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

Each linked list 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).

Hint: put a vertical stroke in the variable/attribute (box) specified on the left side of the = which will receive the reference. Put a circle on the tail of the arrow specified on the right hand side of the =. Copy the value (reference) by making the vertical-stroked box refer to the object the circle-tailed arrow refers to. I will demonstrate in class. At some level, this is no more complicated than picturing \( x = 1 \) followed by \( y = x \).

\[
\begin{align*}
x &= x.next \\
x.next.value &= x.value \\
x.next &= x.next.next \\
x.next.next &= x.next \\
x.next.next &= x \\
x &= LN(7,x.next.next)
\end{align*}
\]
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
```

```
answer

x

|   | 1 | 2 | 3 | 4 | /

ll

|   |

t_m

|   |
```