Linked list leetcode-:

1.design linked list-:

```
class MyLinkedList {

    LinkedList <Integer> list;

    public MyLinkedList() {
        list=new LinkedList<>();
    }

    public int get(int index) {
        if(index>=list.size()) return -1;
        return list.get(index);
    }

    public void addAtHead(int val) {
        list.addFirst(val);
    }

    public void addAtTail(int val) {
        list.addLast(val);
    }

    public void addAtIndex(int index, int val) {
        if(index<list.size()) list.add(index,val);
        else if(index==list.size()) addAtTail(val);
    }

    public void deleteAtIndex(int index) {
        if(index<list.size()){
        list.remove(index);
        }

    }
}
```

2.linked list cycle-:

```
public class Solution {
    public boolean hasCycle(ListNode head) {

        ListNode fast = head;
        ListNode slow = head;
        while (fast != null && fast.next != null) {
            fast=fast.next.next;
            slow=slow.next;
            if(fast==slow){
                return true;
            }
```

```
        }
        return false;
    }
}
```

3.remove nth node from end of list-:

```
class Solution {
    public ListNode removeNthFromEnd(ListNode head, int n) {
        if(head==null) return null;
        ListNode s=head;
        ListNode f=head;

        for(int i=0;i<n;i++){
            f=f.next;
        }
        if(f==null) return head.next;

        while(f!=null && f.next!=null){
            s=s.next;
            f=f.next;
        }

        ListNode temp=s.next.next;
        s.next=temp;
        return head;

    }
}
```