1. Practical

2. Recursion on LL X

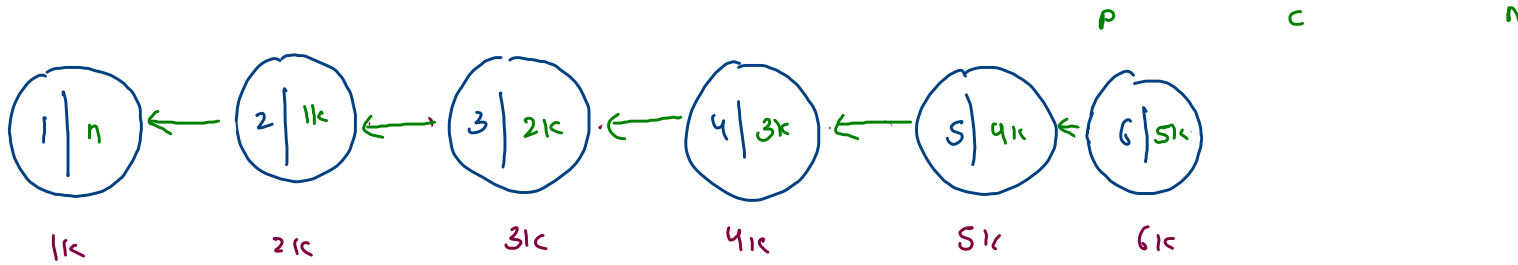
3. Exception

① Reverse of a LL

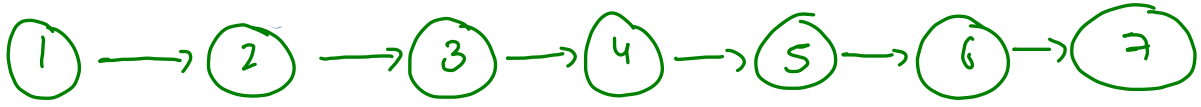
O(n) time

O(1) space

reverse - pointer - iteratively :



(2) mid - node

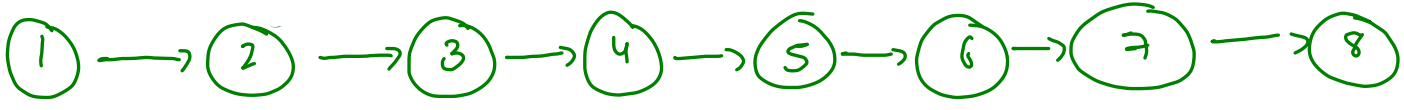


s

f

n = odd

fast.next == null



s

f

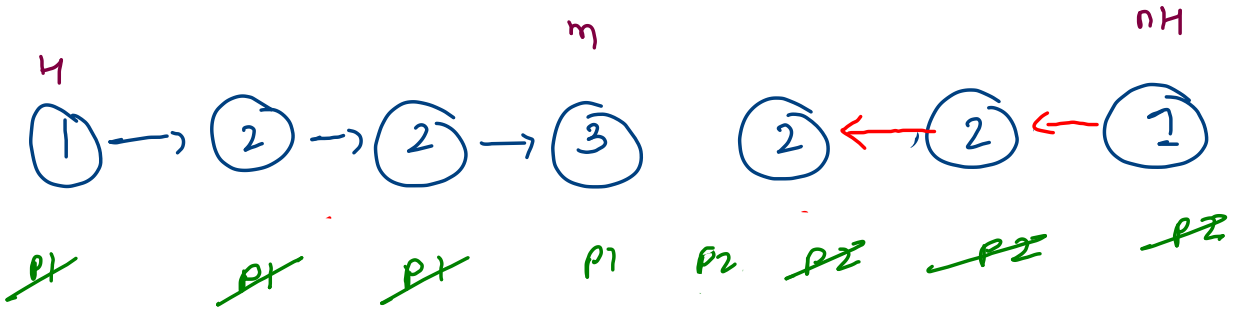
n = even

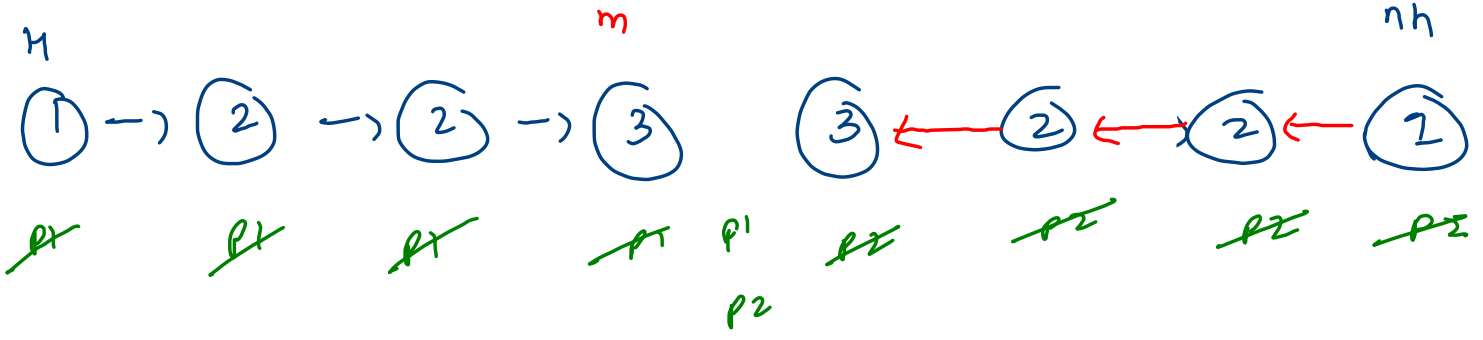
fast.next.next == null

③ Palindromic LL

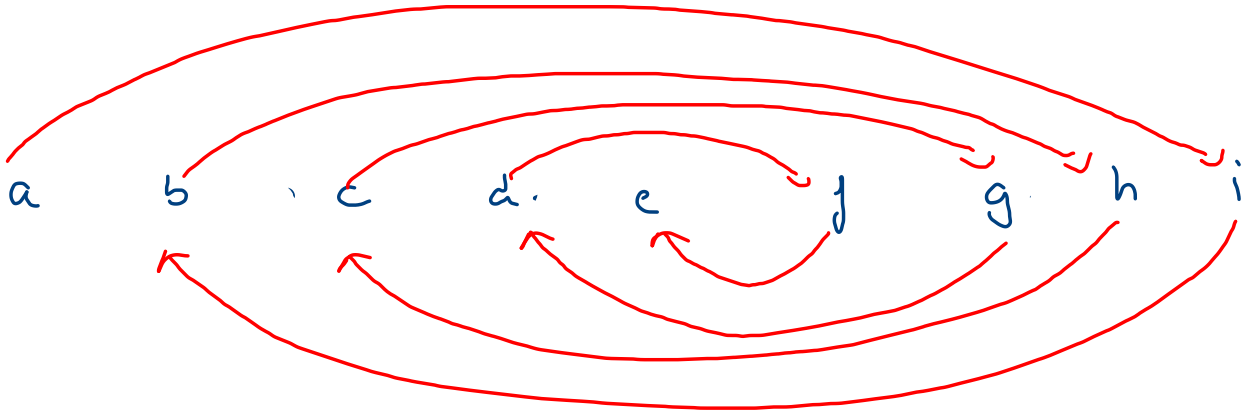
space → O(1)

time → O(n)





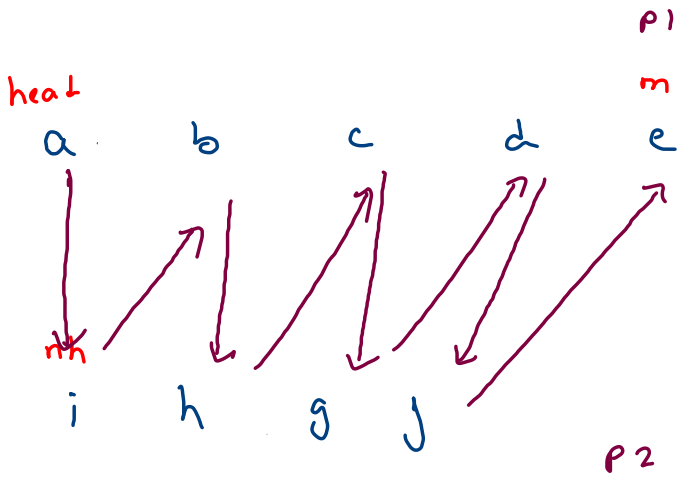
(9) Fold of a LL



Old a -> b -> c -> d -> e -> f -> g -> h -> i

Folded a -> i -> b -> h -> c -> g -> d -> f -> e

Odd



odd

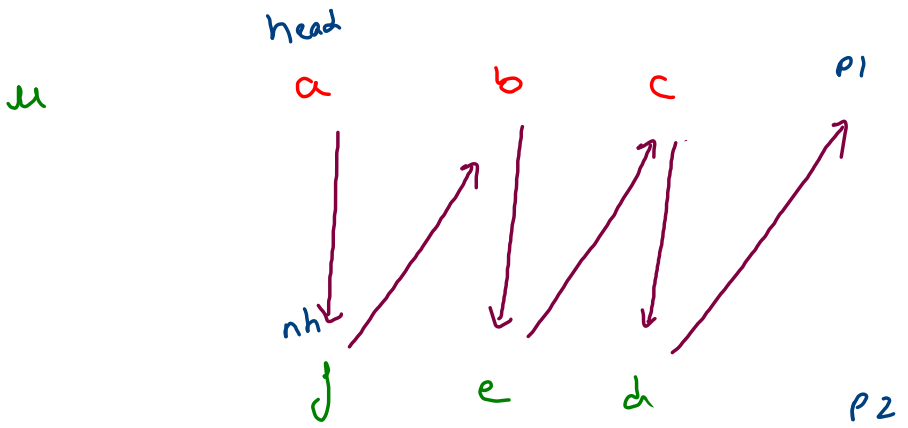
p1.next = p2

p2.next = n1

p1 = n1

p2 = n2

Fold a -> i -> b -> h -> c -> g -> d -> j -> e



p1.next = p2

p2.next = n1

p1 = n1

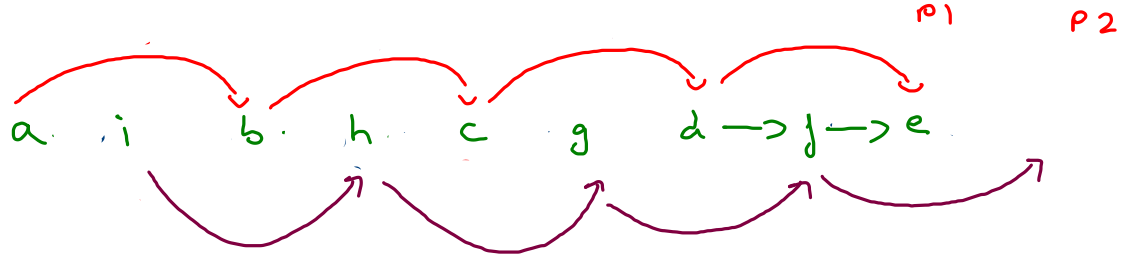
p2 = n2

old a -> j -> b -> e -> c -> d

unfold

```
ListNode h1 = head;
ListNode h2 = head.next;

ListNode p1 = h1;
ListNode p2 = h2;
```



p2 != null && p2.next != null

```
while(p1 != null && p2 != null) {
    //preserve
    ListNode n1 = p1.next;
    ListNode n2 = p2.next;

    //links
    p1.next = n2;
    p2.next = n2.next;

    //move
    p1 = p1.next;
    p2 = p2.next;
}

h2 = reverseOfLL(h2);
p1.next = h2;
```

h1 = a -> b -> c -> d -> e

h2 = f -> g -> h -> i

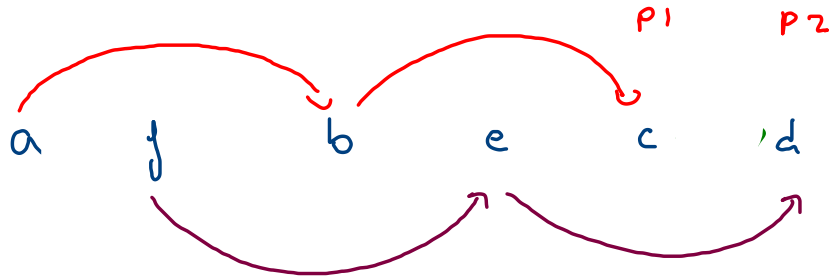
p1.next = null

p1.next = n2

p2.next = n2.next

p1 = p1.next

p2 = p2.next



p2 != null && p2.next != null

```
while(p1 != null && p2 != null) {
    //preserve
    ListNode n1 = p1.next;
    ListNode n2 = p2.next;

    //links
    p1.next = n2;
    p2.next = n2.next;

    //move
    p1 = p1.next;
    p2 = p2.next;
}

h2 = reverseOfLL(h2);
p1.next = h2;
```

h1 = a -> b -> c

h2 = d -> e -> f

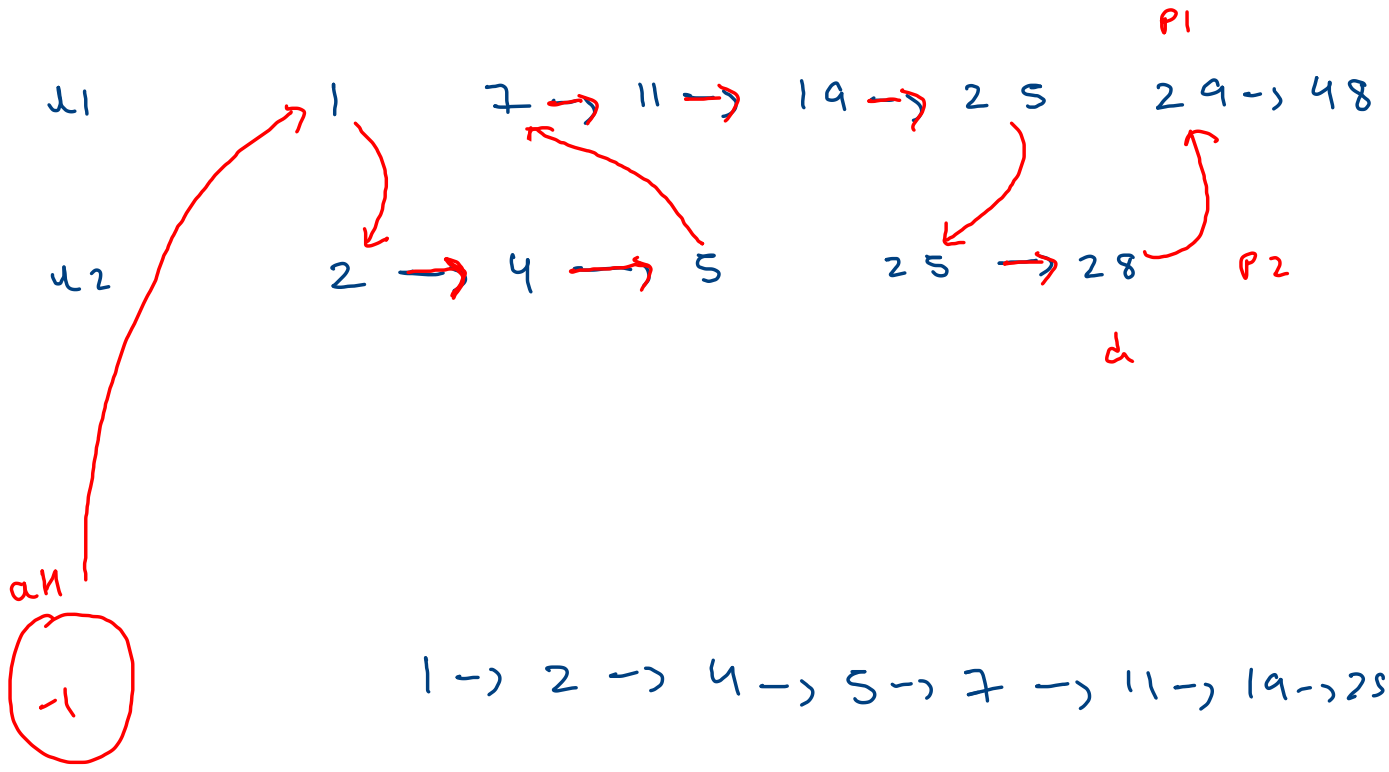
p1.next = null;

p1.next = n2

p2.next = n2.next

p1 = p1.next

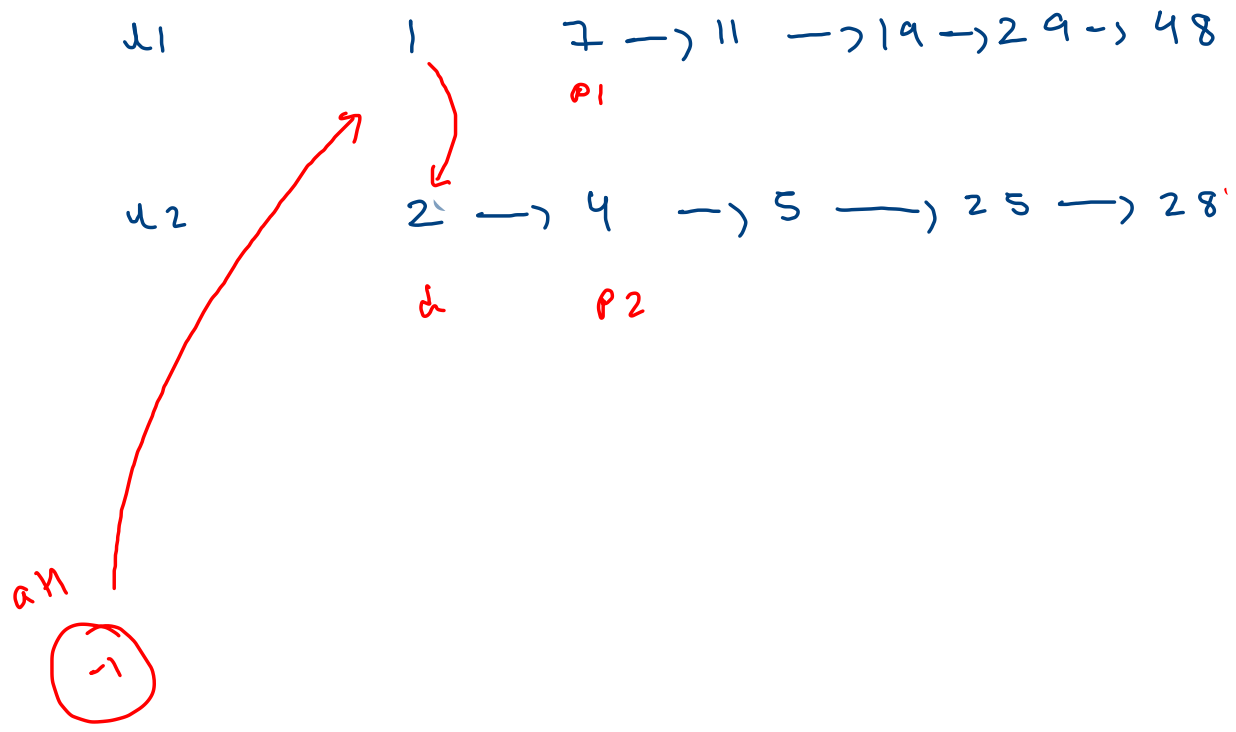
p2 = p2.next



space -> o(1)

time -> o(n+m)

1 -> 2 -> 4 -> 5 -> 7 -> 11 -> 19 -> 25 -> 25 -> 28 -> 29 -> 48


l1
1
7 -> 11 -> 19 -> 29 -> 48
p1
l2
2
4 -> 5 -> 25 -> 28
d
p2
ans
-1

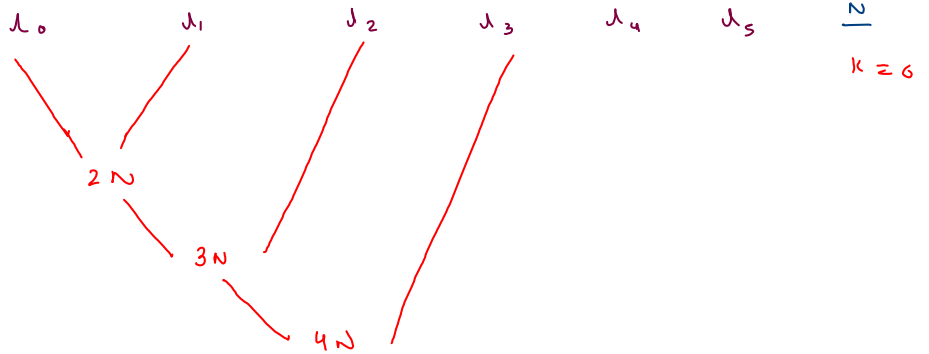
l_0 → 1 → 8 → 11 → 12

l_1 → 3 → 9 → 14

l_2 → 5 → 7 → 8 → 15

space O(1)

time → nk^2

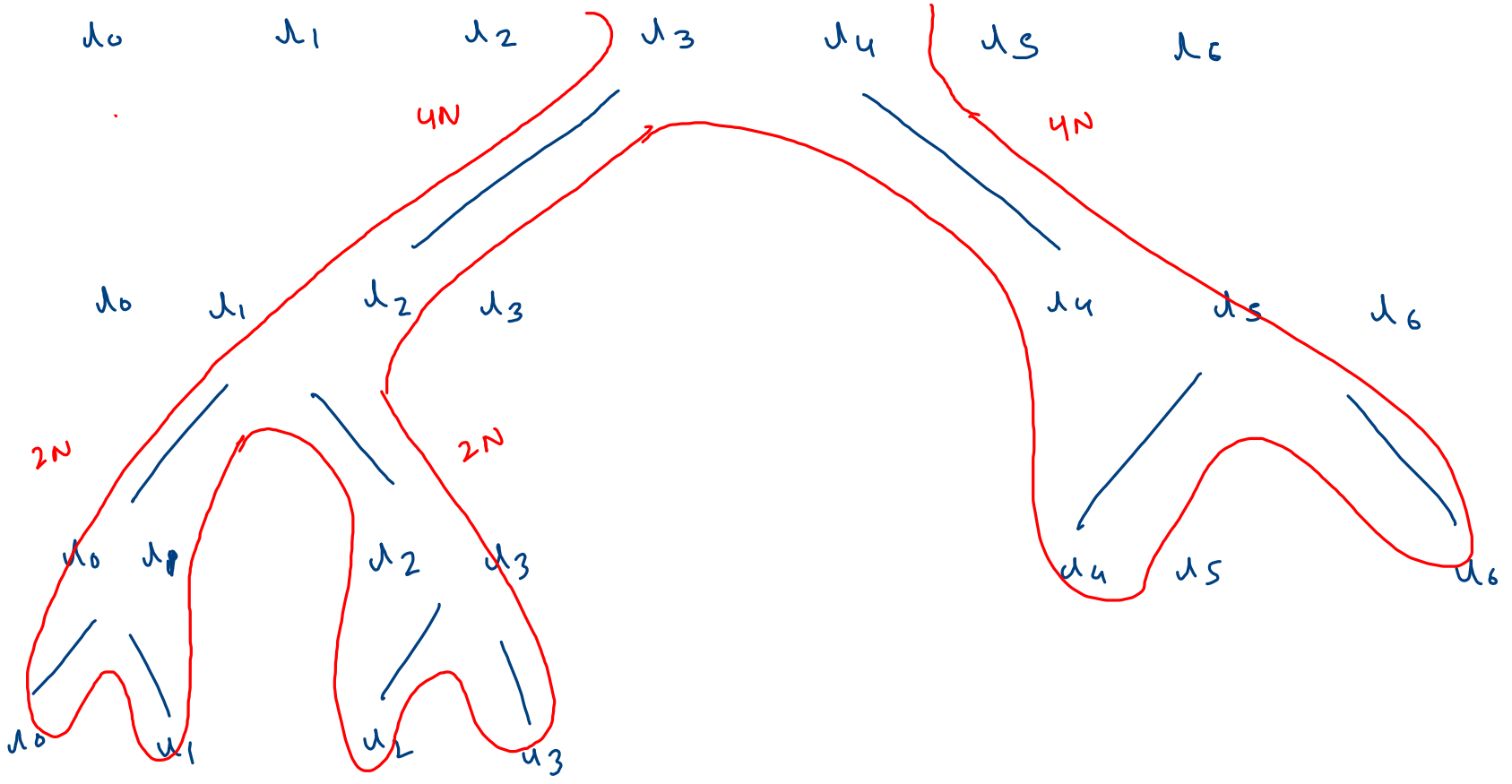


$$T(n) = 2N + 3N + 4N + \dots kN$$

$$= N(2 + 3 + 4 + 5 + 6 \dots k)$$

$$= N\left(\frac{k(k+1)}{2} - 1\right)$$

$$= Nk^2$$


d_0 d_1 d_2 d_3 d_4 d_5 d_6
4N
4N
d_0 d_1 d_2 d_3 d_4 d_5 d_6
2N
2N
d_0 d_1 d_2 d_3 d_4 d_5 d_6
d_0 d_1 d_2 d_3

```java
public static ListNode mergeKListsHelper(ListNode[]lists,int lo,int hi) {
    if(lo == hi) {
        return lists[lo];
    }

    int mid = (lo + hi)/2;

    ListNode left = mergeKListsHelper(lists,lo,mid);
    ListNode right = mergeKListsHelper(lists,mid+1,hi);

    ListNode ml = mergeTwoSortedLists(left,right);
    return ml;
}
```

k -> total lists

n -> avg. of nodes / LL

$T(k) = 2T\left(\frac{k}{2}\right) + nk$

$2T\left(\frac{k}{2}\right) = 4T\left(\frac{k}{4}\right) + \frac{nk}{2} \times 2$

$4T\left(\frac{k}{4}\right) = 8T\left(\frac{k}{8}\right) + \frac{nk}{4} \times 4$

$\vdots$

$T(1) = 0 + \frac{nk}{2^{p-1}} \times 2^{p-1}$

$+$

$T(k) = nkp$

$p = \log_2 k$

$= nk \log_2 k$

$k, \frac{k}{2}, \frac{k}{4} \ldots 1$

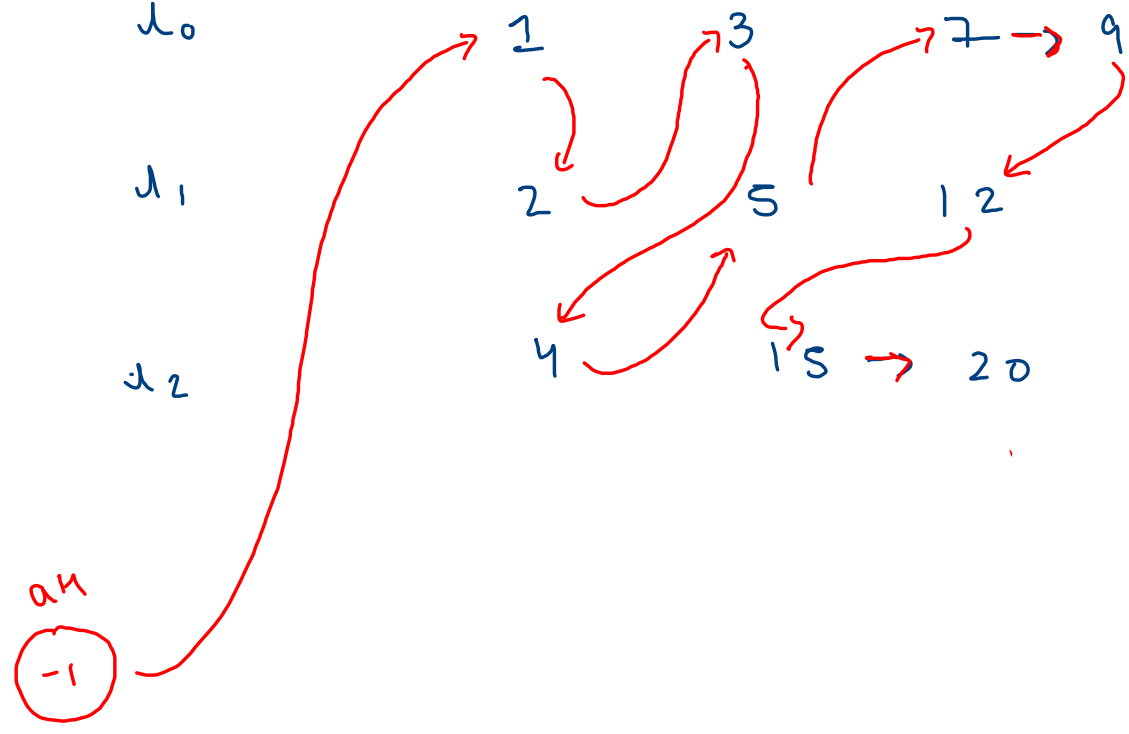
$a = k, \quad r = \frac{1}{2}$

$1 = ar^{p-1}$

$1 = k\left(\frac{1}{2}\right)^{p-1}$

$2^{p-1} = k$

$p-1 = \log_2 k$


d_0
d_1
d_2
1
3
7
9
2
5
12
4
15
20
ah
-1

```java
for(int i=0; i < lists.length;i++) {
    if(lists[i] != null) {
        pq.add(lists[i]);
    }
}

ListNode dm = new ListNode(-1); // dummy node
ListNode ansH = dm;

while(pq.size() > 0) {
    ListNode node = pq.remove();

    dm.next = node;
    dm = dm.next;

    if(node.next != null) {
        pq.add(node.next);
    }

}
```

klogk

space -> O(k)

time -> nk * logk