class LN:
    def __init__(self: "LN", value: object, next: "LN" = None):
        self.value = value
        self.next = next

Each linked lists is created from the class LN. Starting with each list as shown, indicate what state(s) change(s) when the statement to its left is executed. Cross out any values that are replaced and Write in new boxes or values (text or arrows).

\[
\begin{align*}
    x &= LN(7, x.next.next) \\
    x &= x.next \\
    x.next.value &= x.value \\
    x.next &= x.next.next \\
    x.next.next &= x.next \\
    x.next.next &= x
\end{align*}
\]
Assume $x$ refers to a linked list with the values shown below. What is the result of executing $x = \text{magic}(x)$ using the following code (which does something interesting with the linked list)?

```python
def magic(ll):
    answer = None
    while ll != None:
        t_m = ll
        ll = ll.next
        t_m.next = answer
        answer = t_m
    return answer
```

```python
def magic(ll):
    answer = None
    while ll != None:
        t_m = ll
        ll = ll.next
        t_m.next = answer
        answer = t_m
    return answer
```

![Diagram of linked list](image)

1. $x$
2. $ll$
3. $t_m$