CSC148 Summer 2016 Quiz 07 (15 minutes)

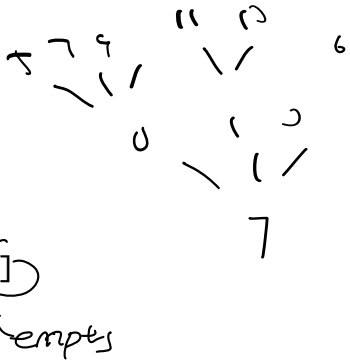
First_name ..... Last_name ..... Lab_room# .....

Read the declaration of class Tree and its __init__ method, then implement count_internal.

```
class Tree:
    """
    A bare-bones Tree ADT that identifies the root with the entire tree.
    """

    def __init__(self, value=None, children=None):
        """
        Create Tree self with content value and 0 or more children
        :param value: value contained in this tree
        :type value: object
        :param children: possibly-empty list of children
        :type children: list[Tree]
        :rtype: None
        """

        self.value = value
        # copy children if not None
        self.children = children.copy() if children else []
```

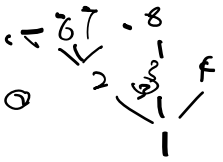


empty

Here is the header for count_internal:

```
def count_internal(t):
    """
    Return number of internal nodes of t.
    :param t: tree to count internal nodes of
    :type t: Tree
    :rtype: int
    >>> t = Tree(0)
    >>> count_internal(t)
    0
    >>> t = descendants_from_list(Tree(0), [1, 2, 3, 4, 5, 6, 7, 8], 3)
    >>> count_internal(t)
    3
    """
```

nodes with 1 / more children



no children

base case. root leaf.

return 0.

general case: not leaf.

count_internal(left) + 1 + count_internal(right)

not BST.

no left / right.

```
if len(t.children) == 0:
    return 0

else:
    return 1 + sum([count_internal(x)
                    for x in t.children])
```