CODING SHUTTLE

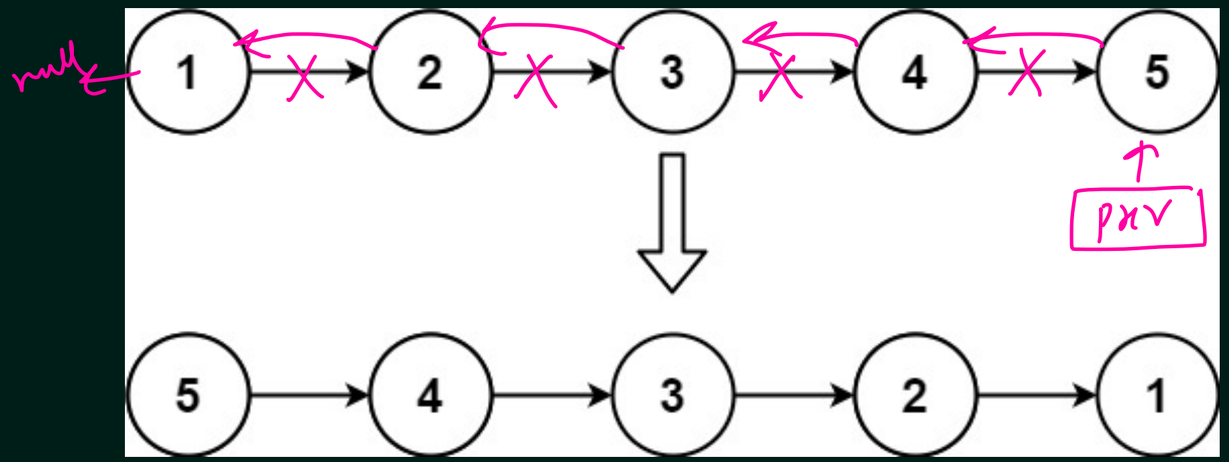
Linked List - 3

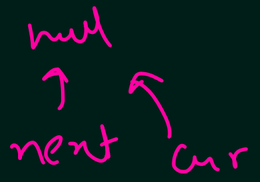
In This Lecture

1. Reverse a Linked List
2. Check if a LinkedList is Palindrome

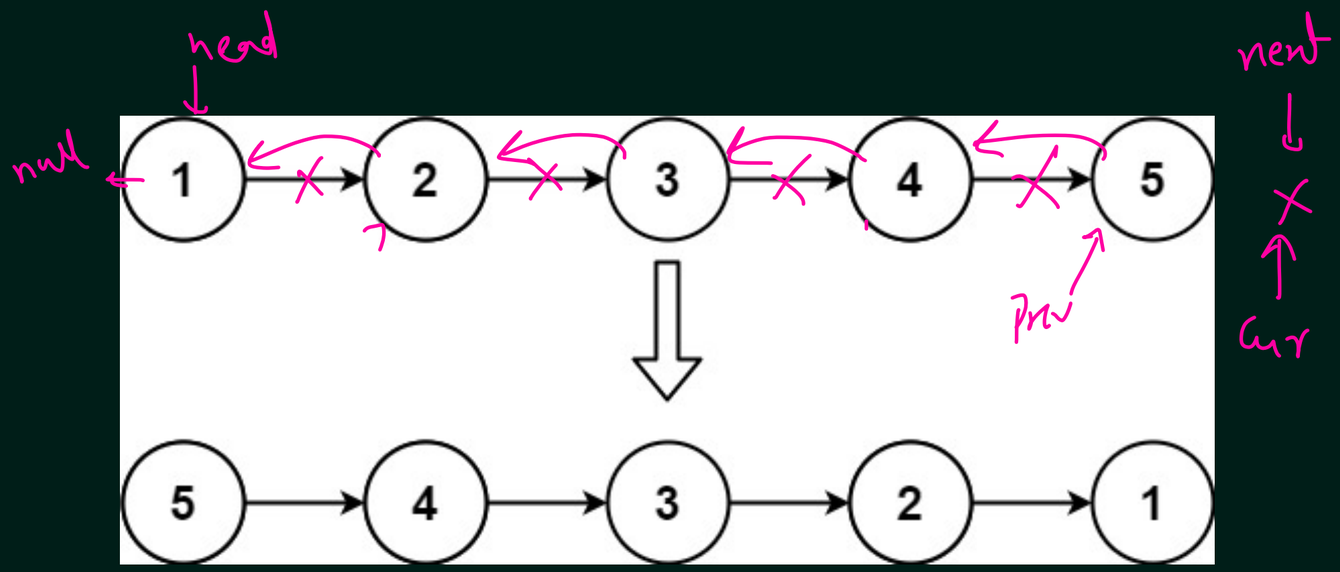
Reverse a Linked List

Given the head of a singly linked list, reverse the list, and return the reversed list.



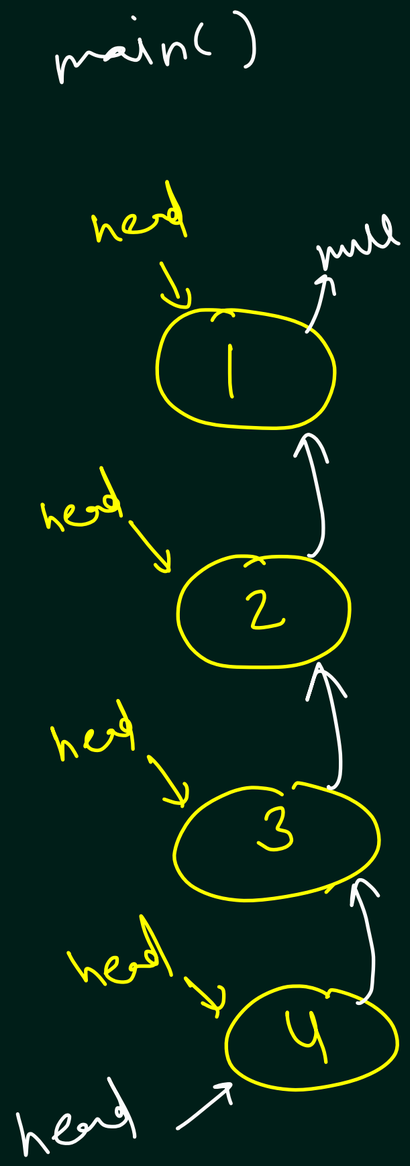


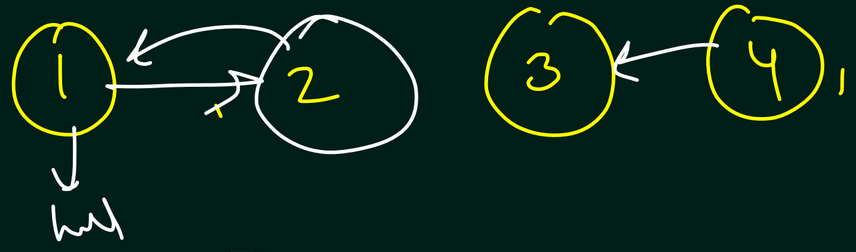
Reverse a Linked List




```java
static Node reverseLinkedList(Node head) {
    if(head == null || head.next == null) return head;
    Node prev = head;
    Node cur = head.next;
    head.next = null;

    while(cur != null) {
        Node next = cur.next;
        cur.next = prev;
        prev = cur;
        cur = next;
    }

    return prev;
}
```

Reverse a Linked List

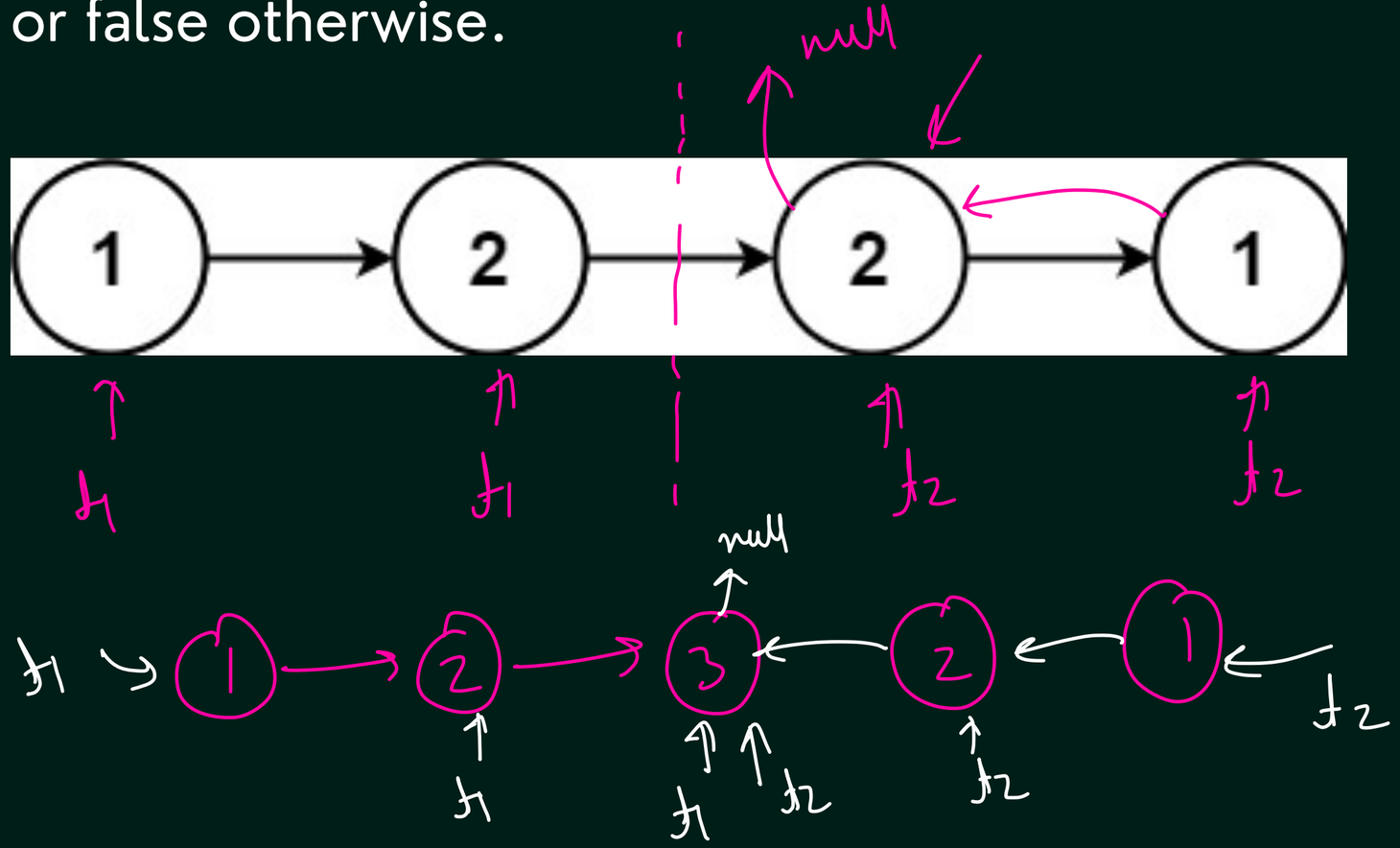




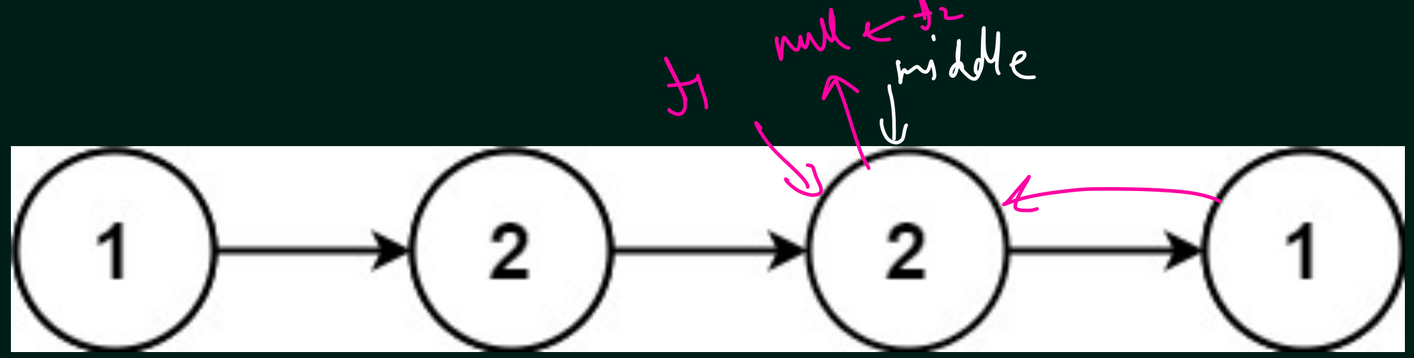

```java
static Node reverseLLRecursively(Node head) {
    if(head == null || head.next == null) return head;

    Node headOfSubProblem = reverseLLRecursively(head.next);

    head.next.next = head;
    head.next = null;

    return headOfSubProblem;
}
```

Check if a LinkedList is Palindrome

Given the head of a singly linked list, return true if it is a palindrome or false otherwise.



Check if a LinkedList is Palindrome




```
static boolean isLLPalindrome(Node head) {
    Node middle = findMiddle(head);
    Node t2 = reverseLinkedList(middle);
    Node t1 = head;

    while(t2 != null) {
        if(t1.data != t2.data) {
            return false;
        }
        t1 = t1.next;
        t2 = t2.next;
    }
    return true;
}
```