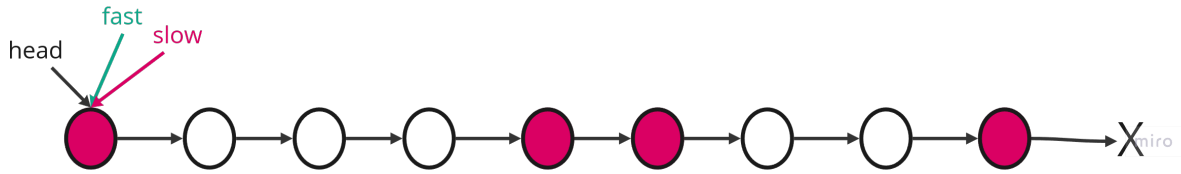
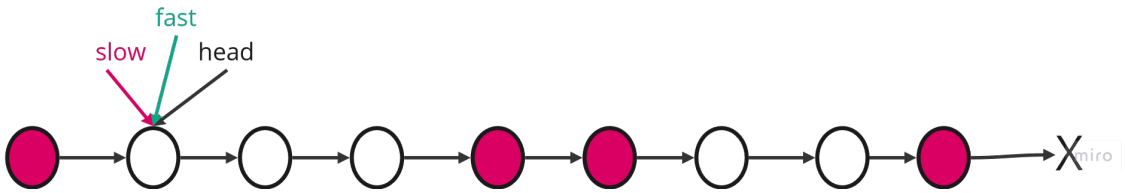
# Straight forward approach (memory leaks)

## If head... else...

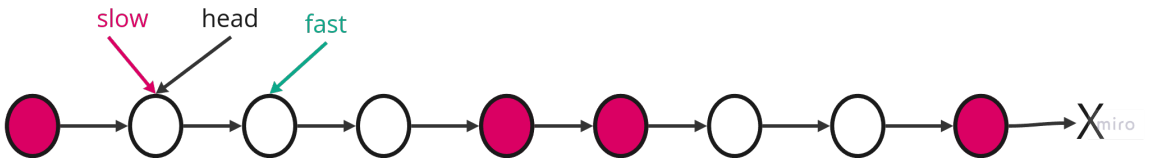
```
/*
    Sorting+Binary search (Memory leaks)
    if head... else...

    Time complexity: O(nlogn)
    Space complexity: O(logn)
*/
typedef std::vector<int> vi;

class Solution {
    public:
        ListNode* modifiedList(vi& nums, ListNode* head){
            std::sort(nums.begin(),nums.end());

            ListNode* fast=head;
            ListNode* slow=head;
            while(fast){
                if(std::binary_search(nums.begin(),nums.end(),fast->val)){
                    if(fast==head) {
                        head=head->next;
                        fast=slow=head;
                    }
                    else{
                        slow->next=fast->next;
                        fast=fast->next;
                    }
                }
                else{
                    slow=fast;
                    fast=fast->next;
                }
            }
            return head;
        }
};
```



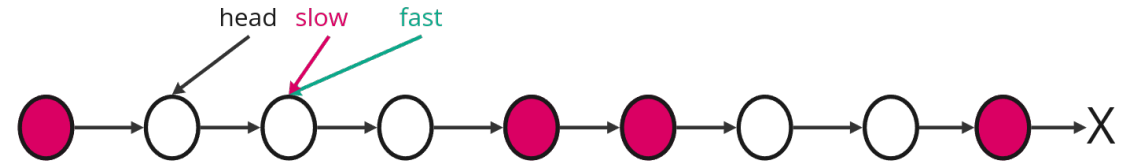

fast points to the node to delete and fast is equal to head:

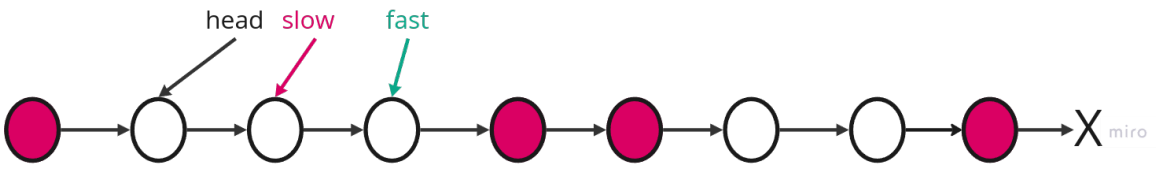



fast does not point to the node to delete:

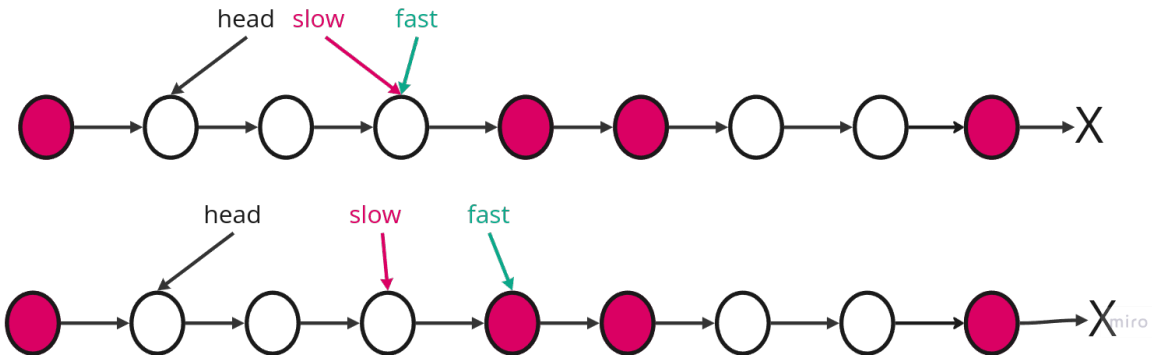



fast does not point to the node to delete:

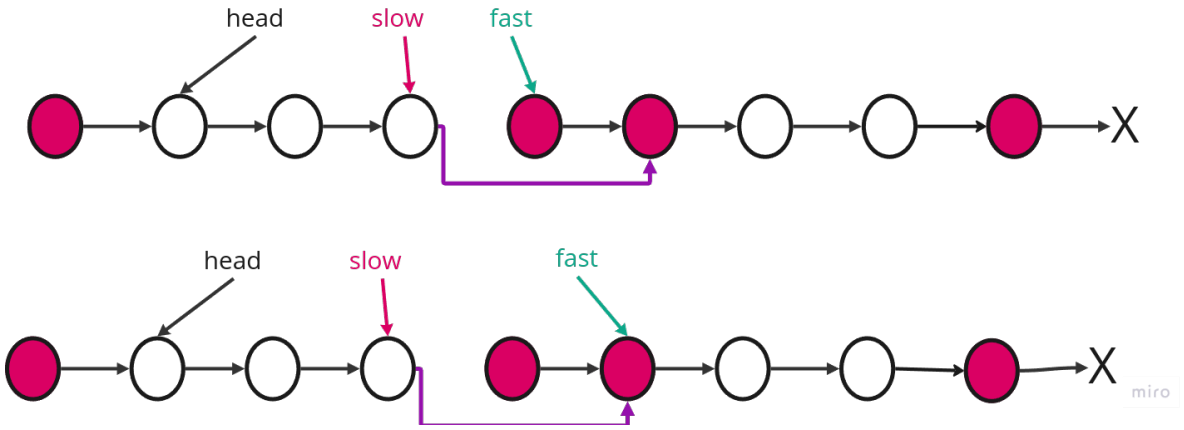


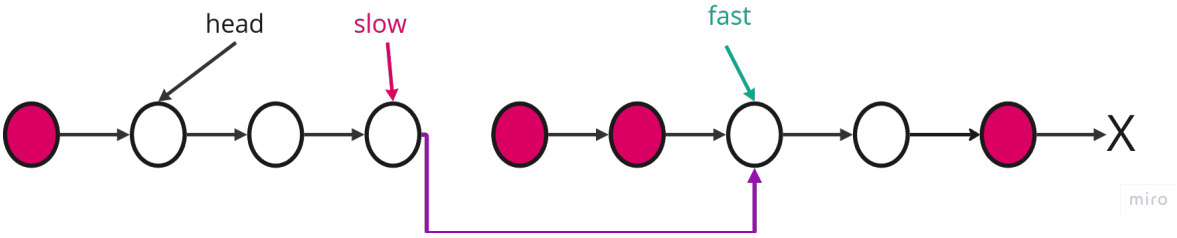
fast does not point to the node to delete:



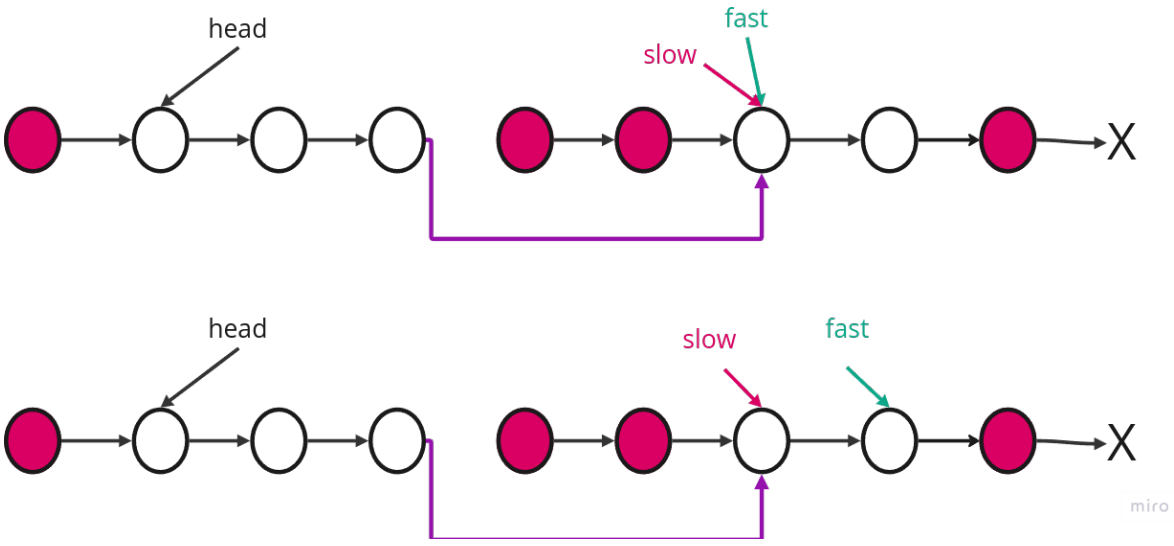

fast points to the node to delete and fast is not equal to head:



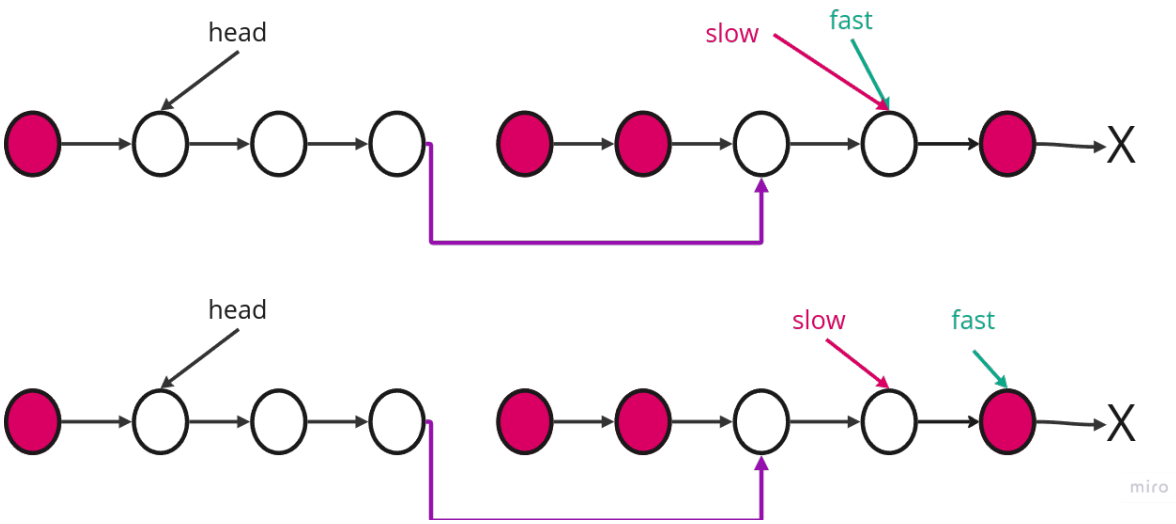

fast points to the node to delete and fast is not equal to head:



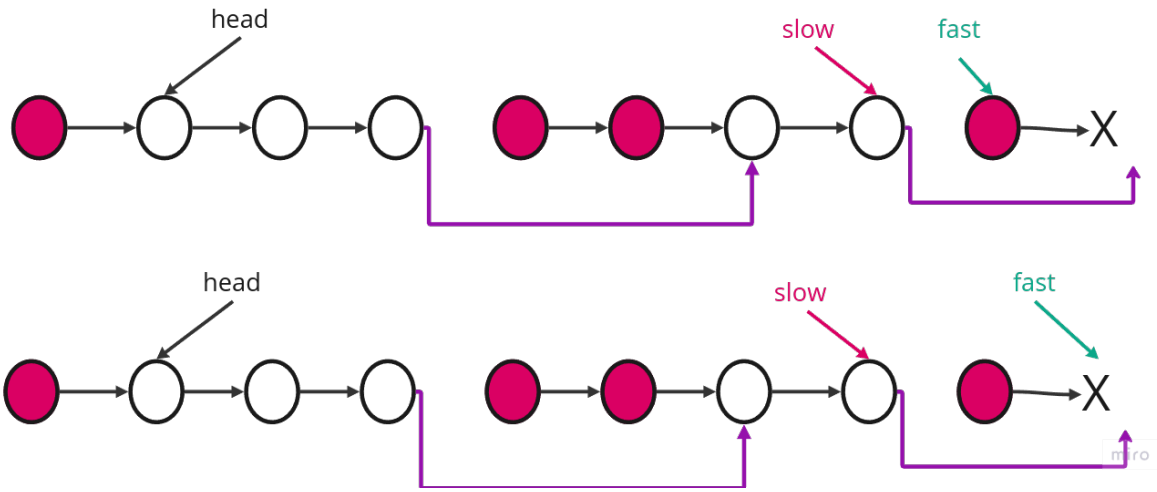

fast does not point to the node to delete:



fast does not point to the node to delete:




fast points to the node to delete and fast is not equal to head:




At end the we get: