Read over the declaration of class `Tree`, and the header and docstring for function `height`:

class Tree:
    """A recursive tree data structure."""
    
    # === Private Attributes ===
    # @type _root: object | None
    #    The item stored at the root of the tree, or None if the tree
    #    is empty.
    # @type _subtrees: list[Tree]
    #    A list of all subtrees of the tree.
    ...

def height(self):
    """Return the height of this tree.
    
    Remember that the height of a tree is the *longest* distance
    from the root of the tree to one of its leaves. The empty tree
    has height 0.
    
    @type self: Tree
    @rtype: int
    """
    pass

Below is a picture of a larger `Tree`, with several levels:

```
        17
       / \  
      /   \ 
     2     3
    / \   / \ 
   5   6  7  8
      \ / \   
       9
```
1. What is the height of the tree on the previous page?

2. Draw each of the subtrees of this tree, and below each one write down its height.

3. Explain, in English, how you could determine the height of the full tree given the height of each of its subtrees.

4. Implement the `height` method in the space below. You may access the `_root` and `_subtrees` attributes, and the `is_empty` Tree method. Do not use any other methods.

```python
def height(self):
    pass
```