For the purposes of this challenge, we define a *binary search tree* to be a *binary tree* with the following ordering properties:

* The  value of every node in a node's left subtree is *less than* the data value of that node.
* The  value of every node in a node's right subtree is *greater than* the data value of that node.

Given the root node of a binary tree, can you determine if it's also a binary search tree?

Complete the function in your editor below, which has  parameter: a pointer to the root of a binary tree. It must return a *boolean* denoting whether or not the binary tree is a binary search tree. You may have to write one or more helper functions to complete this challenge.

**Note:** We do not consider a binary tree to be a binary search tree if it contains duplicate values.

**Input Format**

You are not responsible for reading any input from std

def is\_bst(node, key\_min=None, key\_max=None):

"""Checks is the subtree rooted at this node is a BST."""

if node is None:

return True # By definiton

if key\_max is not None and node.key >= key\_max:

return False

if key\_min is not None and node.key <= key\_min:

return False

return is\_bst(node.left, None, node.key) and is\_bst(node.right, node.key, None)