Linked List Practice Questions (Easy, Medium, Hard)

# Easy Level:

1. Reverse a Linked List  
Problem: Reverse a singly linked list.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5  
Expected Output: 5 -> 4 -> 3 -> 2 -> 1

2. Find the Length of a Linked List  
Problem: Given a linked list, find its length.  
Input: head = 1 -> 2 -> 3 -> 4 -> NULL  
Expected Output: 4

3. Check if a Linked List is Palindrome  
Problem: Given a singly linked list, check if it is a palindrome.  
Input: head = 1 -> 2 -> 3 -> 2 -> 1  
Expected Output: True

4. Detect a Cycle in a Linked List  
Problem: Given a linked list, determine if it contains a cycle.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5 -> 2 (cycle)  
Expected Output: True (cycle exists)

5. Delete a Node in a Linked List  
Problem: Given a linked list and a node to be deleted, delete the node from the list.  
Input: head = 1 -> 2 -> 3 -> 4, node = 3  
Expected Output: 1 -> 2 -> 4

6. Merge Two Sorted Linked Lists  
Problem: Merge two sorted linked lists into one sorted linked list.  
Input: l1 = 1 -> 2 -> 4, l2 = 1 -> 3 -> 4  
Expected Output: 1 -> 1 -> 2 -> 3 -> 4 -> 4

7. Find the Middle of a Linked List  
Problem: Given a singly linked list, find the middle of the list.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5  
Expected Output: 3

8. Remove Duplicates from a Sorted Linked List  
Problem: Given a sorted linked list, remove all duplicate elements.  
Input: head = 1 -> 1 -> 2 -> 3 -> 3  
Expected Output: 1 -> 2 -> 3

9. Find the N-th Node from the End of the Linked List  
Problem: Given a linked list, return the n-th node from the end of the list.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, n = 2  
Expected Output: 4

10. Merge Two Sorted Lists into a New List  
Problem: Given two sorted linked lists, merge them into a new list.  
Input: l1 = 2 -> 3 -> 5, l2 = 1 -> 4 -> 6  
Expected Output: 1 -> 2 -> 3 -> 4 -> 5 -> 6

11. Delete the Head Node of a Linked List  
Problem: Delete the head node of a given linked list and return the new head.  
Input: head = 1 -> 2 -> 3 -> 4  
Expected Output: 2 -> 3 -> 4

12. Print a Linked List  
Problem: Given a linked list, print the values of all the nodes.  
Input: head = 1 -> 2 -> 3 -> NULL  
Expected Output: 1 2 3

13. Add Two Numbers Represented by Linked Lists (Simple)  
Problem: Add two numbers where each number is represented by a linked list in reverse order. Return the sum as a linked list in reverse order.  
Input: l1 = 2 -> 4 -> 3, l2 = 5 -> 6 -> 4  
Expected Output: 7 -> 0 -> 8

14. Flatten a Linked List  
Problem: Flatten a linked list where each node may have a child pointer.  
Input: head = 1 -> 2 -> 3 -> NULL with child pointers.  
Expected Output: Flattened list with child nodes included.

15. Remove N-th Node from the End of the List  
Problem: Remove the n-th node from the end of the list in a single pass.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, n = 2  
Expected Output: 1 -> 2 -> 3 -> 5

Medium Level:

1. Reverse a Linked List in Groups of K  
Problem: Given a linked list and a number k, reverse every k nodes in the list.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, k = 3  
Expected Output: 3 -> 2 -> 1 -> 4 -> 5

2. Remove All Occurrences of a Value in a Linked List  
Problem: Given a linked list and a value, remove all occurrences of that value from the list.  
Input: head = 1 -> 2 -> 6 -> 3 -> 4 -> 5 -> 6, val = 6  
Expected Output: 1 -> 2 -> 3 -> 4 -> 5

3. Intersection Point of Two Linked Lists  
Problem: Find the intersection point of two linked lists.  
Input: head1 = 1 -> 2 -> 3 -> 4, head2 = 5 -> 6 -> 3 -> 4  
Expected Output: 3

4. Clone a Linked List with Random Pointer  
Problem: Clone a linked list where each node has a random pointer in addition to the next pointer.  
Input: head = 1 -> 2 -> 3 -> NULL with random pointers.  
Expected Output: Cloned list with same random pointers.

5. Add Two Numbers Represented by Linked Lists  
Problem: Add two numbers represented by linked lists and return the sum as a linked list.  
Input: l1 = 7 -> 1 -> 6, l2 = 5 -> 9 -> 2  
Expected Output: 2 -> 1 -> 9

6. Sort a Linked List (Merge Sort)  
Problem: Sort a linked list using merge sort.  
Input: head = 4 -> 2 -> 1 -> 3  
Expected Output: 1 -> 2 -> 3 -> 4

7. Reorder List  
Problem: Reorder a linked list in the form: L0 -> Ln -> L1 -> Ln-1 -> L2 -> Ln-2 ...  
Input: head = 1 -> 2 -> 3 -> 4 -> 5  
Expected Output: 1 -> 5 -> 2 -> 4 -> 3

8. Rotate a Linked List  
Problem: Rotate a linked list by k places.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, k = 2  
Expected Output: 4 -> 5 -> 1 -> 2 -> 3

9. Palindrome Linked List  
Problem: Determine whether a linked list is a palindrome.  
Input: head = 1 -> 2 -> 3 -> 2 -> 1  
Expected Output: True

10. Detect Cycle in a Linked List (Floyd's Tortoise and Hare)  
Problem: Detect a cycle in a linked list using two pointers.  
Input: head = 1 -> 2 -> 3 -> 4 -> 2 (cycle)  
Expected Output: True (cycle detected)

Hard Level:

1. Merge k Sorted Linked Lists  
Problem: Merge k sorted linked lists into a single sorted linked list.  
Input: lists = [[1,4,5], [1,3,4], [2,6]]  
Expected Output: 1 -> 1 -> 2 -> 3 -> 4 -> 4 -> 5 -> 6

2. Copy List with Random Pointer  
Problem: Clone a linked list where each node has a random pointer in addition to the next pointer.  
Input: head = 1 -> 2 -> 3 -> NULL with random pointers.  
Expected Output: Cloned list with random pointers intact.

3. Flatten a Multilevel Doubly Linked List  
Problem: Flatten a multilevel doubly linked list into a single level list.  
Input: head with multiple levels of next and child pointers.  
Expected Output: Flattened list.

4. Reverse Nodes in k-Group  
Problem: Reverse nodes of a linked list in k-sized groups.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, k = 3  
Expected Output: 3 -> 2 -> 1 -> 4 -> 5

5. Merge Two Binary Search Trees  
Problem: Merge two binary search trees into a balanced binary search tree.  
Input: BST1 = 1 -> 2 -> 3, BST2 = 3 -> 4 -> 5  
Expected Output: Merged BST.

6. Remove Duplicates from an Unsorted Linked List  
Problem: Remove duplicates from an unsorted linked list.  
Input: head = 1 -> 2 -> 2 -> 3 -> 1  
Expected Output: 1 -> 2 -> 3

7. Longest Palindromic Subsequence in a Linked List  
Problem: Find the longest palindromic subsequence in a linked list.  
Input: head = 1 -> 2 -> 3 -> 2 -> 1  
Expected Output: 1 -> 2 -> 3 -> 2 -> 1

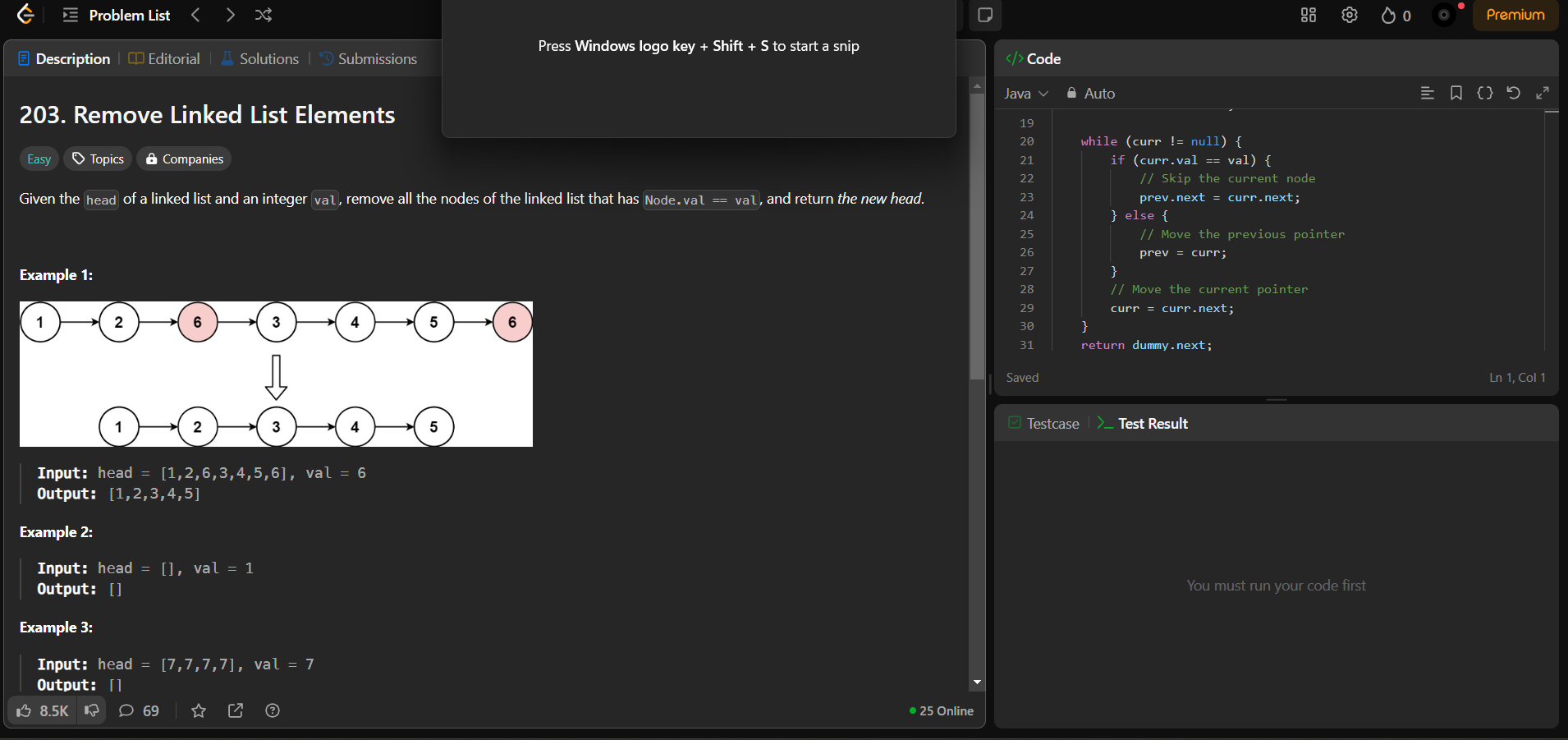
8. Design Linked List with Get, Set, Insert, Delete Operations  
Problem: Implement a linked list with get, set, insert, and delete operations.  
Input: Various operations on a linked list.  
Expected Output: Correct behavior of all operations.

9. K-th Smallest Element in a Linked List  
Problem: Find the k-th smallest element in a linked list.  
Input: head = 1 -> 3 -> 5 -> 7 -> 9, k = 2  
Expected Output: 3

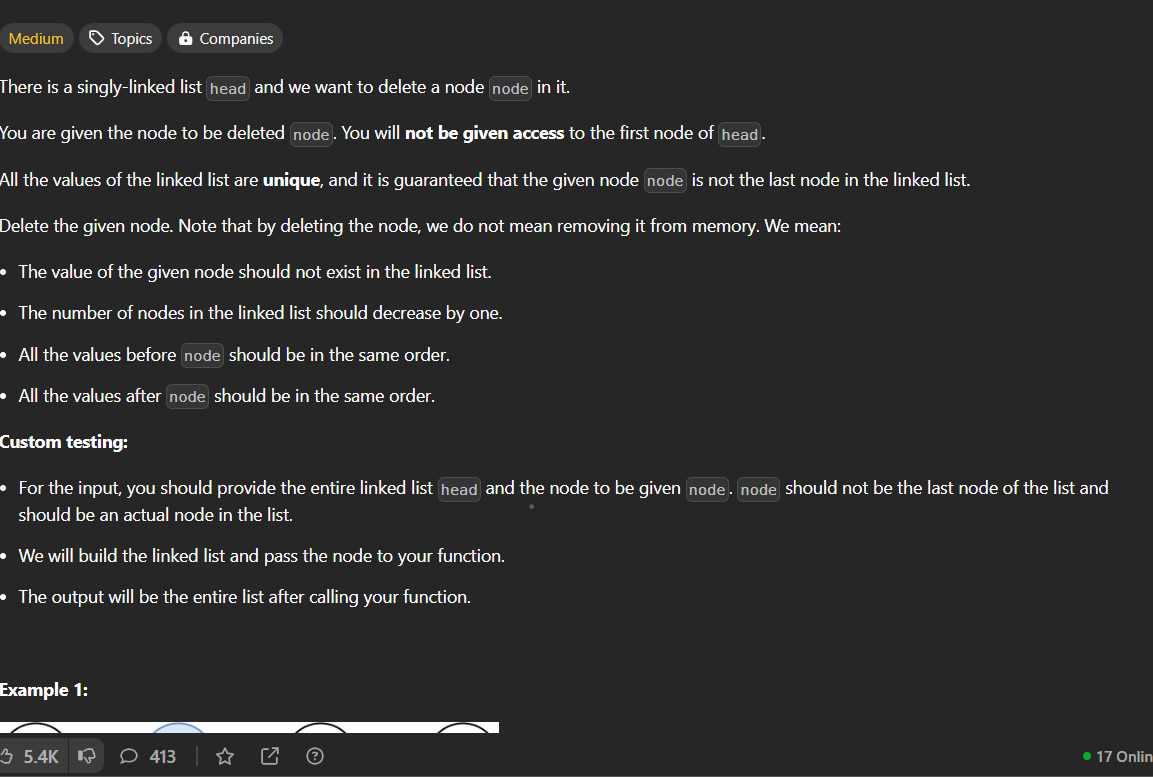
10. Reverse Alternate K Nodes in Linked List  
Problem: Reverse every alternate k nodes in the linked list.  
Input: head = 1 -> 2 -> 3 -> 4 -> 5, k = 2  
Expected Output: 2 -> 1 -> 4 -> 3 -> 5

LEETCODE:

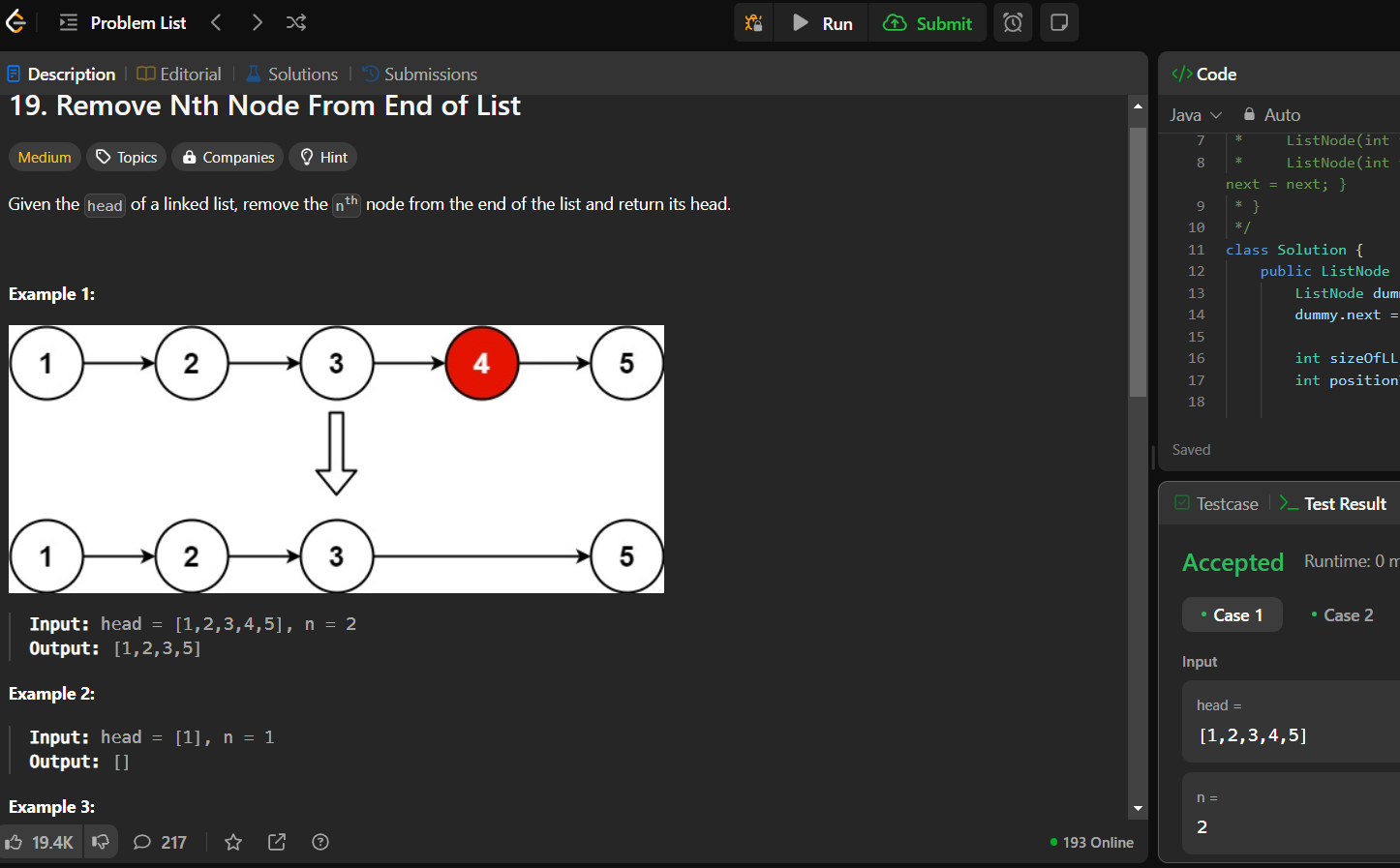
## Remove Linked List Element.



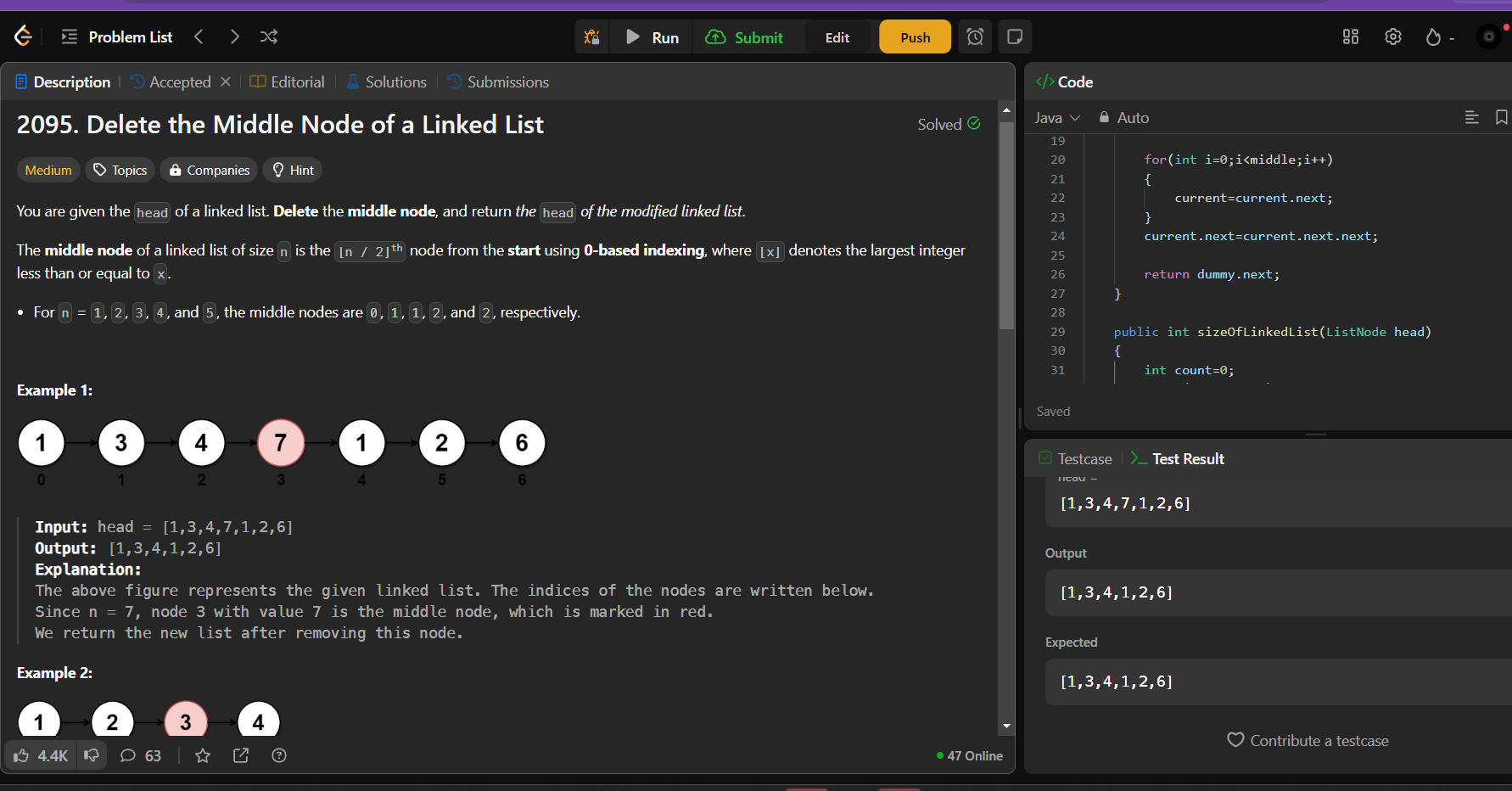
# DELETE A NODE



## 3.Remove nth element from end:



## 4.Delete middle Node in linked list



# 5. Middle of Linked List.

