Experiment 3

Student Name: Ansh **UID: 22BCS13469**
Branch: CSE **Section/Group: 637-B**
Semester: 6th **Date of Performance:7/3/25**
Subject Name: Advanced Programming - 2 **Subject Code: 22CSH-351**

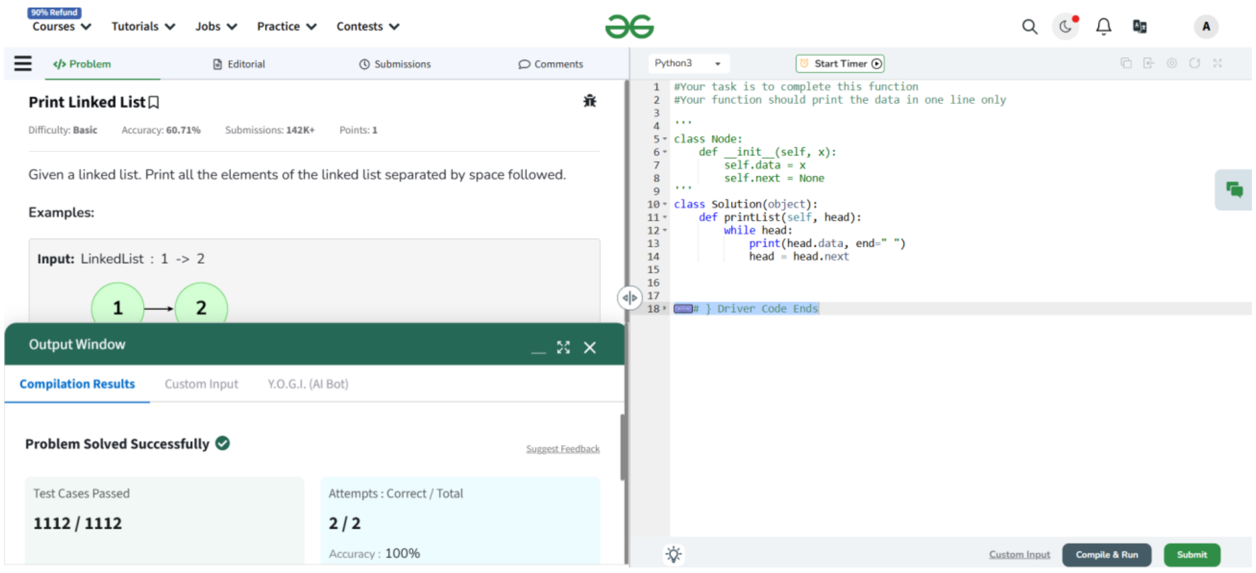
Ques 1:

Aim: Print Linked List:

Code:

```
class Solution(object):
    def printList(self, head):
        while head:
            print(head.data, end=" ")
            head = head.next
```

Submission Screenshot:



Print Linked List

Difficulty: Basic Accuracy: 60.71% Submissions: 142K+ Points: 1

Given a linked list. Print all the elements of the linked list separated by space followed.

Examples:

Input: LinkedList : 1 -> 2

```
#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(object):
    def printList(self, head):
        while head:
            print(head.data, end=" ")
            head = head.next



# } Driver Code Ends
```

Problem Solved Successfully

Test Cases Passed
1112 / 1112

Attempts : Correct / Total
2 / 2
Accuracy : 100%

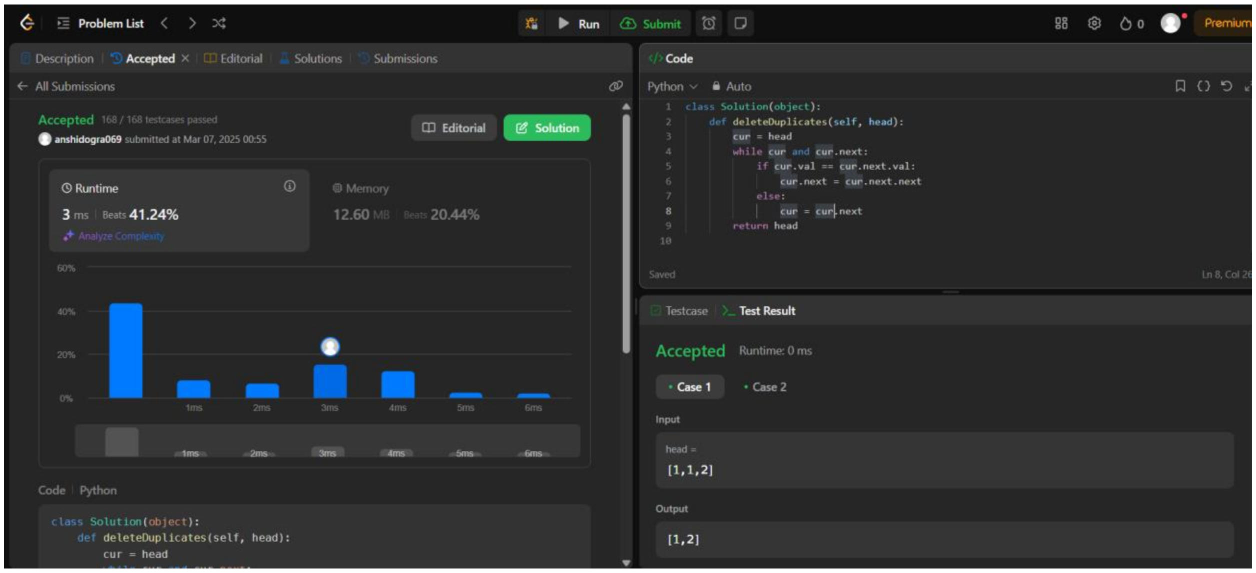
Ques 2:

Aim: Remove duplicates from a sorted list:

Code:

```
class Solution(object):
    def deleteDuplicates(self, head):
        cur = head
        while cur and cur.next:
            if cur.val == cur.next.val:
                cur.next = cur.next.next
            else:
                cur = cur.next
        return head
```

Submission Screenshot:



Accepted 168 / 168 testcases passed
anshidogra069 submitted at Mar 07, 2025 00:55

Runtime: 3 ms | Beats 41.24%
Memory: 12.60 MB | Beats 20.44%

```
class Solution(object):
    def deleteDuplicates(self, head):
        cur = head
        while cur and cur.next:
            if cur.val == cur.next.val:
                cur.next = cur.next.next
            else:
                cur = cur.next
        return head
```

Saved Ln 8, Col 26

Accepted Runtime: 0 ms

Input

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

Output

```
[1,2]
```

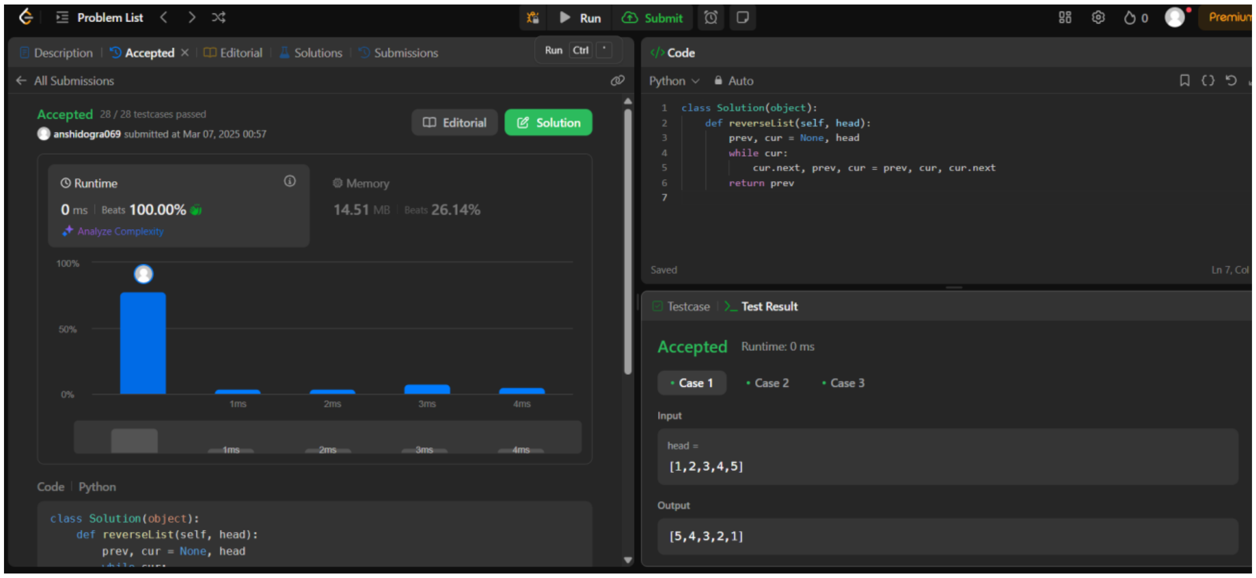
Ques 3:

Aim: Reverse a linked list:

Code:

```
class Solution(object):
    def reverseList(self, head):
        prev, cur = None, head
        while cur:
            cur.next, prev, cur = prev, cur, cur.next
        return prev
```

Submission Screenshot:



Accepted 28 / 28 testcases passed
anshidogra069 submitted at Mar 07, 2025 00:57

Runtime: 0 ms | Beats 100.00%
Memory: 14.51 MB | Beats 26.14%

Code | Python

```
class Solution(object):
    def reverseList(self, head):
        prev, cur = None, head
        while cur:
            cur.next, prev, cur = prev, cur, cur.next
        return prev
```

Accepted Runtime: 0 ms

Input

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

Output

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

Ques 4:

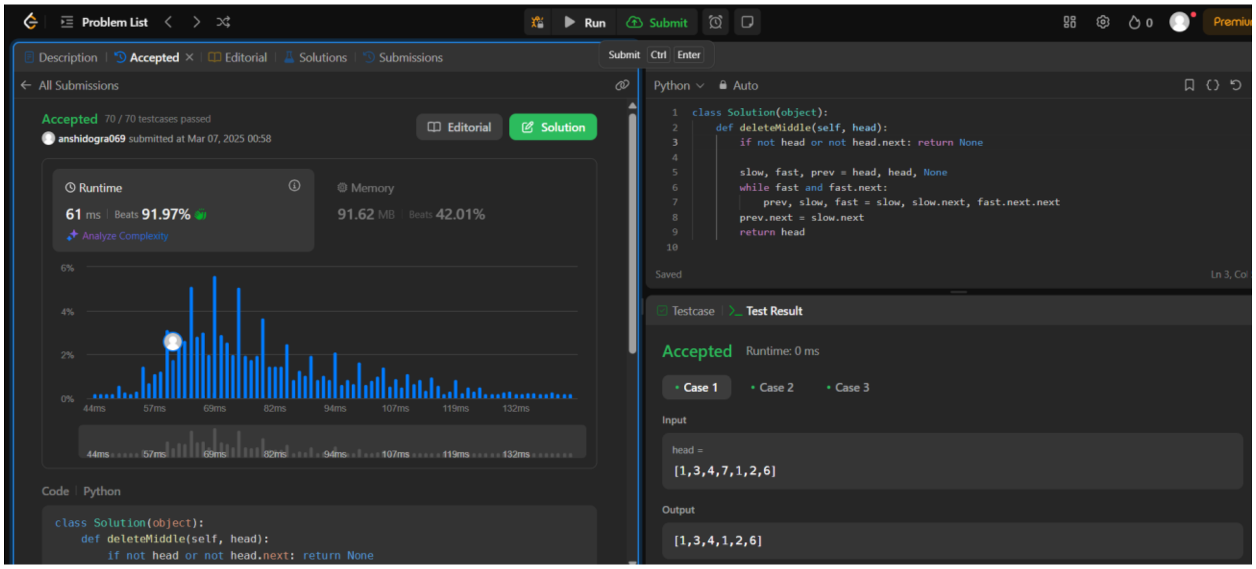
Aim: Delete middle node of a linked list:

Code:

```
class Solution(object):
    def deleteMiddle(self, head):
        if not head or not head.next: return None

        slow, fast, prev = head, head, None
        while fast and fast.next:
            prev, slow, fast = slow, slow.next, fast.next.next
        prev.next = slow.next
        return head
```

Submission Screenshot:



Accepted 70 / 70 testcases passed
anshidogra069 submitted at Mar 07, 2025 00:58

Runtime
61 ms | Beats 91.97%

Memory
91.62 MB | Beats 42.01%

Code | Python

```
class Solution(object):
    def deleteMiddle(self, head):
        if not head or not head.next: return None

        slow, fast, prev = head, head, None
        while fast and fast.next:
            prev, slow, fast = slow, slow.next, fast.next.next
        prev.next = slow.next
        return head
```

Saved
Ln 3, Col

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 · Case 2 · Case 3

Input

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

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

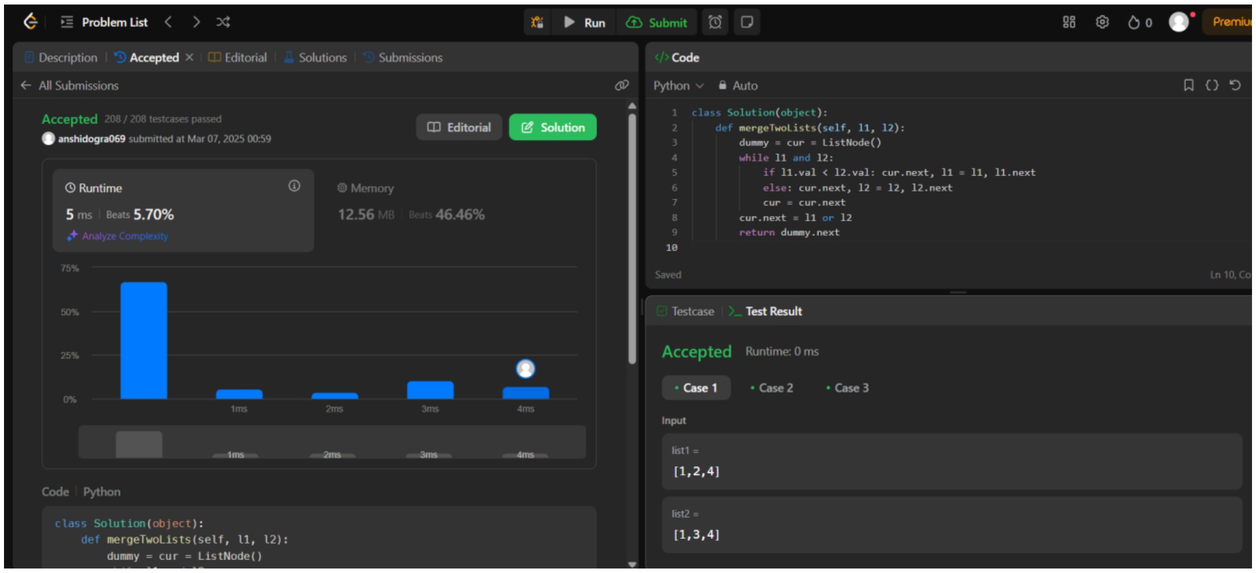
Ques 5:

Aim: Merge two sorted linked lists:

Code:

```
class Solution(object):
    def mergeTwoLists(self, l1, l2):
        dummy = cur = ListNode()
        while l1 and l2:
            if l1.val < l2.val: cur.next, l1 = l1, l1.next
            else: cur.next, l2 = l2, l2.next
            cur = cur.next
        cur.next = l1 or l2
        return dummy.next
```

Submission Screenshot:



Accepted 208 / 208 testcases passed
anshidogra069 submitted at Mar 07, 2025 00:59

Runtime
5 ms | Beats 5.70%

Memory
12.56 MB | Beats 46.46%

```
class Solution(object):
    def mergeTwoLists(self, l1, l2):
        dummy = cur = ListNode()
        while l1 and l2:
            if l1.val < l2.val: cur.next, l1 = l1, l1.next
            else: cur.next, l2 = l2, l2.next
            cur = cur.next
        cur.next = l1 or l2
        return dummy.next
```

Accepted Runtime: 0 ms

```
list1 =
[1,2,4]
list2 =
[1,3,4]
```

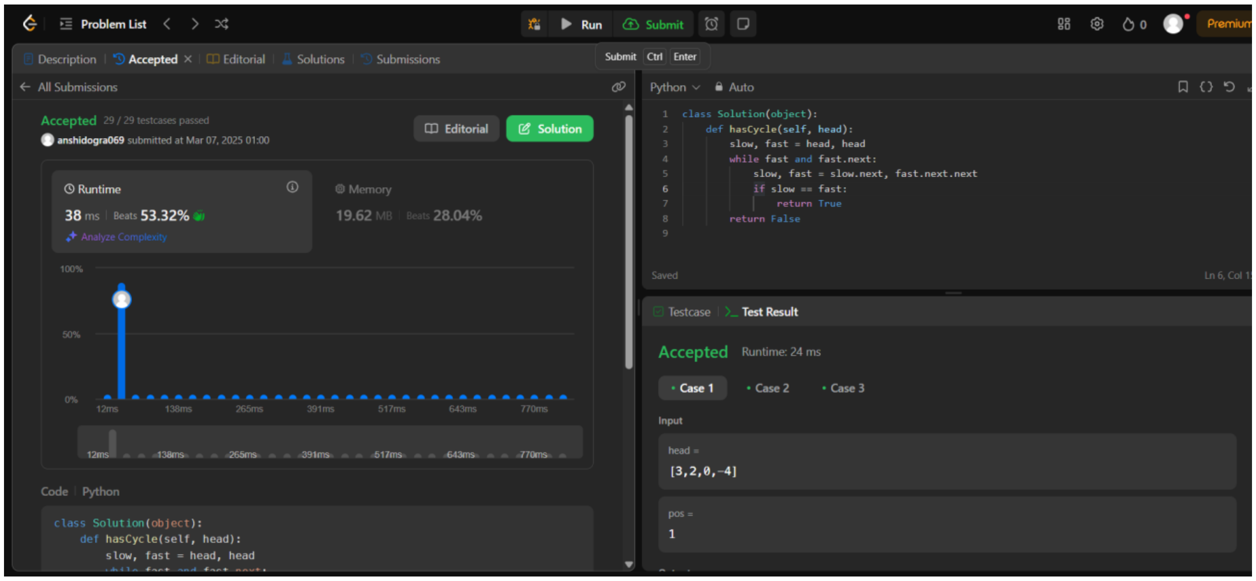
Ques 6:

Aim: Detect a cycle in a linked list:

Code:

```
class Solution(object):
    def hasCycle(self, head):
        slow, fast = head, head
        while fast and fast.next:
            slow, fast = slow.next, fast.next.next
            if slow == fast:
                return True
        return False
```

Submission Screenshot:



Accepted 29 / 29 testcases passed
anshidogra069 submitted at Mar 07, 2025 01:00

Runtime
38 ms | Beats 53.32%

Memory
19.62 MB | Beats 28.04%

Code | Python

```
class Solution(object):
    def hasCycle(self, head):
        slow, fast = head, head
        while fast and fast.next:
            slow, fast = slow.next, fast.next.next
            if slow == fast:
                return True
        return False
```

Accepted Runtime: 24 ms

Input

head =
[3,2,0,-4]

pos =
1

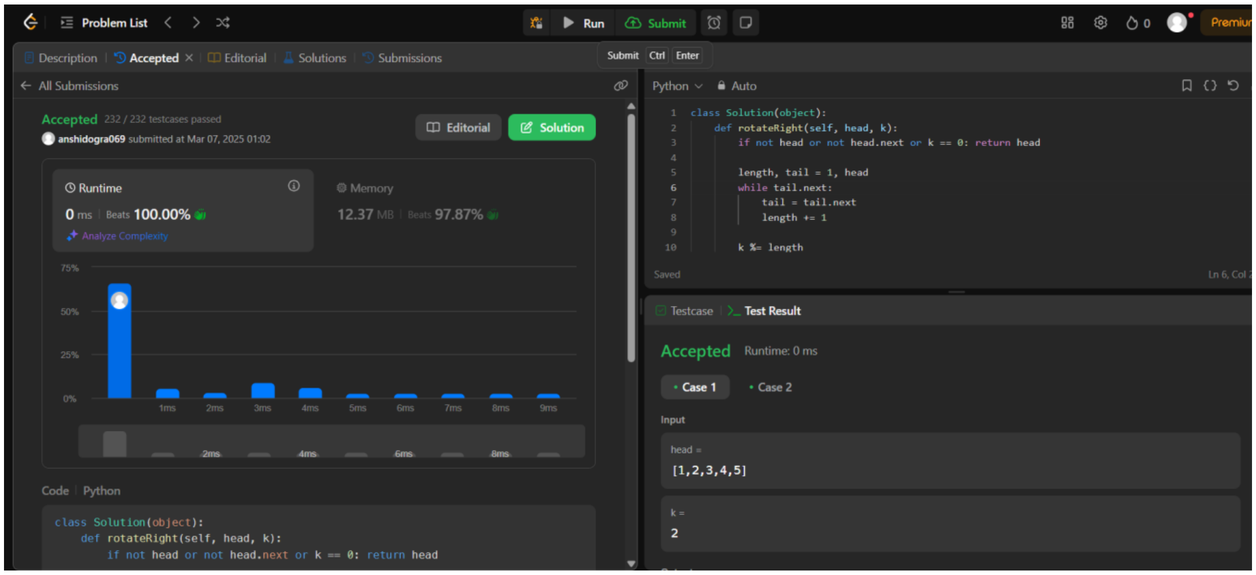
Ques 7:

Aim: Rotate a list:

Code:

```
class Solution(object):
    def rotateRight(self, head, k):
        if not head or not head.next or k == 0: return head
        length, tail = 1, head
        while tail.next:
            tail = tail.next
            length += 1
        k %= length
        if k == 0: return head
        new_tail = head
        for _ in range(length - k - 1):
            new_tail = new_tail.next
        new_head, new_tail.next, tail.next = new_tail.next, None, head
        return new_head
```

Submission Screenshot:



Accepted 232 / 232 testcases passed
anshidogra069 submitted at Mar 07, 2025 01:02

Runtime
0 ms | Beats 100.00%

Memory
12.37 MB | Beats 97.87%

Code | Python

```
class Solution(object):
    def rotateRight(self, head, k):
        if not head or not head.next or k == 0: return head
```

```
class Solution(object):
    def rotateRight(self, head, k):
        if not head or not head.next or k == 0: return head

        length, tail = 1, head
        while tail.next:
            tail = tail.next
            length += 1

        k %= length
```

Accepted Runtime: 0 ms

Input

head =
[1,2,3,4,5]

k =
2

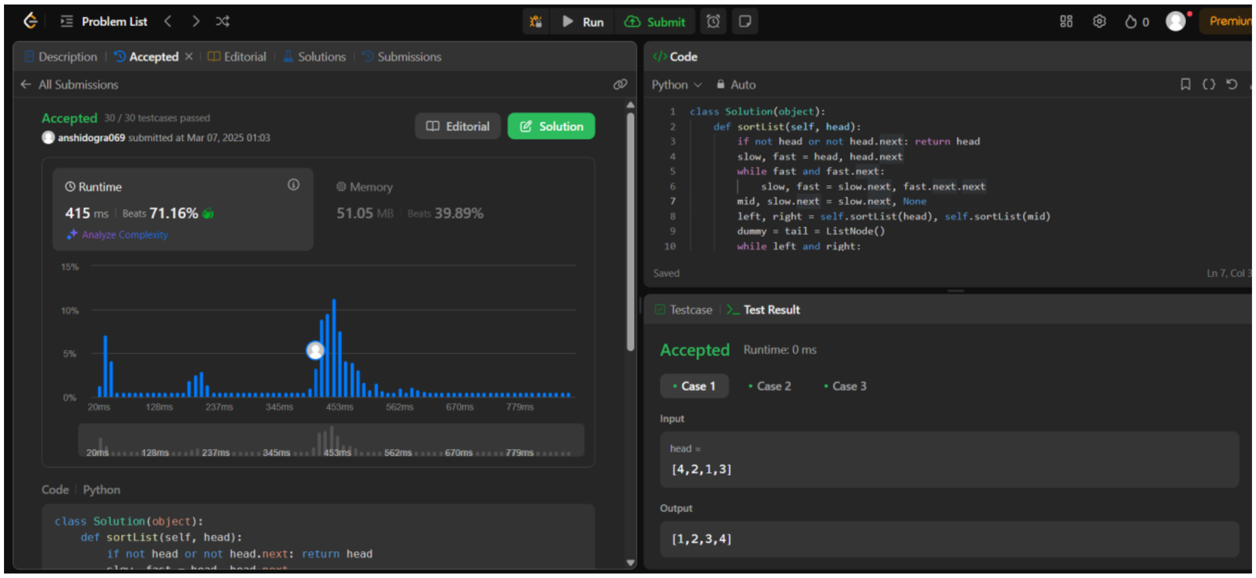
Ques 8:

Aim: Sort List:

Code:

```
class Solution(object):
    def sortList(self, head):
        if not head or not head.next: return head
        slow, fast = head, head.next
        while fast and fast.next:
            slow, fast = slow.next, fast.next.next
        mid, slow.next = slow.next, None
        left, right = self.sortList(head), self.sortList(mid)
        dummy = tail = ListNode()
        while left and right:
            if left.val < right.val:
                tail.next, left = left, left.next
            else:
                tail.next, right = right, right.next
            tail = tail.next
        tail.next = left or right
        return dummy.next
```

Submission Screenshot:



Accepted 30 / 30 testcases passed
anshidogra069 submitted at Mar 07, 2025 01:03

Runtime
415 ms | Beats 71.16%

Memory
51.05 MB | Beats 39.89%

Code | Python

```
class Solution(object):
    def sortList(self, head):
        if not head or not head.next: return head
```

```
class Solution(object):
    def sortList(self, head):
        if not head or not head.next: return head
        slow, fast = head, head.next
        while fast and fast.next:
            slow, fast = slow.next, fast.next.next
        mid, slow.next = slow.next, None
        left, right = self.sortList(head), self.sortList(mid)
        dummy = tail = ListNode()
        while left and right:
```

Accepted Runtime: 0 ms

Input
head =
[4,2,1,3]

Output
[1,2,3,4]

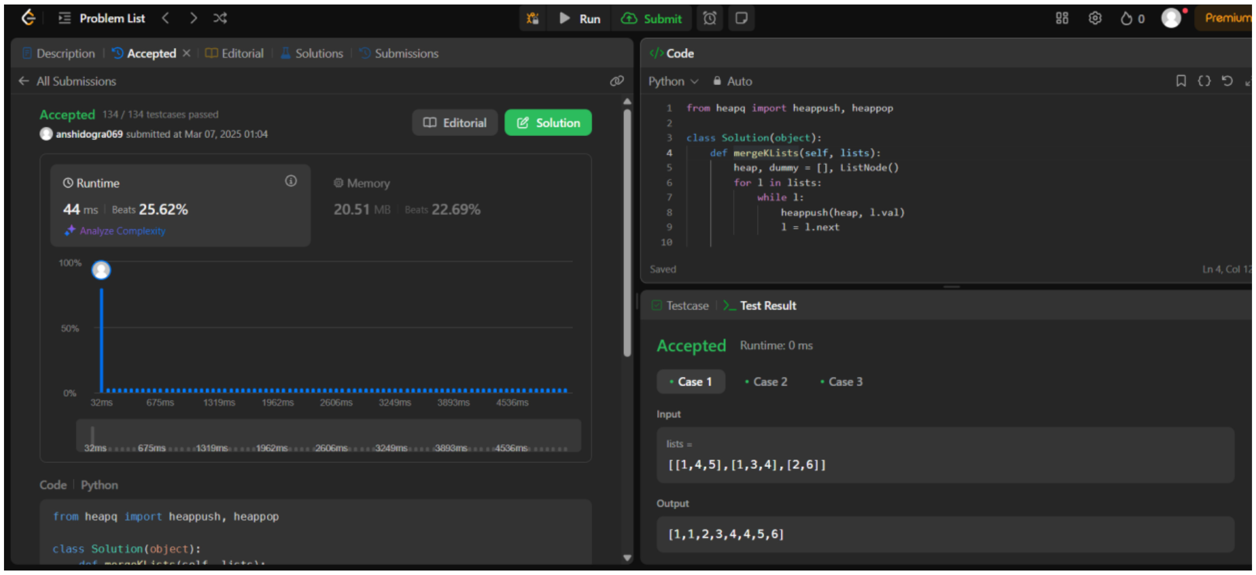
Ques 9:

Aim: Merge k sorted lists:

Code:

```
from heapq import heappush, heappop
class Solution(object):
    def mergeKLists(self, lists):
        heap, dummy = [], ListNode()
        for l in lists:
            while l:
                heappush(heap, l.val)
                l = l.next
        tail = dummy
        while heap:
            tail.next = ListNode(heappop(heap))
            tail = tail.next
        return dummy.next
```

Submission Screenshot:



Accepted 134 / 134 testcases passed
anshidogra069 submitted at Mar 07, 2025 01:04

Runtime 44 ms | Beats 25.62%
Memory 20.51 MB | Beats 22.69%

Code | Python

```
from heapq import heappush, heappop

class Solution(object):
```

```
from heapq import heappush, heappop

class Solution(object):
    def mergeKLists(self, lists):
        heap, dummy = [], ListNode()
        for l in lists:
            while l:
                heappush(heap, l.val)
                l = l.next
```

Accepted Runtime: 0 ms

Input

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

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