```python
# CMPT 145: Binary trees
# CMPT 145 201801 Assignment 8 Question 2

# The original example
# It's poor because of the tricky use of return values
# This technique is NECESSARY in languages where only
# one return value is possible.
# In Python, we can do better, with tuples!
# Also the function does more work than necessary!
# If the left subtree is not complete, there is no
# value in checking the right subtree.  So don't!

def bad_complete(tnode):
    def cmplt(tnode):
        if tnode is None:
            return 0
        else:
            ldepth = cmplt(tn.get_left(tnode))
            rdepth = cmplt(tn.get_right(tnode))
            if ldepth == rdepth:
                return rdepth+1
            else:
                return -1
    result = cmplt(tnode)
    return result > 0


def complete(tnode):
    """
    Purpose:
        A tree is complete if all nodes have 2 non-empty
        subtrees, except for the nodes at maximum level,
        which are all leaf nodes.
    Pre-conditions:
        :param tnode: a treenode
    Return
        :return: True if the tree is complete, False otherwise.
    """
    def cmplt(tnode):
        """
        :return: (True, height) if tnode is complete
                 (False, None) is tnode is not complete
        """
        if tnode is None:
            return True, 0
        else:
            lflag, ldepth = cmplt(tn.get_left(tnode))
            if not lflag:
                return False, None
            rflag, rdepth = cmplt(tn.get_right(tnode))
            if not rflag:
                return False, None
            if ldepth == rdepth:
                return True, rdepth+1
            else:
                return False, None
    result, _ = cmplt(tnode)
    return result
```

This study source was downloaded by 100000845056399 from CourseHero.com on 07-27-2022 05:54:47 GMT -05:00

https://www.coursehero.com/file/54458606/a8q3py/