255. Verify Preorder Sequence in Binary Search Tree

Created	@June 14, 2021 3:09 PM
Tags	Medium
link	https://leetcode.com/problems/verify-preorder-sequence-in-binary-search-tree/

Description

Given an array of unique integers preorder, return true if it is the correct preorder traversal sequence of a binary search tree.

Approach

- The preorder is Root-Left-Right
- In a Binary Search Tree, each value to the left of a given node must be less than the value of the given node, and each value to the right must be greater.
- In preorder traversal, we go all the way down the left subtree until reaching a leaf node, and then work up one level and traverse the right subtree, and so on.
- We can simulate this traversal by keeping track of both a `leftSubtreeValues` stack and a `minimumSubtreeValue`.

1. If the `value`s are consistently decreasing, then we are traversing down a left subtree.
2. Once we read a value greater than the value at the top of the stack, we know we have entered a right subtree.
    - This greater node **will belong to one of the values** previously seen.
    - To determine which one is the parent, we pop values off the `leftSubtreeValues` stack until the current value is the greatest.

3. This final popped value will become the lower bound or `minimumSubtreeValue` that each successive value must be greater than.
4. If we encounter a value that is not greater than `minimumSubtreeValue`, then the list provided by `preorder` is not a valid preorder traversal.

```
class Solution {
public:
    bool verifyPreorder(vector<int>& preorder) {
        stack<int> s;

        int current_root = INT_MIN;

        for (auto n : preorder) {
            if (n < current_root) return false;

            while (!s.empty() && s.top() < n) {
                current_root = s.top();
                s.pop();
            }
            s.push(n);
        }

        return true;
    }
};
```