Assignment 3

Student Name: Nitin Arora **UID:** 22BCS12542
Branch: BE-CSE **Section/Group:** 22BCS_IOT-607-B

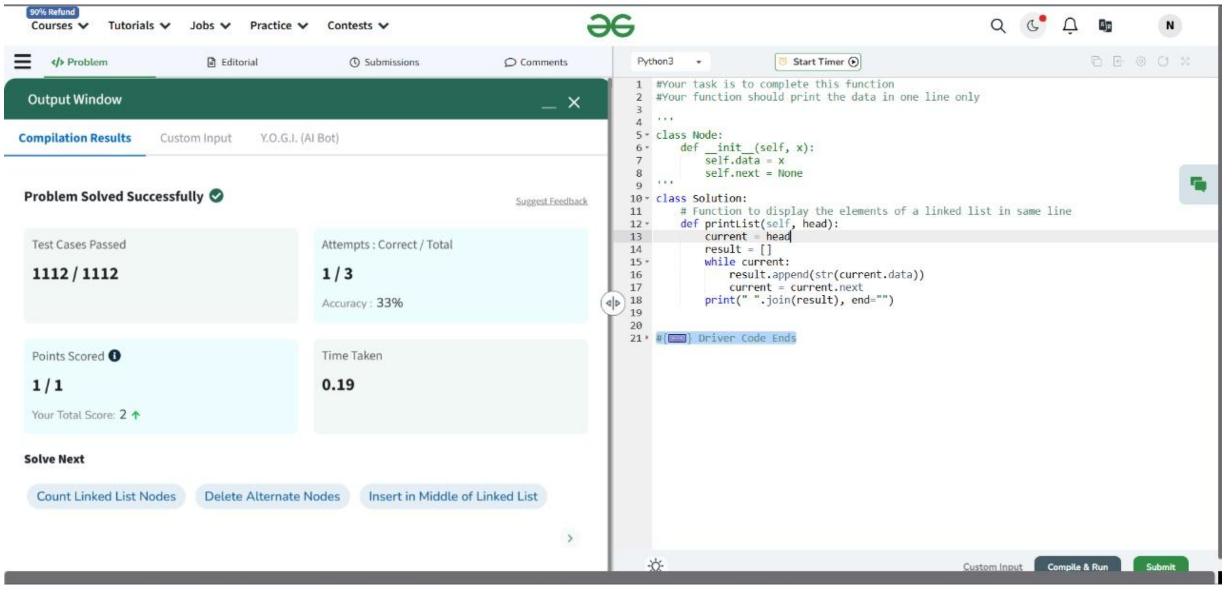
Question 1. Print Linked List

Problem Link: https://www.geeksforgeeks.org/problems/print-linked-list-elements/0

Code:

```
class Solution:
    def printList(self, head):
        current = head
        result = []
        while current:
            result.append(str(current.data))
            current = current.next
        print(" ".join(result))
```

Output:



Problem Solved Successfully

Test Cases Passed
1112 / 1112

Attempts : Correct / Total
1 / 3
Accuracy : 33%

Points Scored
1 / 1
Your Total Score: 2

Time Taken
0.19

Solve Next
Count Linked List Nodes | Delete Alternate Nodes | Insert in Middle of Linked List

```
#Your task is to complete this function
#Your function should print the data in one line only

'''
class Node:
    def __init__(self, x):
        self.data = x
        self.next = None
'''
class Solution:
    # Function to display the elements of a linked list in same line
    def printList(self, head):
        current = head
        result = []
        while current:
            result.append(str(current.data))
            current = current.next
        print(" ".join(result), end="")


#{ Driver Code Ends
```

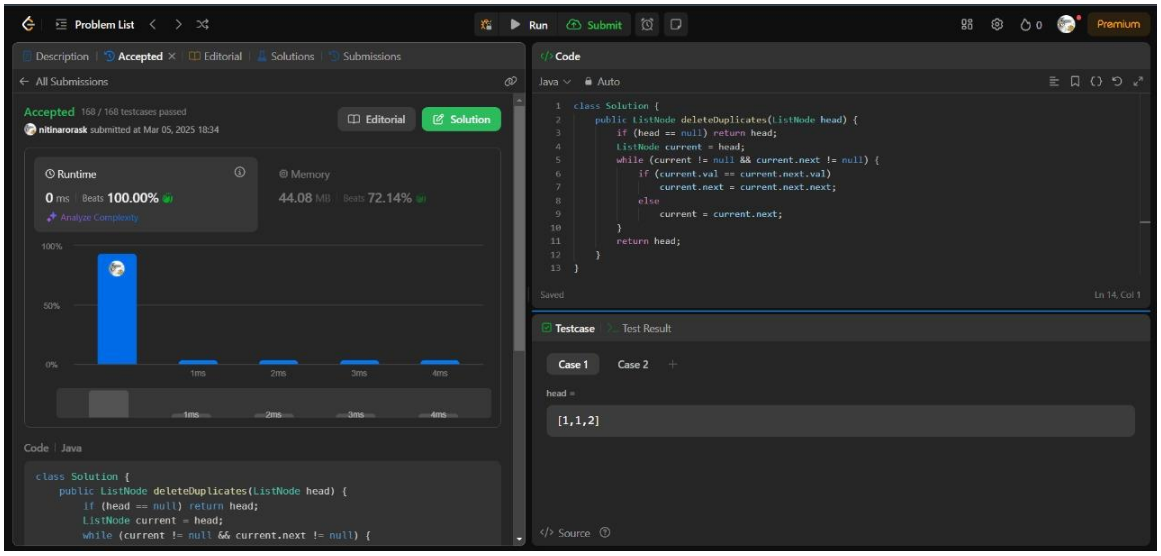
Question 2. Remove duplicates from a sorted list

Problem Link: https://leetcode.com/problems/remove-duplicates-from-sorted-list/description/

Code:

```
class Solution {
    public ListNode deleteDuplicates(ListNode head) {
        ListNode current = head;
        while (current != null && current.next != null) {
            if (current.val == current.next.val) {
                current.next = current.next.next;
            } else {
                current = current.next; }}
        return head; }}
```

Output:



Problem List · Run · Submit · Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 168 / 168 testcases passed
nitinarorask submitted at Mar 05, 2025 18:34

Runtime: 0 ms | Beats 100.00%
Memory: 44.08 MB | Beats 72.14%

Code | Java

```
class Solution {
    public ListNode deleteDuplicates(ListNode head) {
        if (head == null) return head;
        ListNode current = head;
        while (current != null && current.next != null) {
```

Code — Java — Auto

```
class Solution {
    public ListNode deleteDuplicates(ListNode head) {
        if (head == null) return head;
        ListNode current = head;
        while (current != null && current.next != null) {
            if (current.val == current.next.val)
                current.next = current.next.next;
            else
                current = current.next;
        }
        return head;
    }
}
```

Saved · Ln 14, Col 1

Testcase | Test Result

Case 1 · Case 2

```
head =
[1,1,2]
```

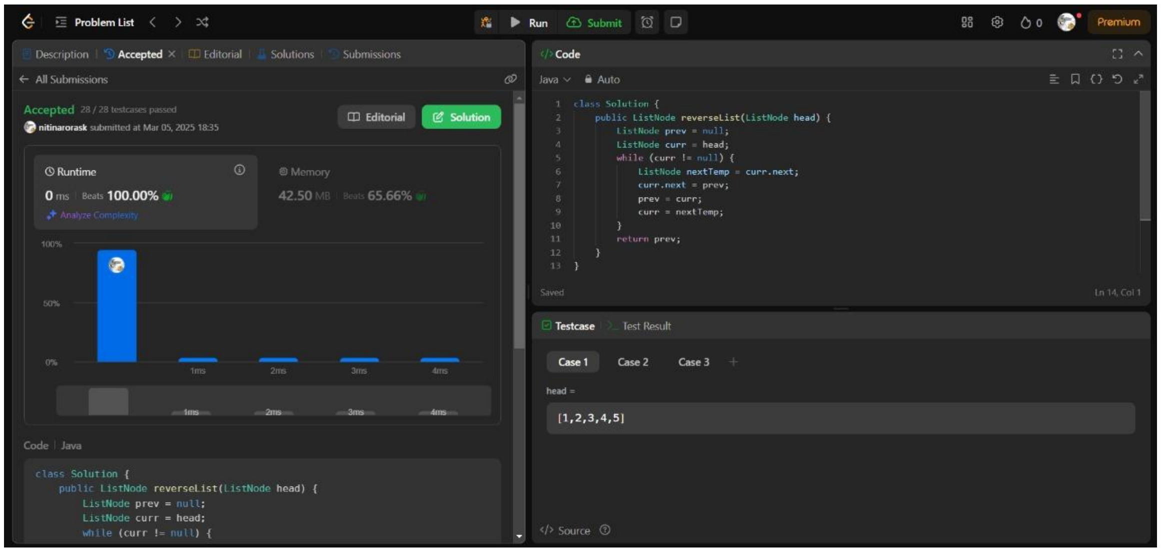
Question 3. Reverse a linked list

Problem Link: https://leetcode.com/problems/reverse-linked-list/description/

Code:

```
class Solution {
    public ListNode reverseList(ListNode head) {
        ListNode prev = null, curr = head, next = null;
        while (curr != null) {
            next = curr.next;
            curr.next = prev;
            prev = curr;
            curr = next; }
        return prev; }}
```

Output:



Accepted 28 / 28 testcases passed
nitinarorask submitted at Mar 05, 2025 18:35

Runtime 0 ms | Beats 100.00%
Memory 42.50 MB | Beats 65.66%

```
class Solution {
    public ListNode reverseList(ListNode head) {
        ListNode prev = null;
        ListNode curr = head;
        while (curr != null) {
            ListNode nextTemp = curr.next;
            curr.next = prev;
            prev = curr;
            curr = nextTemp;
        }
        return prev;
    }
}
```

Testcase — Case 1

head =

```
[1,2,3,4,5]
```

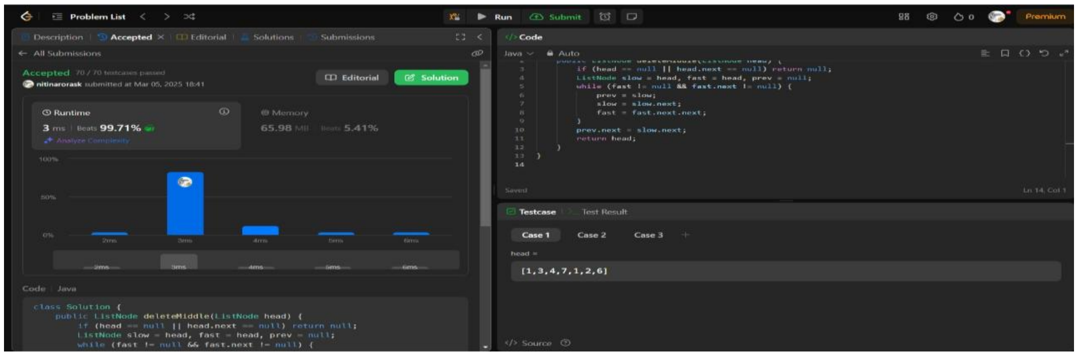
Question 4. Delete middle node of a list

Problem Link: https://leetcode.com/problems/delete-the-middle-node-of-a-linked-list/description/

Code:

```
class Solution {
    public ListNode deleteMiddle(ListNode head) {
        if (head == null || head.next == null) return null;
        ListNode slow = head, fast = head, prev = null;
        while (fast != null && fast.next != null) {
            prev = slow;
            slow = slow.next;
            fast = fast.next.next; }
        prev.next = slow.next;
        return head; }}
```

Output:



Accepted 70 / 70 testcases passed
nitinarorask submitted at Mar 05, 2025 18:41

Runtime 3 ms | Beats 99.71%
Memory 65.98 MB | Beats 5.41%

```
class Solution {
    public ListNode deleteMiddle(ListNode head) {
        if (head == null || head.next == null) return null;
        ListNode slow = head, fast = head, prev = null;
        while (fast != null && fast.next != null) {
            prev = slow;
            slow = slow.next;
            fast = fast.next.next;
        }
        prev.next = slow.next;
        return head;
    }
}
```

Testcase — Case 1

head =

```
[1,3,4,7,1,2,6]
```

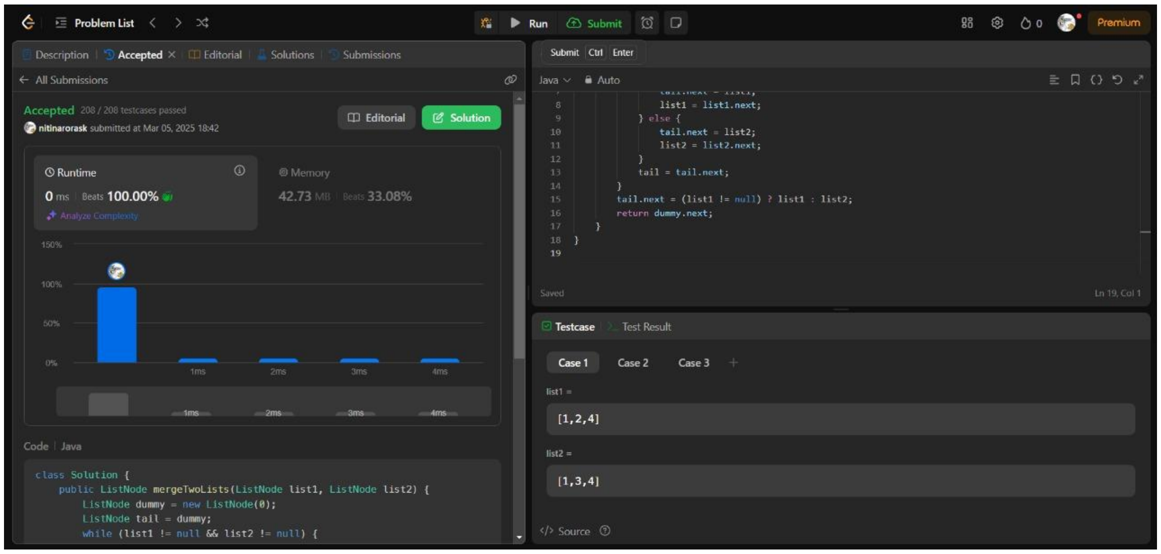
Question 5. Merge two sorted linked lists

Problem Link: https://leetcode.com/problems/merge-two-sorted-lists/description/

Code:

```
class Solution {
    public ListNode mergeTwoLists(ListNode list1, ListNode list2) {
        if (list1 == null) return list2;
        if (list2 == null) return list1;
        if (list1.val < list2.val) {
            list1.next = mergeTwoLists(list1.next, list2);
            return list1;
        } else {
            list2.next = mergeTwoLists(list1, list2.next);
            return list2; }}}
```

Output:



Accepted 208 / 208 testcases passed
nitinarorask submitted at Mar 05, 2025 18:42

Runtime
0 ms | Beats 100.00%

Memory
42.73 MB | Beats 33.08%

Code | Java

```
class Solution {
    public ListNode mergeTwoLists(ListNode list1, ListNode list2) {
        ListNode dummy = new ListNode(0);
        ListNode tail = dummy;
        while (list1 != null && list2 != null) {
```

```
                list1 = list1.next;
            } else {
                tail.next = list2;
                list2 = list2.next;
            }
            tail = tail.next;
        }
        tail.next = (list1 != null) ? list1 : list2;
        return dummy.next;
    }
}
```

Saved
Ln 19, Col 1

list1 =
[1,2,4]

list2 =
[1,3,4]

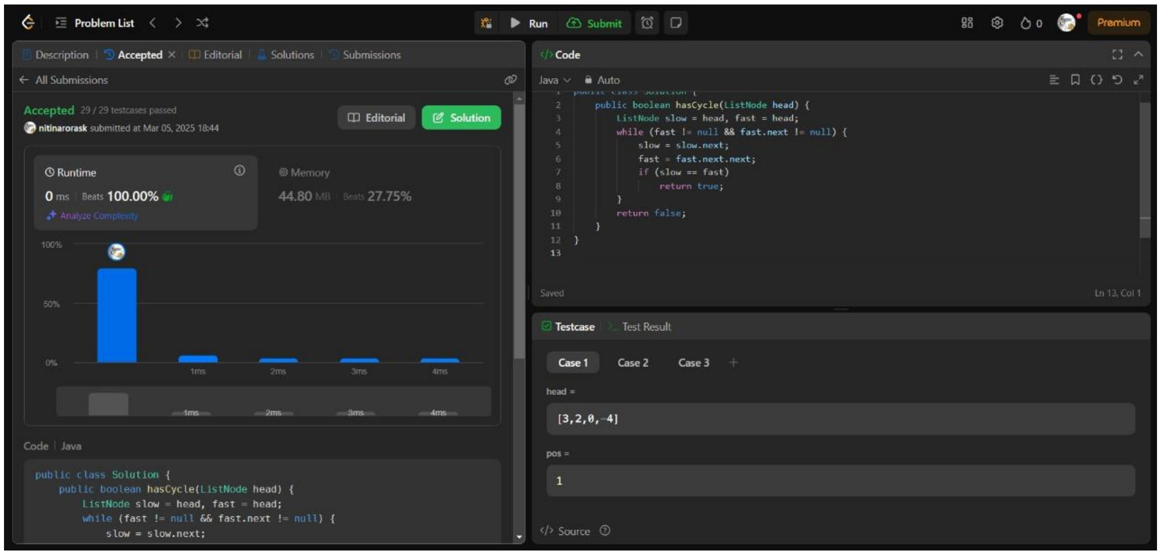
Question 6. Detect a cycle in a linked list

Problem Link: https://leetcode.com/problems/linked-list-cycle/description/

Code:

```
public class Solution {
    public boolean hasCycle(ListNode head) {
        ListNode slow = head, fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
            fast = fast.next.next;
            if (slow == fast) return true;
        }
        return false;
    }
}
```

Output:



Problem List · Run · Submit · Premium

Description | Accepted | Editorial | Solutions | Submissions

← All Submissions

Accepted 29 / 29 testcases passed
nitinarorask submitted at Mar 05, 2025 18:44

Editorial · Solution

Runtime
0 ms | Beats 100.00%
Analyze Complexity

Memory
44.80 MB | Beats 27.75%

Code | Java

```
public class Solution {
    public boolean hasCycle(ListNode head) {
        ListNode slow = head, fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
```

Code

Java · Auto

```
    public boolean hasCycle(ListNode head) {
        ListNode slow = head, fast = head;
        while (fast != null && fast.next != null) {
            slow = slow.next;
            fast = fast.next.next;
            if (slow == fast)
                return true;
        }
        return false;
    }
}
```

Saved · Ln 13, Col 1

Testcase | Test Result

Case 1 · Case 2 · Case 3

head =

```
[3,2,0,-4]
```

pos =

```
1
```

Source

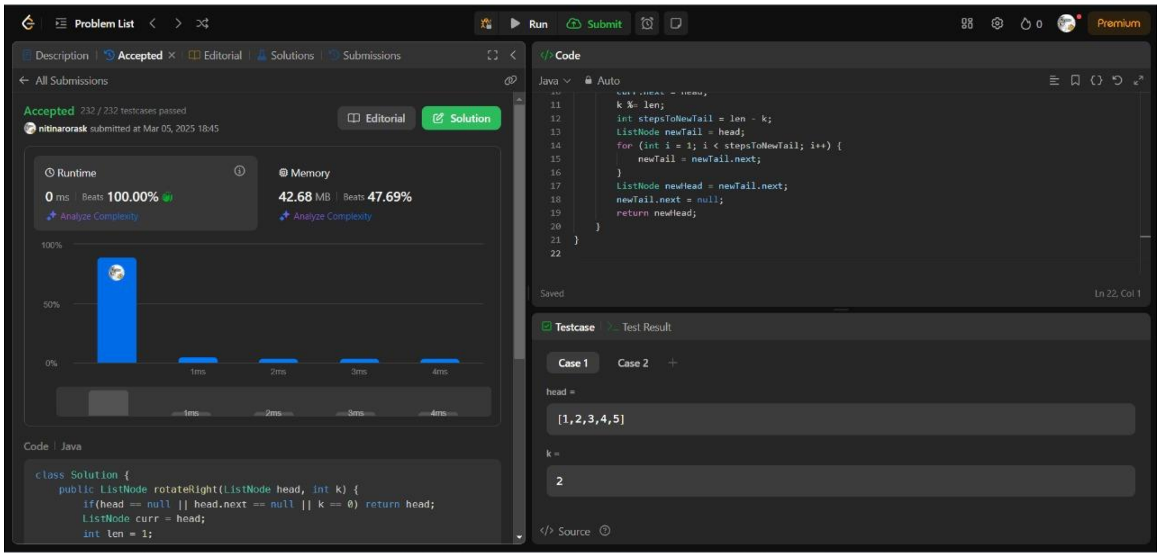
Question 7. Rotate a list

Problem Link: https://leetcode.com/problems/rotate-list/description/

Code:

```
class Solution {
    public ListNode rotateRight(ListNode head, int k) {
        if (head == null || head.next == null || k == 0) return head;
        ListNode temp = head;
        int len = 1;
        while (temp.next != null) {
            temp = temp.next;
            len++;
        }
        temp.next = head;
        k = k % len;
        int steps = len - k;
        while (steps-- > 0) temp = temp.next;
        head = temp.next;
        temp.next = null;
        return head;
    }
}
```

Output:



Accepted 232 / 232 testcases passed
nitinarorask submitted at Mar 05, 2025 18:45

Runtime
0 ms | Beats 100.00%

Memory
42.68 MB | Beats 47.69%

Code | Java

```
class Solution {
    public ListNode rotateRight(ListNode head, int k) {
        if(head == null || head.next == null || k == 0) return head;
        ListNode curr = head;
        int len = 1;
```

```
        k %= len;
        int stepsToNewTail = len - k;
        ListNode newTail = head;
        for (int i = 1; i < stepsToNewTail; i++) {
            newTail = newTail.next;
        }
        ListNode newHead = newTail.next;
        newTail.next = null;
        return newHead;
    }
}
```

head =

```
[1,2,3,4,5]
```

k =

```
2
```

Question 8. Sort List

Problem Link: https://leetcode.com/problems/sort-list/description/

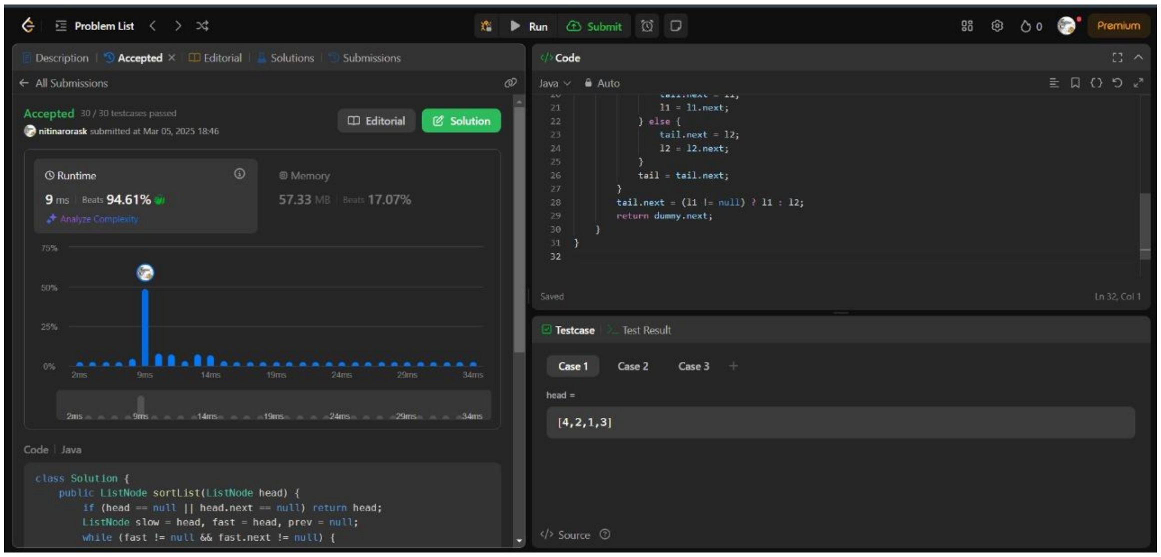
Code:

```
class Solution {
    public ListNode sortList(ListNode head) {
        if (head == null || head.next == null) return head;
        ListNode mid = getMid(head);
        ListNode left = sortList(head);
        ListNode right = sortList(mid);
        return merge(left, right);
    }

    private ListNode getMid(ListNode head) {
        ListNode midPrev = null;
        while (head != null && head.next != null) {
            midPrev = (midPrev == null) ? head : midPrev.next;
            head = head.next.next;
        }
        ListNode mid = midPrev.next;
        midPrev.next = null;
        return mid;
    }

    private ListNode merge(ListNode list1, ListNode list2) {
        ListNode dummyHead = new ListNode();
        ListNode current = dummyHead;
        while (list1 != null && list2 != null) {
            if (list1.val < list2.val) {
                current.next = list1;
                list1 = list1.next;
            } else {
                current.next = list2;
                list2 = list2.next;
            }
            current = current.next;
        }
        current.next = (list1 != null) ? list1 : list2;
        return dummyHead.next;
    }
}
```

Output:



Accepted 30 / 30 testcases passed
nitinarorask submitted at Mar 05, 2025 18:46

Runtime: 9 ms | Beats 94.61%
Memory: 57.33 MB | Beats 17.07%

Code | Java

```
class Solution {
    public ListNode sortList(ListNode head) {
        if (head == null || head.next == null) return head;
        ListNode slow = head, fast = head, prev = null;
        while (fast != null && fast.next != null) {
```

```
                l1 = l1.next;
            } else {
                tail.next = l2;
                l2 = l2.next;
            }
            tail = tail.next;
        }
        tail.next = (l1 != null) ? l1 : l2;
        return dummy.next;
    }
}
```

head =
[4,2,1,3]

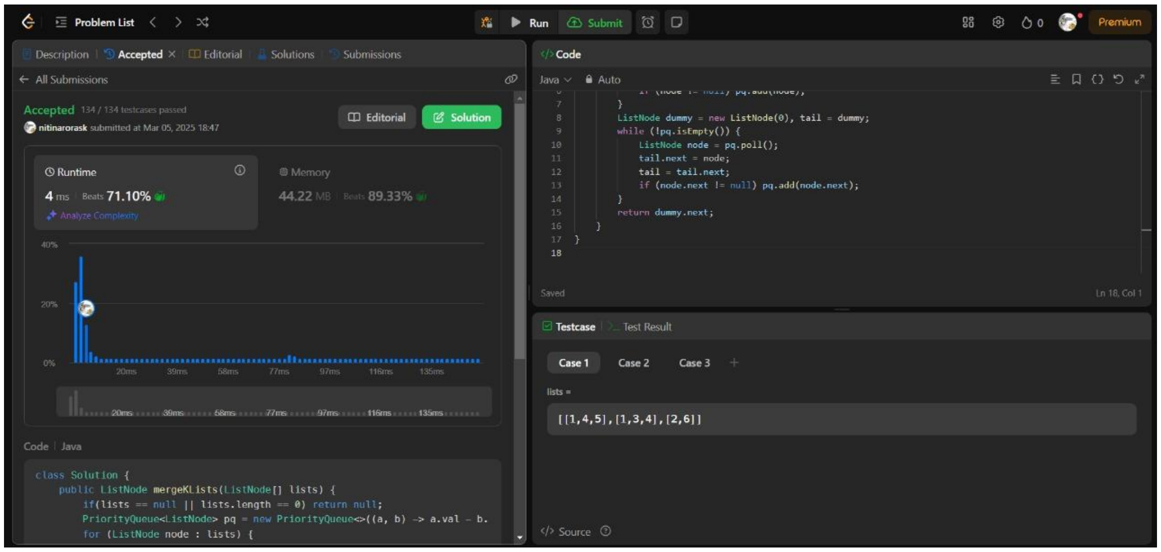
Question 9. Merge k sorted lists

Problem Link: https://leetcode.com/problems/merge-k-sorted-lists/description/

Code:

```
class Solution {
   public ListNode mergeKLists(ListNode[] lists) {
      PriorityQueue<ListNode> pq = new PriorityQueue<>((a, b) -> a.val - b.val);
      for (ListNode node : lists) {
         if (node != null) pq.offer(node);
      }
      ListNode dummy = new ListNode(0), tail = dummy;
      while (!pq.isEmpty()) {
         tail.next = pq.poll();
         tail = tail.next;
         if (tail.next != null) pq.offer(tail.next);
      }
      return dummy.next;
   }
}
```

Output:



Accepted 134 / 134 testcases passed
nitinarorask submitted at Mar 05, 2025 18:47

Runtime
4 ms | Beats 71.10%

Memory
44.22 MB | Beats 89.33%

Code | Java

```
class Solution {
    public ListNode mergeKLists(ListNode[] lists) {
        if(lists == null || lists.length == 0) return null;
        PriorityQueue<ListNode> pq = new PriorityQueue<>((a, b) -> a.val - b.
        for (ListNode node : lists) {
```

```
            }
        ListNode dummy = new ListNode(0), tail = dummy;
        while (!pq.isEmpty()) {
            ListNode node = pq.poll();
            tail.next = node;
            tail = tail.next;
            if (node.next != null) pq.add(node.next);
        }
        return dummy.next;
    }
}
```

Saved
Ln 18, Col 1

lists =

```
[[1,4,5],[1,3,4],[2,6]]
```