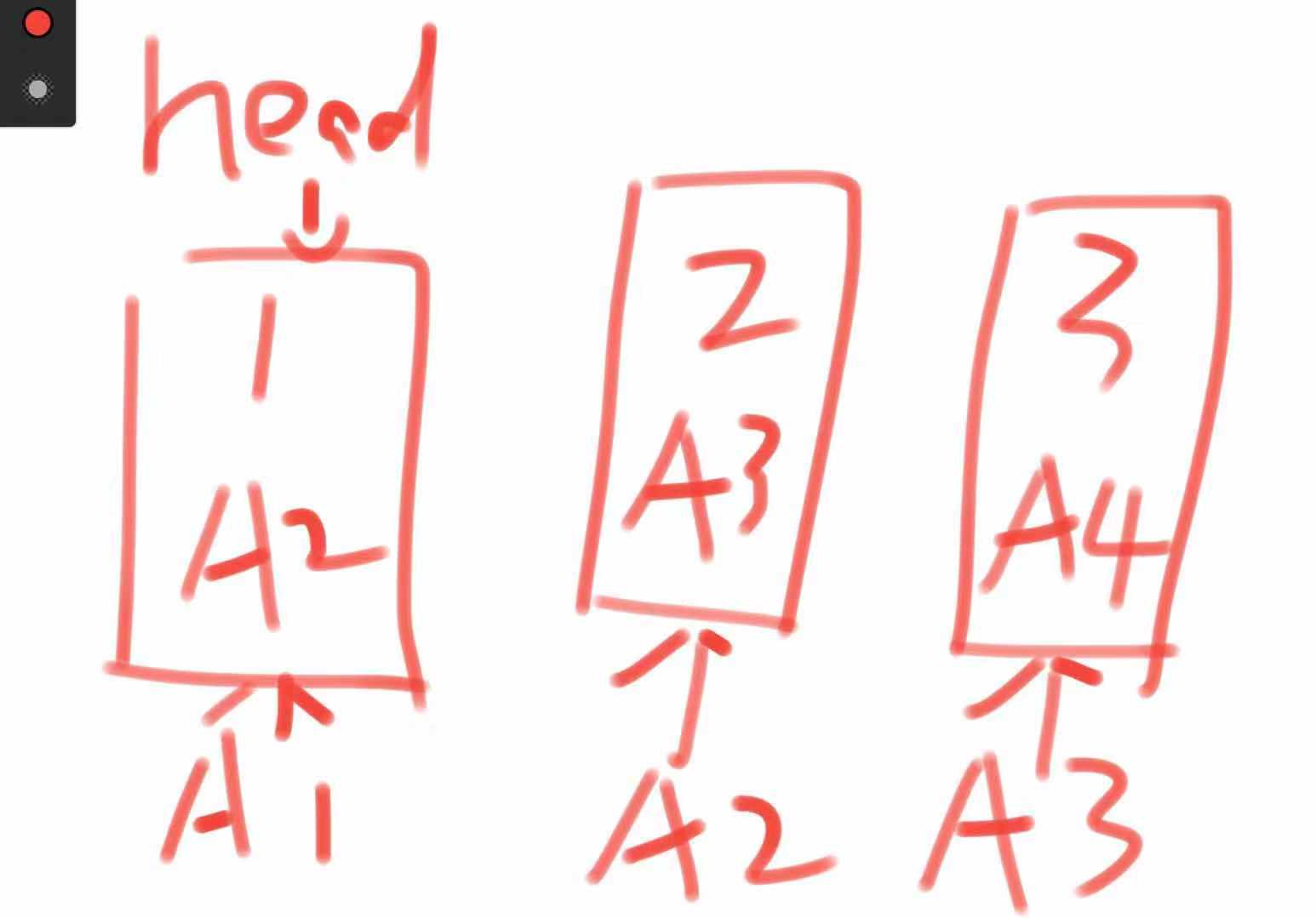
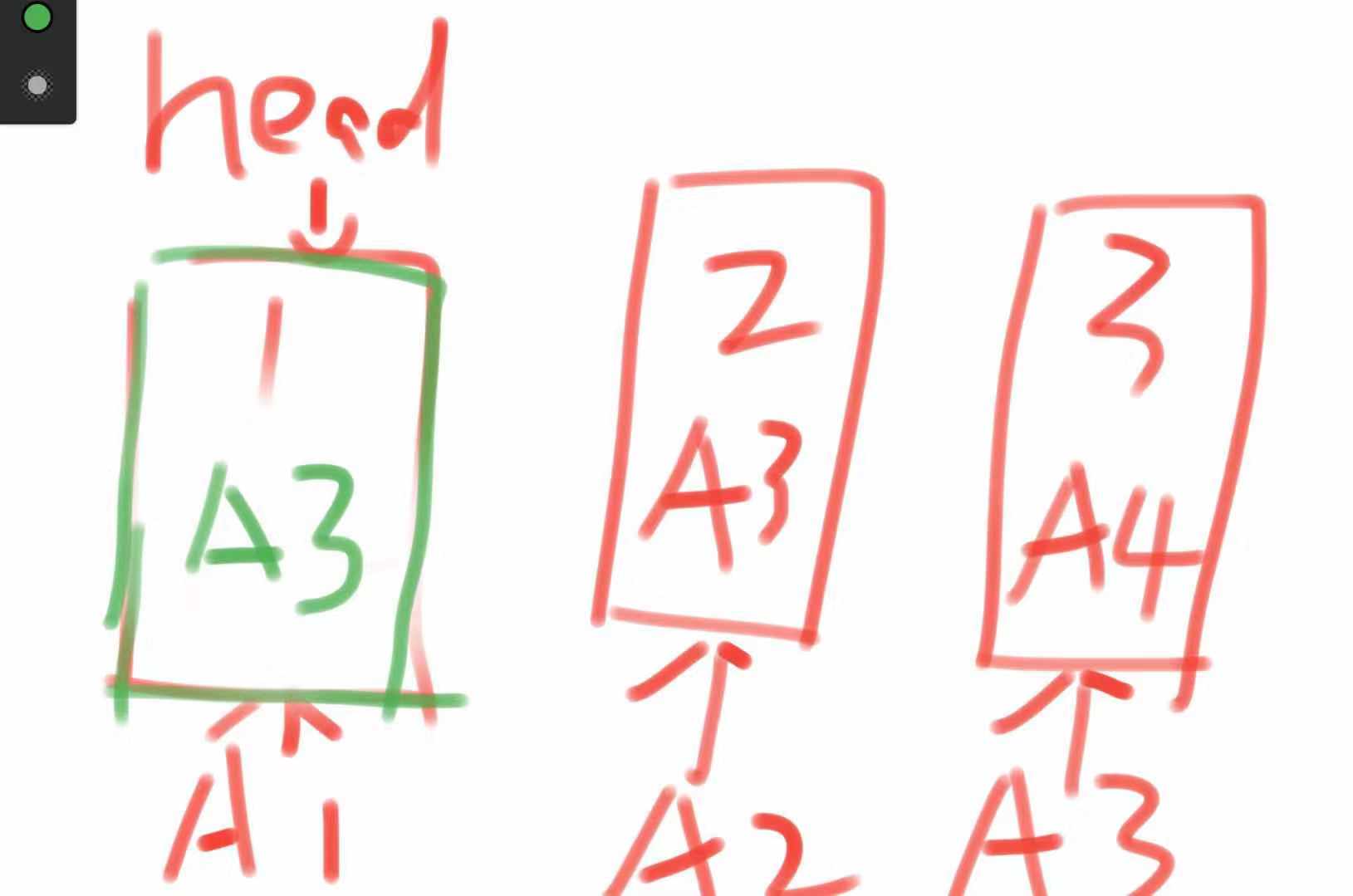
第一次提交就0ms太感动了。

还是这两幅图（感谢公公）





和之前说的一样，有链表的要在前面加一个节点。

然后我这个代码里面first->next就是A1，second->next就是A2，third->next就是A3。/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

\* ListNode(int x) : val(x), next(NULL) {}

\* };

\*/

class Solution {

public:

ListNode\* swapPairs(ListNode\* head) {

ListNode newhead (0);

newhead.next = head;

ListNode\* first = &newhead;

while (first -> next && first -> next -> next){

ListNode\* second = first -> next;

ListNode\* third = first -> next -> next;

ListNode\* temp3 = third -> next;

ListNode\* temp2 = second -> next;

third -> next = first -> next;

second -> next = temp3;

first -> next = temp2;

first = first -> next -> next;

}

return newhead.next;

}

};

在discussion里看到一个代码，觉得挺好

class Solution {

public:

ListNode\* swapPairs(ListNode\* head) {

if (head == nullptr || head->next == nullptr) return head;

ListNode\* new\_head = head->next;// 2 is new head, 1 is head

ListNode\* third = head->next->next;// store 3

new\_head->next = head;

**head->next = swapPairs(third);**

return new\_head;

}

};

最重要的是中间那句标粗的。

只剩一个或者没有的话就直接return，return后就接在head->next（上一层head是A3），然后return两两交换后的new\_head（比如4和3交换就是A3），return之后就是上一层的head->next，因为上一层的head是A1所以就是1的那个节点next是A4！