Exercise: Checking the BST property

Challenge yourself with an exercise in which you'll have to check the BST property for a Binary Search Tree!

WE'LL COVER THE FOLLOWING

Problem

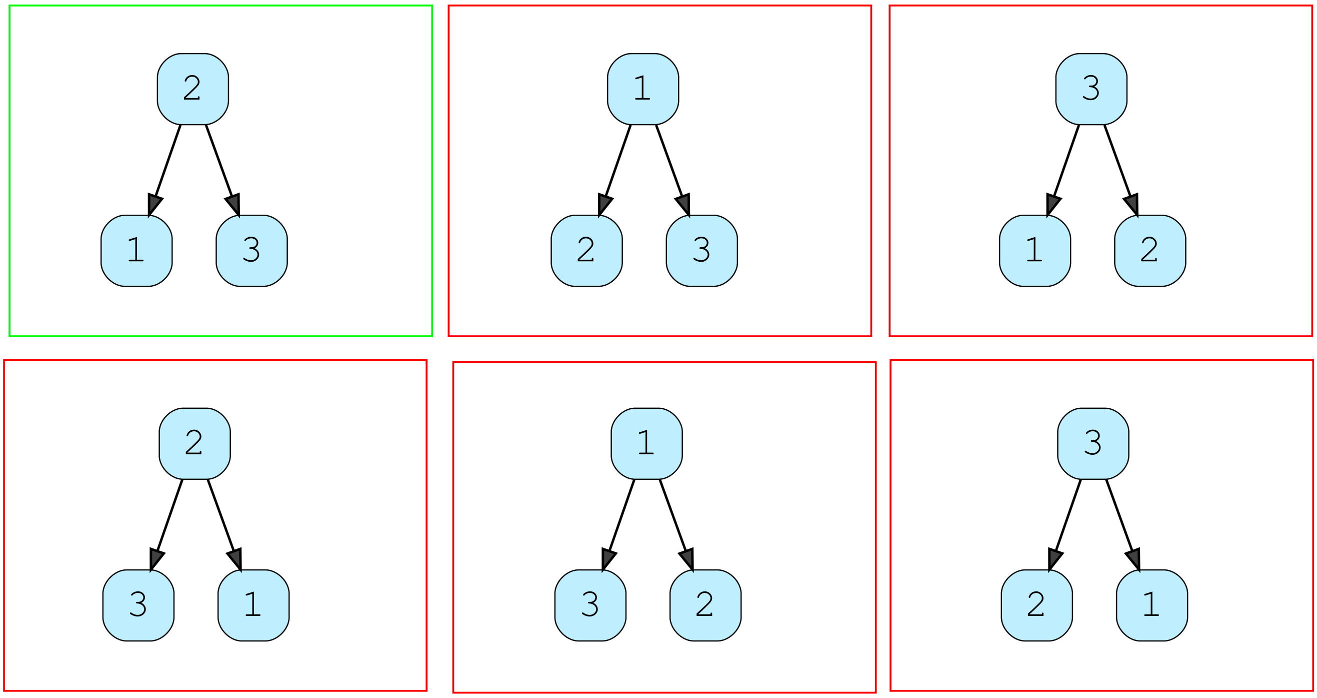
You are required to check and determine whether a tree satisfies the BST property. First of all, let’s define the BST property.

BST Property

The BST property states that every node on the right subtree has to be larger than the current node, and every node on the left subtree has to be smaller than the current node.

The binary search tree property (BST property) is a global property that every binary search tree must satisfy.

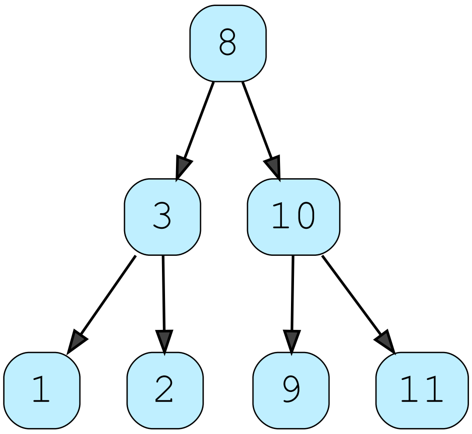
Below are some examples that show which trees satisfy the BST property:



Examples

Here are some other examples where we check for the BST property:

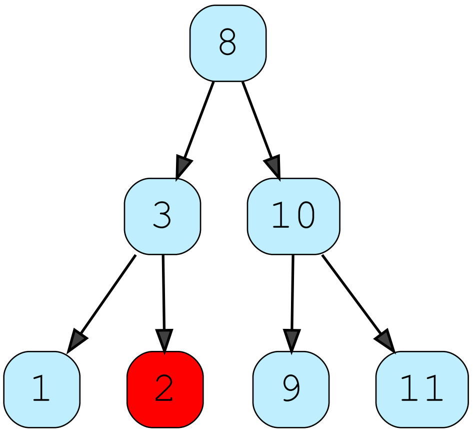
Checking BST Property



Does this tree satisfy the BST Property?

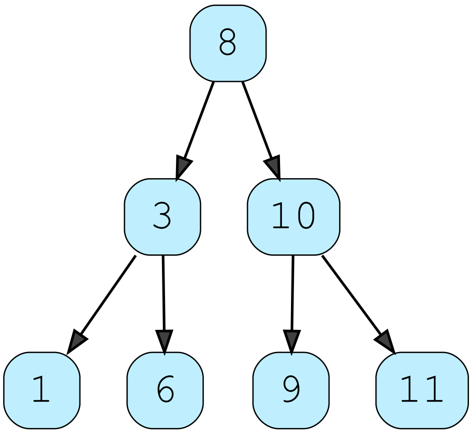
1 of 6

Checking BST Property



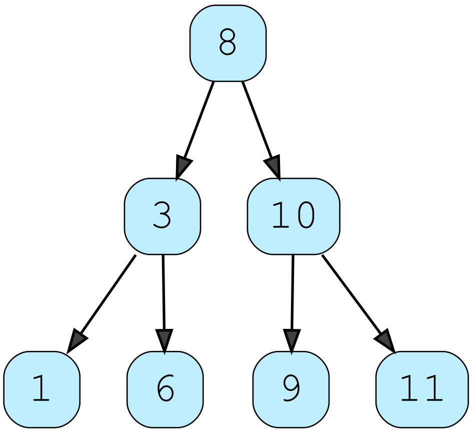
Does this tree satisfy the BST Property?
No! 2 is less than 3, so it can't be the right child of 3.

Checking BST Property



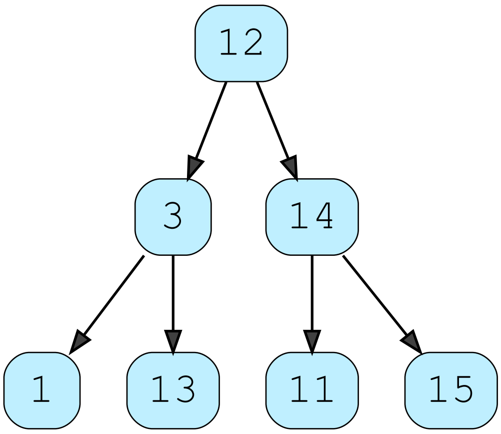
Does this tree satisfy the BST Property?

Checking BST Property

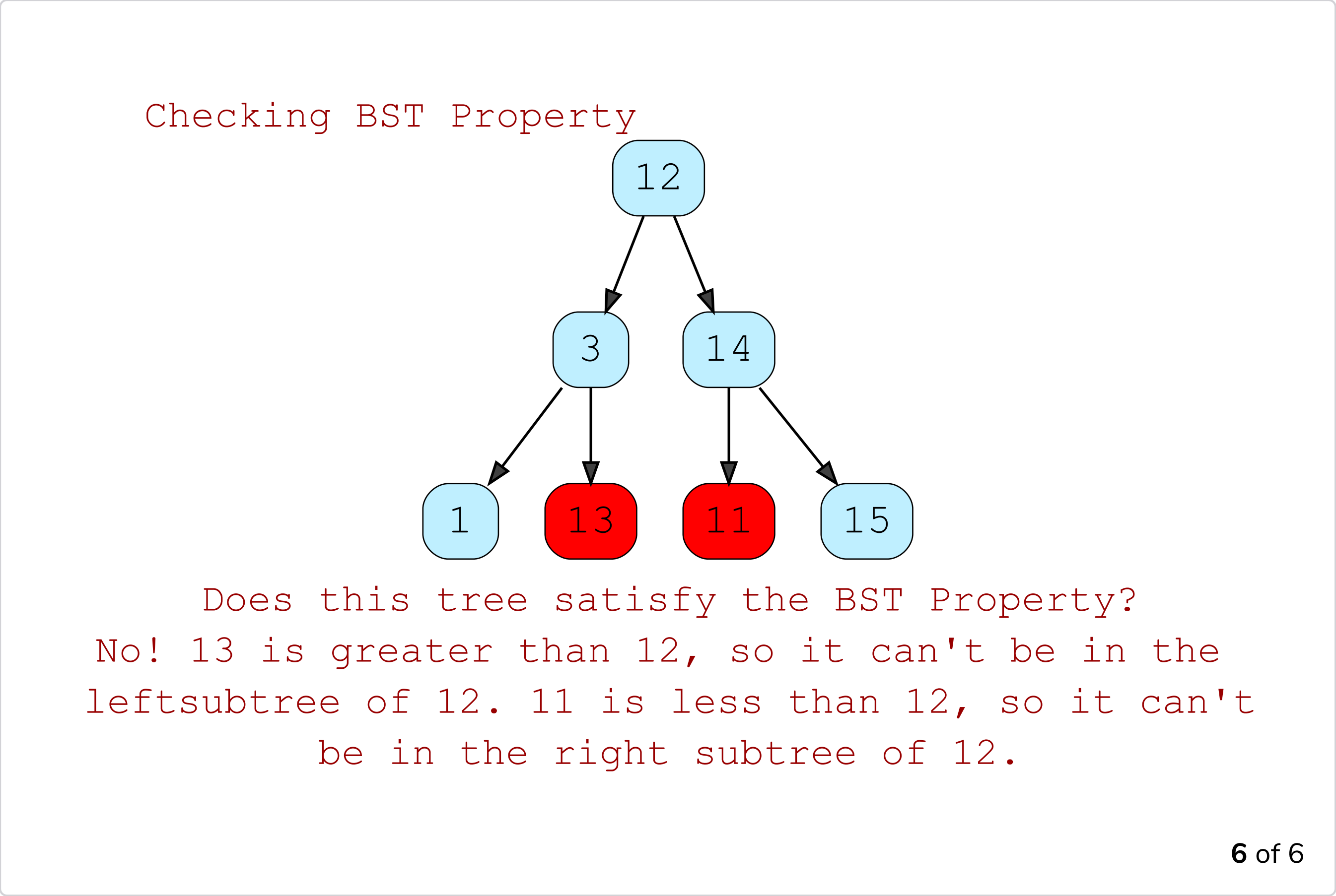


Does this tree satisfy the BST Property?
Yes!

Checking BST Property



Does this tree satisfy the BST Property?



Checking BST Property

Does this tree satisfy the BST Property?
No! 13 is greater than 12, so it can't be in the leftsubtree of 12. 11 is less than 12, so it can't be in the right subtree of 12.

6 of 6

Coding Time!

In the code below, `is_bst_satisfied` is a class method of the `BST` class. You cannot see the rest of the code as it is hidden. As `is_bst_satisfied` is a class method, please make sure that you don't change the indentation of the code provided to you. You are required to write your solution under the method prototype and return `True` or `False` from the method.

Good luck!

```
    def is_bst_satisfied(self):
      pass
```