```
class LinkedListNode{
    constructor(value){
        this.value = value;
        this.next = null;
    }
}

// let head = new LinkedListNode(5);
// head.next = new LinkedListNode(6);
// head.next.next = new LinkedListNode(6);

//convert this array to Linkelist

const func = () => {

    let arr = [2,4,5];
    let temp = null;
    for(let i=0;i<arr.length;i++){
        if(i ==0)
            {
                temp = new LinkedListNode(arr[i]);
            }
        else{
            temp.next = new LinkedListNode(arr[i]);
            temp = temp.next;
        }
        console.log(temp);
    }
}

console.log(func());







//https://leetcode.com/problems/reverse-linked-list/

const reverseLL = (head) => {

    //base case
    if(head == null|| head.next == null)
```

```
        return head;

    //general case
    let reversedLLHead = reverse(head.next);
    head.next.next = head;
    head.next = null;
    return reversedLLHead

}
```

https://leetcode.com/problems/middle-of-the-linked-list/description/

```
var middleNode = function(head) {

    let slow = head;
    let fast = head;

    while(fast != null && fast.next != null){

            slow = slow.next;
            fast = fast.next.next;
    }

    return slow;
};
```

```
https://leetcode.com/problems/merge-two-sorted-lists/description/
var mergeTwoLists = function(list1, list2) {
    let newHead = new ListNode(0);
    let temp = newHead;

    while(list1 != null && list2 != null){

        if(list1.val < list2.val)
            {
                temp.next = list1;
```

```
                list1 = list1.next;
            }

        else{
            temp.next = list2;
            list2 = list2.next;
        }

        temp = temp.next;
    }

    while(list1 != null){
        temp.next = list1;
        list1 = list1.next;
        temp = temp.next;
    }

       while(list2 != null){
        temp.next = list2;
        list2 = list2.next;
        temp = temp.next;
    }
    return newHead.next;

//for very large dataset
    //TC: o(n+m)= linear= O(n)
    //SC: o(n+m)
};
```

Home Work

```
//find loop in linkedlist
```