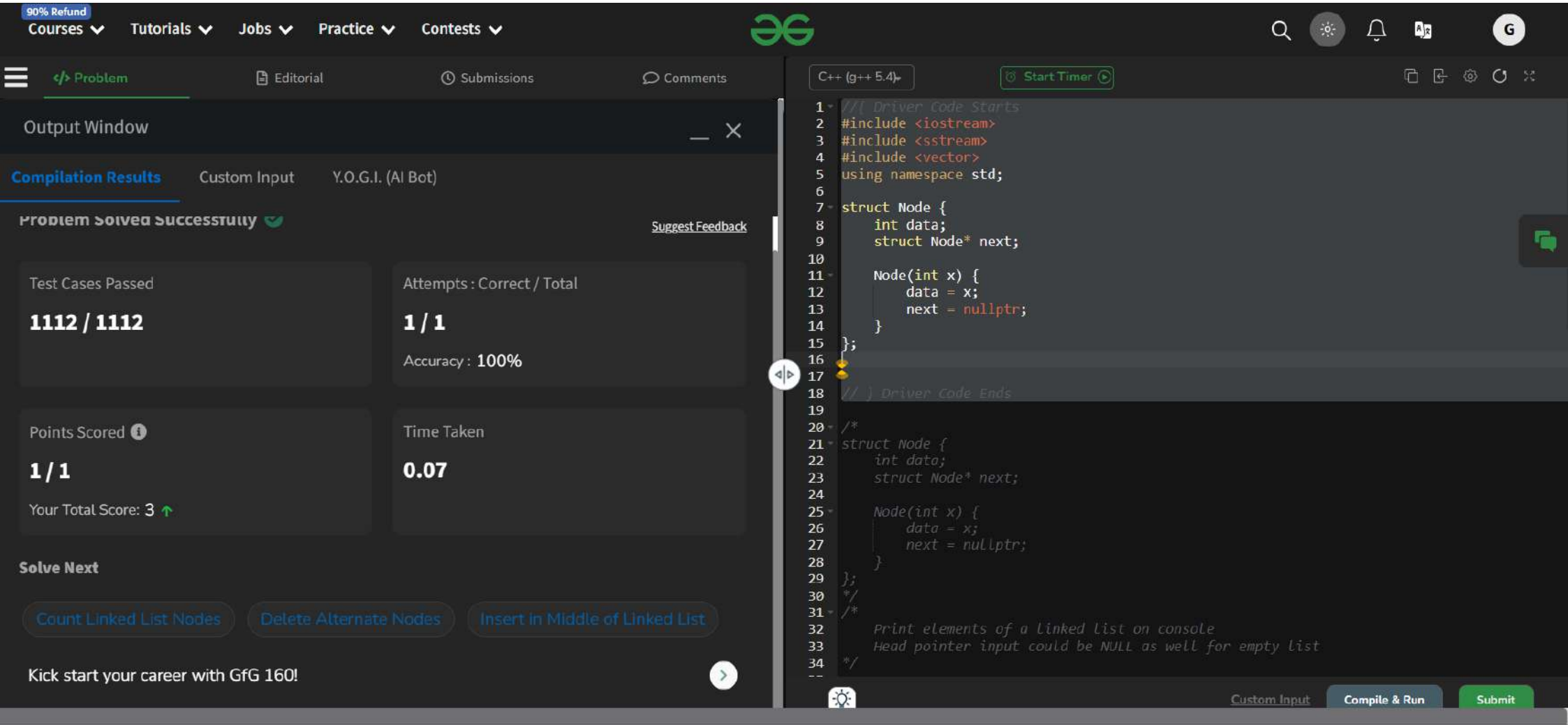

90% Refund
Courses
Tutorials
Jobs
Practice
Contests
Problem
Editorial
Submissions
Comments
Output Window
Compilation Results
Custom Input
Y.O.G.I. (AI Bot)
Problem Solved Successfully
Suggest Feedback
Test Cases Passed
1112 / 1112
Attempts : Correct / Total
1 / 1
Accuracy : 100%
Points Scored
1 / 1
Your Total Score: 3
Time Taken
0.07
Solve Next
Count Linked List Nodes
Delete Alternate Nodes
Insert in Middle of Linked List
Kick start your career with GfG 160!
C++ (g++ 5.4)
Start Timer
/*{ Driver Code Starts
#include <iostream>
#include <sstream>
#include <vector>
using namespace std;
struct Node {
int data;
struct Node* next;
Node(int x) {
data = x;
next = nullptr;
}
};
// } Driver Code Ends
/*
struct Node {
int data;
struct Node* next;
Node(int x) {
data = x;
next = nullptr;
}
};
*/
/*
Print elements of a linked list on console
Head pointer input could be NULL as well for empty list
*/
Custom Input
Compile & Run
Submit

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 168 / 168 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:22

Editorial | Solution

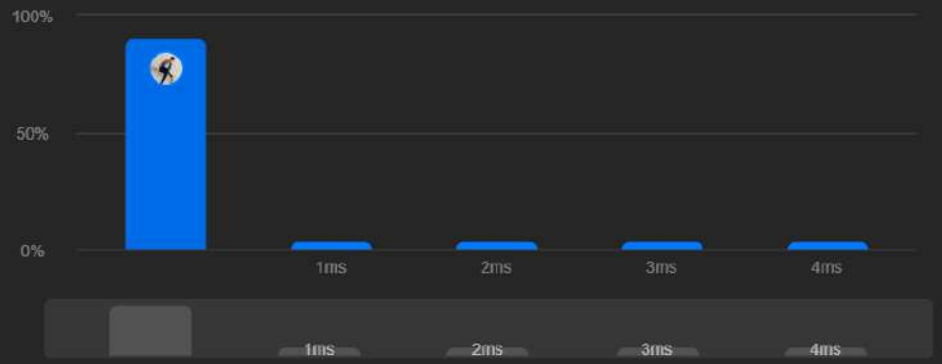
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

15.98 MB | Beats 99.30%



Code | C++

```cpp
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
```

Code

C++ | Auto

```cpp
class Solution {
public:
    ListNode* deleteDuplicates(ListNode* head) {
        ListNode* current = head;
        while (current && current->next) {
            if (current->val == current->next->val) {
                current->next = current->next->next;
            } else {
                current = current->next;
            }
```

Saved

Ln 1, Col 1

Testcase | Test Result

Case 1 | Case 2 | +

head =

[1,1,2,3,3]

Source

Problem List

Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 28 / 28 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:23

Editorial | Solution

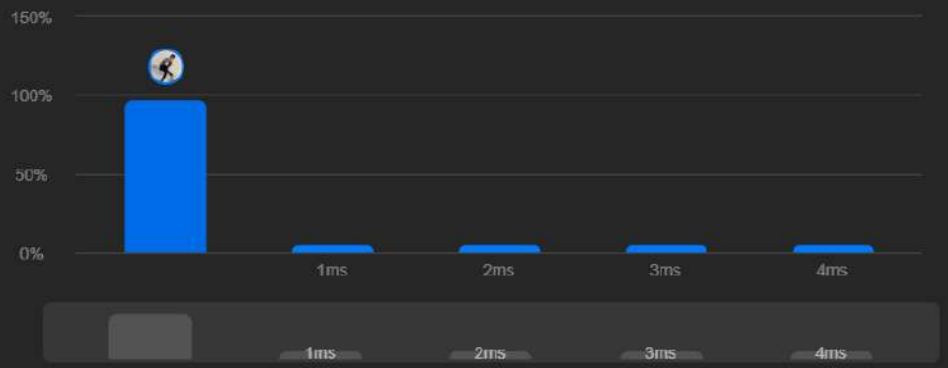
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

13.46 MB | Beats 39.77%



Code | C++

```
class Solution {
public:
    ListNode* reverseList(ListNode* head) {
```

Code

C++ | Auto

```
1  class Solution {
2  public:
3      ListNode* reverseList(ListNode* head) {
4          // Initialize pointers
5          ListNode* prev = nullptr; // Previous node starts as NULL
6          ListNode* next = nullptr; // Next node
7          ListNode* curr = head;    // Current node starts at the head
8
9          // Traverse the list
10         while (curr != nullptr) {
```

Saved

Ln 1, Col 1

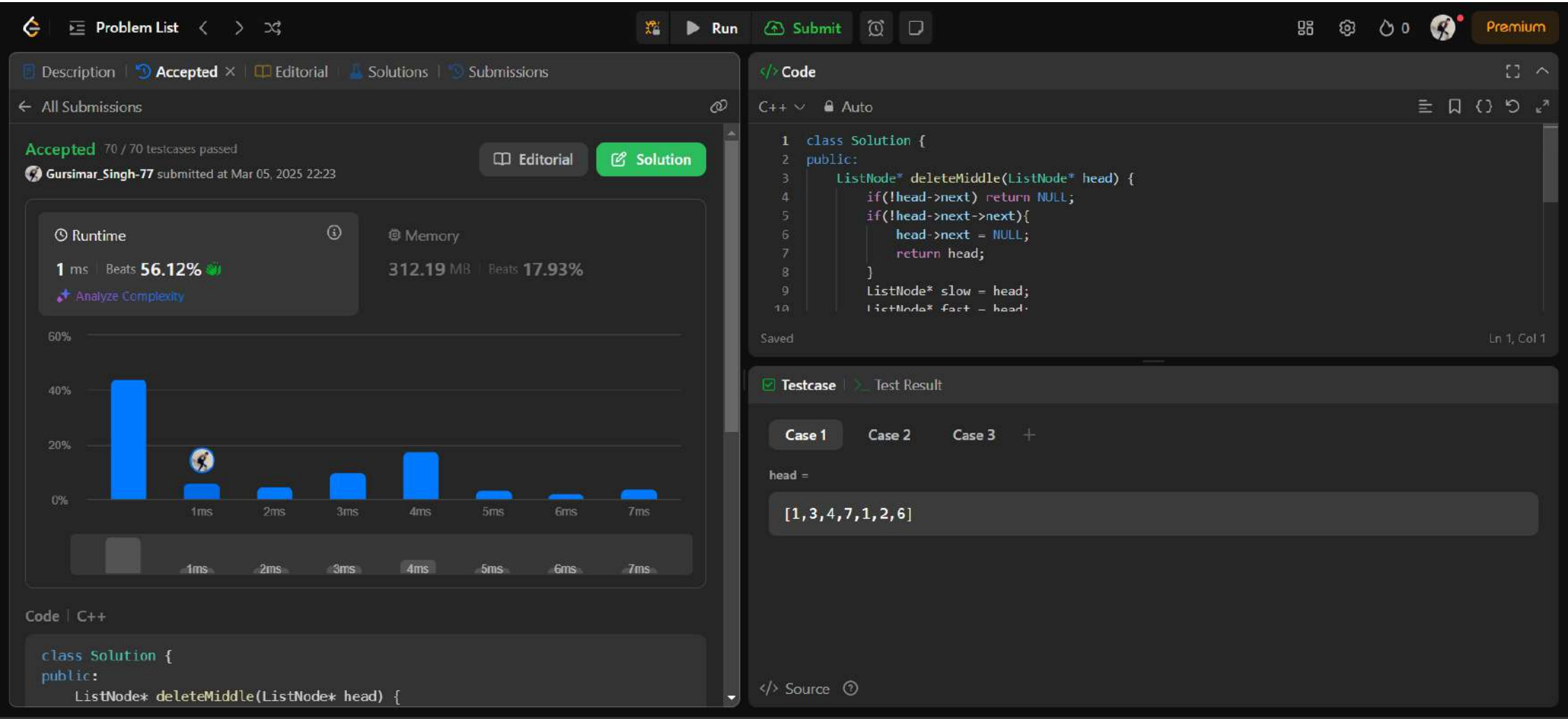
Testcase | Test Result

Case 1 | Case 2 | Case 3

head =

[1,2,3,4,5]

Source


Problem List
Run
Submit
Premium
Description
Accepted
Editorial
Solutions
Submissions
All Submissions
Accepted 70 / 70 testcases passed
Gursimar_Singh-77 submitted at Mar 05, 2025 22:23
Editorial
Solution
Runtime
1 ms | Beats 56.12%
Analyze Complexity
Memory
312.19 MB | Beats 17.93%
60%
40%
20%
0%
1ms
2ms
3ms
4ms
5ms
6ms
7ms
Code | C++
class Solution {
public:
ListNode* deleteMiddle(ListNode* head) {
Code
C++
Auto
class Solution {
public:
ListNode* deleteMiddle(ListNode* head) {
if(!head->next) return NULL;
if(!head->next->next){
head->next = NULL;
return head;
}
ListNode* slow = head;
Saved
Ln 1, Col 1
Testcase | Test Result
Case 1
Case 2
Case 3
head =
[1,3,4,7,1,2,6]
Source

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 208 / 208 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:23

Editorial | Solution

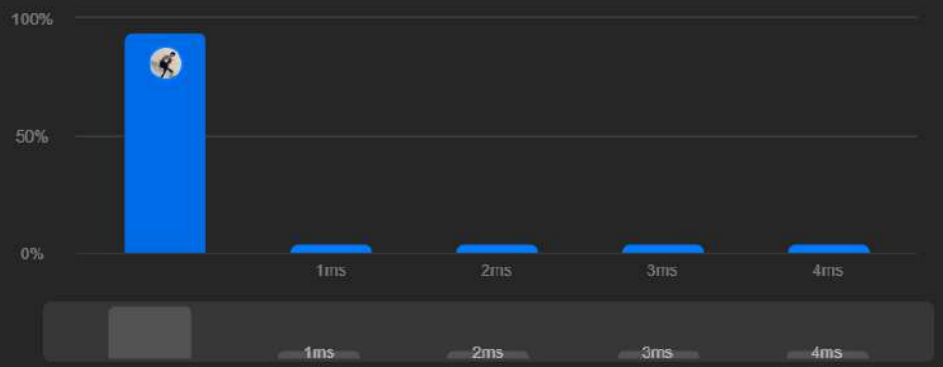
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

19.43 MB | Beats 62.23%



Code | C++

```
class Solution {
public:
```

Code

C++ | Auto

```
1
2  class Solution {
3  public:
4      ListNode* mergeTwoLists(ListNode* l1, ListNode* l2)
5      {
6          // if list1 happen to be NULL
7          // we will simply return list2.
8          if(l1 == NULL)
9          {
10             return l2;
```

Saved | Ln 1, Col 1

Testcase | Test Result

Case 1 | Case 2 | Case 3 +

list1 =

```
[1,2,4]
```

list2 =

```
[1,3,4]
```

Source

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

Submit Ctrl Enter

All Submissions

Accepted 29 / 29 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:24

Editorial | Solution

Runtime

12 ms | Beats 40.49%

Analyze Complexity

Memory

11.86 MB | Beats 54.00%

Code | C++

```
class Solution {
public:
    bool hasCycle(ListNode* head) {
```

C++ | Auto

```
1  class Solution {
2  public:
3      bool hasCycle(ListNode* head) {
4          if (head == NULL || head->next == NULL) {
5              return false;
6          }
7          ListNode* slow = head;
8          ListNode* fast = head->next;
9          while (fast != slow) {
```

Saved

Ln 1, Col 1

Testcase | Test Result

Case 1 | Case 2 | Case 3

head =

[3,2,0,-4]

pos =

1

Source

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 232 / 232 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:24

Editorial | Solution

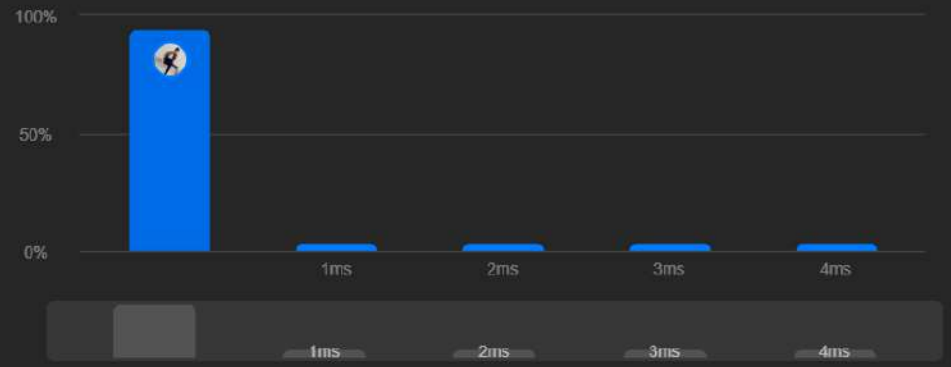
Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

16.36 MB | Beats 64.53%



Code | C++

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
```

Code

C++ | Auto

```
/**
 * 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) {}
 * };
 */
```

Saved | Ln 1, Col 1

Testcase | Test Result

Case 1 | Case 2 | +

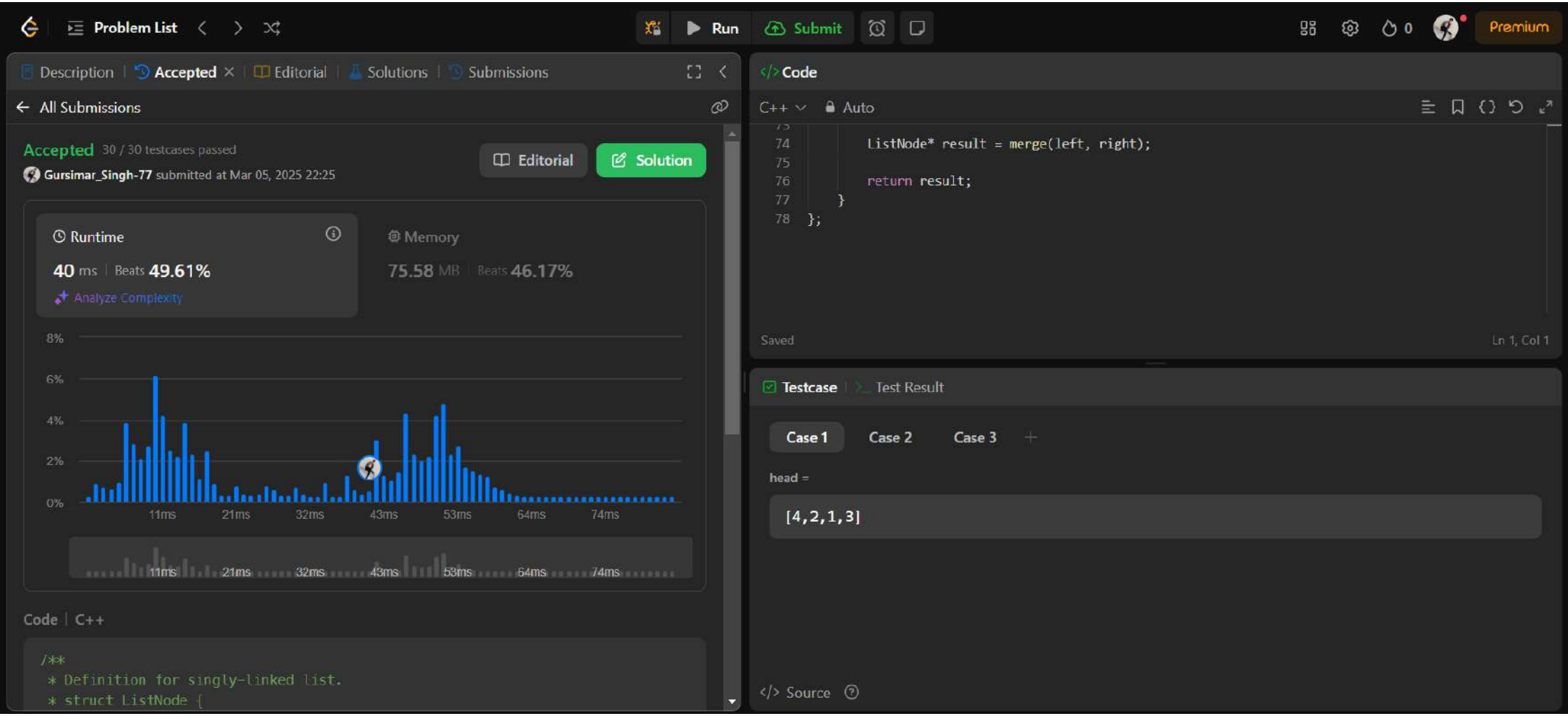
head =

[1,2,3,4,5]

k =

2

Source


Problem List
Run
Submit
Premium
Description | Accepted | Editorial | Solutions | Submissions
All Submissions
Accepted 30 / 30 testcases passed
Gursimar_Singh-77 submitted at Mar 05, 2025 22:25
Editorial
Solution
Runtime
40 ms | Beats 49.61%
Analyze Complexity
Memory
75.58 MB | Beats 46.17%
8%
6%
4%
2%
0%
11ms
21ms
32ms
43ms
53ms
64ms
74ms
Code | C++
/**
* Definition for singly-linked list.
* struct ListNode {
Code
C++
Auto
ListNode* result = merge(left, right);
return result;
}
};
Saved
Ln 1, Col 1
Testcase | Test Result
Case 1
Case 2
Case 3
head =
[4,2,1,3]
Source

Problem List | Run | Submit | Premium

Description | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 134 / 134 testcases passed

Gursimar_Singh-77 submitted at Mar 05, 2025 22:25

Editorial | Solution

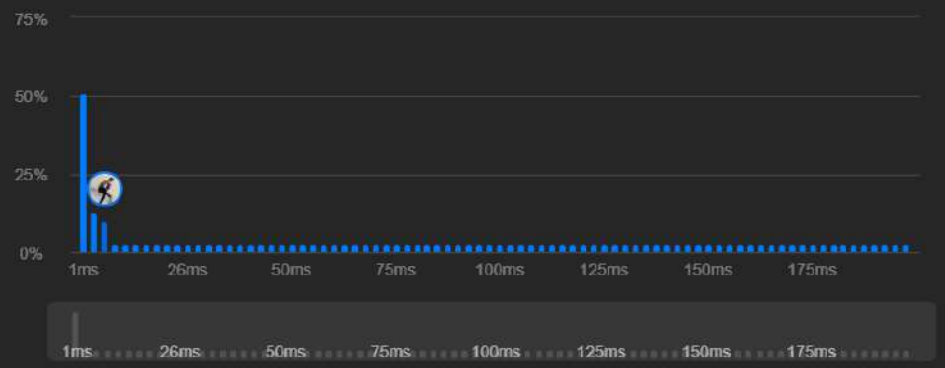
Runtime

6 ms | Beats 36.04%

Analyze Complexity

Memory

23.98 MB | Beats 5.82%



Code | C++

```
class Solution {
public:
    ListNode* mergeKLists(vector<ListNode*>& lists) {
```

Code

C++ | Auto

```
1  class Solution {
2  public:
3      ListNode* mergeKLists(vector<ListNode*>& lists) {
4          if (lists.empty()) {
5              return nullptr;
6          }
7          return mergeKListsHelper(lists, 0, lists.size() - 1);
8      }
9
10     ListNode* mergeKListsHelper(vector<ListNode*>& lists, int start, int end) {
```

Saved | Ln 1, Col 1

Testcase | Test Result

Case 1 | Case 2 | Case 3 | +

lists =

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

Source