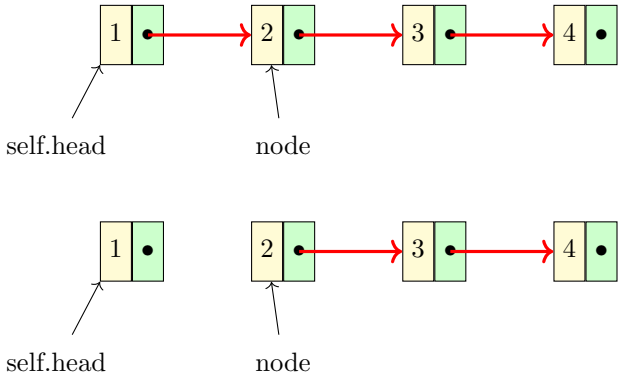
NOTES ON REVERSING A LINKED LIST

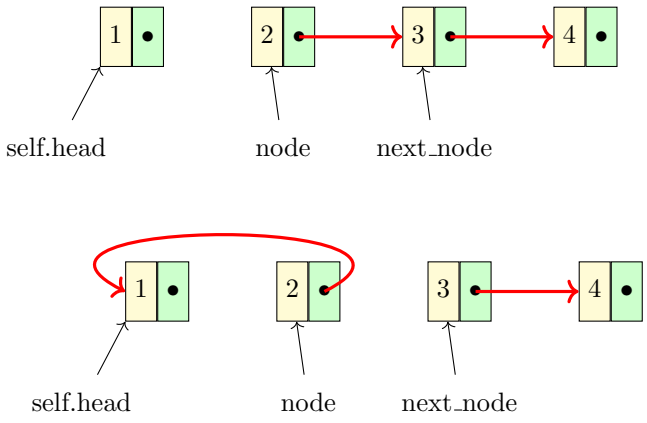
ERIC MARTIN

1. Iterative version (linear)

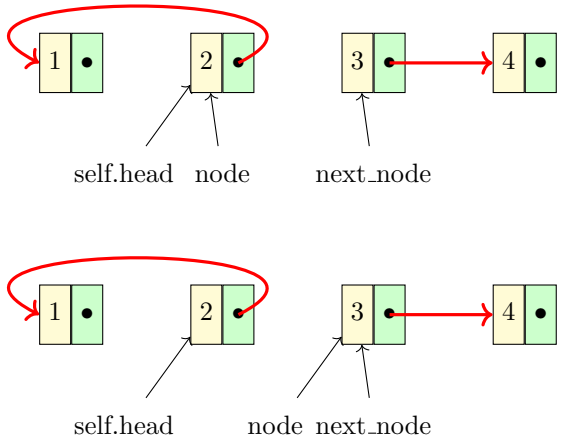
```
if not self.head:
    return
node = self.head.next_node
self.head.next_node = None
while node:
    next_node = node.next_node
    node.next_node = self.head
    self.head = node
    node = next_node
```



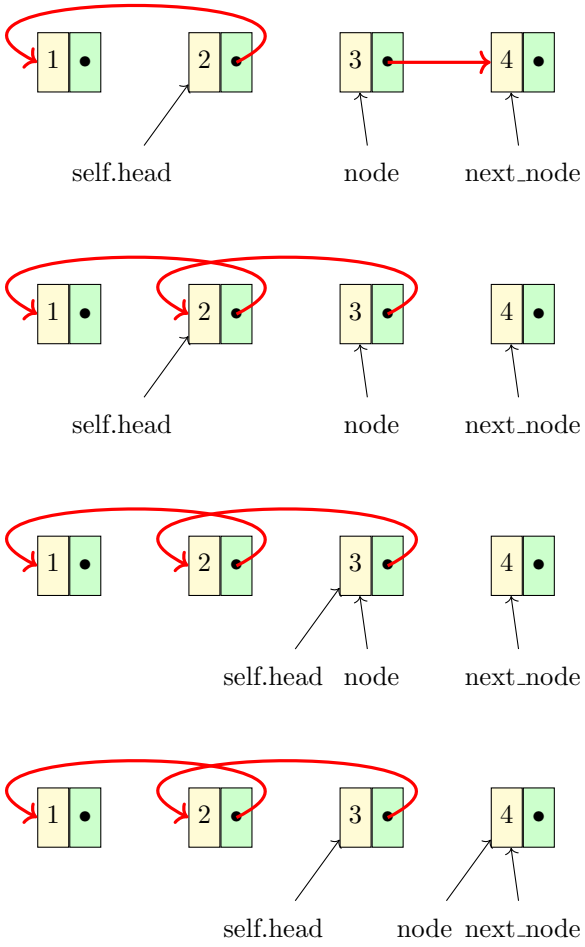
First execution of the loop



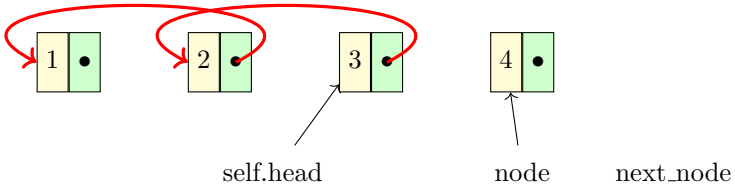
Date: Session 2, 2017.

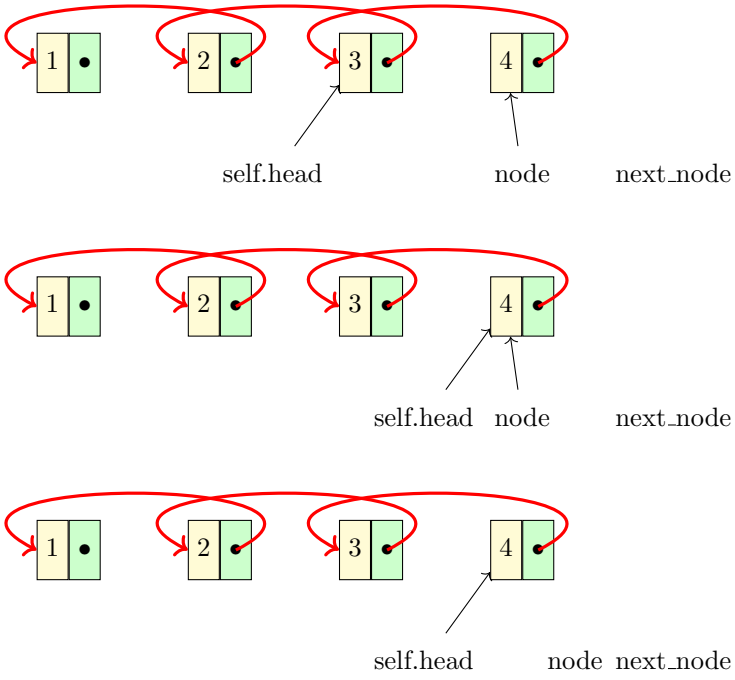


Second execution of the loop



Third execution of the loop

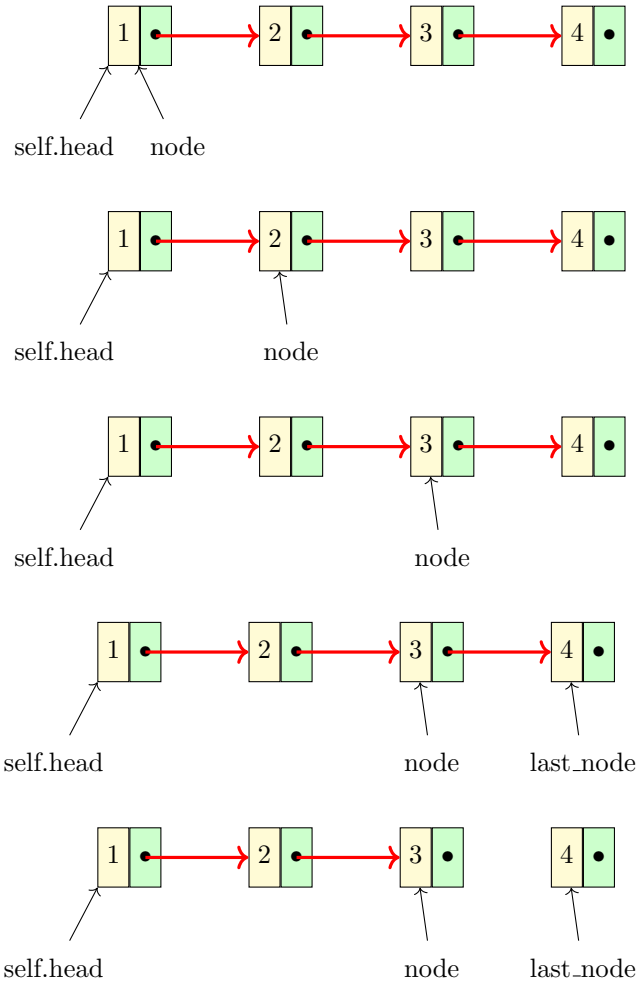



1
2
3
4
self.head
node
next_node
1
2
3
4
self.head
node
next_node
1
2
3
4
self.head
node
next_node

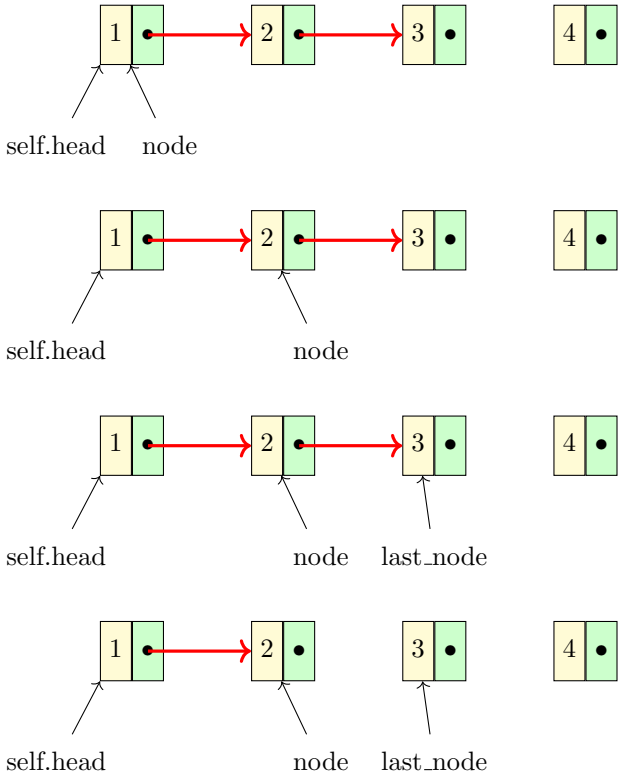
2. Recursive version (quadratic)

```
def recursive_reverse(self):
    if not self.head or not self.head.next_node:
        return
    node = self.head
    while node.next_node.next_node:
        node = node.next_node
    last_node = node.next_node
    node.next_node = None
    self.recursive_reverse()
    last_node.next_node = self.head
    self.head = last_node
```

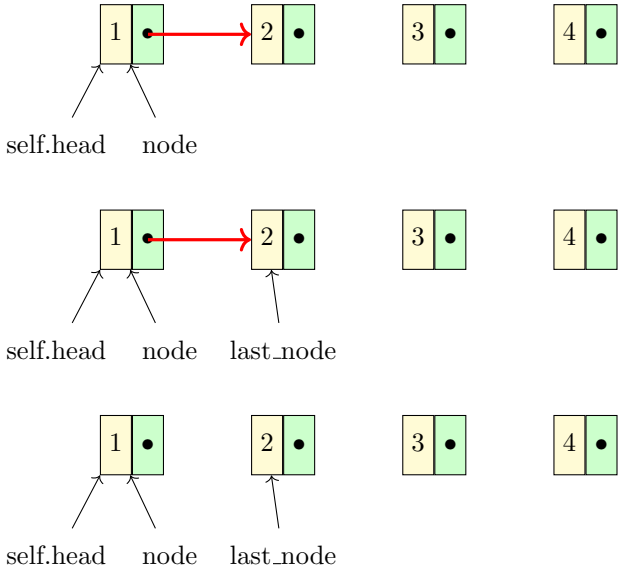
First call to `reverse()`



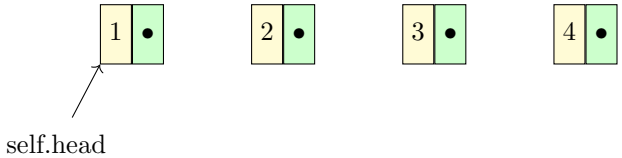
Second call to `reverse()`



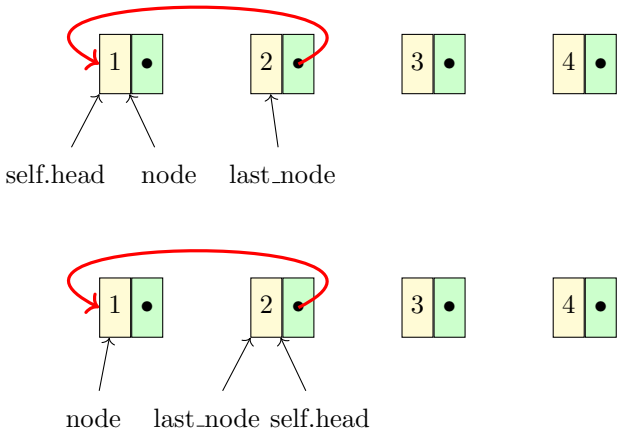
Third call to `reverse()`



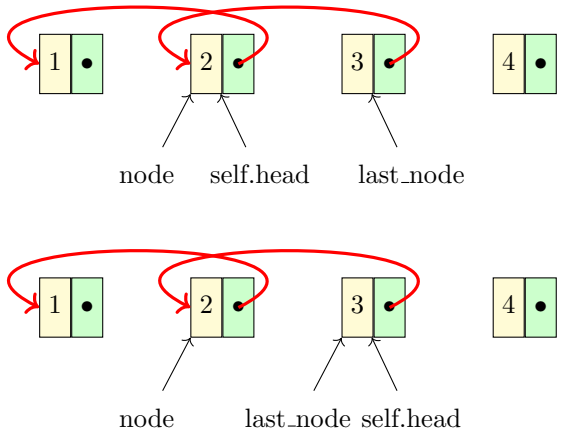
Fourth call to `reverse()`



Back to third call to `reverse()`



Back to second call to `reverse()`



Back to first call to `reverse()`