7th August Morning

1. Multiplication → Addition
2. Merge k sorted Divide and conq.
3. Reverse of linked list using addFirst
4. Remove duplicates from sorted list → Add Last / Normal
5. Remove all duplicates
6. Clone a linkedlist with Random pointer
   ↳ O(1) → without using space

7th August Evening

1. Mathematical proof of cyclic linkedlist
2. Clone a linkedlist using Hash Map
3. Segregate odd-Even
4. segregate 01 → swap data / Nodes arrangement
5. segregate 012 → swap data / Nodes arrangement

8th August (Morning)

1. Segregate node of Linkedlist over last index
2. Segregate Node of Linked list over pivot index
3. quicksort in linkedlist

8th August (Evening) Doubly linkedlist

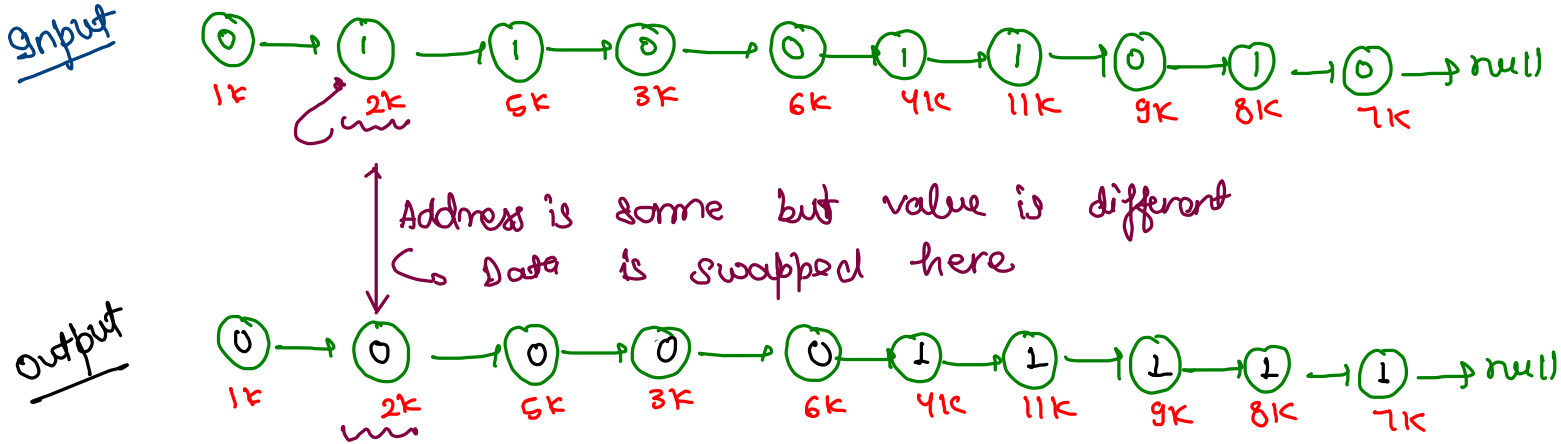
1. Get First + Get Last + Get At
2. Add At
3. Remove At
4. Add Before + Add After
5. Remove Before + Remove After
6. Remove Node in DLL.

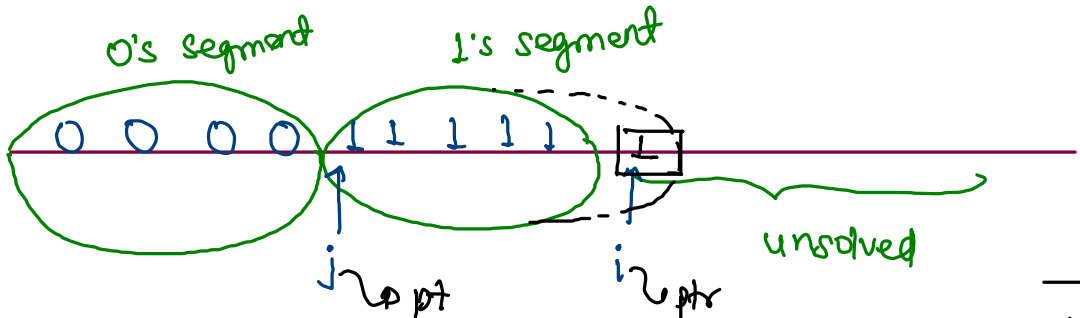
→ Creation of doubly linked list

10th August

1. Display Forward and Backward
2. LRU cache → Application of DLL
3. Miscellaneous

Segregate 0 1 (Data Swapping) :



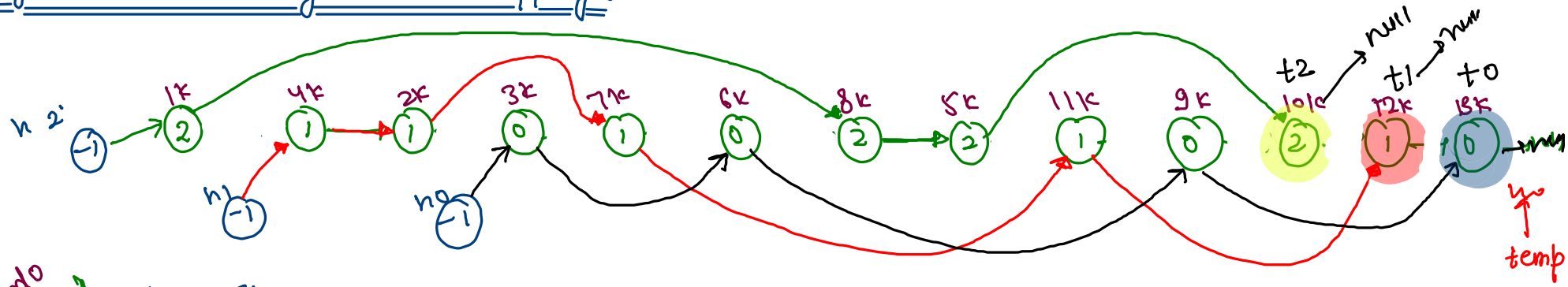


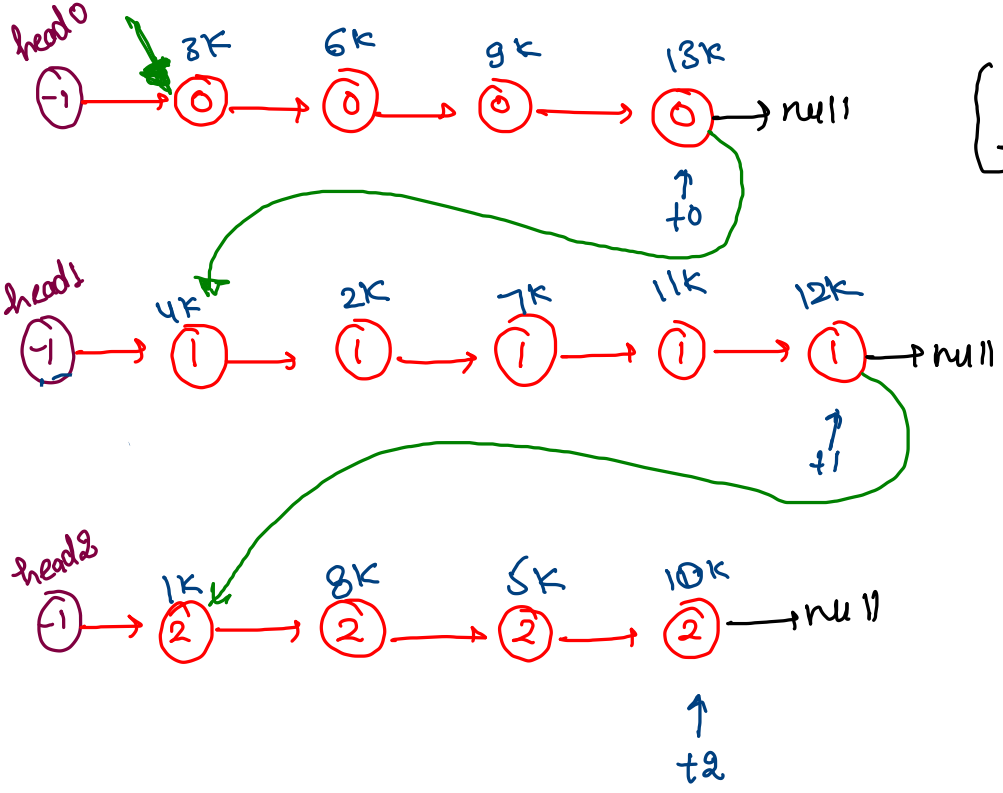
j → First one

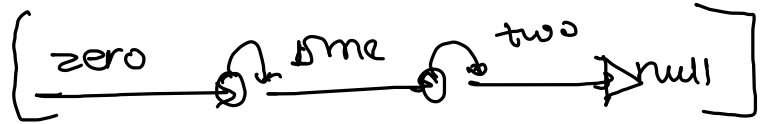
i → first unsolved

i.val == 0	i.val == 1
swap(i,j) i = i.next j = j.next	i = i.next;

Segregate 0 1 2 using Node Swapping :








```
t0.next = null;
t1.next = null;
t2.next = null;
```

```
t0.next = head1.next;
t1.next = head2.next;
```

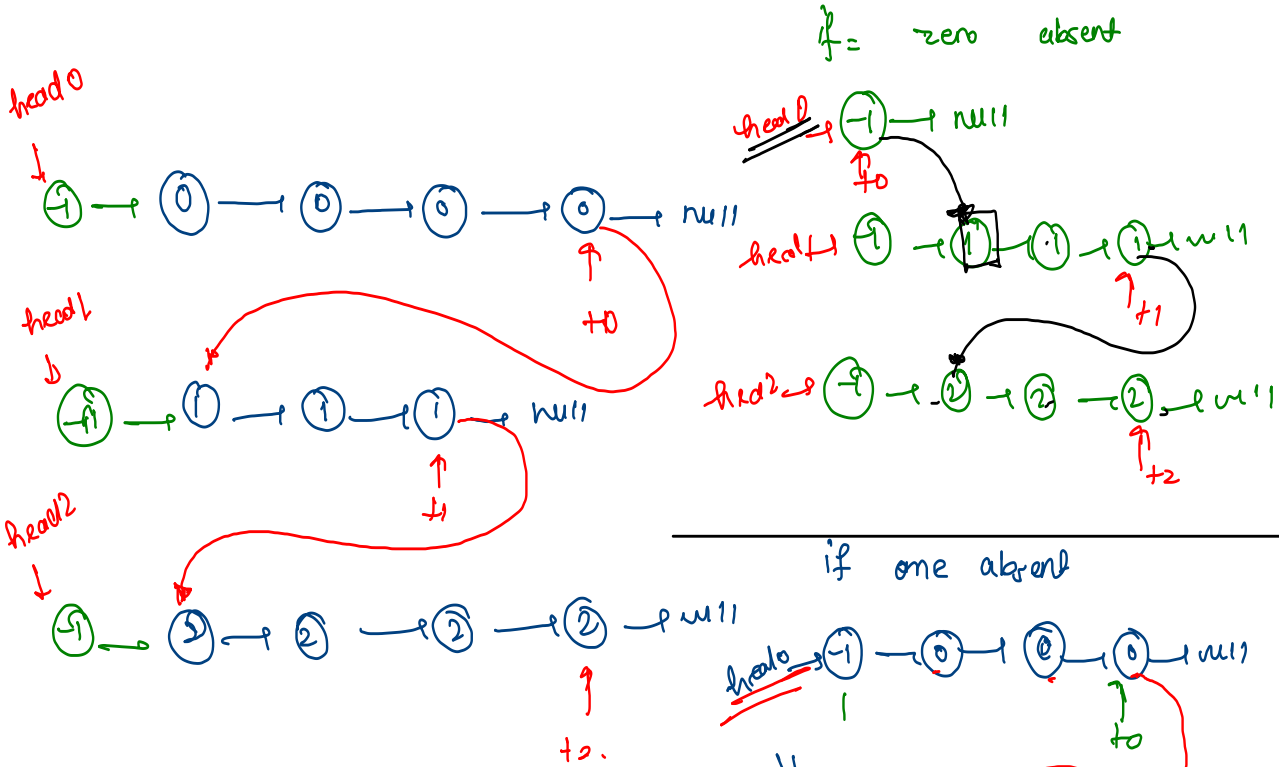
```
return head0.next;
```

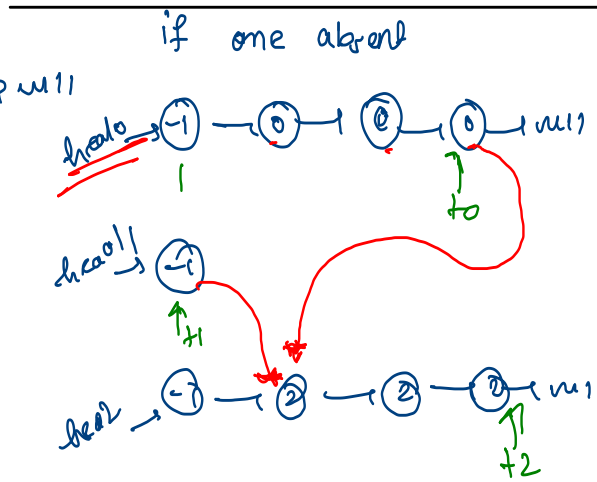
Edge case

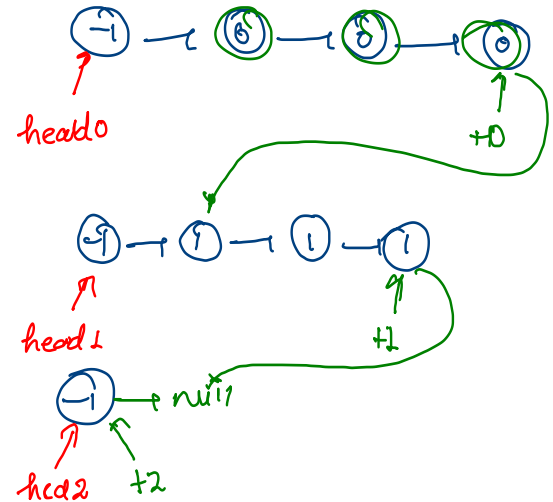
If 0 is absent

If 1 is not present

If 2 is not present



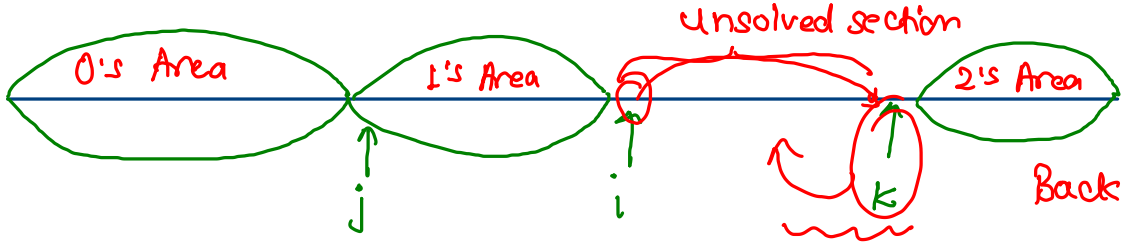




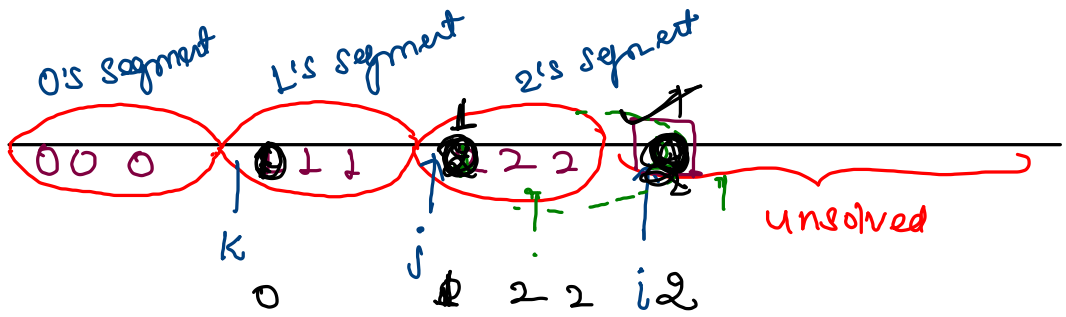
Dependencies

highest >>> lowest

Segregate 0 1 2 using data swapping :



Back traversal is difficult in linked list

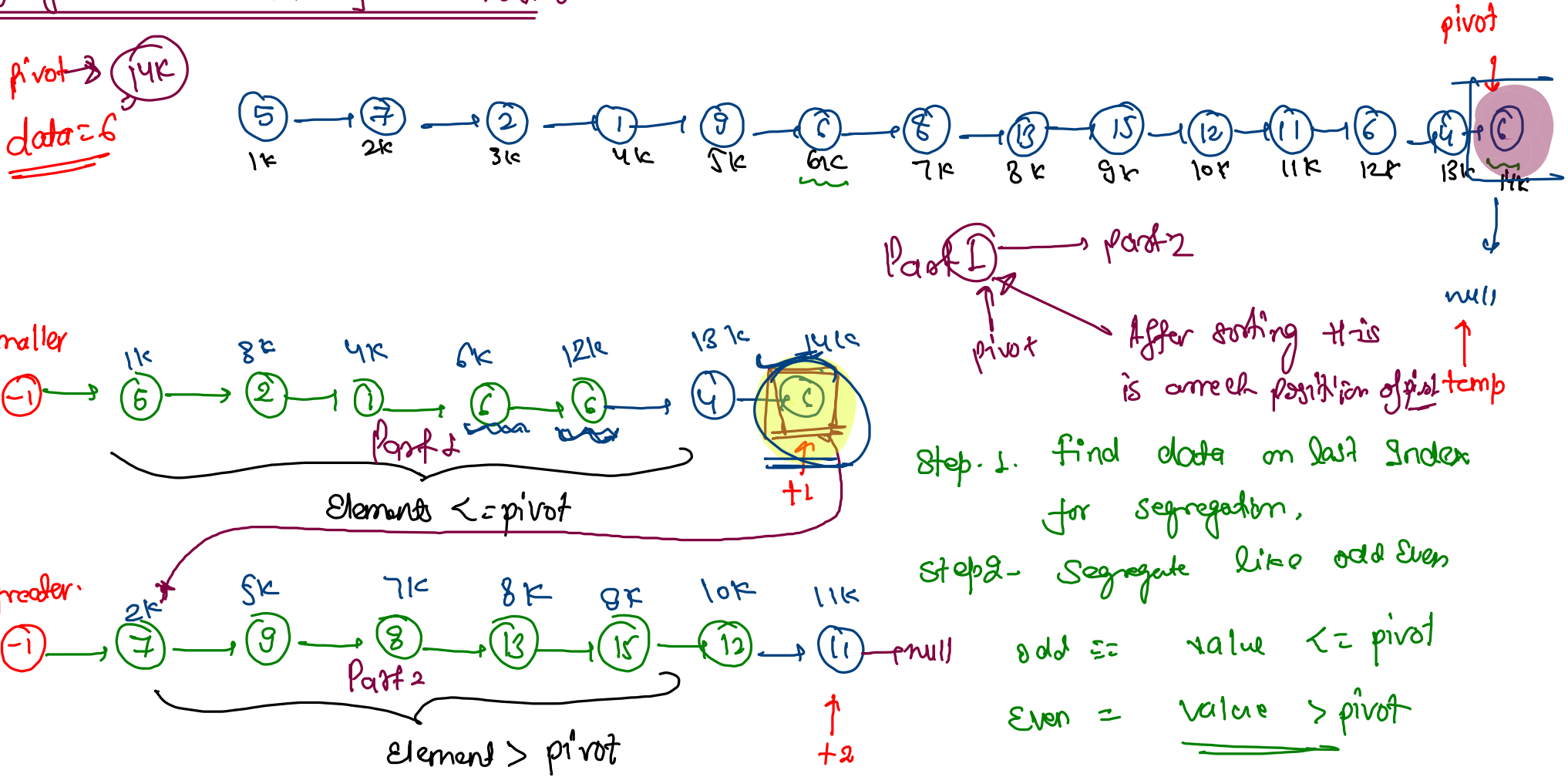


k → first 1
j → first 2.
i → first unsolved

i.val == 2	i.val == 1	i.val == 0

i = i.next	swap(i, j);
	i = i.next;
	j = j.next
		swap(i, j)
		swap(j, k)
		i = i.next;
		j = j.next;
		k = k.next;

Segregate on the basis of last index :



Step. 1. find data on last index for segregation,

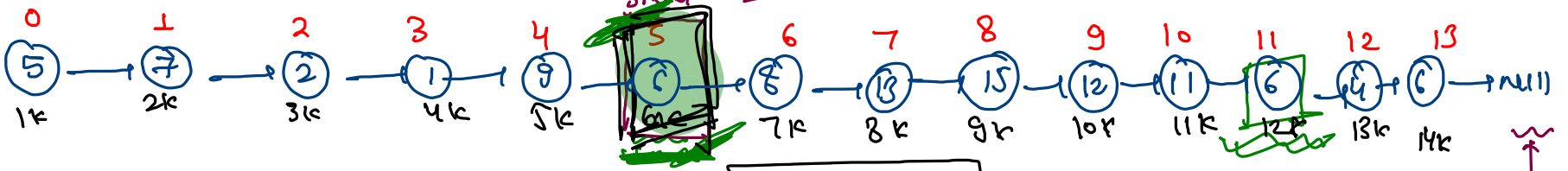
Step 2- Segregate like odd Even

odd == value <= pivot

Even = value > pivot

odd Even

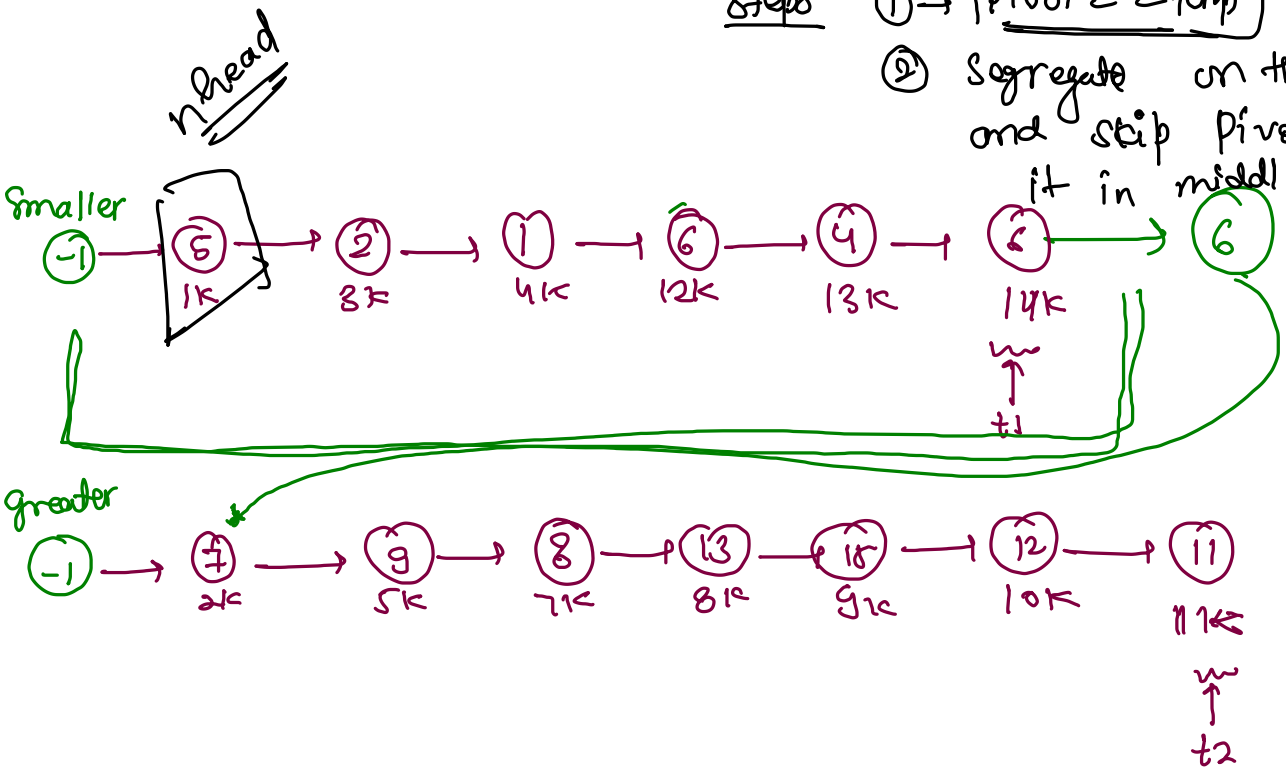
Segregate Node Of Linkedlist Over Pivot Index



Steps: ① → pivot = = temp → address

② Segregate on the basis of data and skip pivot Node to add it in middle later.

pivot = 6 (5.) 6K




```
t1.next = pivot;
pivot.next = greater.next;
t2.next = null;
return smaller.next
```