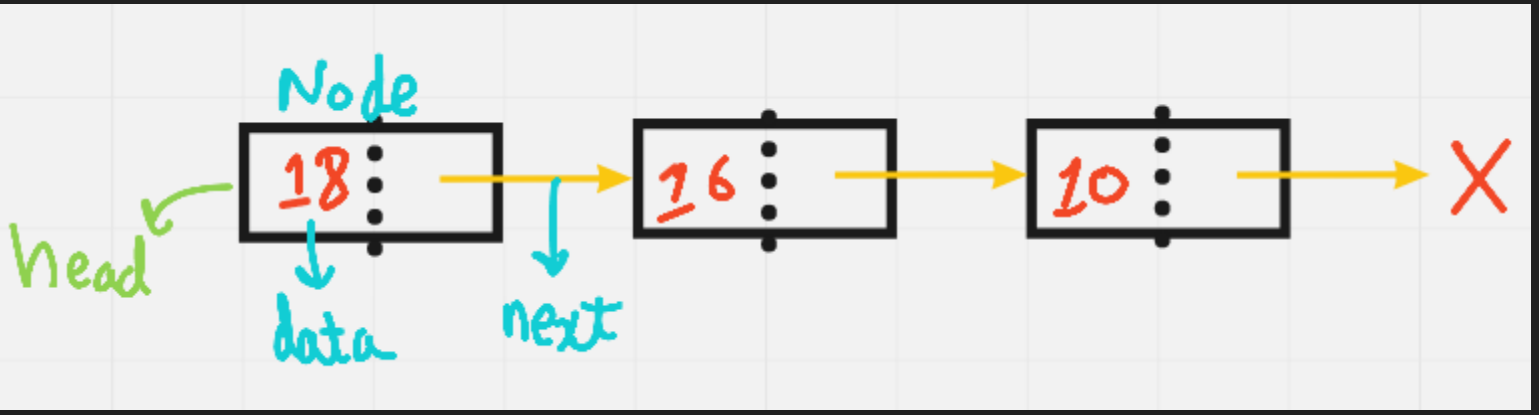
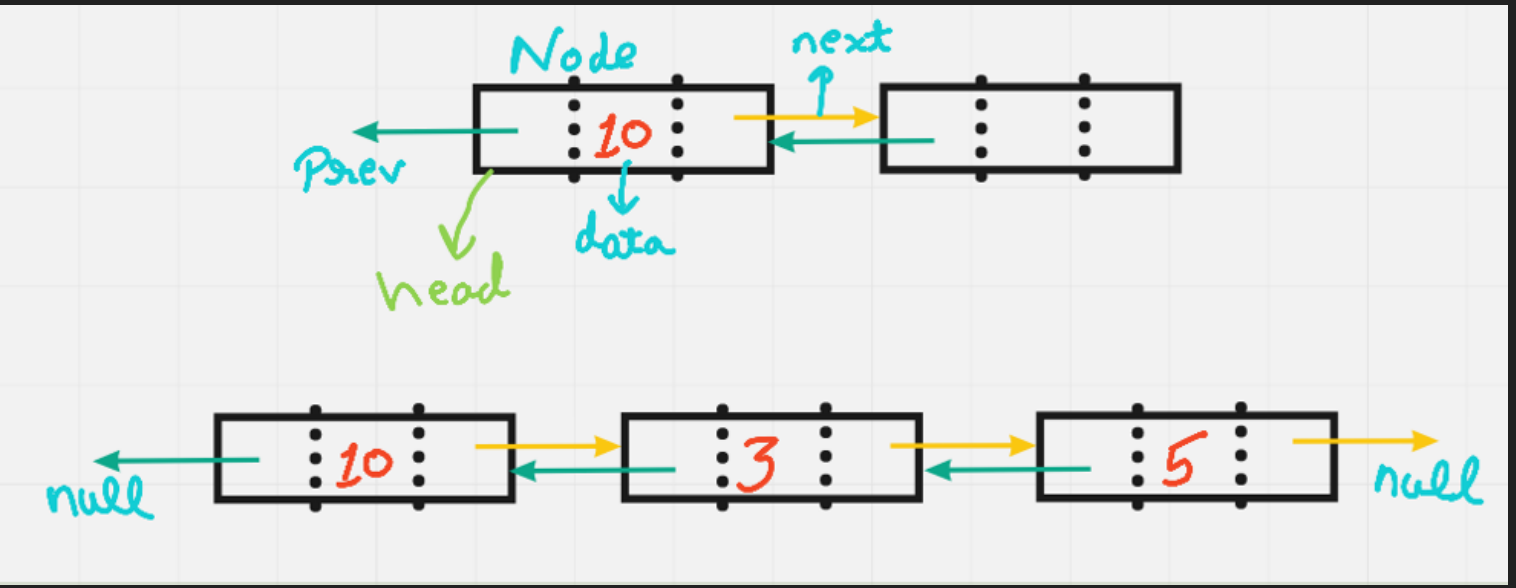
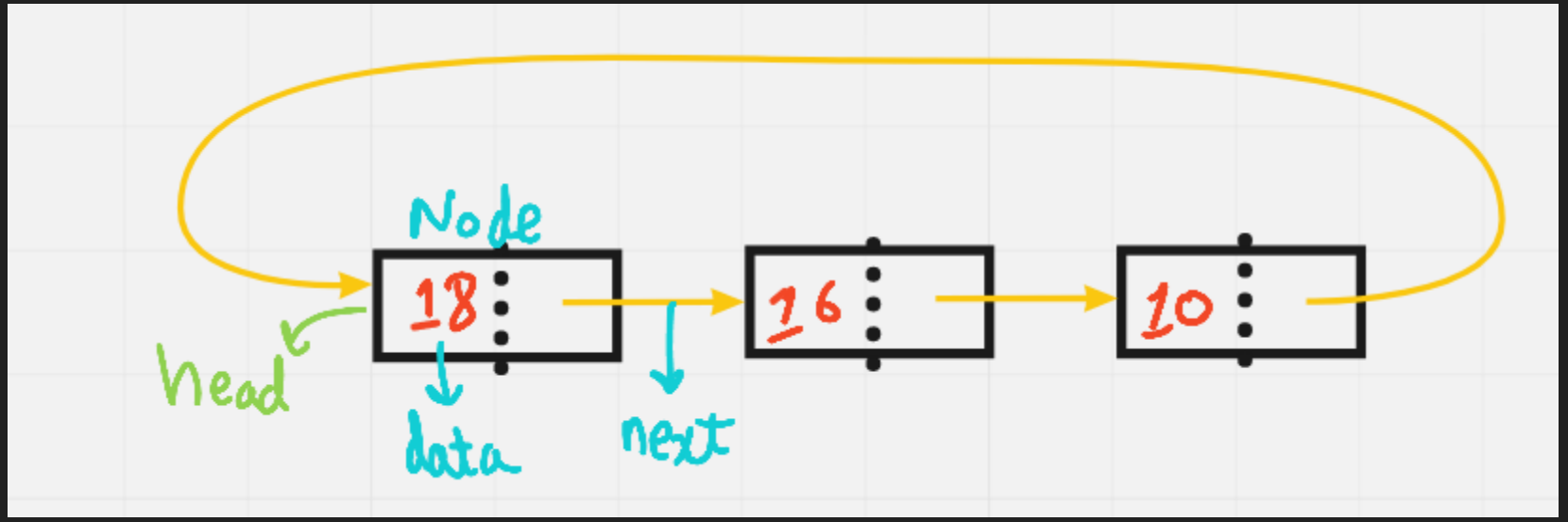
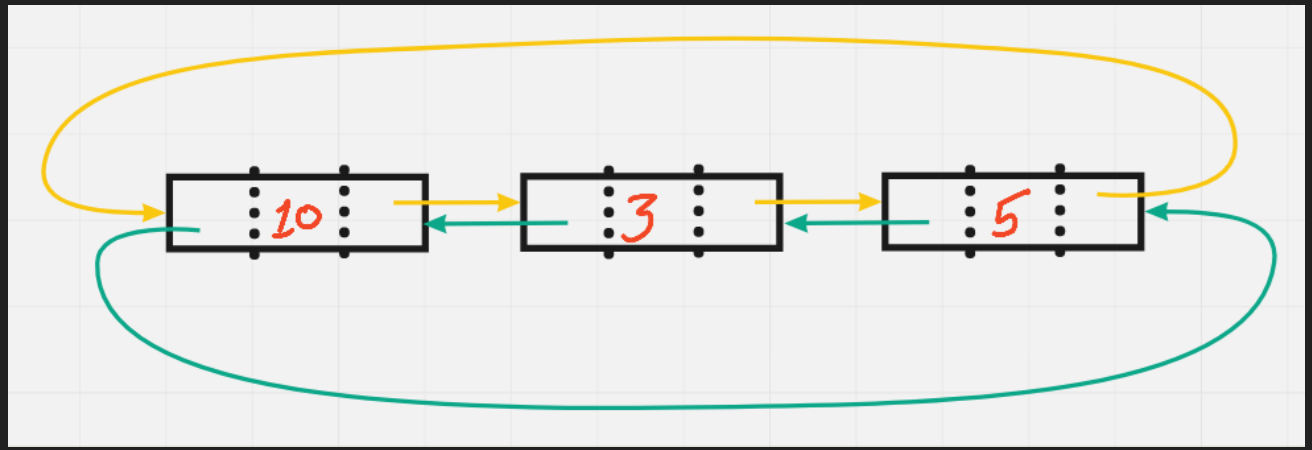
Linked list revision

https://leetcode.com/discuss/study-guide/1800120/become-master-in-linked-list


Node
18
16
10
X
head
data
next


Node
next
10
Prev
data
head
null
10
3
5
null


Node
18
16
10
head
data
next


10
3
5

```
class Node{
  int data;
  int next;
  node (int data){
    this.data = data;
  }
}
```


```
class Node<TreeNode>{
  TreeNode data;
  Node next;
  Node (TreeNode data){
    this.data = data;
  }
}

void main() {
  traverse(head);
}

void traverse(Node head){
  Node curr = head;

  while(curr != null){
    print(curr.data);
    curr = curr.next;
  }
}
```

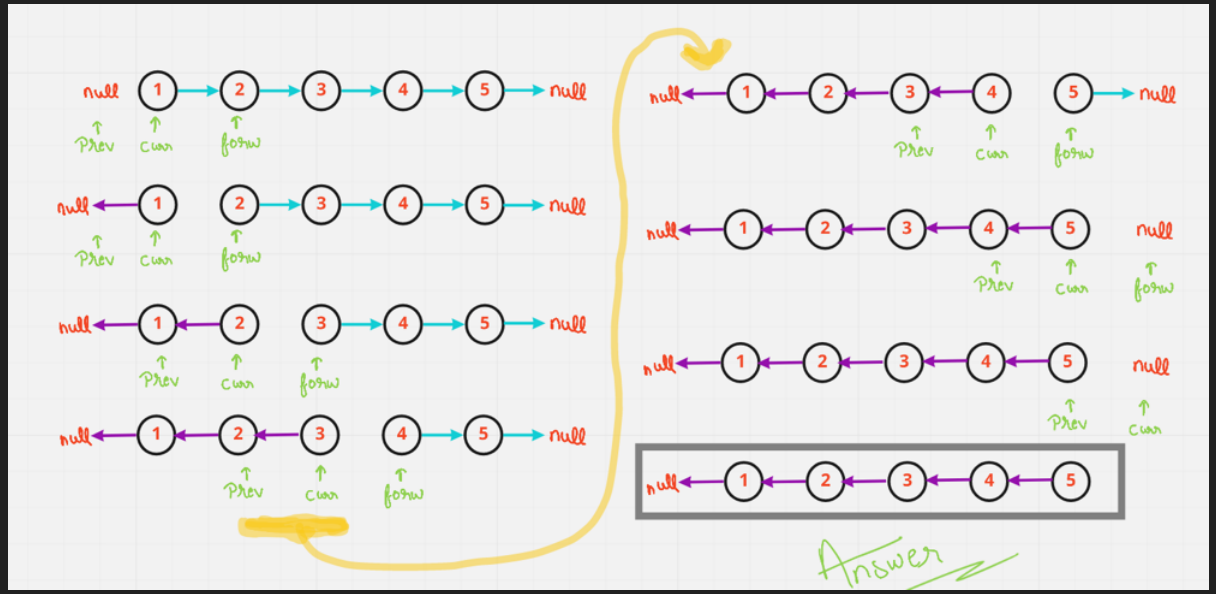
```
class Node{
  int data;
  int next;
  node (int data){
    this.data = data;
  }
}

void main() {
  Node n1 = new Node(10);
  Node n2 = new Node(20);
  Node n3 = new Node(30);

  Node head = n1;
  head.next = n2;
  n2.next = n3;
  n3.next = null;
}
```

```
void main() {
  insert(30, head, 3);
}

void insert(int data, Node head, int pos){
  Node toAdd = new node(data);
  // Base Condition
  if(pos == 0){
    toAdd.next = head;
    head = toAdd;
    return;
  }

  Node prev = head;

  for(int i = 0; i < pos - 1; i++){
    prev = prev.next;
  }
  toAdd.next = prev.next;
  prev.next = toAdd;
}
```

Reverse

Solution :-

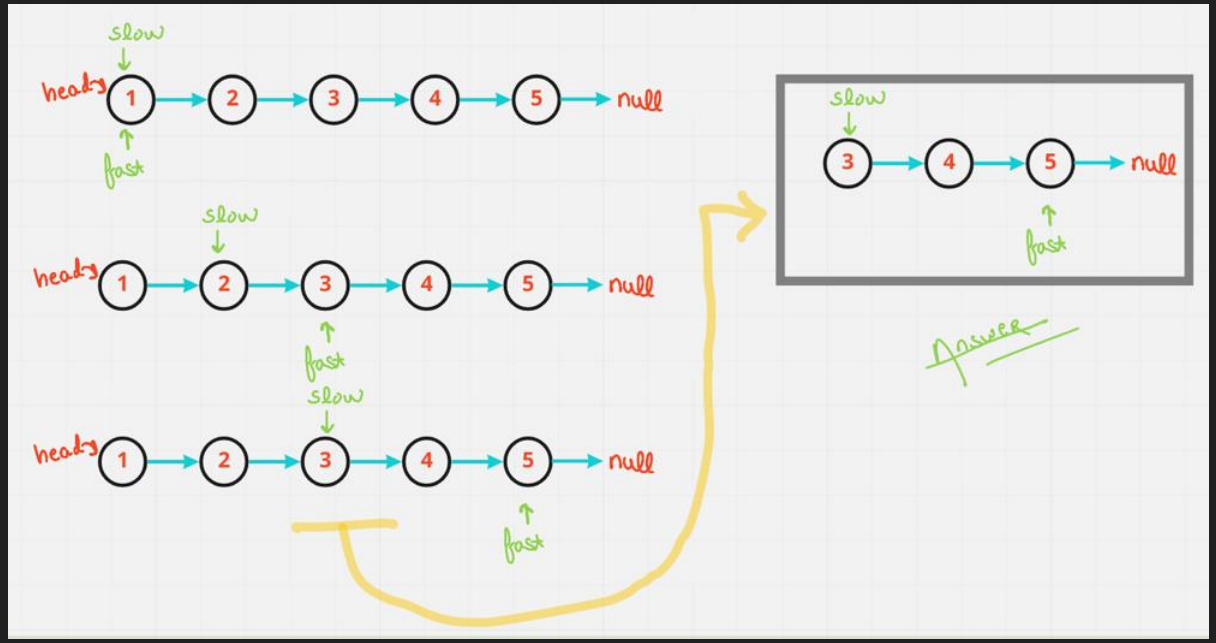
```
class Solution {
    public ListNode reverseList(ListNode head) {
        ListNode prev = null;
        ListNode curr = head;
        ListNode forw = null;

        while(curr != null){
            forw = curr.next;
            curr.next = prev;
            prev = curr;
            curr = forw;
        }
        return prev;
    }
}
```



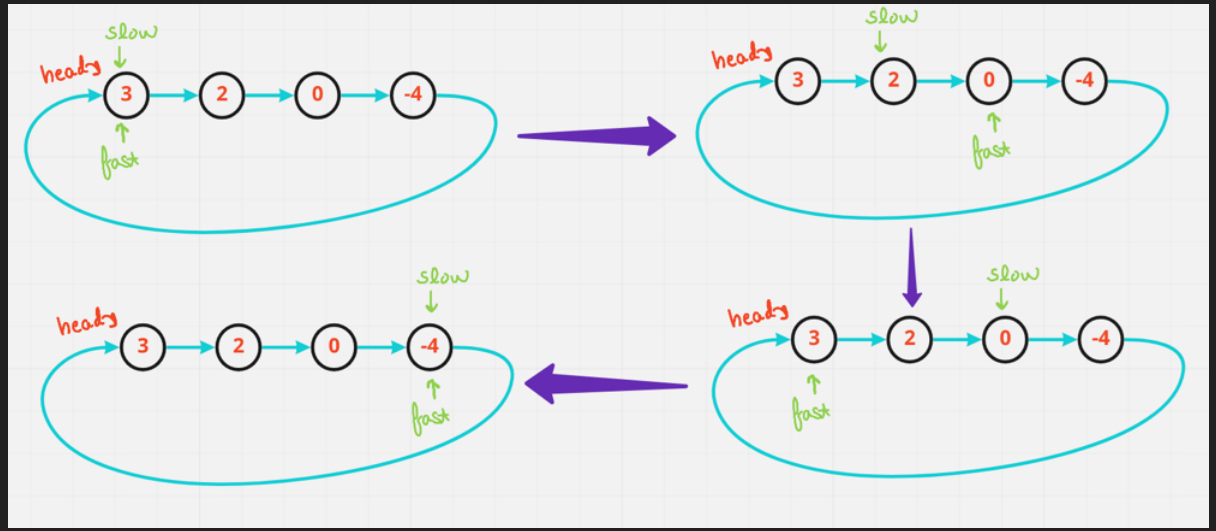
Middle

Solution:-

```
class Solution {
    public ListNode middleNode(ListNode head) {
        // Base Condition
        if(head.next == null) return head;

        ListNode slow = head;
        ListNode fast = head;

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



Cycle - same

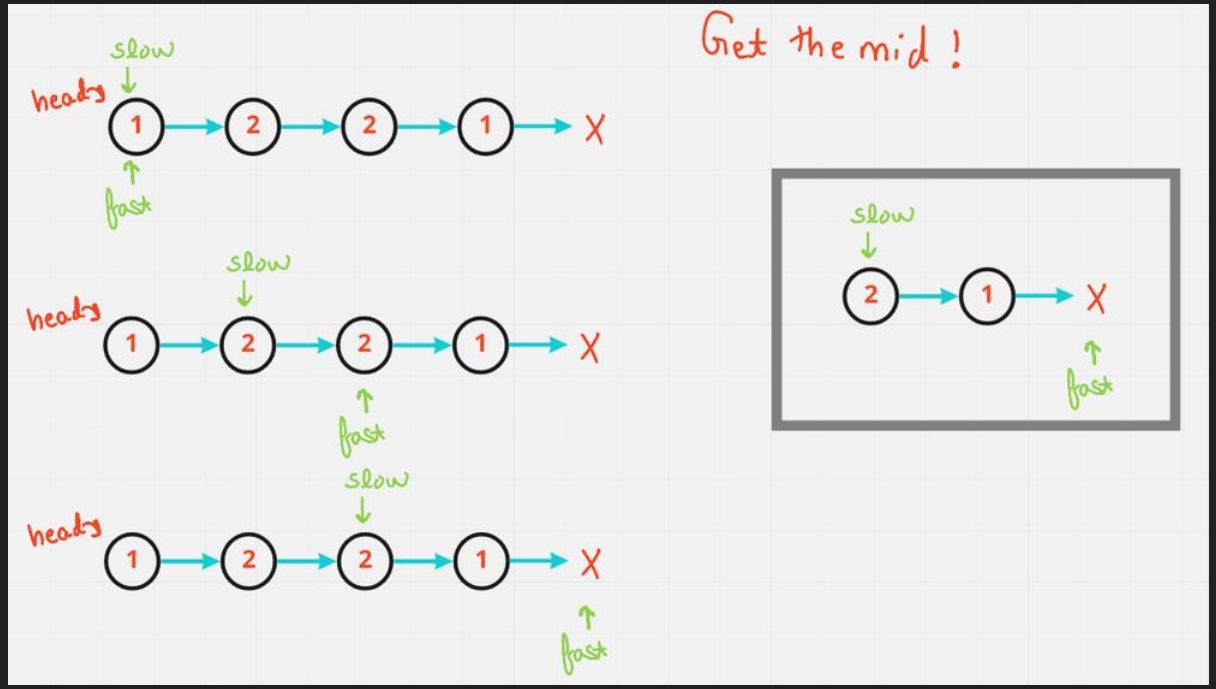

slow
head
3
2
0
-4
fast
slow
head
3
2
0
-4
fast
slow
head
3
2
0
-4
fast
slow
head
3
2
0
-4
fast

Palindrome

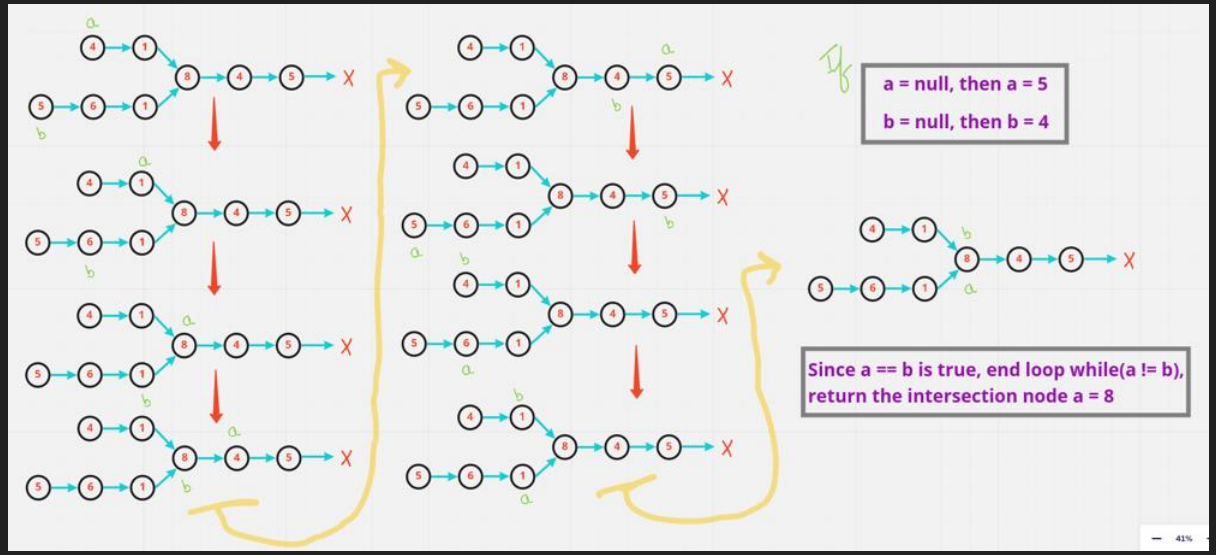
Middle

Reverse

Find



Intersection



Solution:-

```
public class Solution {
    public ListNode getIntersectionNode(ListNode headA, ListNode headB) {
        ListNode a = headA;
        ListNode b = headB;

        while(a != b){
            a = a == null ? headB : a.next;
            b = b == null ? headA : b.next;
        }
        return b;
    }
}
```

Leetcode article

https://leetcode.com/discuss/study-guide/1800120/become-master-in-linked-list