



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

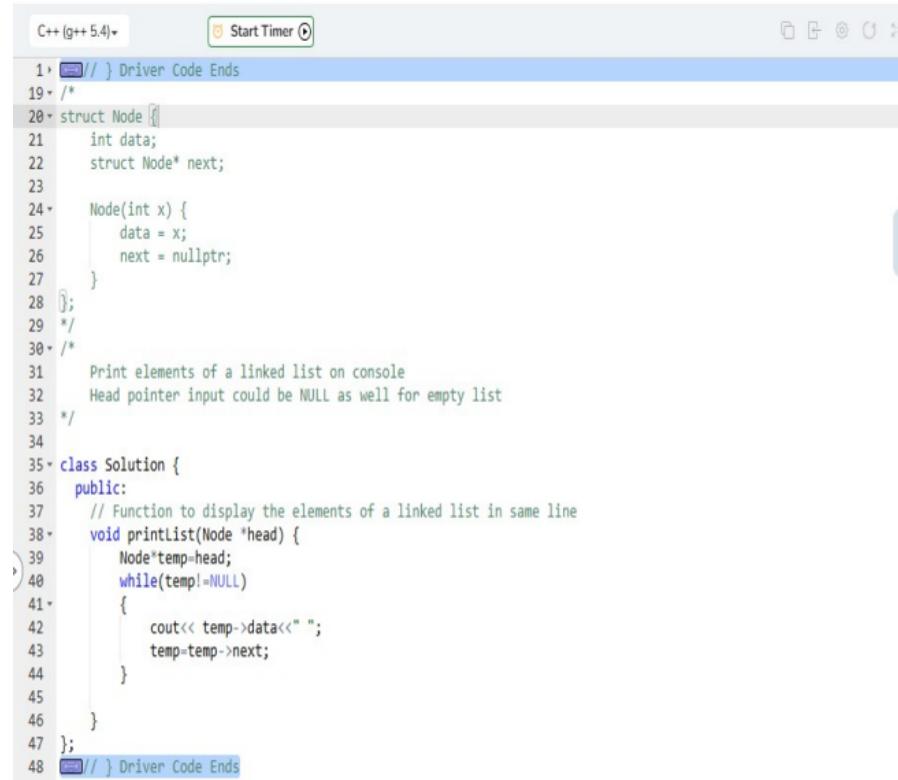
Discover. Learn. Empower.

## Experiment 3

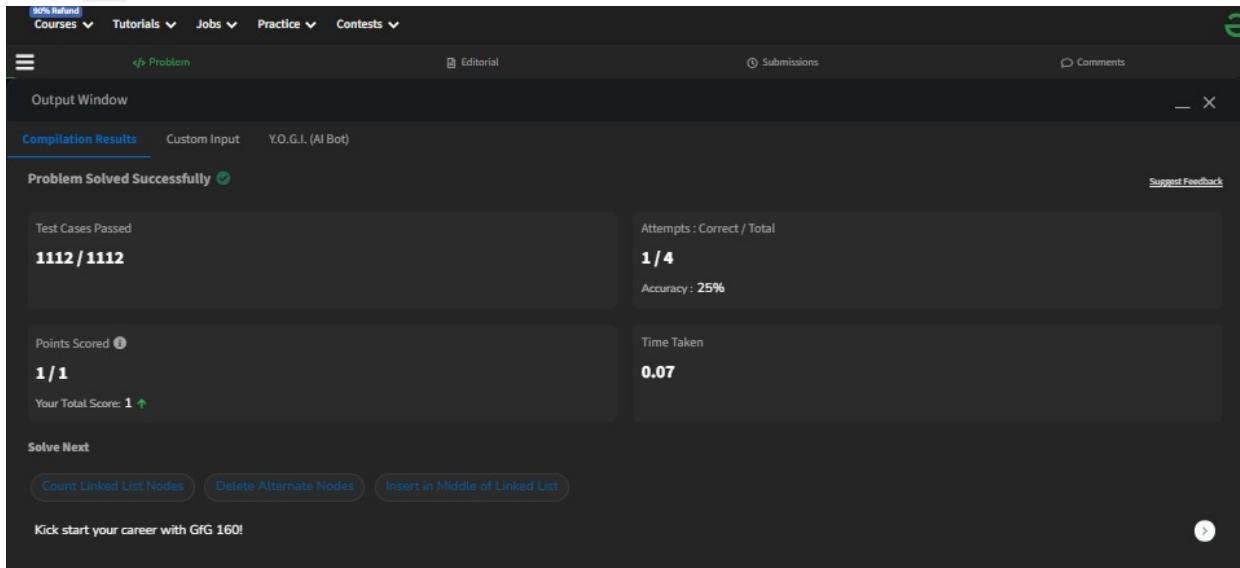
**Student Name:** Saurav Ashiwal  
**Branch:** CSE  
**Semester:** 6  
**Subject Name:** AP lab-2

**UID:** 22BCS13250  
**Section/Group:** 614/B  
**Date of Performance:** 14/02/2025  
**Subject Code:** 22CSP-351

**Q1.**



```
C++ (g++ 5.4) Start Timer
1 // Driver Code Ends
19 /*
20 struct Node {
21     int data;
22     struct Node* next;
23
24     Node(int x) {
25         data = x;
26         next = nullptr;
27     }
28 };
29 */
30 /*
31 Print elements of a linked list on console
32 Head pointer input could be NULL as well for empty list
33 */
34
35 class Solution {
36 public:
37     // Function to display the elements of a linked list in same line
38     void printList(Node *head) {
39         Node*temp=head;
40         while(temp!=NULL)
41         {
42             cout<< temp->data<< " ";
43             temp=temp->next;
44         }
45     }
46 };
47 // Driver Code Ends
```



50% Refined 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 / 4 Accuracy : 25%

Points Scored 1 / 1 Time Taken 0.07

Your Total Score: 1 ↑

Solve Next

Count Linked List Nodes Delete Alternate Nodes Insert in Middle of Linked List

Kick start your career with GFG 160!



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Q 2.

The screenshot shows a programming competition interface. At the top, there are tabs for 'Problem List', 'Run', 'Submit', and others. Below the tabs, it says 'Accepted' with '168 / 168 testcases passed'. A user 'Saurav\_20062' submitted the solution at 'Feb 14, 2025 18:24'. There are two tabs: 'Editorial' and 'Solution' (which is selected). On the left, there's a performance chart for 'Runtime' and 'Memory'. The runtime chart shows a single bar at 0 ms, labeled 'Beats 100.00%'. The memory chart shows 16.37 MB, labeled 'Beats 11.26%'. Below the chart is a complexity analysis button. On the right, the code is displayed in C++:

```
ListNode* deleteDuplicates(ListNode* head) {
    ListNode* curr_pos = head;
    while (curr_pos != NULL && curr_pos->next != NULL) {
        if (curr_pos->val == curr_pos->next->val) {
            ListNode* temp = curr_pos->next;
            curr_pos->next = curr_pos->next->next;
            delete temp;
        } else {
            curr_pos = curr_pos->next;
        }
    }
    return head;
}
```

Below the code, it says 'Saved'. In the 'Test Result' section, the expected output is '[1,2]' and the actual output is also '[1,2]'. There is a 'Contribute a testcase' button.

Q 3.

The screenshot shows a programming competition interface. At the top, there are tabs for 'Problem List', 'Run', 'Submit', and others. Below the tabs, it says 'Accepted' with '28 / 28 testcases passed'. A user 'Saurav\_20062' submitted the solution at 'Feb 14, 2025 18:31'. There are two tabs: 'Editorial' and 'Solution' (selected). On the left, there's a performance chart for 'Runtime' and 'Memory'. The runtime chart shows a single bar at 0 ms, labeled 'Beats 100.00%'. The memory chart shows 13.37 MB, labeled 'Beats 70.61%'. Below the chart is a complexity analysis button. On the right, the code is displayed in C++:

```
#include <iostream>
using namespace std;

class Solution {
public:
    ListNode* reverseList(ListNode* head) {
        ListNode* curr = head;
        ListNode* prev = nullptr;
        ListNode* next = nullptr;

        while (curr != nullptr) {
            next = curr->next; // Store the next node
            curr->next = prev; // Reverse the link
            prev = curr; // Move prev forward
            curr = next; // Move curr forward
        }
        return prev; // New head of reversed list
    }
}
```

Below the code, it says 'Saving...'. In the 'Test Result' section, the expected output is '[5,4,3,2,1]' and the actual output is also '[5,4,3,2,1]'. There is a 'Contribute a testcase' button.



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Q 4.

The screenshot shows a programming environment with the following details:

- Problem List:** Accepted.
- Description:** Accepted 70 / 70 testcases passed.
- Editorial:** Solution
- Runtime:** 6 ms | Beats 14.41%.
- Memory:** 312.01 MB | Beats 55.19%.
- Complexity Analysis:** Analyze Complexity
- Performance Chart:** A bar chart showing execution times for various test cases. The Y-axis ranges from 0% to 60%, and the X-axis shows times from 1ms to 7ms. The largest bar is at 1ms (~40%), followed by 4ms (~18%). Other bars are much smaller.
- Code:** C++ code for deleting the middle node of a singly-linked list.

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode() : val(0), next(nullptr) {}
7  *     ListNode(int x) : val(x), next(nullptr) {}
8  *     ListNode(int x, ListNode *next) : val(x), next(next) {}
9  * };
10 */
11 class Solution {
12 public:
13     ListNode* deleteMiddle(ListNode* head) {
14         if (head == nullptr || head->next == nullptr) {
15             return nullptr;
16         }
17
18         ListNode* fast = head;
19         ListNode* slow = head;
20         ListNode* pre = nullptr;
```

- Test Result:** Accepted | Testcase | Test Result
- Runtime:** 0 ms

Q 5.

The screenshot shows a programming environment with the following details:

- Problem List:** Accepted.
- Description:** Accepted 208 / 208 testcases passed.
- Editorial:** Solution
- Runtime:** 0 ms | Beats 100.00%.
- Memory:** 19.47 MB | Beats 62.56%.
- Complexity Analysis:** Analyze Complexity
- Performance Chart:** A bar chart showing execution times for various test cases. The Y-axis ranges from 0% to 100%, and the X-axis shows times from 1ms to 4ms. The largest bar is at 1ms (~100%), followed by 2ms (~10%). Other bars are very small.
- Code:** C++ code for merging two sorted linked lists.

```
11 class Solution {
12 public:
13     ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
14         ListNode* dummy = new ListNode(-1);
15         ListNode* ptr1 = list1;
16         ListNode* ptr2 = list2;
17         ListNode* ptr3 = dummy;
18
19         while (ptr1 != NULL && ptr2 != NULL) {
20             if (ptr1->val < ptr2->val) {
21                 ptr3->next = ptr1;
22                 ptr1 = ptr1->next;
23             } else {
24                 ptr3->next = ptr2;
25                 ptr2 = ptr2->next;
26             }
27             ptr3 = ptr3->next;
28         }
29
30         if (ptr1 != NULL) {
```

- Test Result:** Accepted | Testcase | Test Result
- Runtime:** 0 ms



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Q 6.

Problem List | Run | Submit | All Submissions

Accepted 166 / 166 testcases passed  
Saurav\_20062 submitted at Feb 14, 2025 18:43

Runtime: 0 ms | Beats 100.00% | Memory: 15.42 MB | Beats 99.72%

Analyze Complexity

100%  
50%  
0%  
1ms 2ms 3ms 4ms

Code | C++

```
/*  
 * Definition for singly-linked list.  
 * struct ListNode {  
 *     int val;  
 * };  
 */
```

Code

```
25 while (curr) {  
26     // If the current node has duplicates  
27     if (curr->next && curr->val == curr->next->val) {  
28         // Move curr forward to the last duplicate node  
29         while (curr->next && curr->val == curr->next->val) {  
30             curr = curr->next;  
31         }  
32         // Skip all duplicates  
33         prev->next = curr->next;  
34     } else {  
35         // No duplicate, just move prev forward  
36         prev = prev->next;  
37     }  
38     curr = curr->next;  
39 }  
40  
41 return dummy->next;  
42 }  
43  
44 }  
45 }
```

Saved | Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Q 7.

Problem List | Run | Submit | All Submissions

Accepted 29 / 29 testcases passed  
Saurav\_20062 submitted at Feb 14, 2025 18:46

Runtime: 12 ms | Beats 40.45% | Memory: 11.91 MB | Beats 24.62%

Analyze Complexity

40%  
20%  
0%  
3ms 5ms 7ms 9ms 11ms 13ms 15ms 17ms 19ms

Code | C++

```
/*  
 * Definition for singly-linked list.  
 * struct ListNode {  
 *     int val;  
 * };  
 */
```

Code

```
9 class Solution {  
10 public:  
11     bool hasCycle(ListNode *head) {  
12         if(head == NULL) {  
13             return false;  
14         }  
15         ListNode* fast = head;  
16         ListNode* slow = head;  
17         while(fast != NULL && fast->next != NULL) {  
18             fast = fast->next->next;  
19             slow = slow->next;  
20             if(fast == slow) {  
21                 return true;  
22             }  
23         }  
24         return false;  
25     }  
26 };
```

Saved | Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Q 8.

Problem List < > ✎ Run Submit 🔍

Description Accepted Editorial Solutions Submissions

All Submissions

Accepted 44 / 44 testcases passed Saurav\_20062 submitted at Feb 14, 2025 19:03

Runtime Memory

0 ms Beats 100.00% 11.18 MB Beats 73.55%

Analyze Complexity

150%  
100%  
50%  
0%

1ms 2ms 3ms 4ms

Code C++

```
class Solution {
public:
    // Function to reverse a linked list
    ListNode* reverse(ListNode* head) {
```

Code C++

```
ListNode* next = nullptr;
ListNode* prevReverse = nullptr;

for (int i = left; i <= right; i++) {
    next = current->next;
    current->next = prevReverse;
    prevReverse = current;
    current = next;
}

// Connect the reversed sublist back to the list
prev->next->next = current;
prev->next = prevReverse;

return dummy.next;
}
```

Saved

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

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

Q 9.

Problem List < > ✎ Run Submit Ctrl Enter 🔍

Description Accepted Editorial Solutions Submissions

All Submissions

Accepted 232 / 232 testcases passed Saurav\_20062 submitted at Feb 14, 2025 19:06

Runtime Memory

0 ms Beats 100.00% 16.30 MB Beats 93.81%

Analyze Complexity

100%  
50%  
0%

1ms 2ms 3ms 4ms

Code C++

```
/* Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 * }
```

Code C++

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

        int n = 1;
        ListNode* tail = head;
        while (tail->next != NULL) {
            n++;
            tail = tail->next;
        }

        tail->next = head;
        k = k % n;
        if (k == 0) {
            ...
```

Saved

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

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



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Q 10.

Problem List | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 30 / 30 testcases passed

Saurav\_20062 submitted at Feb 14, 2025 19:10

Runtime: 54 ms | Beats 20.84% | Analyze Complexity

Memory: 75.81 MB | Beats 9.15%

Runtime Distribution (ms): 11ms, 22ms, 32ms, 43ms, 54ms, 65ms, 75ms

Code | C++

```
class Solution {
public:
    ListNode* sortList(ListNode* head) {
        if (head == NULL || head->next == NULL) {
            return head;
        }
        // Find the middle of the list
        ListNode* mid = Mid(head);
        ListNode* right = mid->next;
        mid->next = NULL; // Split the list into two halves
        // Recursively sort both halves
        ListNode* leftSorted = sortList(head);
        ListNode* rightSorted = sortList(right);
        // Merge both sorted halves
        return MergeSortedList(leftSorted, rightSorted);
    }
    ListNode* Mid(ListNode* head) {
```

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

head = [4,2,1,3]

Q 11.

Problem List | Accepted | Editorial | Solutions | Submissions

All Submissions

Accepted 18 / 18 testcases passed

Saurav\_20062 submitted at Feb 14, 2025 19:12

Runtime: 0 ms | Beats 100.00% | Analyze Complexity

Memory: 11.32 MB | Beats 53.90%

Runtime Distribution (ms): 3ms, 5ms, 7ms, 9ms, 11ms, 13ms

Code | C++

```
/*
class Solution {
public:
    ListNode *detectCycle(ListNode *head) {
        if (head == NULL) {
            return head;
        }
        ListNode *fast = head;
        ListNode *slow = head;
        while (fast != NULL && fast->next != NULL) {
            fast = fast->next->next;
            slow = slow->next;
            if (fast == slow) {
                break;
            }
        }
        if (fast == NULL || fast->next == NULL) {
```

Testcase | Test Result

Accepted Runtime: 2 ms

Case 1 Case 2 Case 3

Input

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