Predecessor and Successor

There is BST given with the root node with the key part as an integer only. You need to find the in-order **successor** and **predecessor** of a given key. If either predecessor or successor is not found, then set it to **NULL**.

Note:- In an inorder traversal the number just **smaller** than the target is the predecessor and the number just **greater** than the target is the successor.

Example 1:

```
Input:
         10
       /    \
      2     11
    /   \
   1     5
       /   \
      3     6
       \
        4
key = 8
Output:
6 10
Explanation:
In the given BST the inorder predecessor of 8 is 6 and
inorder successor of 8 is 10.
```

Example 2:

```
Input:
      8
    /   \
   1     9
    \     \
     4    10
    /
   3
key = 11
Output:
10 -1
Explanation:
In given BST, the inorder predecessor of 11 is 10 whereas
it does not have any inorder successor.
```

Your Task: You don't need to print anything. You need to update **pre** with the predecessor of the key or **NULL** if the predecessor doesn't exist and **succ** to the successor of the key or **NULL** if the successor doesn't exist. pre and succ are passed as an argument to the function **findPreSuc()**.

Expected Time Complexity: O(Height of the BST).
Expected Auxiliary Space: O(Height of the BST).

Constraints:
1 <= Number of nodes <= 10^4
1 <= key of node <= 10^7
1 <= key <= 10^7