Write recursive contains method

first...

Read over the __init__ method for class BinaryTree:

class BinaryTree:
    
    """
    A Binary Tree, i.e. arity 2.
    """

    def __init__(self, data, left=None, right=None):
        """
        Create BinaryTree self with data and children left and right.
        """

        @param BinaryTree self: this binary tree
        @param object data: data of this node
        @param BinaryTree|None left: left child
        @param BinaryTree|None right: right child
        @rtype: None

        self.data, self.left, self.right = data, left, right

next...

Now, read the header and docstring for the function contains, and then answer the questions that follow it.

def contains(node, value):
    """
    Return whether tree rooted at node contains value.
    """

    @param BinaryTree|None node: binary tree to search for value
    @param object value: value to search for
    @rtype: bool

    >>> contains(None, 5)
    False
    >>> contains(BinaryTree(5, BinaryTree(7), BinaryTree(9)), 7)
    True

1. One of the examples in contains docstring is simple enough not to require recursion (a base case). Write an if... expression that checks for this case, and then returns the correct thing. Include an else... for when the tree is less easy to deal with.

2. Another docstring example is a typical one which can benefit from recursion. Write code that returns the correct value for this case.

Now implement the body of contains