CODING SHUTTLE

Week 8 LIVE

LinkedList Problems And Doubts Session

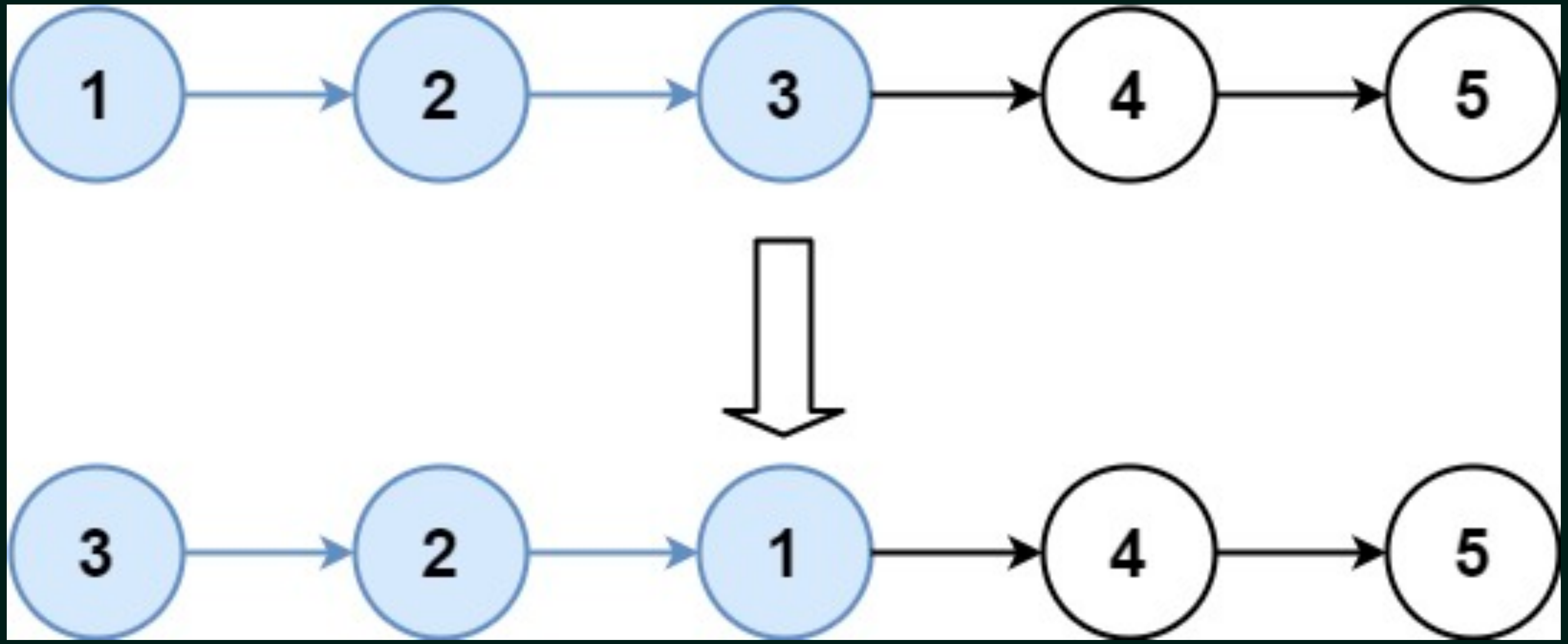
In This Lecture

1. Reverse K Lists
2. Circular Linked List

Reverse K Linked List

Given the head of a linked list, reverse the nodes of the list k at a time, and return the modified list.

k is a positive integer and is less than or equal to the length of the linked list. If the number of nodes is not a multiple of k then left-out nodes, in the end, should remain as it is.

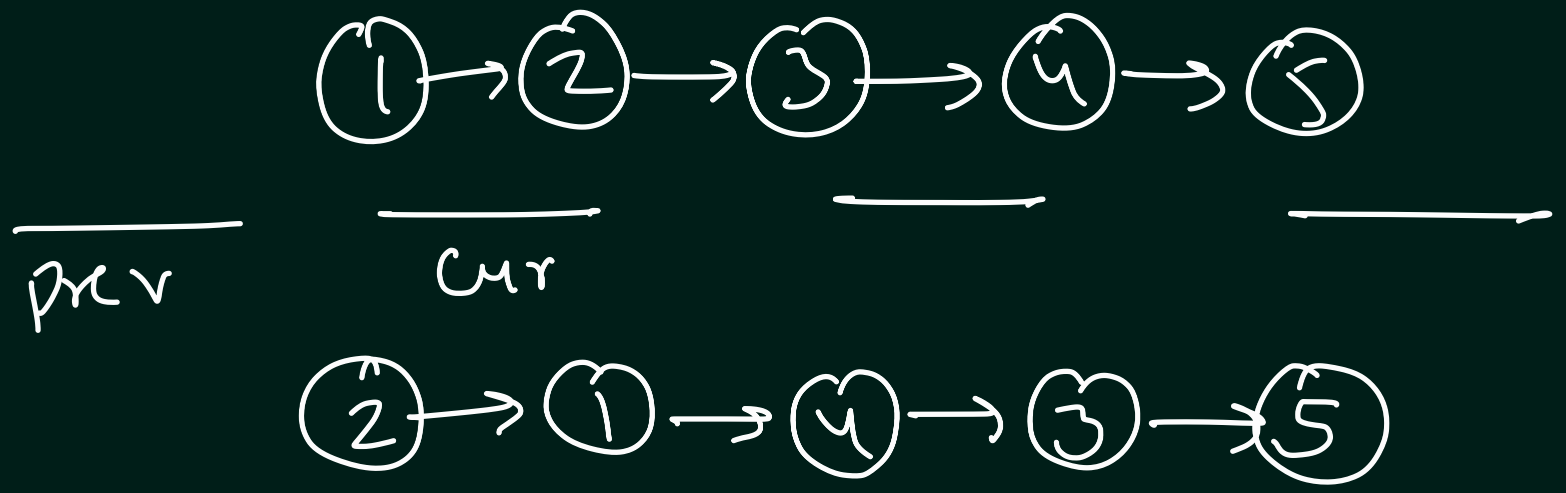


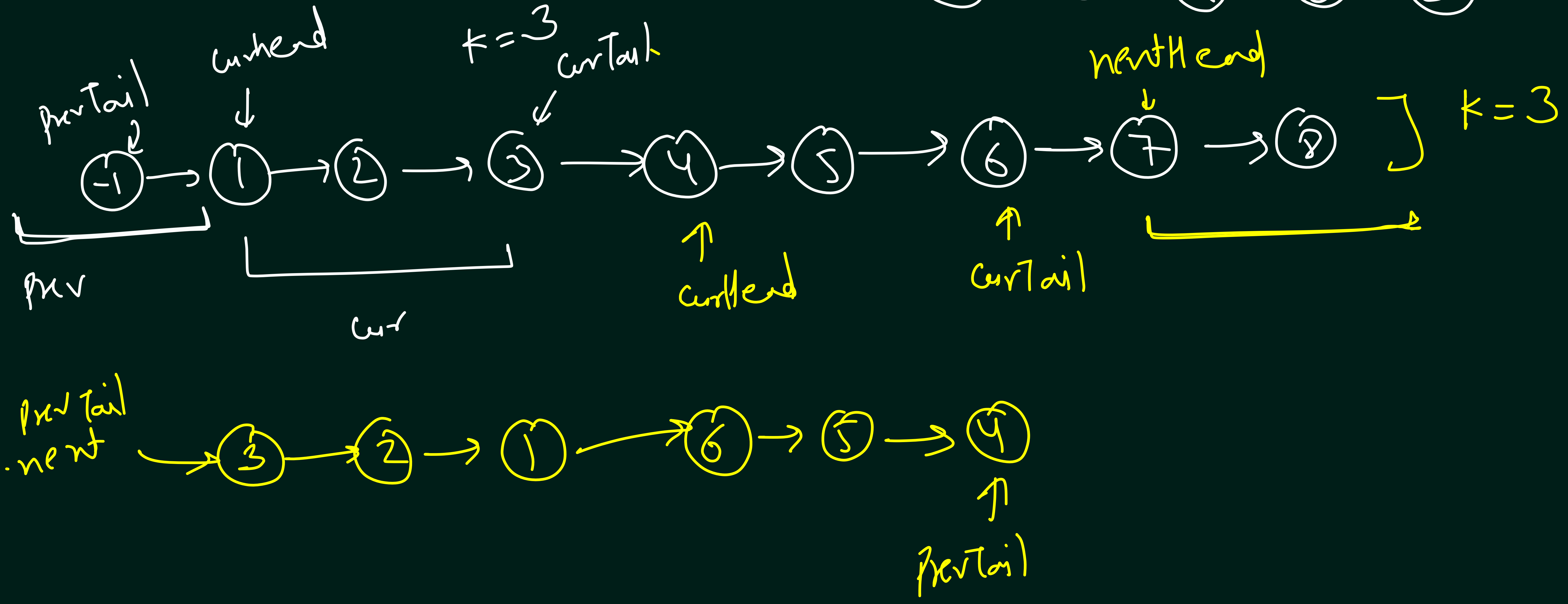
K=3

Reverse K Linked List

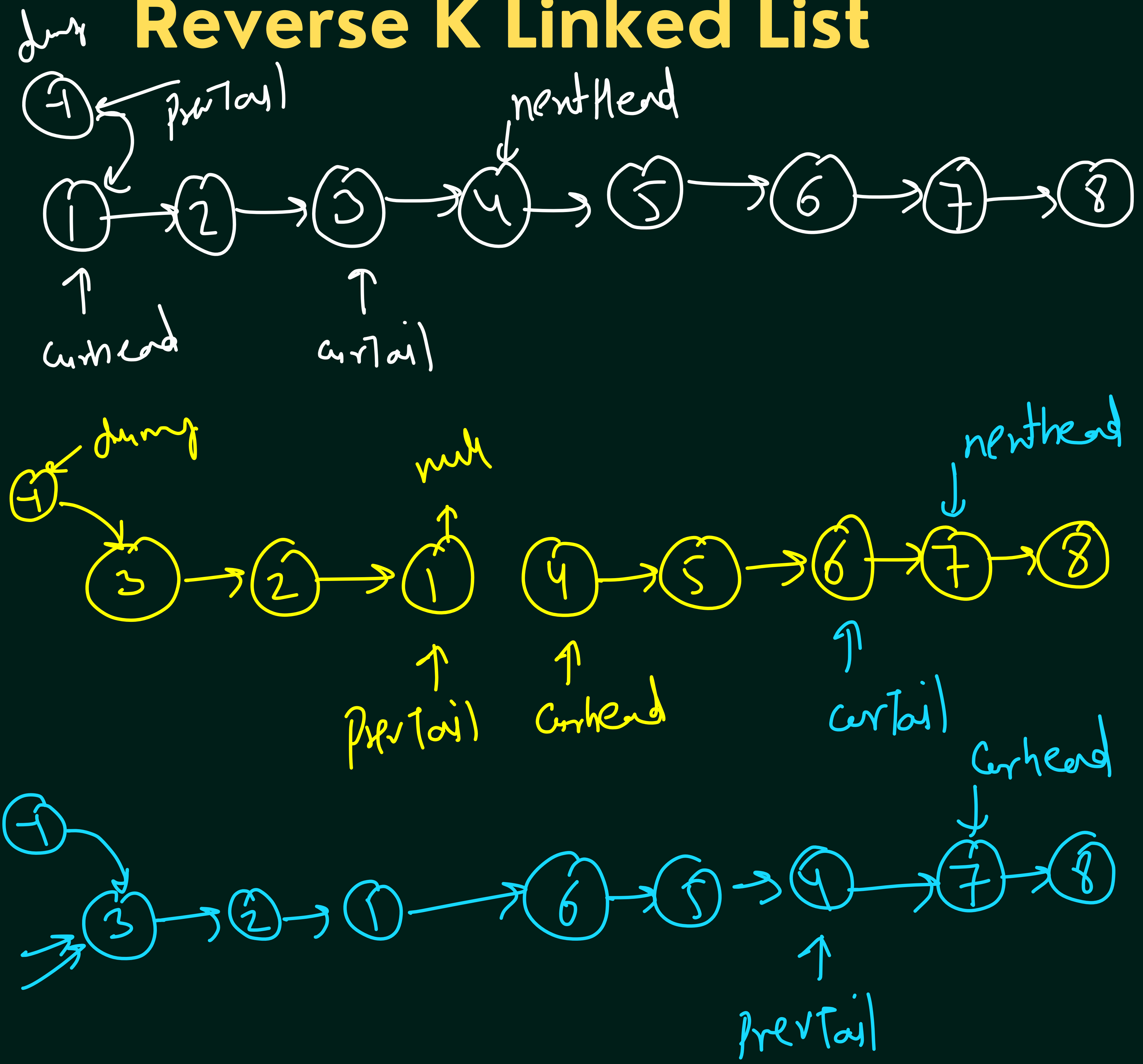
Input: head = [1,2,3,4,5], k = 2

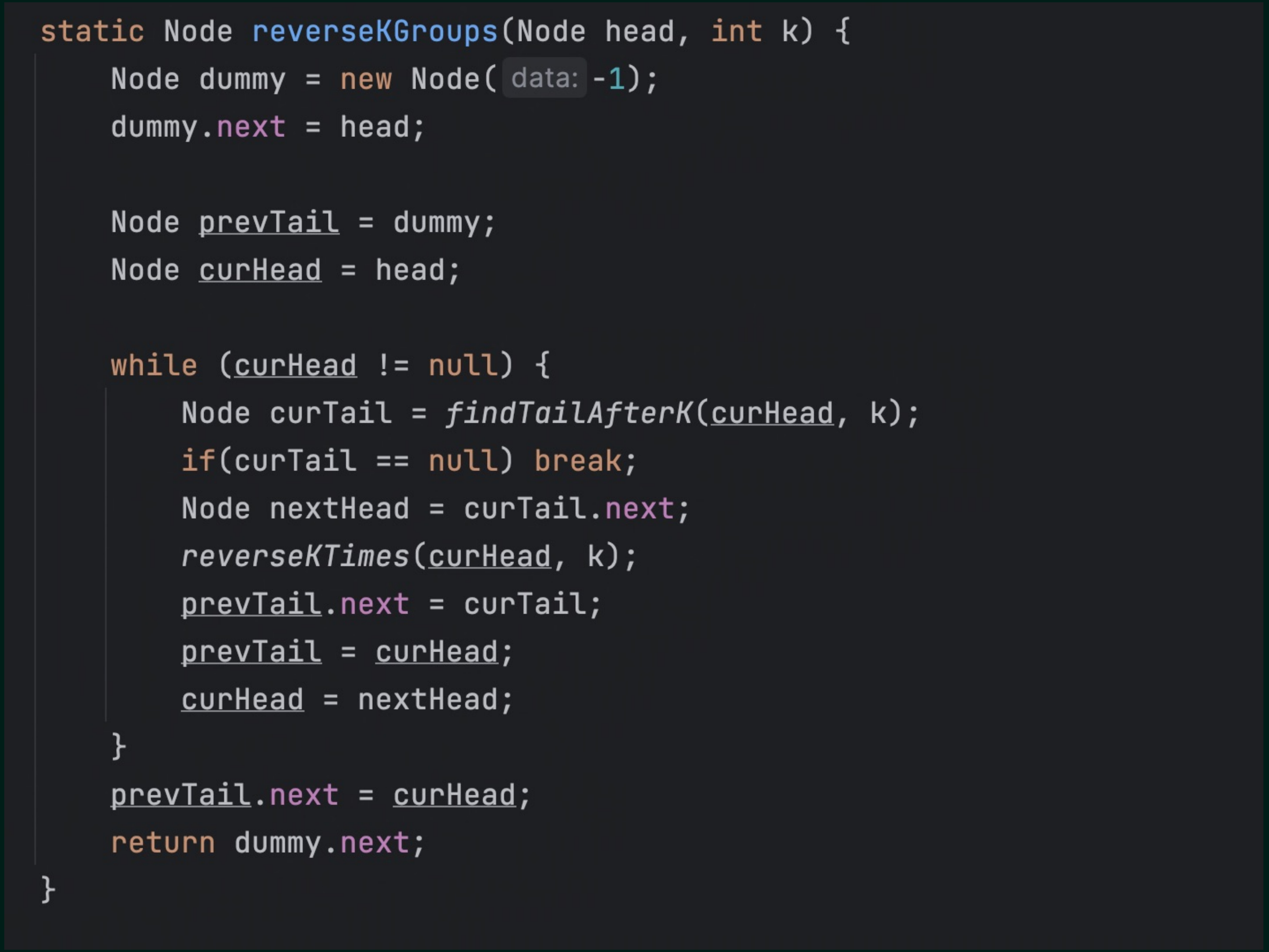
Output: [2,1,4,3,5]





Reverse K Linked List





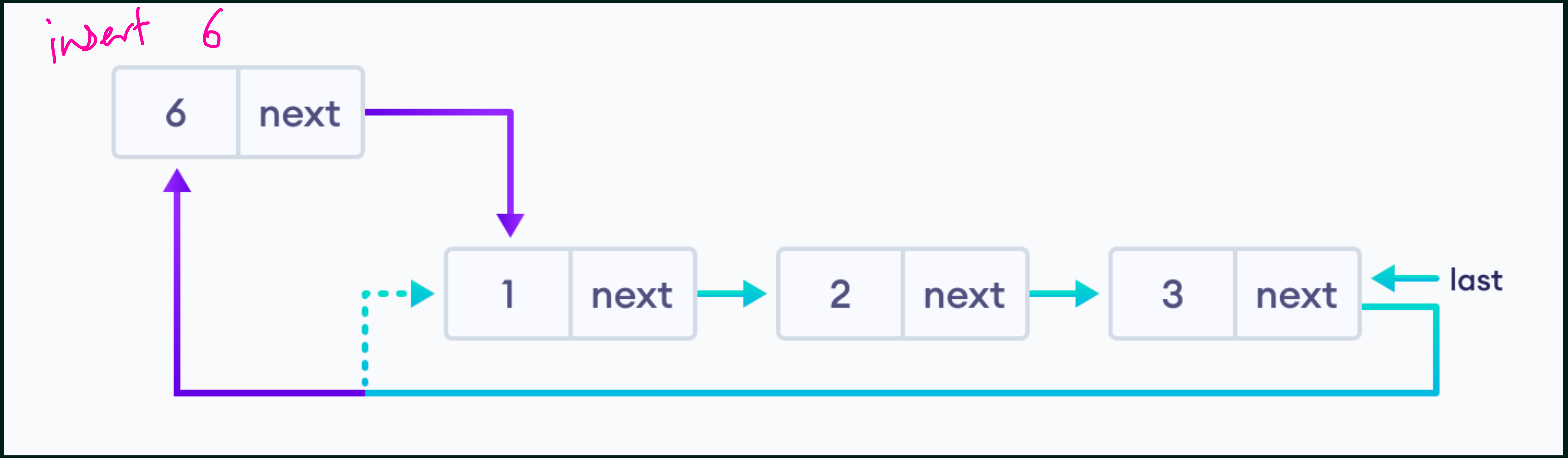
```java
static Node reverseKGroups(Node head, int k) {
    Node dummy = new Node( data: -1);
    dummy.next = head;

    Node prevTail = dummy;
    Node curHead = head;

    while (curHead != null) {
        Node curTail = findTailAfterK(curHead, k);
        if(curTail == null) break;
        Node nextHead = curTail.next;
        reverseKTimes(curHead, k);
        prevTail.next = curTail;
        prevTail = curHead;
        curHead = nextHead;
    }
    prevTail.next = curHead;
    return dummy.next;
}
```

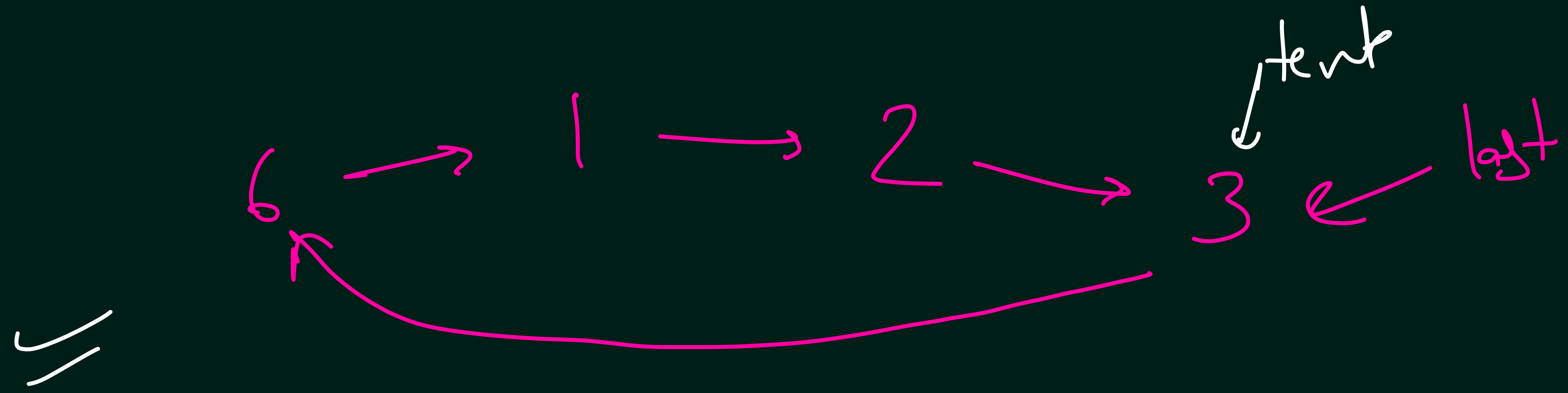
Circular Linked List

A circular linked list is a type of linked list in which the first and the last nodes are also connected to each other to form a circle.



Why Circular Linked List?

1. The NULL assignment is not required because a node always points to another node.
2. The starting point can be set to any node.
3. Traversal from the first node to the last node is quick.



$$\begin{tikzcd} 6 \arrow[r] & 1 \arrow[r] & 2 \arrow[r] & 3 \arrow[lll, bend left] \end{tikzcd}$$