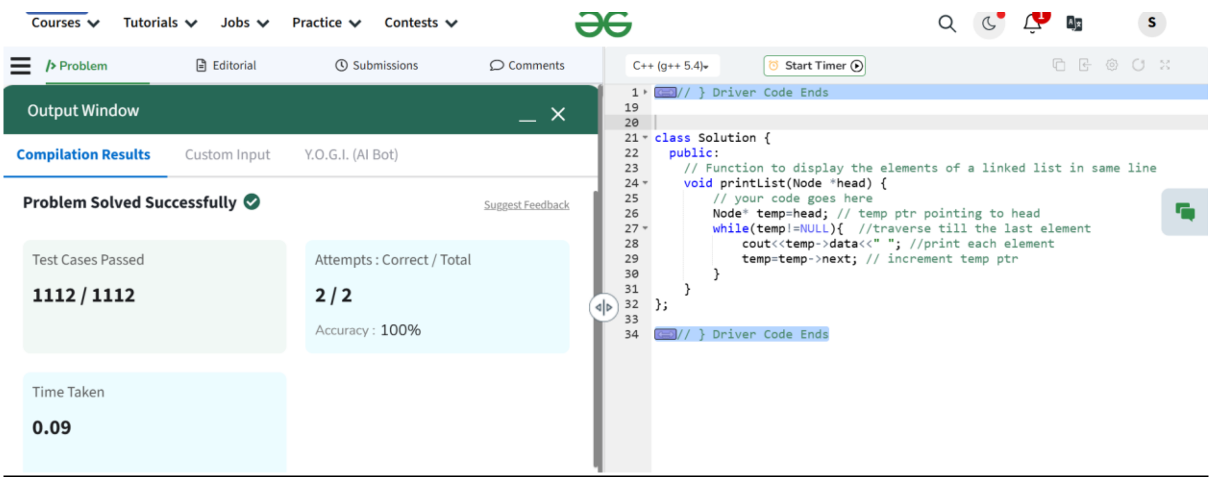
AP- ASSIGNMENT

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



Problem Solved Successfully

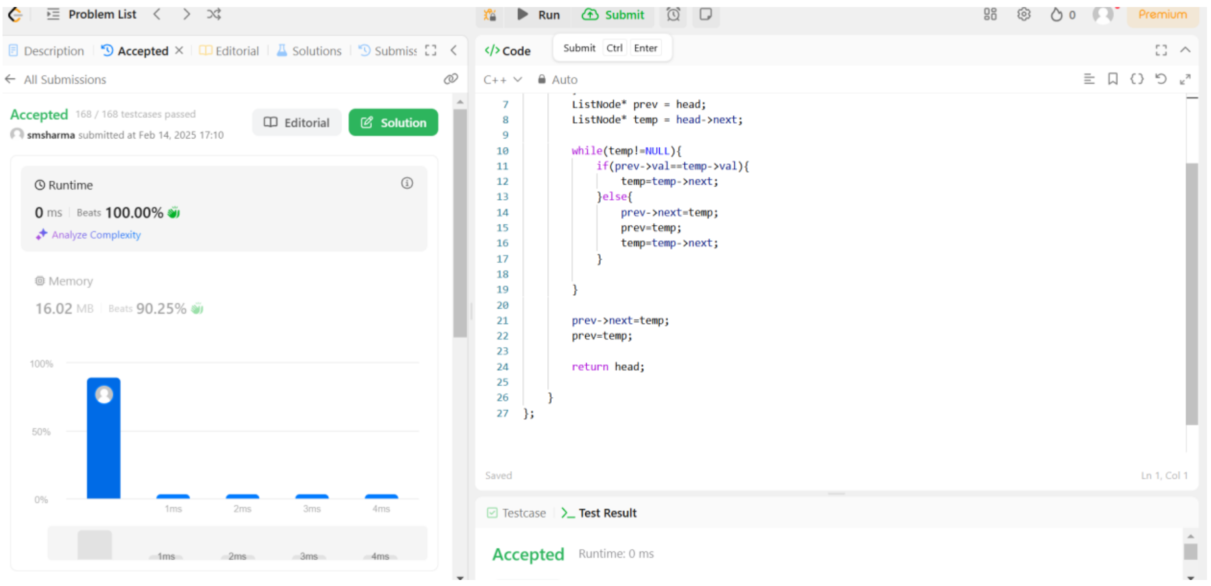
Test Cases Passed
1112 / 1112

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

Time Taken
0.09

```
// } Driver Code Ends


class Solution {
  public:
    // Function to display the elements of a linked list in same line
    void printList(Node *head) {
        // your code goes here
        Node* temp=head; // temp ptr pointing to head
        while(temp!=NULL){  //traverse till the last element
            cout<<temp->data<<" "; //print each element
            temp=temp->next; // increment temp ptr
        }
    }
};

// } Driver Code Ends
```

2. https://leetcode.com/problems/remove-duplicates-from-sorted-list/submissions/



Accepted 168 / 168 testcases passed
smsharma submitted at Feb 14, 2025 17:10

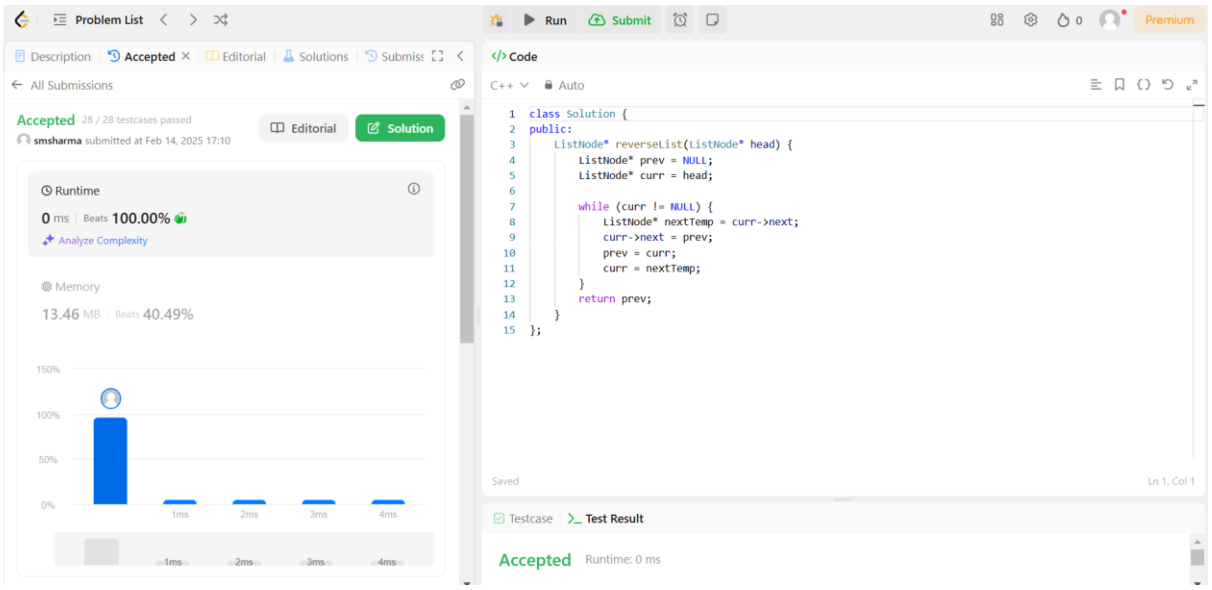
Runtime
0 ms | Beats 100.00%

Memory
16.02 MB | Beats 90.25%

```
        ListNode* prev = head;
        ListNode* temp = head->next;

        while(temp!=NULL){
            if(prev->val==temp->val){
                temp=temp->next;
            }else{
                prev->next=temp;
                prev=temp;
                temp=temp->next;
            }

        }

        prev->next=temp;
        prev=temp;

        return head;

    }
};
```

Accepted Runtime: 0 ms

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



Accepted 28 / 28 testcases passed

smsharma submitted at Feb 14, 2025 17:10

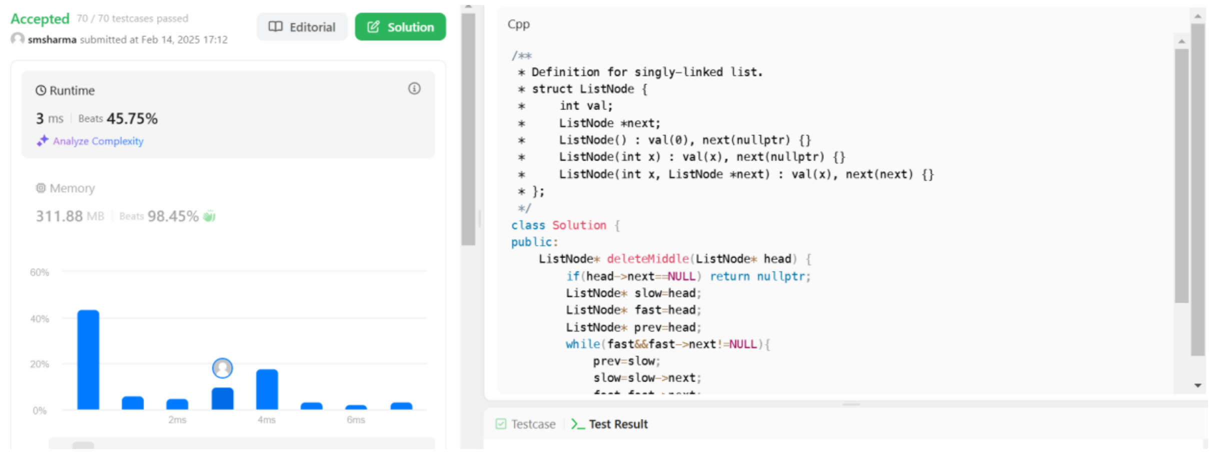
Runtime 0 ms Beats 100.00%

Memory 13.46 MB Beats 40.49%

```cpp
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;
    }
};
```

Accepted Runtime: 0 ms

4. https://leetcode.com/problems/delete-the-middle-node-of-a-linked-list/submissions/



Accepted 70 / 70 testcases passed

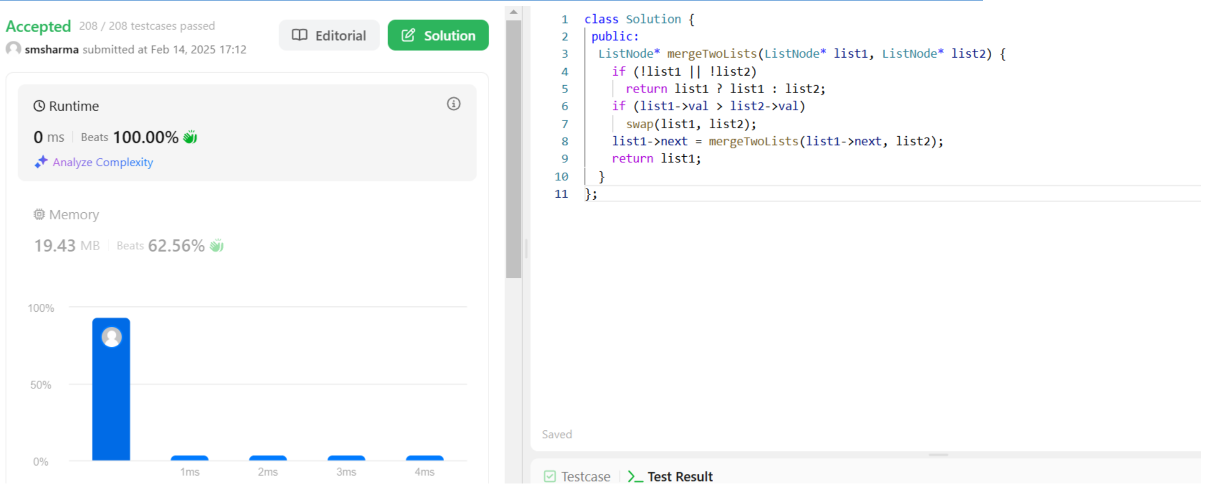
smsharma submitted at Feb 14, 2025 17:12

Runtime 3 ms Beats 45.75%

Memory 311.88 MB Beats 98.45%

```cpp
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* deleteMiddle(ListNode* head) {
        if(head->next==NULL) return nullptr;
        ListNode* slow=head;
        ListNode* fast=head;
        ListNode* prev=head;
        while(fast&&fast->next!=NULL){
            prev=slow;
            slow=slow->next;
```

5. https://leetcode.com/problems/merge-two-sorted-lists/submissions/1542046736/

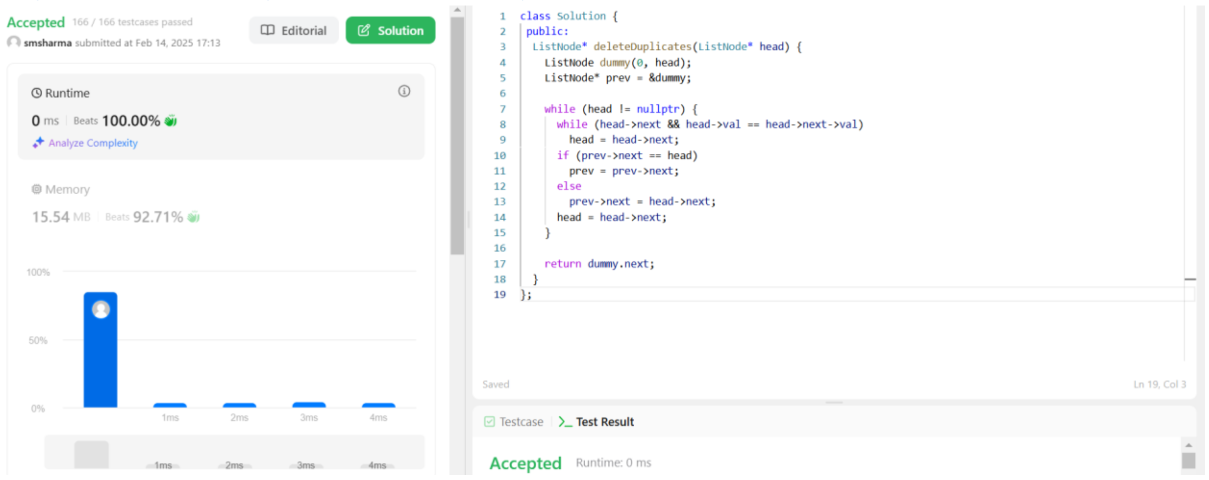


Accepted 208 / 208 testcases passed

smsharma submitted at Feb 14, 2025 17:12

```cpp
class Solution {
public:
  ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
    if (!list1 || !list2)
      return list1 ? list1 : list2;
    if (list1->val > list2->val)
      swap(list1, list2);
    list1->next = mergeTwoLists(list1->next, list2);
    return list1;
  }
};
```

6. https://leetcode.com/problems/remove-duplicates-from-sorted-list-ii /



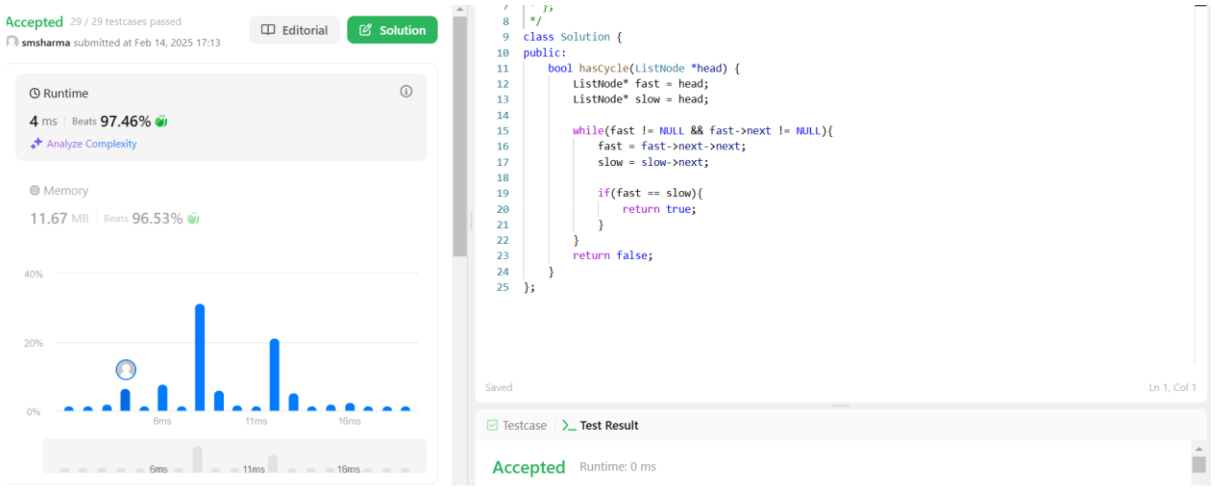
Accepted 166 / 166 testcases passed

smsharma submitted at Feb 14, 2025 17:13

```cpp
class Solution {
public:
  ListNode* deleteDuplicates(ListNode* head) {
    ListNode dummy(0, head);
    ListNode* prev = &dummy;

    while (head != nullptr) {
      while (head->next && head->val == head->next->val)
        head = head->next;
      if (prev->next == head)
        prev = prev->next;
      else
        prev->next = head->next;
      head = head->next;
    }

    return dummy.next;
  }
};
```

Accepted Runtime: 0 ms

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


Accepted 29 / 29 testcases passed
smsharma submitted at Feb 14, 2025 17:13

Runtime: 4 ms | Beats 97.46%

Memory: 11.67 MB | Beats 96.53%

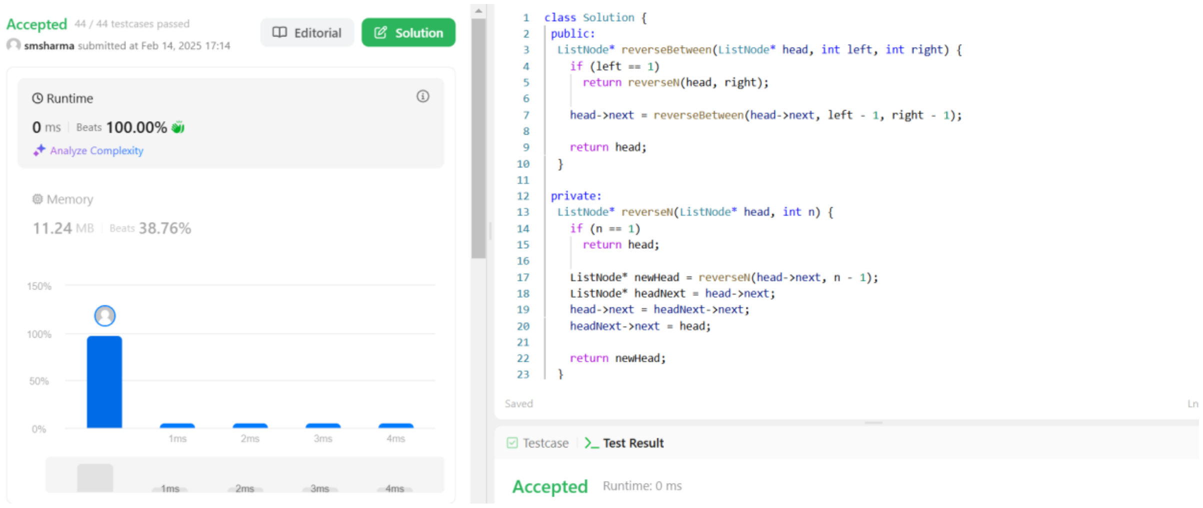
```
 */
class Solution {
public:
    bool hasCycle(ListNode *head) {
        ListNode* fast = head;
        ListNode* slow = head;

        while(fast != NULL && fast->next != NULL){
            fast = fast->next->next;
            slow = slow->next;

            if(fast == slow){
                return true;
            }
        }
        return false;
    }
};
```

Saved | Ln 1, Col 1

Testcase | Test Result

Accepted Runtime: 0 ms

8. https://leetcode.com/problems/reverse-linked-list-ii/


Accepted 44 / 44 testcases passed
smsharma submitted at Feb 14, 2025 17:14

Runtime: 0 ms | Beats 100.00%

Memory: 11.24 MB | Beats 38.76%

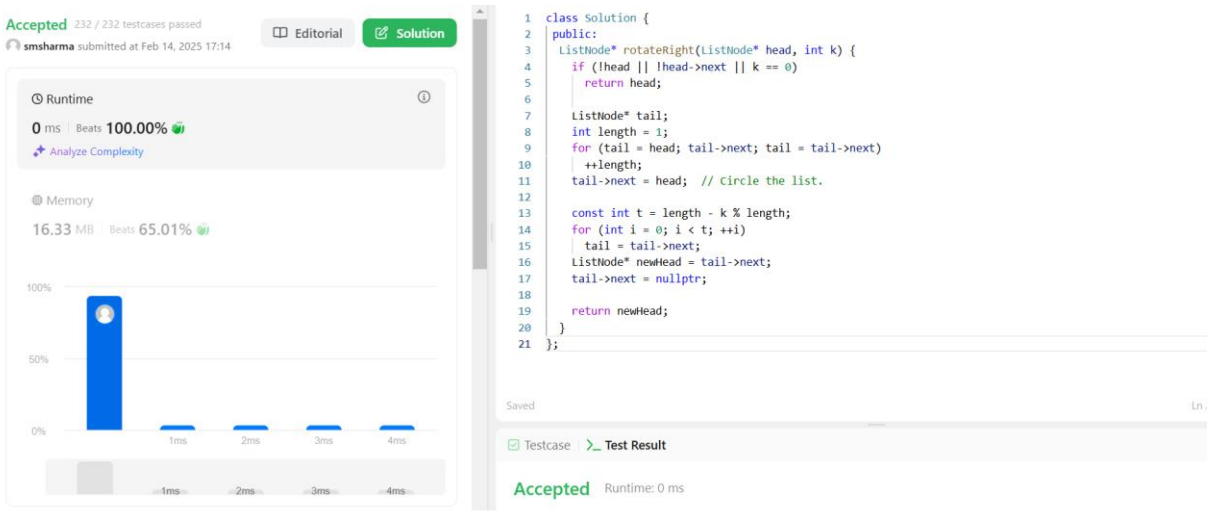
```
class Solution {
public:
 ListNode* reverseBetween(ListNode* head, int left, int right) {
   if (left == 1)
     return reverseN(head, right);

   head->next = reverseBetween(head->next, left - 1, right - 1);

   return head;
 }

private:
 ListNode* reverseN(ListNode* head, int n) {
   if (n == 1)
     return head;

   ListNode* newHead = reverseN(head->next, n - 1);
   ListNode* headNext = head->next;
   head->next = headNext->next;
   headNext->next = head;

   return newHead;
 }
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

9. https://leetcode.com/problems/rotate-list/


Accepted 232 / 232 testcases passed
smsharma submitted at Feb 14, 2025 17:14

Runtime: 0 ms | Beats 100.00%

Memory: 16.33 MB | Beats 65.01%

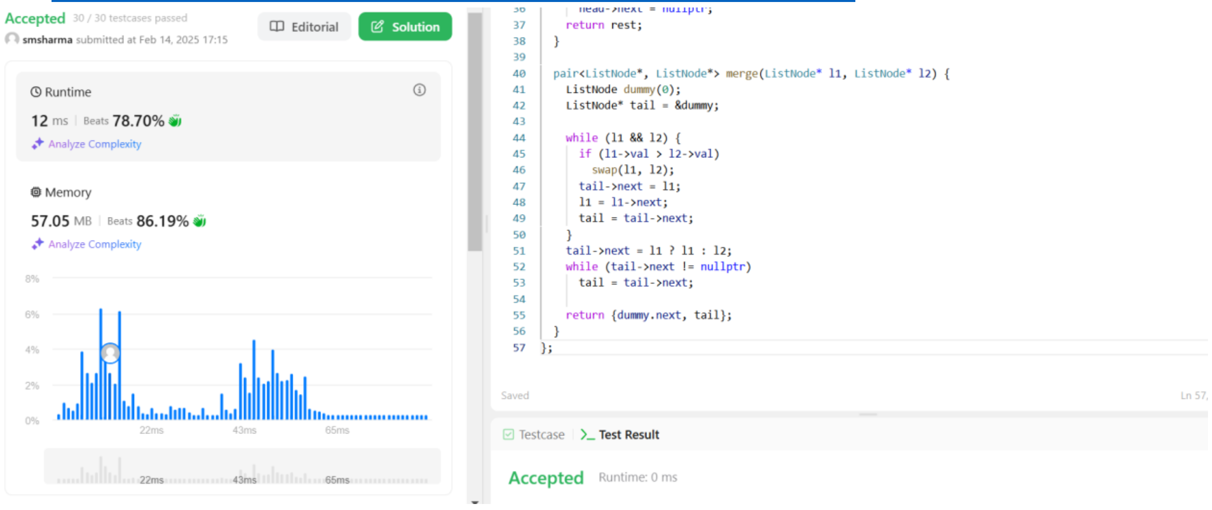
```
class Solution {
public:
 ListNode* rotateRight(ListNode* head, int k) {
   if (!head || !head->next || k == 0)
     return head;

   ListNode* tail;
   int length = 1;
   for (tail = head; tail->next; tail = tail->next)
     ++length;
   tail->next = head;  // Circle the list.

   const int t = length - k % length;
   for (int i = 0; i < t; ++i)
     tail = tail->next;
   ListNode* newHead = tail->next;
   tail->next = nullptr;

   return newHead;
 }
};
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

10. https://leetcode.com/problems/sort-list/submissions/1542851463/


Accepted 30 / 30 testcases passed
smsharma submitted at Feb 14, 2025 17:15
Editorial
Solution

Runtime
12 ms | Beats 78.70%
Analyze Complexity

Memory
57.05 MB | Beats 86.19%
Analyze Complexity

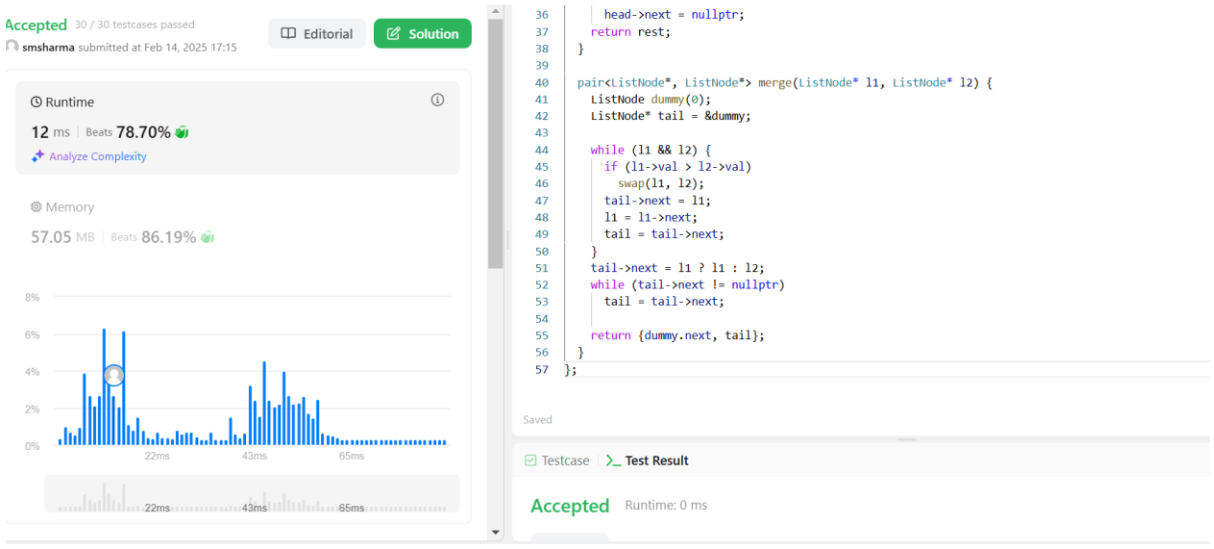
```cpp
      return rest;
    }

    pair<ListNode*, ListNode*> merge(ListNode* l1, ListNode* l2) {
      ListNode dummy(0);
      ListNode* tail = &dummy;

      while (l1 && l2) {
        if (l1->val > l2->val)
          swap(l1, l2);
        tail->next = l1;
        l1 = l1->next;
        tail = tail->next;
      }
      tail->next = l1 ? l1 : l2;
      while (tail->next != nullptr)
        tail = tail->next;

      return {dummy.next, tail};
    }
};
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

11. https://leetcode.com/problems/linked-list-cycle-ii/description/


Accepted 30 / 30 testcases passed
smsharma submitted at Feb 14, 2025 17:15
Editorial
Solution

Runtime
12 ms | Beats 78.70%
Analyze Complexity

Memory
57.05 MB | Beats 86.19%

```cpp
        head->next = nullptr;
      return rest;
    }

    pair<ListNode*, ListNode*> merge(ListNode* l1, ListNode* l2) {
      ListNode dummy(0);
      ListNode* tail = &dummy;

      while (l1 && l2) {
        if (l1->val > l2->val)
          swap(l1, l2);
        tail->next = l1;
        l1 = l1->next;
        tail = tail->next;
      }
      tail->next = l1 ? l1 : l2;
      while (tail->next != nullptr)
        tail = tail->next;

      return {dummy.next, tail};
    }
};
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms