Read over the declaration of class `Tree`, and the header and docstring for method `biggest`:

```python
class Tree:
    """A recursive tree data structure."
    """
    # === Private Attributes ===
    # The item stored at this tree’s root, or None if the tree is empty.
    _root: Optional[object]
    # The list of all subtrees of this tree.
    _subtrees: List['Tree']

    # === Representation Invariants ===
    # - If self._root is None then self._subtrees is an empty list.
    # - self._subtrees may be empty when self._root is not None.
    # - This setting of attributes represents a tree consisting of just one
    #   node.

def biggest(self) -> Union[int, None]:
    """Return a the largest item in the tree, or None if the tree is empty.
    """
    pass
```

Below is a picture of a `Tree` with several levels.
1. What value should the method return for the tree on the previous page?

2. Draw each of the subtrees of this tree, and below each one write down what the method should return.

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

4. Implement the biggest 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.

   def biggest(self):