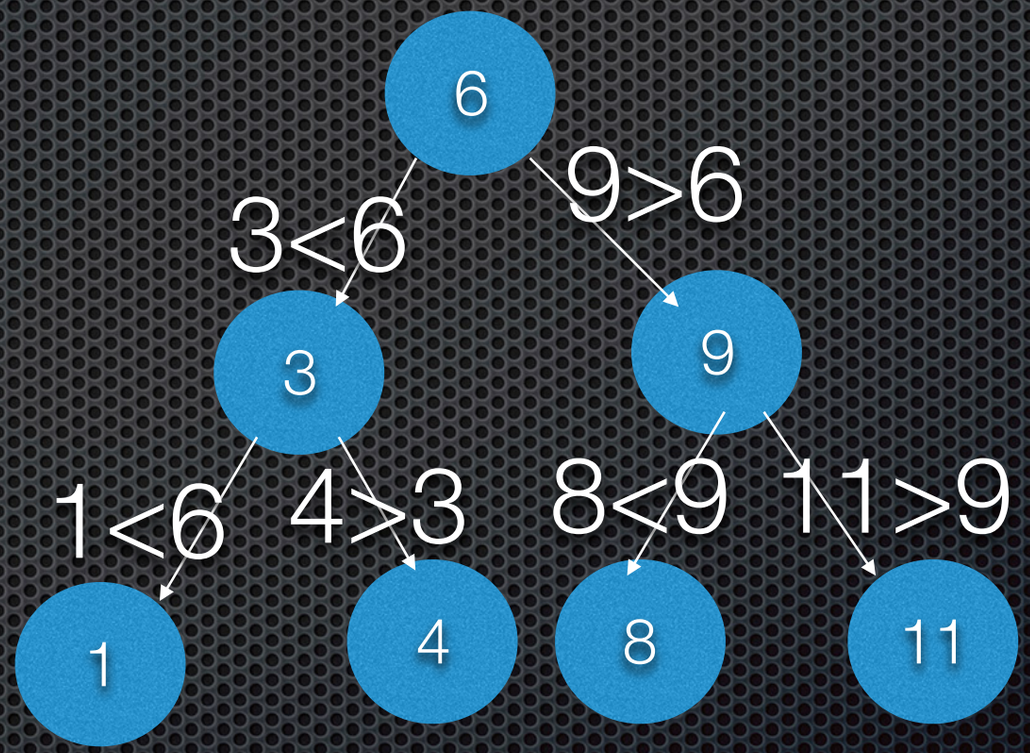
Step 6:

Insert or add 19 :
As 19 >13, so insert 19 to the right of 13.
As 19 >16, so insert 19 to the right of 16.



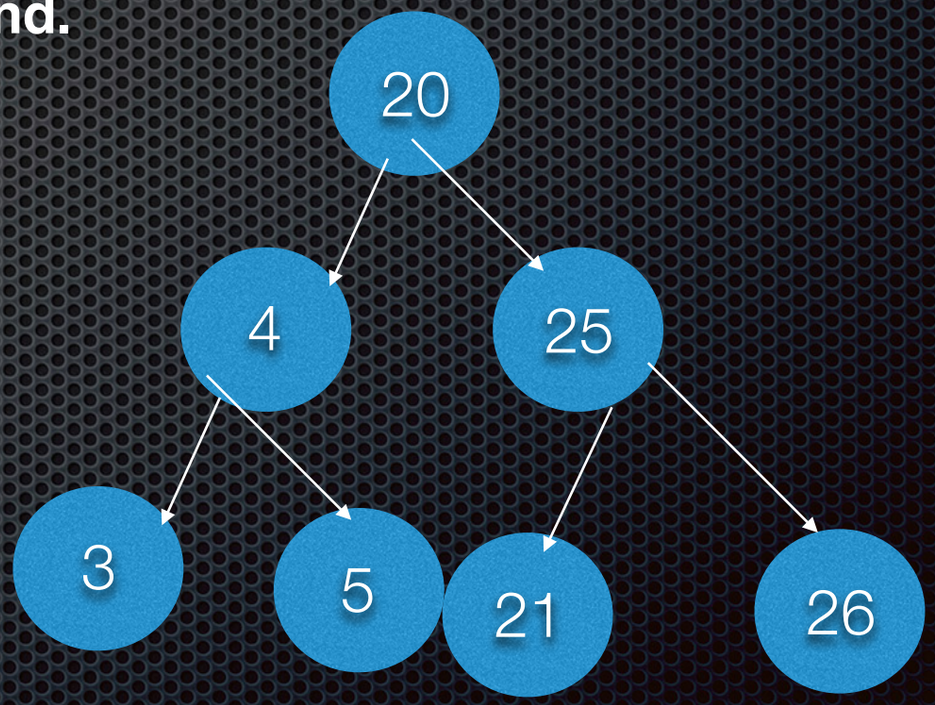
Binary Search Tree is a binary tree in which every node contains only smaller values in its left subtree and only larger values in its right subtree.

6, 3, 9,1, 4, 8,11



Searching element in Binary Search tree

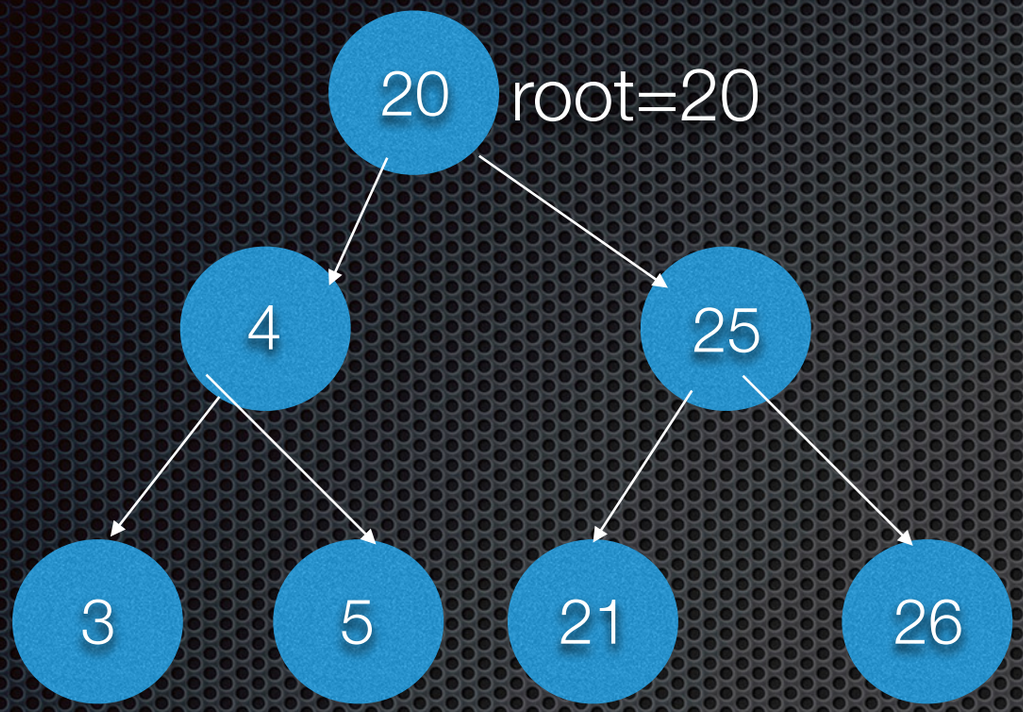
- **Compare the element with the root of the tree.**
- **If the item is matched then return node value.**
- **Otherwise check if item is less than the element present on root, if so then move to the left sub-tree.**
- **If not, then move to the right sub-tree.**
- **Repeat this procedure recursively until match found.**
- **If element is not found then return NULL.**



Search 21

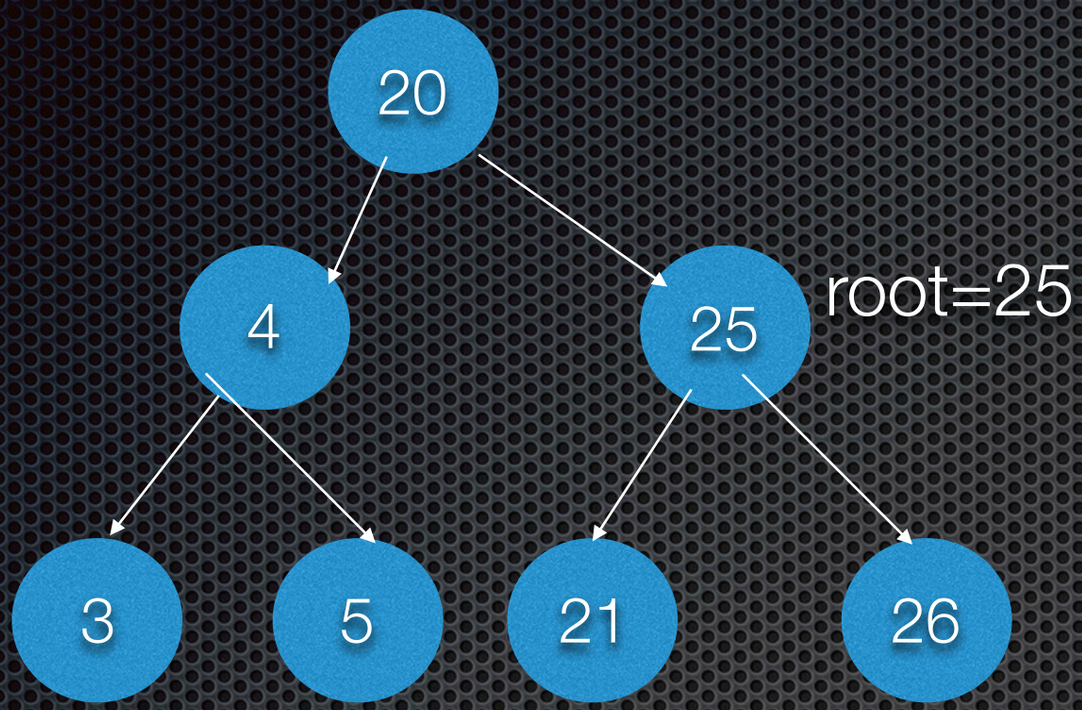
Step 1:

- **Compare 21 with the root**
- **21>20 so move to the right sub-tree.**



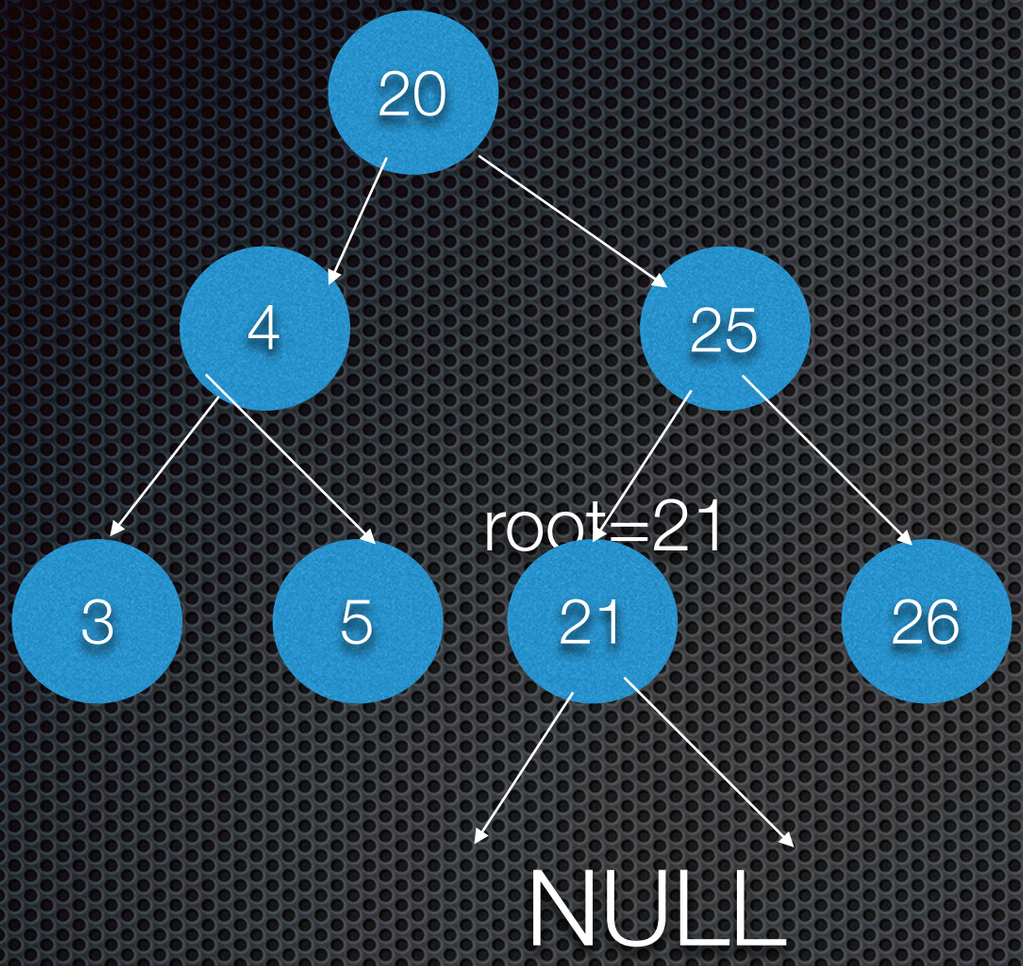
Step 2:

Now in right sub tree 25 is a root
Compare 21 with the root(25)
21<25 so move to the left sub-tree.



Step 3:

Now in left sub tree 21 is a root
Compare 21 with the root(21)
21 == 21 Match(found) return 21



root=20

Search 30

30>20(search in right sub tree)

In Right sub tree root=25

Search 30

30>25

Right sub tree root=37

Search 30

30<37

In Left sub tree root=30

Search 30

30=30(found)