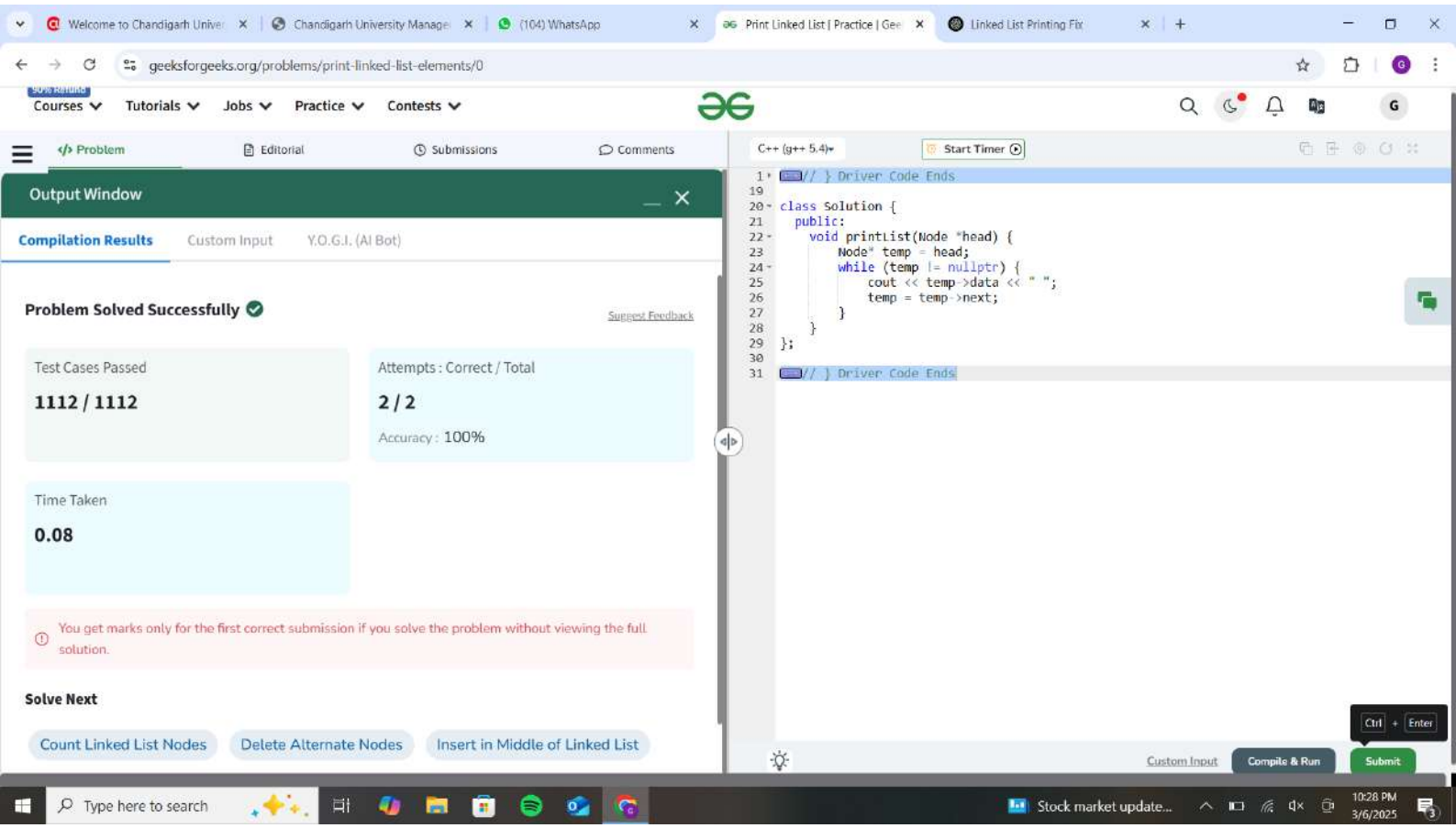

Welcome to Chandigarh Unive
Chandigarh University Manage
(104) WhatsApp
Print Linked List | Practice | Gee
Linked List Printing Fix
geeksforgeeks.org/problems/print-linked-list-elements/0
Courses
Tutorials
Jobs
Practice
Contests
Problem
Editorial
Submissions
Comments
Output Window
Compilation Results
Custom Input
Y.O.G.I. (AI Bot)
Problem Solved Successfully
Suggest Feedback
Test Cases Passed
1112 / 1112
Attempts : Correct / Total
2 / 2
Accuracy : 100%
Time Taken
0.08
You get marks only for the first correct submission if you solve the problem without viewing the full solution.
Solve Next
Count Linked List Nodes
Delete Alternate Nodes
Insert in Middle of Linked List
C++ (g++ 5.4)
Start Timer
// } Driver Code Ends
class Solution {
public:
void printList(Node *head) {
Node* temp = head;
while (temp != nullptr) {
cout << temp->data << " ";
temp = temp->next;
}
}
};
// } Driver Code Ends
Ctrl + Enter
Custom Input
Compile & Run
Submit
Type here to search
Stock market update...
10:28 PM
3/6/2025

Welcome to Chandigarh Univer | Chandigarh University Manage | (104) WhatsApp | Remove Duplicates from Sorte | Linked List Printing Fix

leetcode.com/problems/remove-duplicates-from-sorted-list/submissions/1565103586/

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 168 / 168 testcases passed

Gurvirsinghsekhon submitted at Mar 06, 2025 22:31

Editorial | Solution

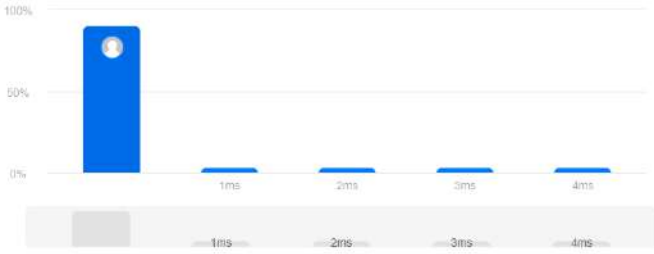
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

16.16 MB | Beats 67.73%



Code | C++

```cpp
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        if (head == nullptr) return head; // If list is empty, return NULL

        ListNode* current = head;
```

Code

C++ | Auto

```cpp
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        if (head == nullptr) return head; // If list is empty, return NULL

        ListNode* current = head;

        while (current != nullptr && current->next != nullptr) {
            if (current->val == current->next->val) {
                // Skip duplicate node
                current->next = current->next->next;
            } else {
                // Move to next distinct node
                current = current->next;
            }
        }

        return head;
    }
};
```

Saved | Ln 21, Col 1

Testcase | Test Result

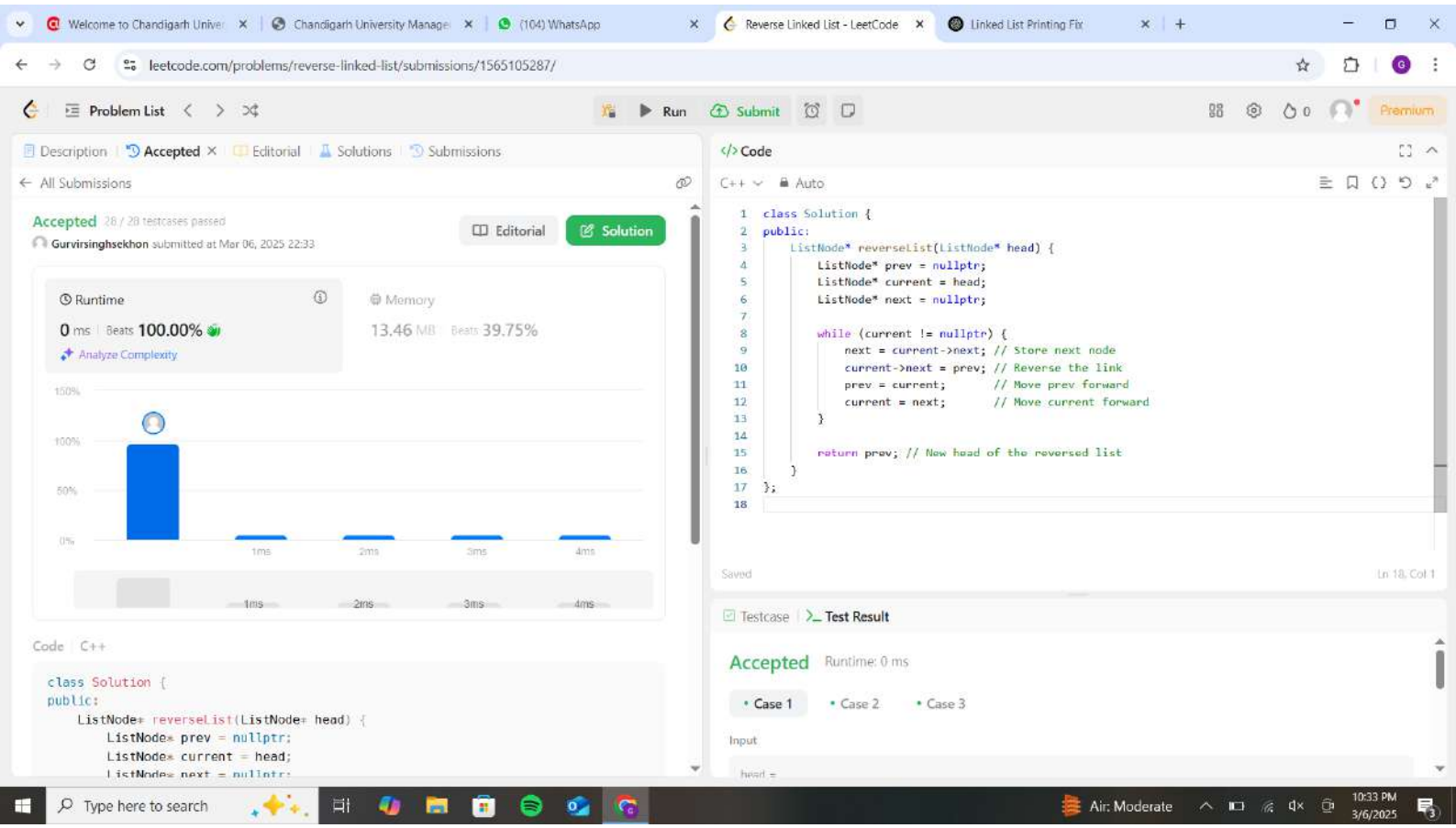
Accepted Runtime: 0 ms

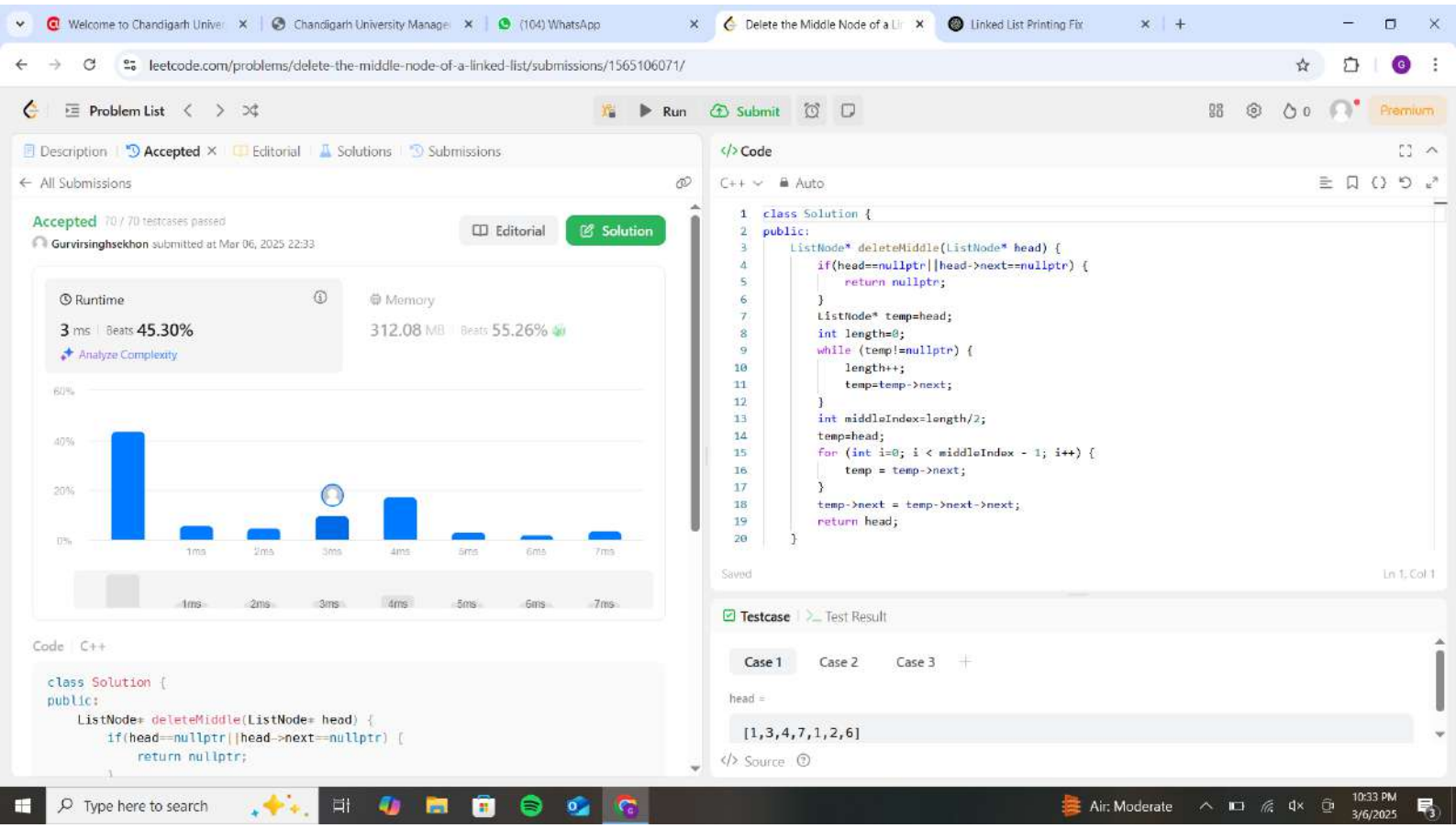
Case 1 | Case 2

Input

head =

Type here to search | Air: Moderate | 10:31 PM 3/6/2025


Welcome to Chandigarh Univer
Chandigarh University Manage
(104) WhatsApp
Reverse Linked List - LeetCode
Linked List Printing Fix
leetcode.com/problems/reverse-linked-list/submissions/1565105287/
Problem List
Run
Submit
Premium
Description
Accepted
Editorial
Solutions
Submissions
All Submissions
Accepted 28 / 28 testcases passed
Gurvirsinghsekhon submitted at Mar 06, 2025 22:33
Editorial
Solution
Runtime
0 ms | Beats 100.00%
Analyze Complexity
Memory
13.46 MB | Beats 39.75%
150%
100%
50%
0%
1ms
2ms
3ms
4ms
Code | C++
class Solution {
public:
ListNode* reverseList(ListNode* head) {
ListNode* prev = nullptr;
ListNode* current = head;
ListNode* next = nullptr;
Code
C++
Auto
class Solution {
public:
ListNode* reverseList(ListNode* head) {
ListNode* prev = nullptr;
ListNode* current = head;
ListNode* next = nullptr;
while (current != nullptr) {
next = current->next; // Store next node
current->next = prev; // Reverse the link
prev = current; // Move prev forward
current = next; // Move current forward
}
return prev; // New head of the reversed list
}
};
Saved
Ln 18, Col 1
Testcase | Test Result
Accepted Runtime: 0 ms
Case 1
Case 2
Case 3
Input
head =
Type here to search
Air: Moderate
10:33 PM
3/6/2025


Welcome to Chandigarh Univer
Chandigarh University Manage
(104) WhatsApp
Delete the Middle Node of a Li
Linked List Printing Fix
leetcode.com/problems/delete-the-middle-node-of-a-linked-list/submissions/1565106071/
Problem List
Run
Submit
Premium
Description
Accepted
Editorial
Solutions
Submissions
All Submissions
Accepted 70 / 70 testcases passed
Gurvirsinghsekhon submitted at Mar 06, 2025 22:33
Editorial
Solution
Runtime
3 ms | Beats 45.30%
Analyze Complexity
Memory
312.08 MB | Beats 55.26%
60%
40%
20%
0%
1ms
2ms
3ms
4ms
5ms
6ms
7ms
Code | C++
class Solution {
public:
ListNode* deleteMiddle(ListNode* head) {
if(head==nullptr||head->next==nullptr) {
return nullptr;
Code
C++
Auto
class Solution {
public:
ListNode* deleteMiddle(ListNode* head) {
if(head==nullptr||head->next==nullptr) {
return nullptr;
}
ListNode* temp=head;
int length=0;
while (temp!=nullptr) {
length++;
temp=temp->next;
}
int middleIndex=length/2;
temp=head;
for (int i=0; i < middleIndex - 1; i++) {
temp = temp->next;
}
temp->next = temp->next->next;
return head;
}
Saved
Ln 1, Col 1
Testcase | Test Result
Case 1
Case 2
Case 3
head =
[1,3,4,7,1,2,6]
Source
Type here to search
Air: Moderate
10:33 PM
3/6/2025

Welcome to Chandigarh Unive | Chandigarh University Manage | (104) WhatsApp | Merge Two Sorted Lists - LeetC | Linked List Printing Fix

leetcode.com/problems/merge-two-sorted-lists/submissions/1565107811/

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 208 / 208 testcases passed

Gurvirsinghsekhon submitted at Mar 06, 2025 22:35

Editorial | Solution

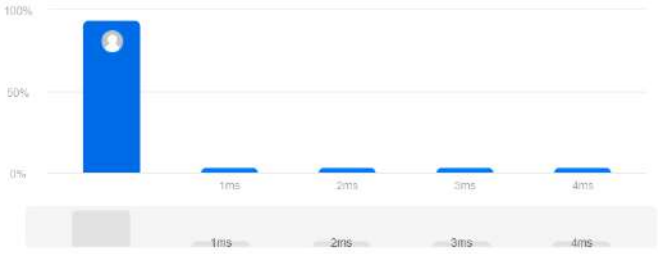
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

19.62 MB | Beats 10.44%



Code | C++

```cpp
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        ListNode dummy(0);  // Dummy node as placeholder
        ListNode* tail = &dummy;  // Pointer to track last node
```

Code

C++ | Auto

```cpp
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        ListNode dummy(0);  // Dummy node as placeholder
        ListNode* tail = &dummy;  // Pointer to track last node

        while (list1 != nullptr && list2 != nullptr) {
            if (list1->val <= list2->val) {
                tail->next = list1;
                list1 = list1->next;
            } else {
                tail->next = list2;
                list2 = list2->next;
            }
            tail = tail->next;  // Move tail forward
        }

        // Attach remaining elements (if any)
        if (list1 != nullptr) tail->next = list1;
        if (list2 != nullptr) tail->next = list2;

        return dummy.next;  // Return merged list head
    }
};
```

Saved

Ln 25, Col 1

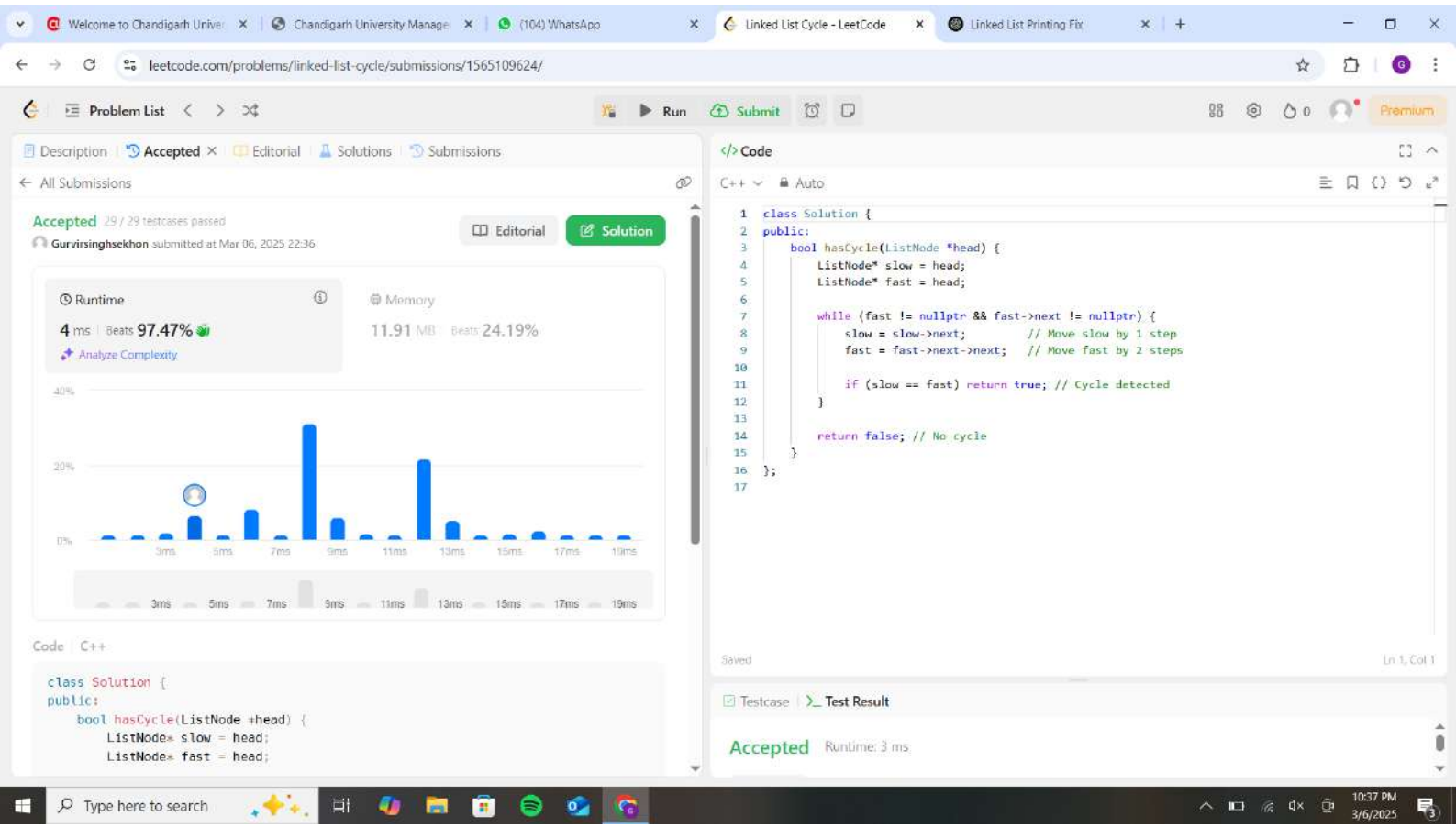
Testcase | Test Result

Accepted Runtime: 0 ms

Type here to search

Air: Moderate

10:35 PM
3/6/2025


Welcome to Chandigarh Univer
Chandigarh University Manage
(104) WhatsApp
Linked List Cycle - LeetCode
Linked List Printing Fix
leetcode.com/problems/linked-list-cycle/submissions/1565109624/
Problem List
Run
Submit
Premium
Description
Accepted
Editorial
Solutions
Submissions
All Submissions
Accepted 29 / 29 testcases passed
Gurvirsinghsekhon submitted at Mar 06, 2025 22:36
Editorial
Solution
Runtime
4 ms | Beats 97.47%
Analyze Complexity
Memory
11.91 MB | Beats 24.19%
40%
20%
0%
3ms 5ms 7ms 9ms 11ms 13ms 15ms 17ms 19ms
Code | C++
class Solution {
public:
bool hasCycle(ListNode *head) {
ListNode* slow = head;
ListNode* fast = head;
Code
C++
Auto
class Solution {
public:
bool hasCycle(ListNode *head) {
ListNode* slow = head;
ListNode* fast = head;
while (fast != nullptr && fast->next != nullptr) {
slow = slow->next; // Move slow by 1 step
fast = fast->next->next; // Move fast by 2 steps
if (slow == fast) return true; // Cycle detected
}
return false; // No cycle
}
};
Saved
Ln 1, Col 1
Testcase
Test Result
Accepted Runtime: 3 ms
Type here to search
10:37 PM
3/6/2025

Welcome to Chandigarh Unive | Chandigarh University Manage | (104) WhatsApp | Rotate List - LeetCode | Linked List Printing Fix

leetcode.com/problems/rotate-list/submissions/1565112881/

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 232 / 232 testcases passed

Gurvirsinghsekhon submitted at Mar 06, 2025 22:39

Editorial | Solution

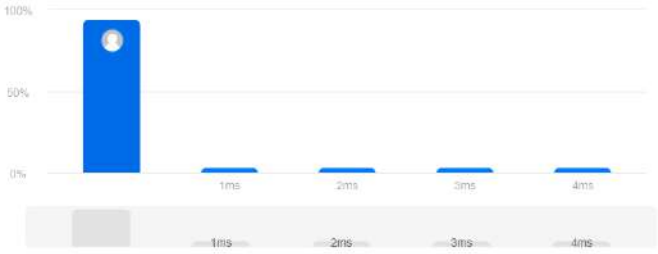
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

16.45 MB | Beats 31.91%



Code | C++

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

        // Step 1: Find the length of the list
```

</> Code

C++ | Auto

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

        // Step 1: Find the length of the list
        int length = 1;
        ListNode* tail = head;
        while (tail->next) {
            tail = tail->next;
            length++;
        }

        // Step 2: Optimize k
        k = k % length;
        if (k == 0) return head; // No rotation needed

        // Step 3: Find the new tail (n - k - 1) and new head (n - k)
        ListNode* new_tail = head;
        for (int i = 0; i < length - k - 1; i++) {
            new_tail = new_tail->next;
        }

        // Step 4: Update pointers
        ListNode* new_head = new_tail->next;
        new_tail->next = nullptr;  // Break the list
        tail->next = head;  // Connect old tail to old head

        return new_head;
```

Saved | Ln 32, Col 1

Testcase | Test Result

Type here to search | Air: Moderate | 10:40 PM 3/6/2025

Welcome to Chandigarh Unive | Chandigarh University Manage | (104) WhatsApp | Sort List - LeetCode | Linked List Printing Fix

leetcode.com/problems/sort-list/submissions/1565114889/

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 30 / 30 testcases passed

Gurvirsinghsekhon submitted at Mar 06, 2025 22:41

Editorial | Solution

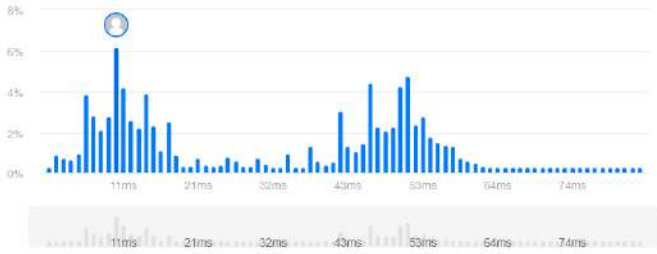
Runtime

10 ms | Beats 87.60%

Analyze Complexity

Memory

57.16 MB | Beats 81.38%



Code | C++

```cpp
class Solution {
public:
    // Function to merge two sorted lists
    ListNode* merge(ListNode* l1, ListNode* l2) {
        ListNode dummy(0);
        ListNode* tail = &dummy;
```

Code

C++ | Auto

```cpp
    // Function to find the middle node
    ListNode* findMid(ListNode* head) {
        ListNode* slow = head;
        ListNode* fast = head->next;

        while (fast && fast->next) {
            slow = slow->next;
            fast = fast->next->next;
        }
        return slow;
    }

    // Main function to sort the linked list
    ListNode* sortList(ListNode* head) {
        if (!head || !head->next) return head; // Base case

        // Step 1: Find the middle and split the list
        ListNode* mid = findMid(head);
        ListNode* rightHead = mid->next;
        mid->next = nullptr; // Break the list into two halves

        // Step 2: Sort both halves recursively
        ListNode* left = sortList(head);
        ListNode* right = sortList(rightHead);

        // Step 3: Merge both sorted halves
        return merge(left, right);
    }
};
```

Saved

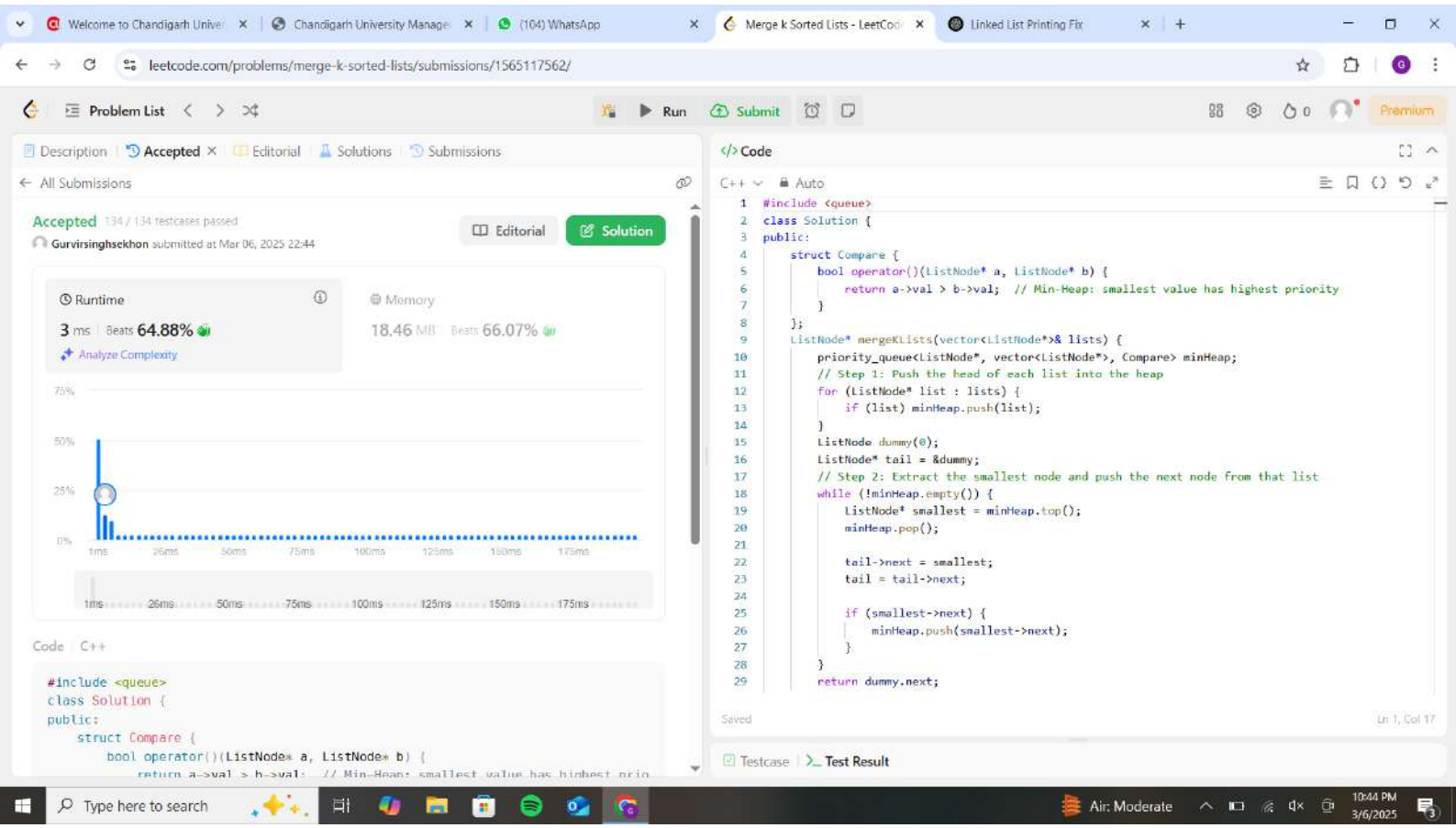
Ln 51, Col 1

Testcase | Test Result

Type here to search

Air: Moderate

10:41 PM
3/6/2025


Welcome to Chandigarh Univer
Chandigarh University Manage
(104) WhatsApp
Merge k Sorted Lists - LeetCod
Linked List Printing Fix
leetcode.com/problems/merge-k-sorted-lists/submissions/1565117562/
Problem List
Run
Submit
Premium
Description
Accepted
Editorial
Solutions
Submissions
All Submissions
Accepted 134 / 134 testcases passed
Gurvirsinghsekhon submitted at Mar 06, 2025 22:44
Editorial
Solution
Runtime
3 ms | Beats 64.88%
Analyze Complexity
Memory
18.46 MB | Beats 66.07%
75%
50%
25%
0%
1ms
26ms
50ms
75ms
100ms
125ms
150ms
175ms
Code | C++
#include <queue>
class Solution {
public:
struct Compare {
bool operator()(ListNode* a, ListNode* b) {
Code
C++
Auto
#include <queue>
class Solution {
public:
struct Compare {
bool operator()(ListNode* a, ListNode* b) {
return a->val > b->val; // Min-Heap: smallest value has highest priority
}
};
ListNode* mergeKLists(vector<ListNode*>& lists) {
priority_queue<ListNode*, vector<ListNode*>, Compare> minHeap;
// Step 1: Push the head of each list into the heap
for (ListNode* list : lists) {
if (list) minHeap.push(list);
}
ListNode dummy(0);
ListNode* tail = &dummy;
// Step 2: Extract the smallest node and push the next node from that list
while (!minHeap.empty()) {
ListNode* smallest = minHeap.top();
minHeap.pop();
tail->next = smallest;
tail = tail->next;
if (smallest->next) {
minHeap.push(smallest->next);
}
}
return dummy.next;
Saved
Ln 1, Col 17
Testcase
Test Result
Type here to search
Air: Moderate
10:44 PM
3/6/2025