Binary Search Tree Iterator (LeetCode 173 - Medium)

Problem Description

Implement an iterator over a binary search tree (BST). Your iterator will be initialized with the root node of a BST.

Calling **next()** will return the next smallest number in the BST.

Note: next() and **hasNext()** should run in average O(1) time and uses O(*h*) memory, where *h* is the height of the tree.

```
* Your BSTIterator will be called like this:
* BSTIterator i = new BSTIterator(root);
* while (i.hasNext()) v[f()] = i.next();
```

Solution

```
//DFS - stack
public class BSTIterator {

    Stack<TreeNode> stack;

    public BSTIterator(TreeNode root) {
        stack = new Stack<TreeNode>();
        pushAllLeft(root);
    }

    /** @return whether we have a next smallest number */
    public boolean hasNext() {
        return !stack.isEmpty();
    }

    /** @return the next smallest number */
    public int next() {
        TreeNode currentTop = stack.pop();
        int rv = currentTop.val;
        pushAllLeft(currentTop.right);
        return rv;
    }

    private void pushAllLeft(TreeNode node) {
       while (node != null) {
              stack.push(node);
              node = node.left;
        }
    }
}
```