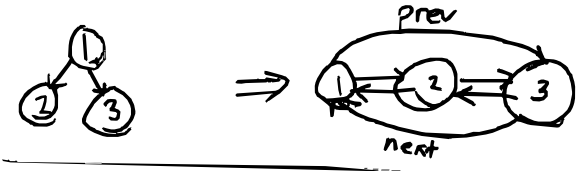
BST to DLL:

1:45pm

left → prev
right → next



1.left/prev = 2 ⟶ 1. left/prev = 3
1.right/next = 3 ⟶ 1. right/next = 2
null ⟶ 2. left/prev = 1
null ⟶ 2. right/next = 3
null ⟶ 3. left/prev = 2
null ⟶ 3. right/next = 1

Order of DLL (in-order) node + node.left + node.right

node.left → BST to DLL (node.right)
node.right → BST to DLL (node.left)
if Leaf (node):

```
def BSTtoDLL(node, first_ptr, last_ptr) -> None:
    if not node:
        return
    BSTtoDLL(node.left)
    if last_ptr.value:
        last_ptr.value.right = node
        node.right = last_ptr.value
    else:
        # no last (first elem)
        first_ptr.value = node
    last_ptr.value = node
    BSTtoDLL(node.right)
def Solution(node: Node or None) -> Opt[None]:
    if not node:
        return None
    first_ptr, last_ptr = ...
    BSTtoDLL(node, ...)
    last.right = first
    first.left = last
    return first
```