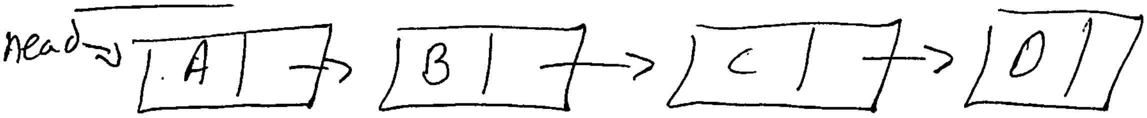
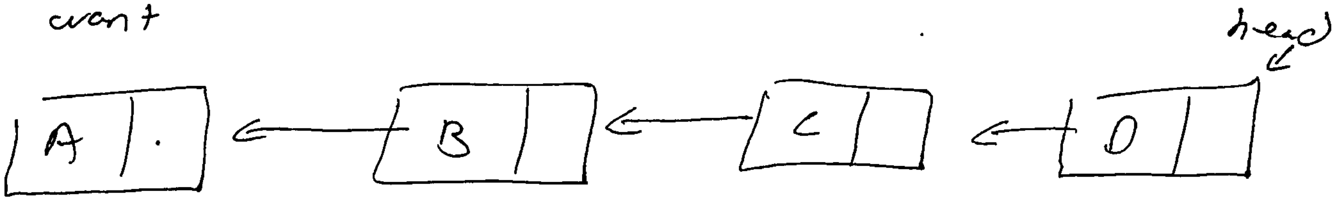
Reverse



Note this algorithm is different from the lab.

we want



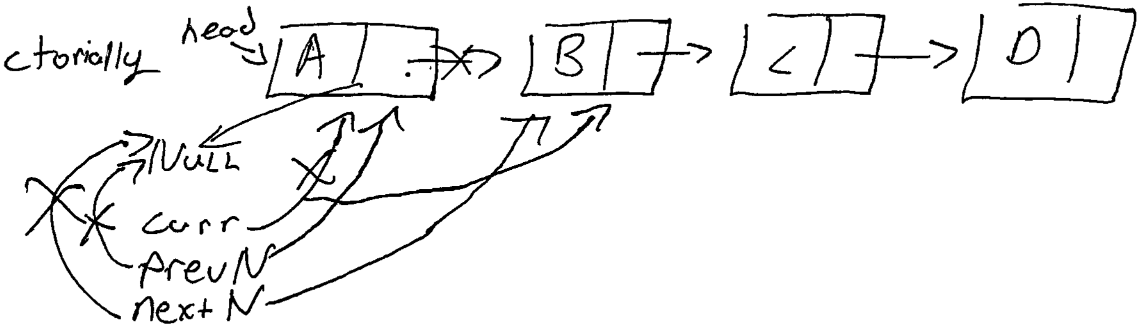
To do this, we use three temporary node variables with these initial values.

```
Node curr = head;
Node prevN = null;
Node nextN = null;
```

The idea is at each iteration we set

curr.next = prevN;

Pictorially



(1) First set ~~curr.next = prevN~~ nextN = curr.next

(2) Then set curr.next = prev (In first iteration this changes A's next reference to null).

(3) Then → prevN = curr

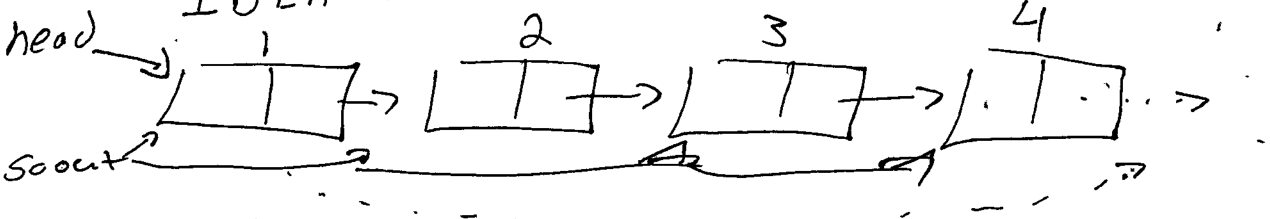
(4) Finally curr = nextN

Steps 1-4 are done in a loop. I'll leave you guys to determine when the loop should terminate.

At the end (after loop) set head = prevN;

Cycle

IDEA: Move firstNode to end of chain

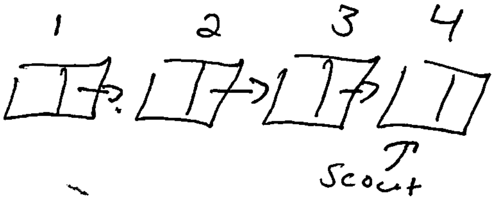


To do this we must traverse to the last node.
Use a scout node

Node ~~Sco~~ Scout = head;
do $\underbrace{\text{scout = scout.next until scout is null.}}_{\text{do this in a loop}}$

there is a case where we do not want to do scout = scout.next. I'll let you guys figure it out.

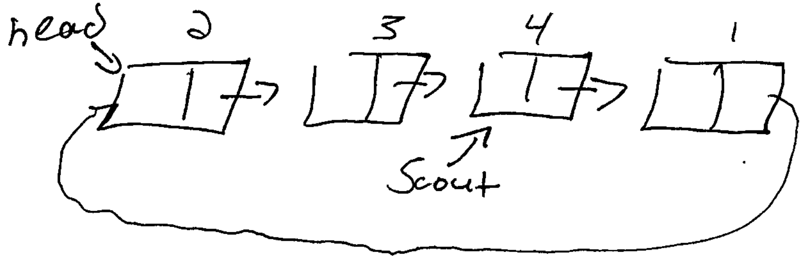
Once scout is at the last node



(1) Simply say scout.next = head;

(2) Must remove the first node now. I'll leave that to you as well. Hint: must modify head.

Caution when we did (1) head's next reference pointed to the second node. So do we have a loop in our chain? In picture form,



You must erase this loop.

Do so using the scout node.