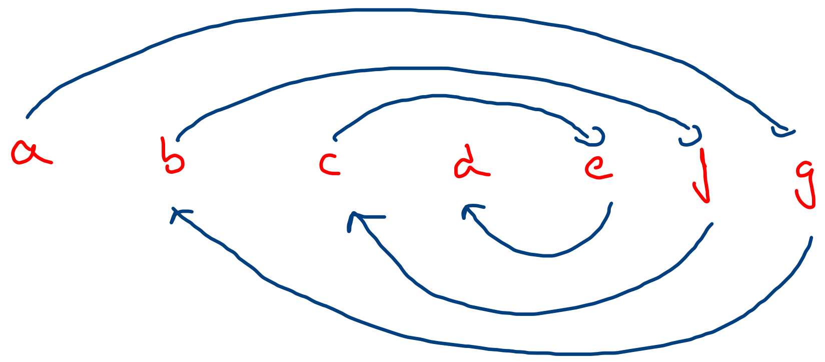
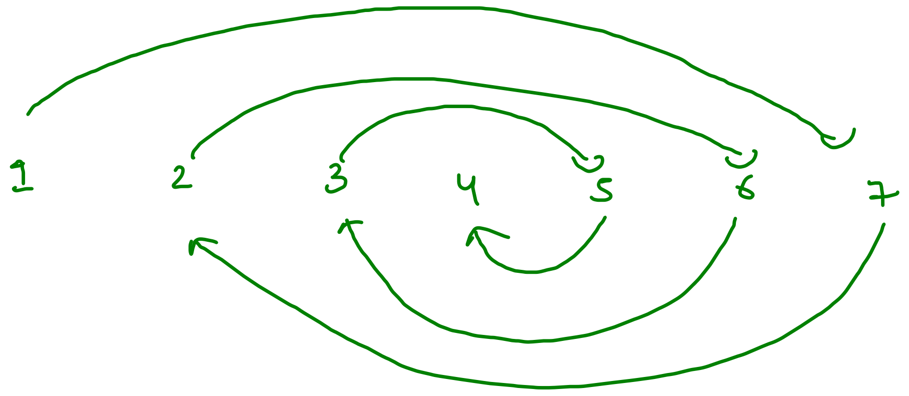
Fold A Linked List

oll $\quad a \longrightarrow b \longrightarrow c \longrightarrow d \longrightarrow e \longrightarrow f$

fll $\quad a \to f \to b \to e \to c \to d$

head $\qquad\qquad\qquad\qquad$ tail





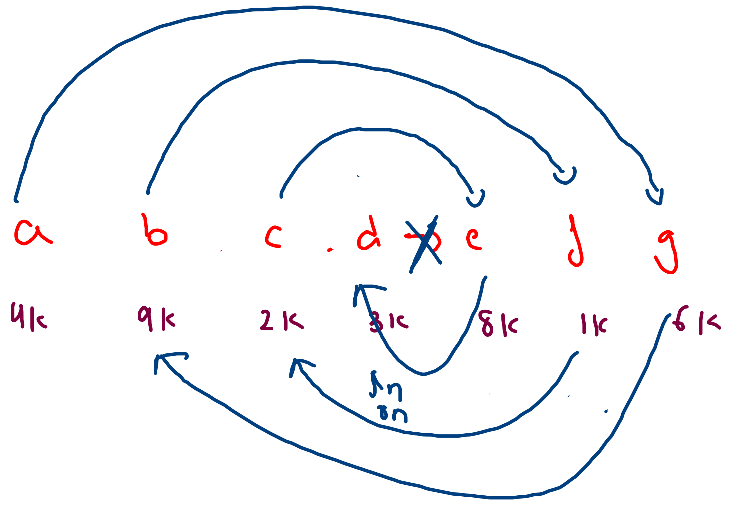
$a \to b \to c \to d \quad \underbrace{e \to f \to g}$

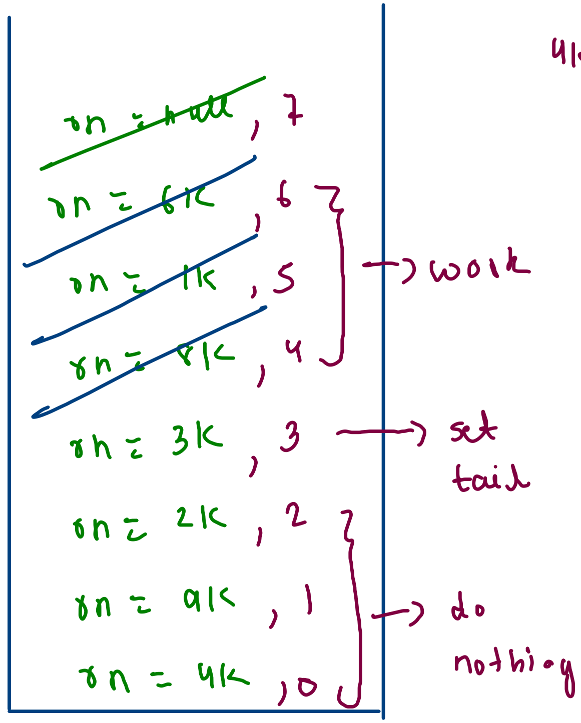
$\downarrow$

rev

$\underline{g} \to f \to e$

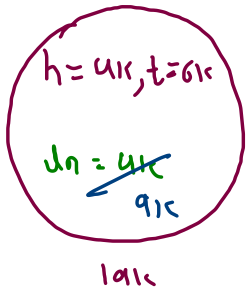
$a \to g \to b \to f \to c \to e \to d$

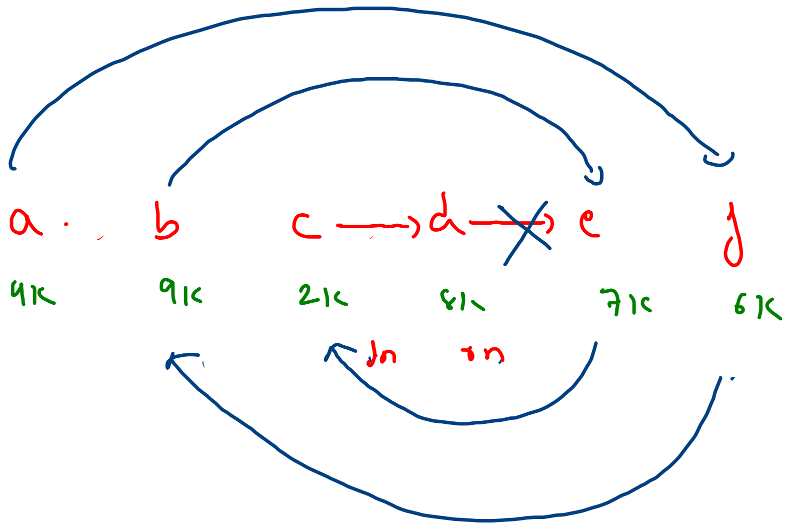


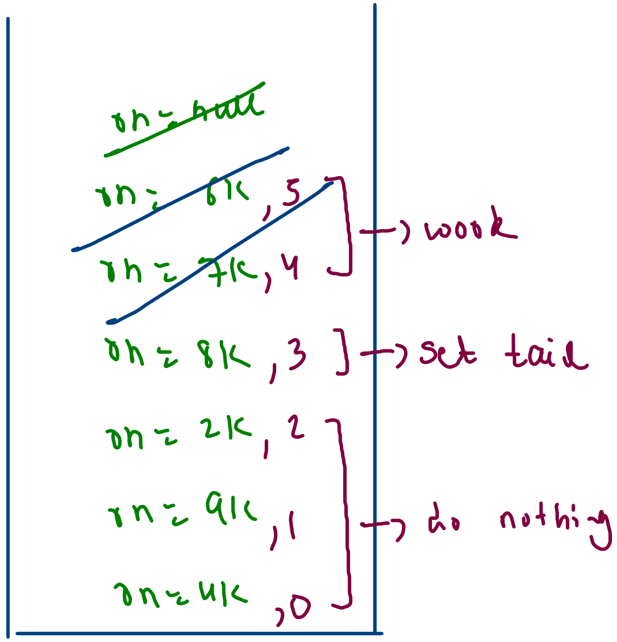


```
Node temp = dn.next;
dn.next = rn;
rn.next = temp;

dn = temp;
```

(i) single parse
(recursion)



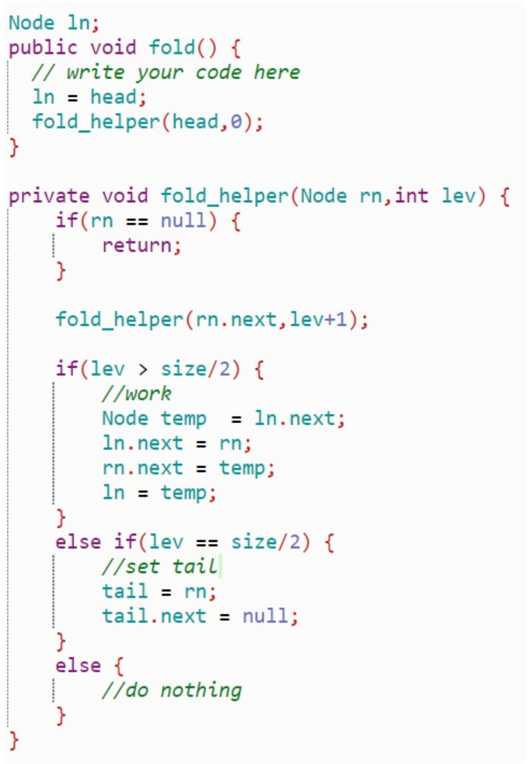




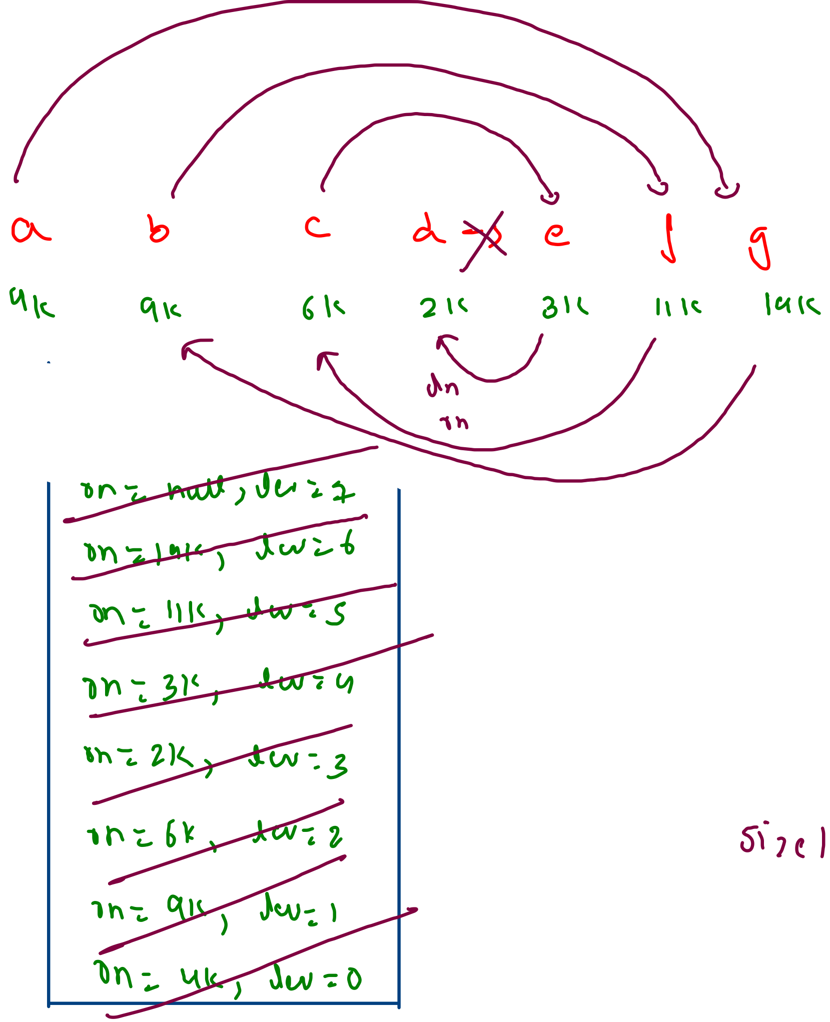
```
temp = ln.next;

ln.next = rn;

rn.next = temp;

ln = temp;
```

ln = ~~4k~~ ~~9k~~

2k



```java
Node ln;
public void fold() {
  // write your code here
  ln = head;
  fold_helper(head,0);
}

private void fold_helper(Node rn,int lev) {
    if(rn == null) {
        return;
    }

    fold_helper(rn.next,lev+1);

    if(lev > size/2) {
        //work
        Node temp  = ln.next;
        ln.next = rn;
        rn.next = temp;
        ln = temp;
    }
    else if(lev == size/2) {
        //set tail
        tail = rn;
        tail.next = null;
    }
    else {
        //do nothing
    }
}
```



h = 4k

t = ~~14k~~

2k

size/2 = 3

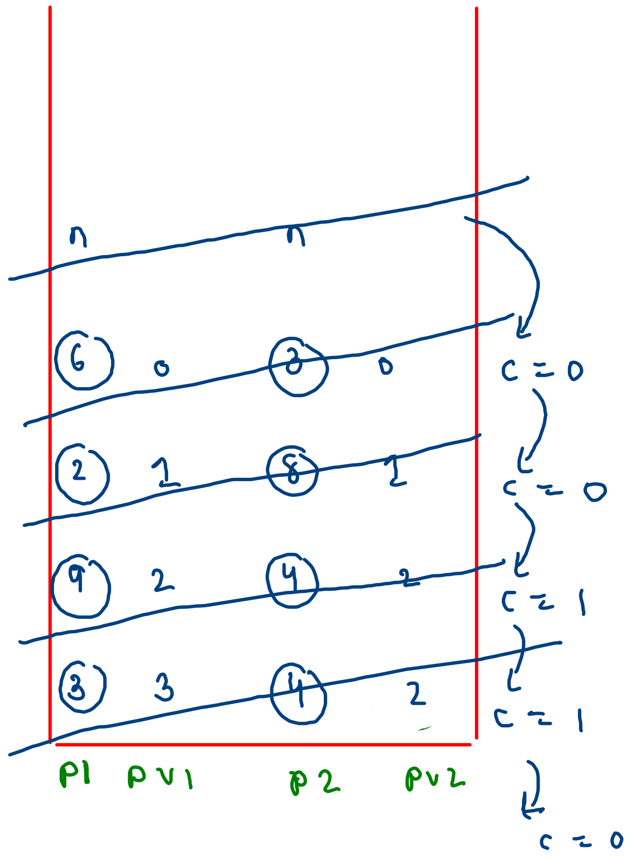
ln = ~~4k~~

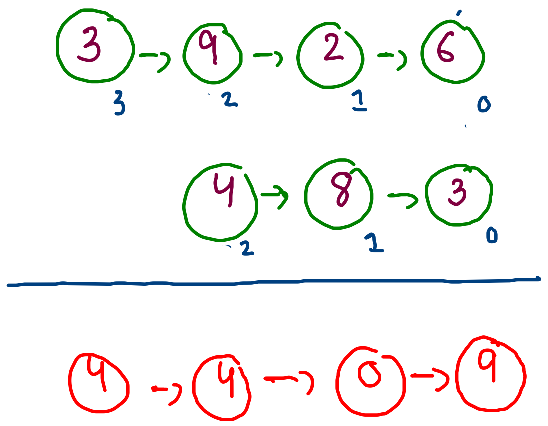
~~9k~~

~~6k~~

2k

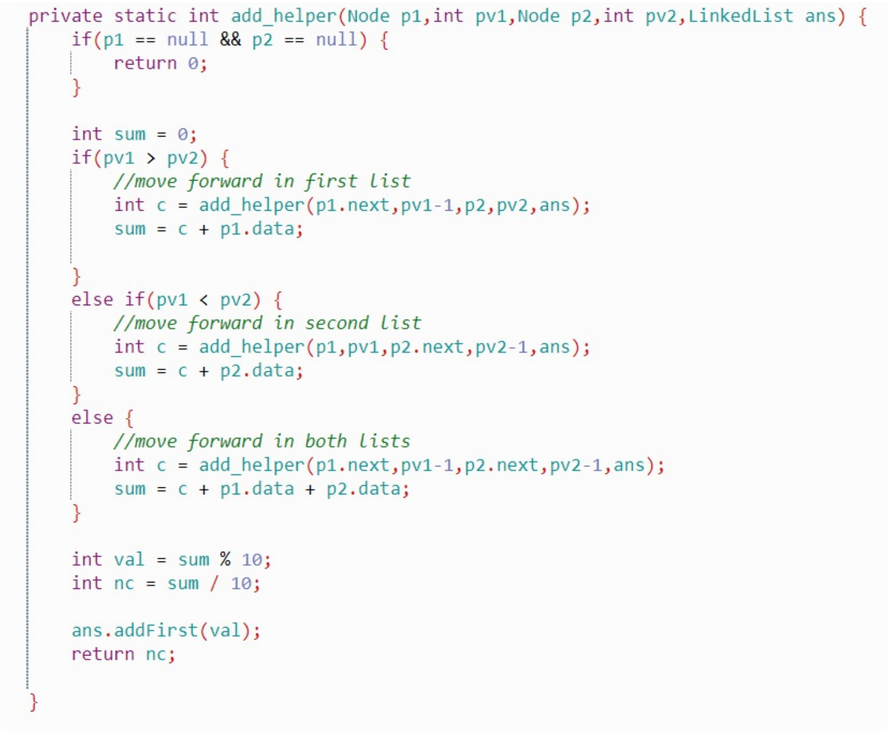
Add Two Linked Lists



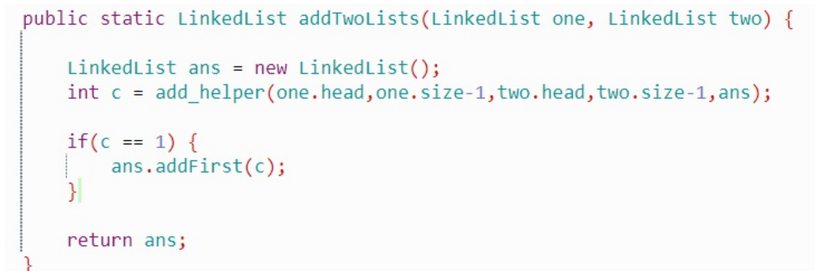


(i) single parse
└ O(n)
└ only one traversal

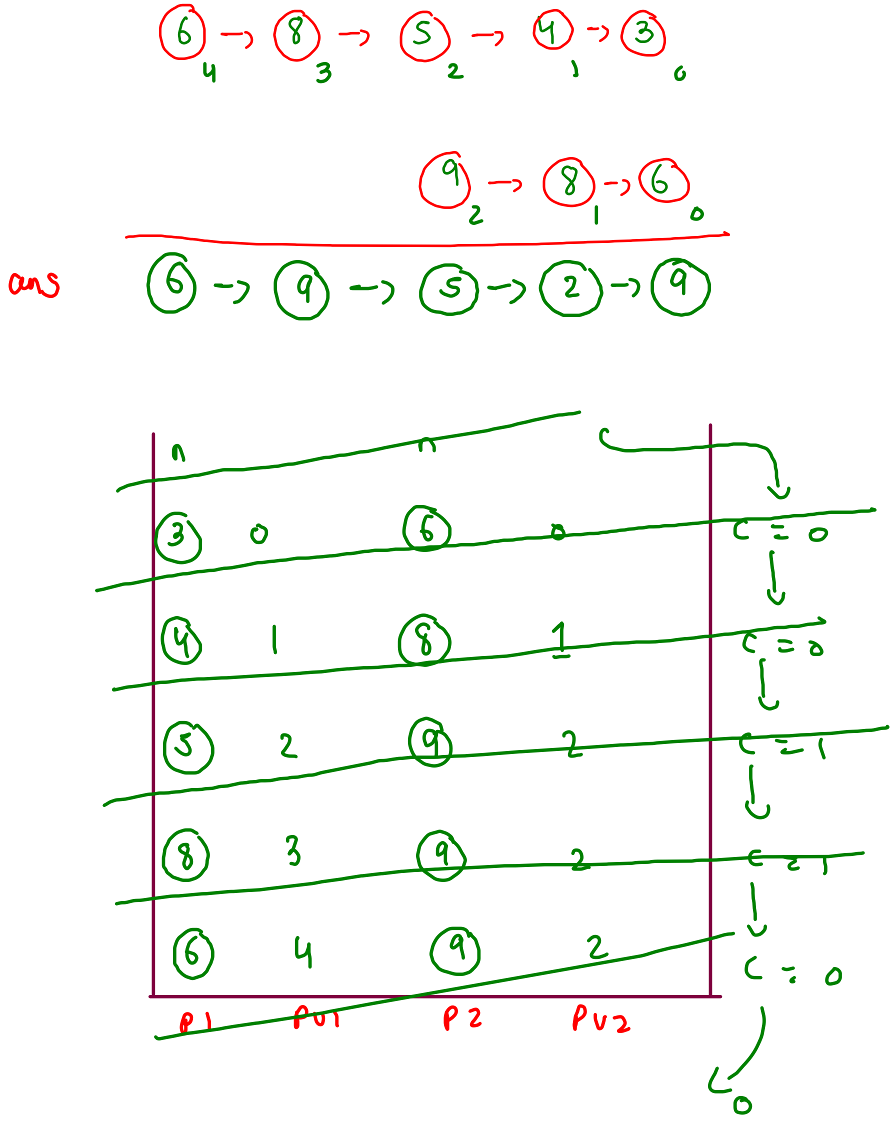
return type → int



```java
private static int add_helper(Node p1,int pv1,Node p2,int pv2,LinkedList ans) {
    if(p1 == null && p2 == null) {
        return 0;
    }

    int sum = 0;
    if(pv1 > pv2) {
        //move forward in first list
        int c = add_helper(p1.next,pv1-1,p2,pv2,ans);
        sum = c + p1.data;

    }
    else if(pv1 < pv2) {
        //move forward in second list
        int c = add_helper(p1,pv1,p2.next,pv2-1,ans);
        sum = c + p2.data;
    }
    else {
        //move forward in both lists
        int c = add_helper(p1.next,pv1-1,p2.next,pv2-1,ans);
        sum = c + p1.data + p2.data;
    }

    int val = sum % 10;
    int nc = sum / 10;

    ans.addFirst(val);
    return nc;

}
```



```java
public static LinkedList addTwoLists(LinkedList one, LinkedList two) {

    LinkedList ans = new LinkedList();
    int c = add_helper(one.head,one.size-1,two.head,two.size-1,ans);

    if(c == 1) {
        ans.addFirst(c);
    }

    return ans;
}
```



6 → 8 → 5 → 4 → 3
4 3 2 1 0
9 → 8 → 6
2 1 0
ans
6 → 9 → 5 → 2 → 9
n
n
3 0 6 0
4 1 8 1
5 2 9 2
8 3 9 2
6 4 9 2
c = 0
c = 0
c = 1
c = 1
c = 0
0
p1
pv1
p2
pv2