98. Validate Binary Search Tree

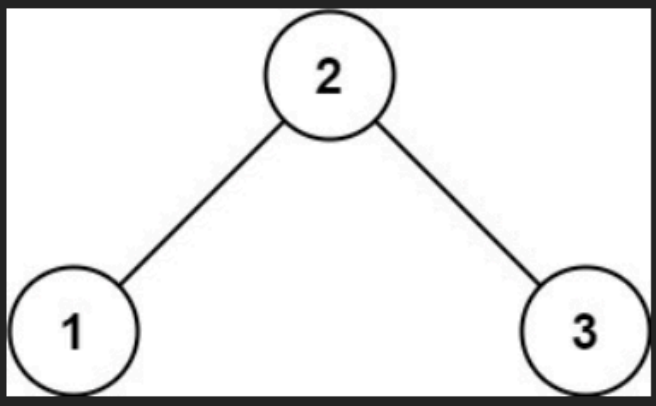
Problem Statement

1. Given the `root` of a binary tree, *determine if it is a valid binary search tree (BST).*

   A **valid BST** is defined as follows:

   - The left subtree of a node contains only nodes with keys **less than** the node's key.
   - The right subtree of a node contains only nodes with keys **greater than** the node's key.
   - Both the left and right subtrees must also be binary search trees.

   Example 1:

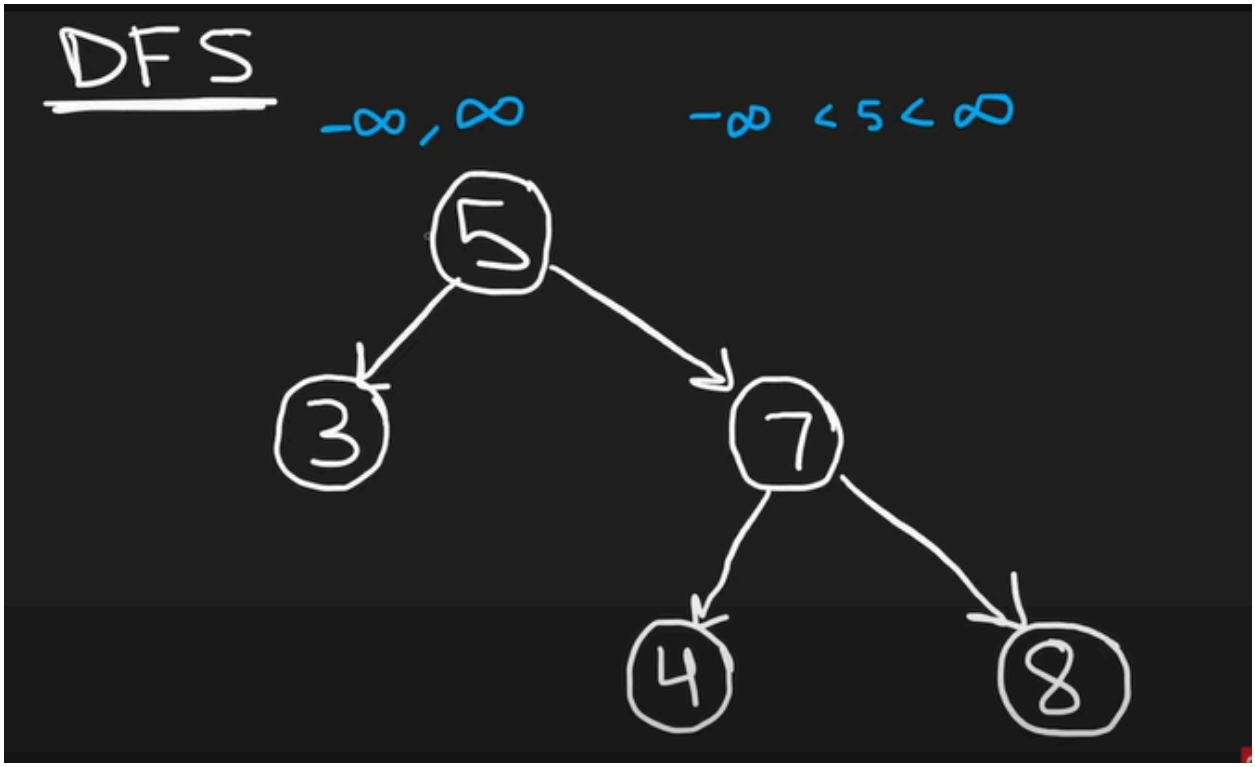
   


   ```
   Input: root = [2,1,3]
   Output: true
   ```

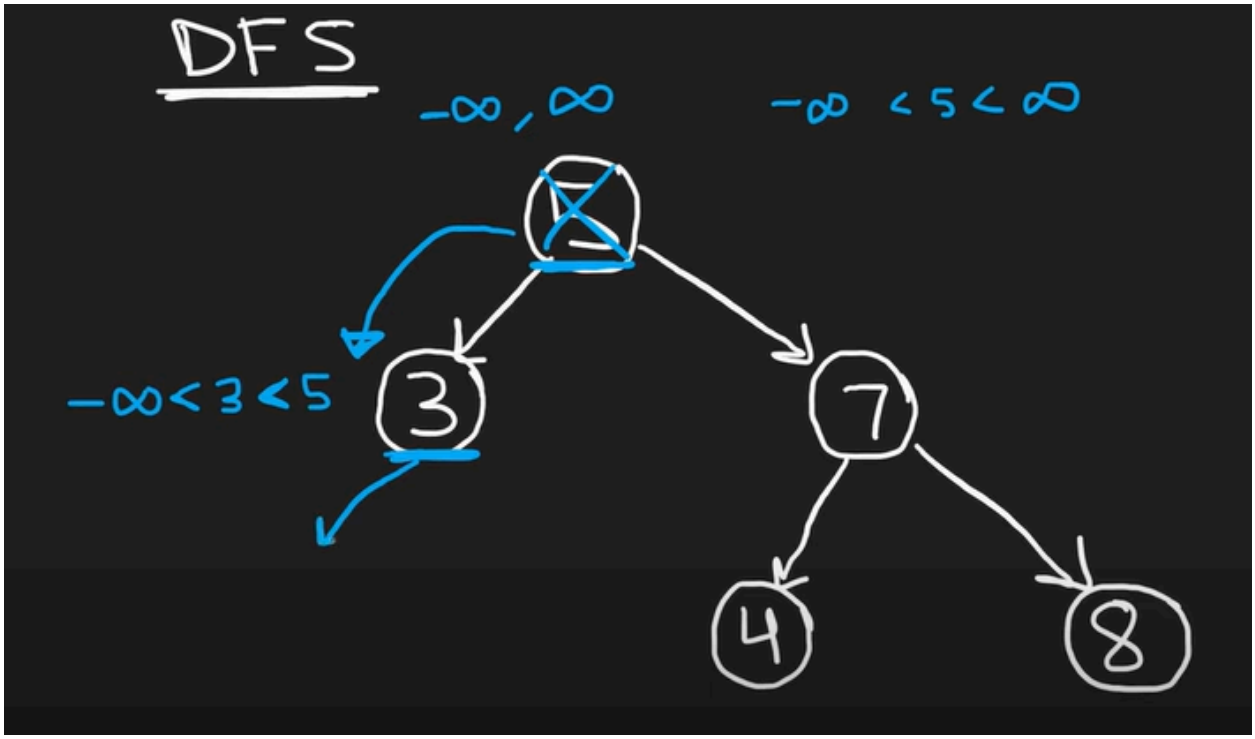
Approach

1. Here we are just comparing the elements with left and right
2. Initially our left and right would be -inf,inf

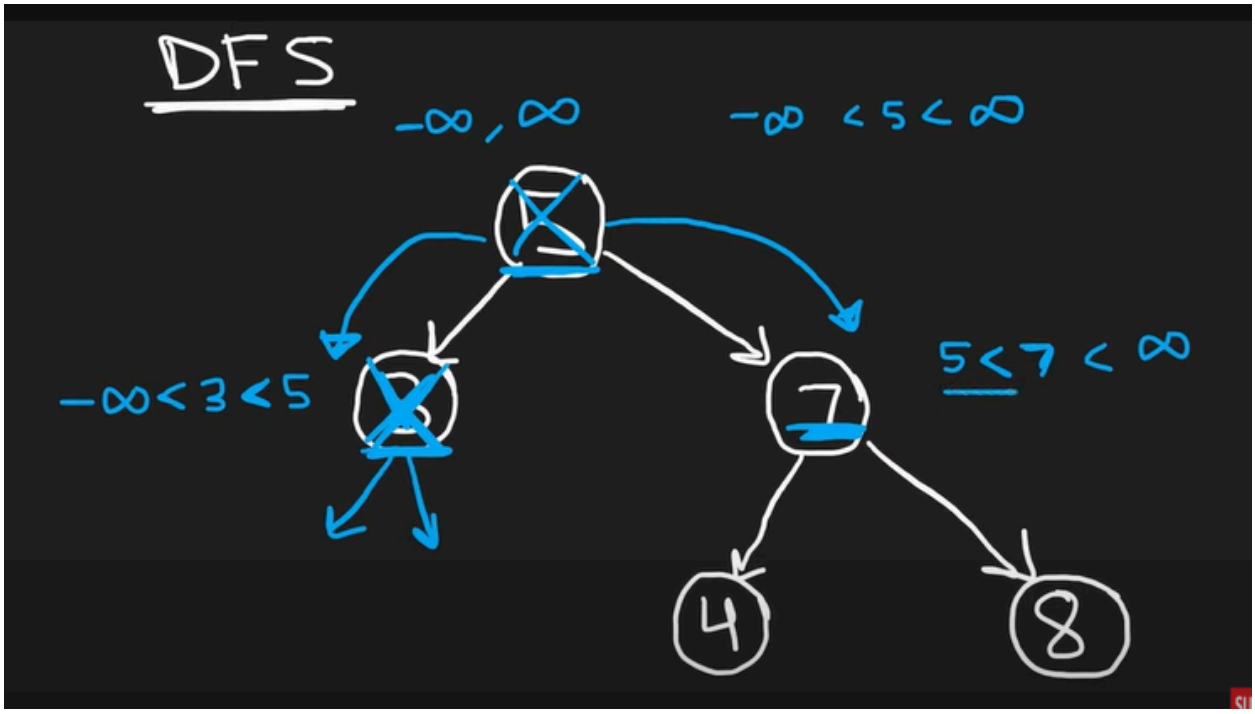
3. We choose extreme values becuz our root can be in any where bw them



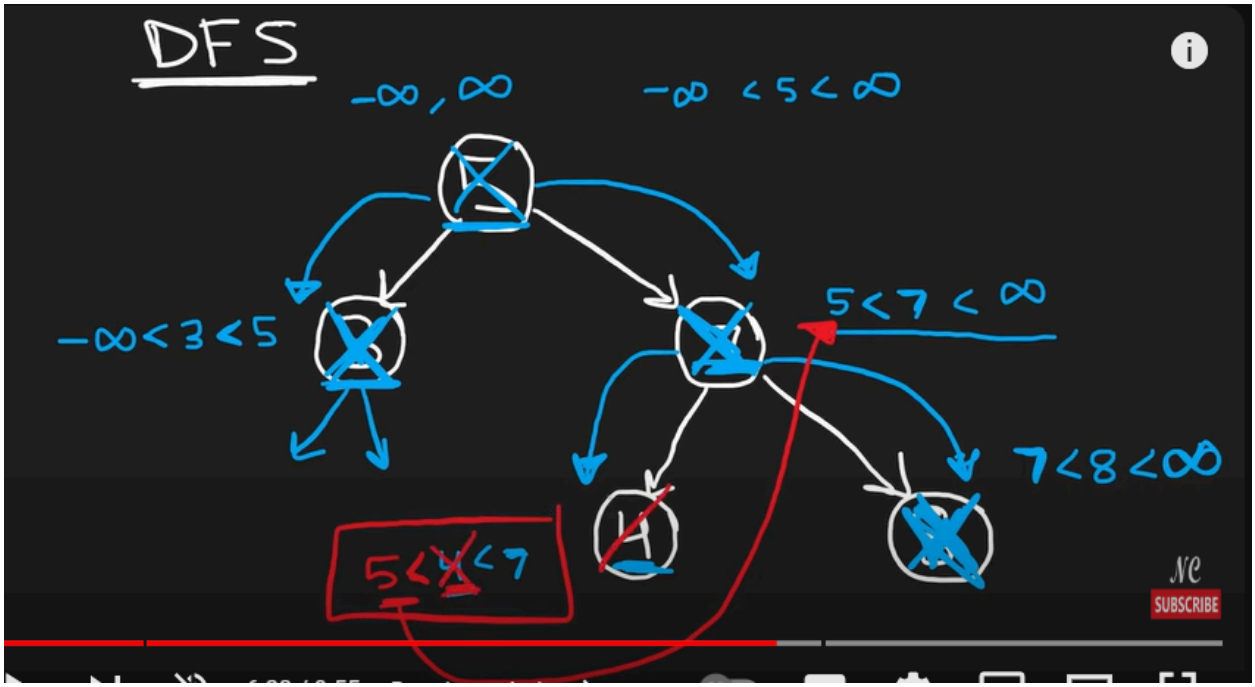
4. Now for the LST(node 3) ..the right value will be 5 and left will be -INF…means it has to be less than root value



5. Similarly for RST(node 7)...the left value will be 5 and right value will be inf

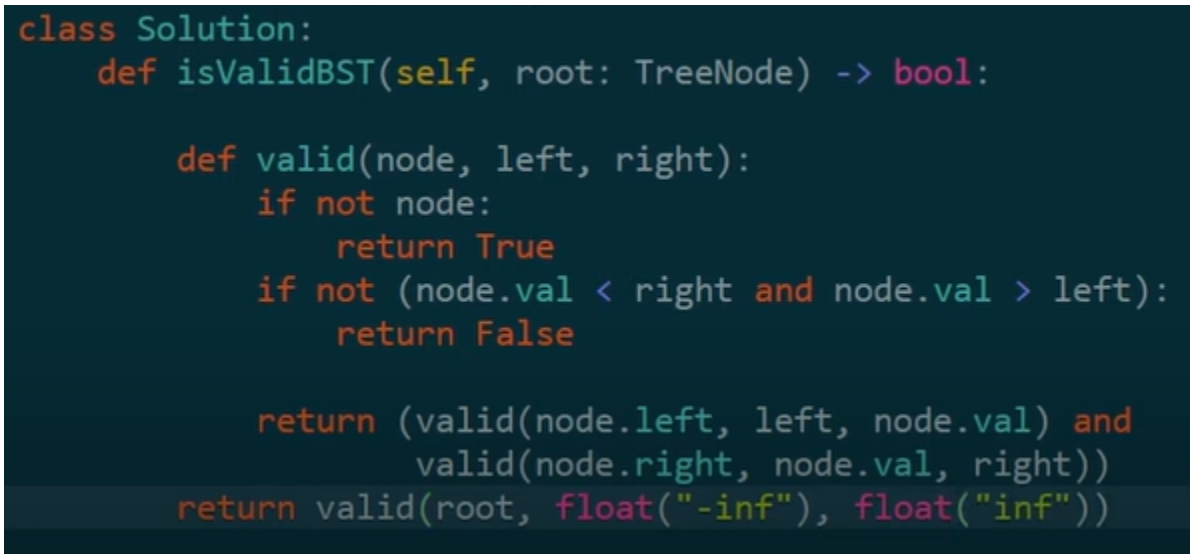


6. Similarly we compare the other nodes



here node 4 does not follow the property of BST…as it has to be greater than 5

Python code :

1. 

```python
class Solution:
    def isValidBST(self, root: TreeNode) -> bool:

        def valid(node, left, right):
            if not node:
                return True
            if not (node.val < right and node.val > left):
                return False

            return (valid(node.left, left, node.val) and
                    valid(node.right, node.val, right))
        return valid(root, float("-inf"), float("inf"))
```