Daily Coding Problem #73

Problem

This problem was asked by Google.

Given the head of a singly linked list, reverse it in-place.

Solution

We can do this recursively and cleverly, using Python's default argument feature. Basically, we call reverse on the node's next, but not before cleaning up some pointers first:

```
def reverse(head, prev=None):
    if not head:
        return prev
    tmp = head.next
    head.next = prev
    return reverse(tmp, head)
```

This runs in O(N) time. But it also runs in O(N) space, since Python doesn't do tail-recursion elimination.

We can improve the space by doing this iteratively, and keeping track of two things: a prev pointer and a current pointer. The current pointer will iterate over through the list and the prev pointer will follow, one node behind. Then, as we move along the list, we'll fix up the current node's next to point to the previous node. Then we update prev and current.

```
def reverse(head):
    prev, current = None, head
    while current is not None:
        tmp = current.next
        current.next = prev
```

```
        prev = current
        current = tmp
    return prev
```

Now this only uses constant space!