

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Assignment-3

Student Name: Ananya Kanwar

Branch: CSE

Semester: 6th

Subject Name: AP Lab

UID: 22BCS10579

Section/Group: IOT-609-B

Date: 05-03-2025

Subject Code: 22CSP-351

Ques 1: Print Linked List

Code: -

```
class Solution {  
public:  
    // Function to display the elements of a linked list in same line  
    void printList(Node *head) {  
        Node *temp=head;  
        while(temp!=NULL){  
            cout<<temp->data<<" ";  
            temp=temp->next;  
        }  
    }  
};
```

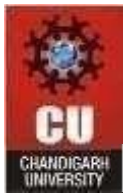
Submission: -

The screenshot shows a C++ IDE with the following content:

```
1 // Print Linked List  
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             cout<<temp->data<<" ";  
42             temp=temp->next;  
43         }  
44     }  
45 }  
46
```

On the left, the 'Compilation Results' tab shows:

- Problem Solved Successfully
- Test Cases Passed: 1112 / 1112
- Attempts: Correct / Total: 2 / 2
- Accuracy: 100%
- Time Taken: 0.07



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

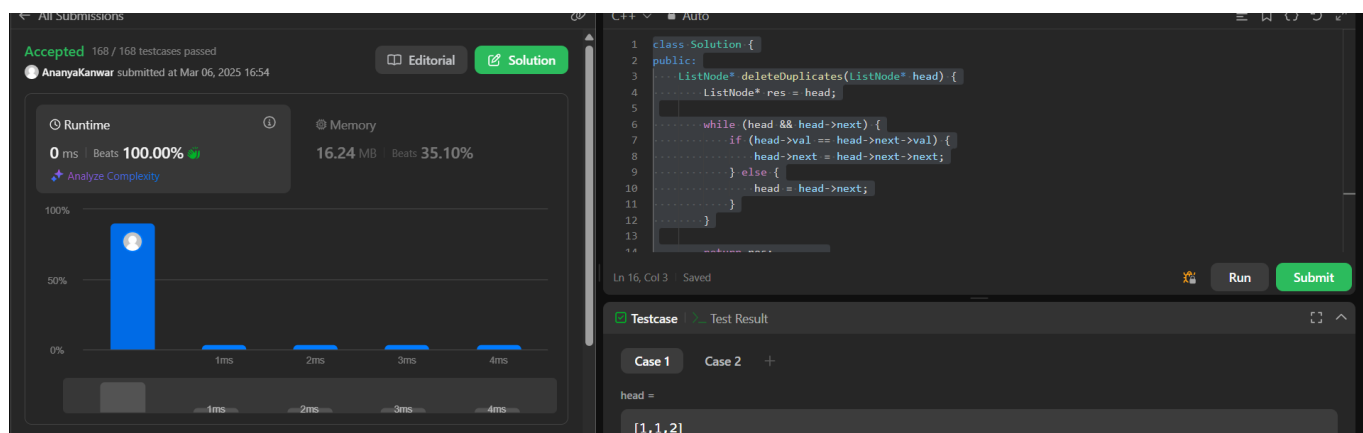
Discover. Learn. Empower.

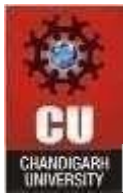
Ques 2: Remove Duplicates from Sorted List

Code: -

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

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

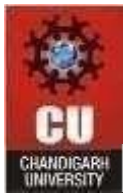
Ques 3: Reverse Linked List

Code: -

```
class Solution {  
public:  
    ListNode* reverseList(ListNode* head) {  
        if(head==NULL){  
            return NULL;  
        }  
        ListNode* curr=head;  
        ListNode* nextNode=curr->next;  
        ListNode* prev=NULL;  
        while(curr!=NULL){  
            curr->next=prev;  
            prev=curr;  
            curr=nextNode;  
            if (nextNode != NULL) {  
                nextNode = nextNode->next;  
            }  
        }  
        return prev;  
    }  
};
```

Submission: -

The screenshot displays a submission interface for a C++ program. On the left, a summary card indicates the submission is 'Accepted' with 28/28 testcases passed, submitted by 'AnanyaKanwar' on Mar 06, 2025 at 16:57. Performance metrics show a runtime of 0 ms (beating 100.00%) and memory usage of 13.35 MB (beating 70.13%). A bar chart shows the runtime performance across different test cases. Below this, the C++ code is displayed, matching the code provided in the previous block. On the right, the code editor shows the same code with line numbers. At the bottom right, the 'Testcase' tab is active, showing 'Case 1' with the input 'head = [1,2,3,4,5]'. Buttons for 'Run' and 'Submit' are visible.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

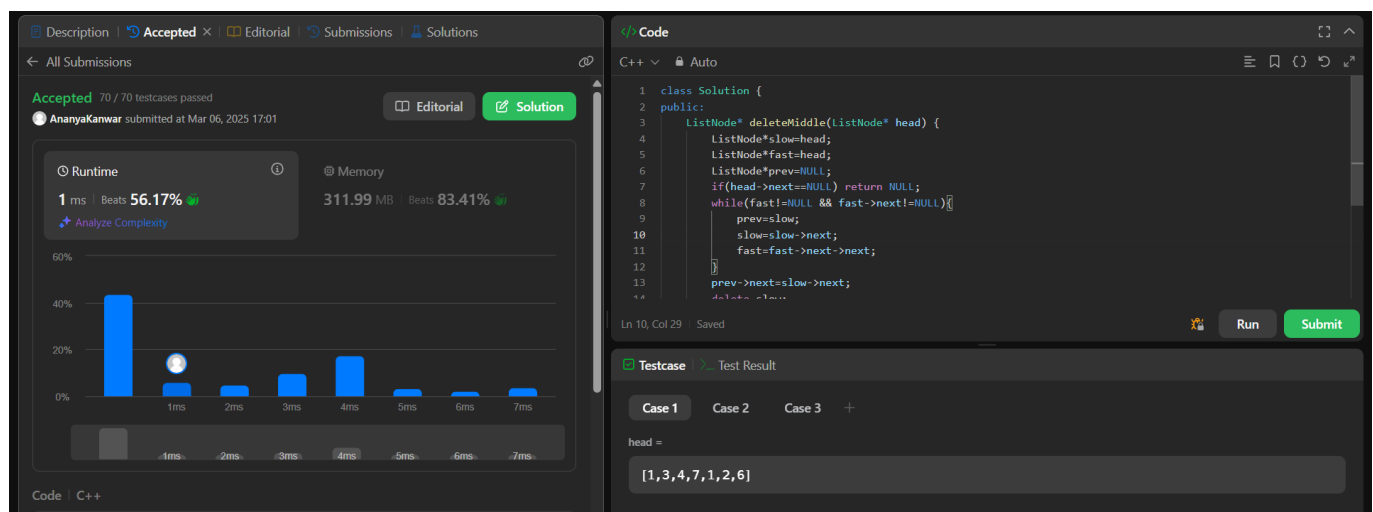
Discover. Learn. Empower.

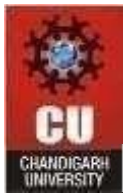
Ques 4: Delete the Middle Node of a Linked List

Code: -

```
class Solution {  
public:  
    ListNode* deleteMiddle(ListNode* head) {  
        ListNode*slow=head;  
        ListNode*fast=head;  
        ListNode*prev=NULL;  
        if(head->next==NULL) return NULL;  
        while(fast!=NULL && fast->next!=NULL){  
            prev=slow;  
            slow=slow->next;  
            fast=fast->next->next;  
        }  
        prev->next=slow->next;  
        delete slow;  
        return head;  
    }  
};
```

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

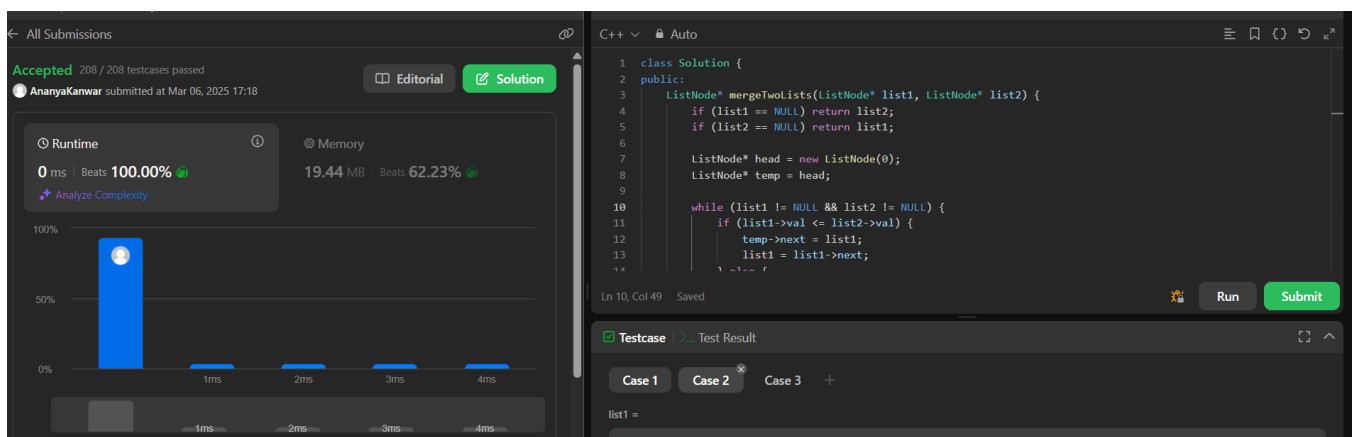
Discover. Learn. Empower.

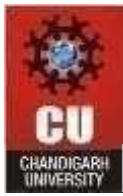
Ques 5: Merge Two Sorted Lists

Code: -

```
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* list1, ListNode* list2) {
        if (list1 == NULL) return list2;
        if (list2 == NULL) return list1;
        ListNode* head = new ListNode(0);
        ListNode* temp = head;
        while (list1 != NULL && list2 != NULL) {
            if (list1->val <= list2->val) {
                temp->next = list1;
                list1 = list1->next;
            } else {
                temp->next = list2;
                list2 = list2->next;
            }
            temp = temp->next;
        }
        if (list1 != NULL) temp->next = list1;
        if (list2 != NULL) temp->next = list2;
        return head->next;
    }
};
```

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

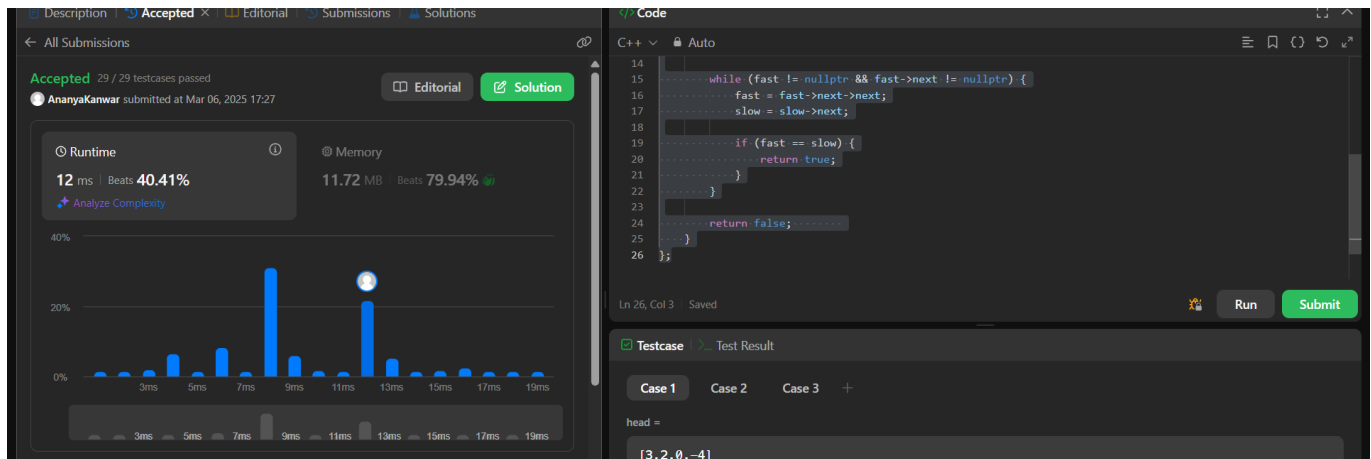
Discover. Learn. Empower.

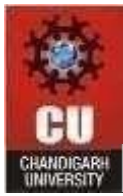
Ques 6: Linked List Cycle

Code: -

```
class Solution {
public:
    bool hasCycle(ListNode *head) {
        ListNode* fast = head;
        ListNode* slow = head;
        while (fast != nullptr && fast->next != nullptr) {
            fast = fast->next->next;
            slow = slow->next;
            if (fast == slow) {
                return true;
            }
        }
        return false;
    }
};
```

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

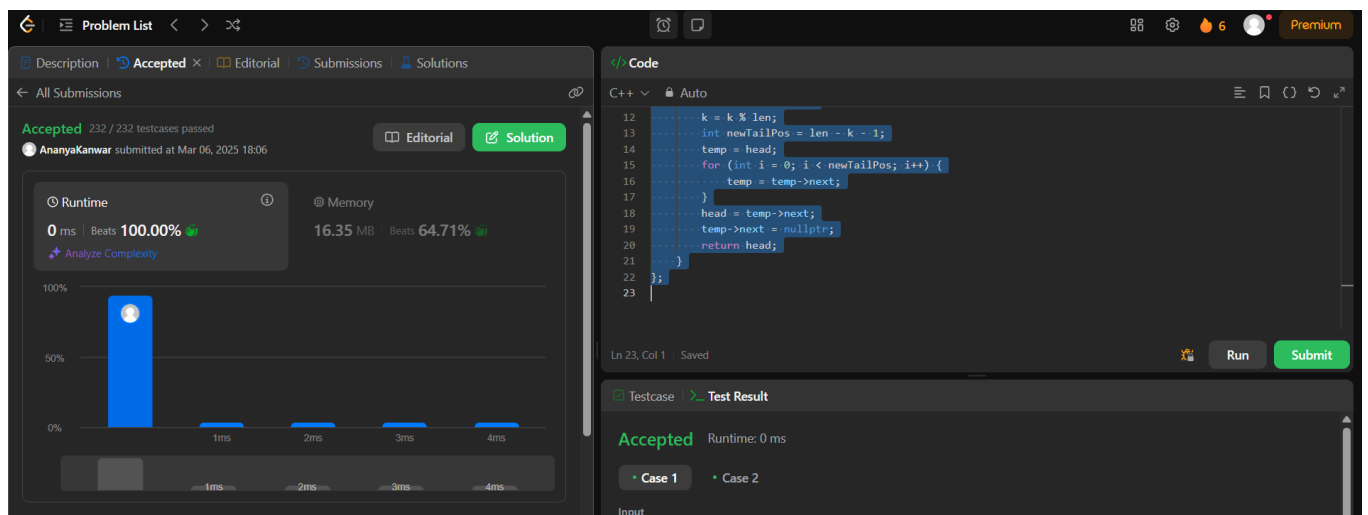
Discover. Learn. Empower.

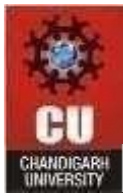
Ques 7: Rotate Lists

Code: -

```
class Solution {
public:
    ListNode* rotateRight(ListNode* head, int k) {
        if (!head || !head->next || k == 0) return head;
        ListNode* temp = head;
        int len = 1;
        while (temp->next) {
            temp = temp->next;
            len++;
        }
        temp->next = head;
        k = k % len;
        int newTailPos = len - k - 1;
        temp = head;
        for (int i = 0; i < newTailPos; i++) {
            temp = temp->next;
        }
        head = temp->next;
        temp->next = nullptr;
        return head;
    }
};
```

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Ques 8: Sort Lists

Code: -

```
class Solution {
public:
    ListNode* merge(ListNode* l1, ListNode* l2) {
        if (!l1) return l2;
        if (!l2) return l1;

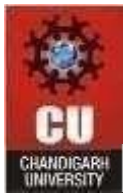
        if (l1->val < l2->val) {
            l1->next = merge(l1->next, l2);
            return l1;
        } else {
            l2->next = merge(l1, l2->next);
            return l2;
        }
    }

    ListNode* findMid(ListNode* head) {
        ListNode* slow = head, *fast = head->next;
        while (fast && fast->next) {
            slow = slow->next;
            fast = fast->next->next;
        }
        return slow;
    }

    ListNode* sortList(ListNode* head) {
        if (!head || !head->next) return head;

        ListNode* mid = findMid(head);
        ListNode* right = mid->next;
        mid->next = nullptr;

        ListNode* leftSorted = sortList(head);
        ListNode* rightSorted = sortList(right);
```

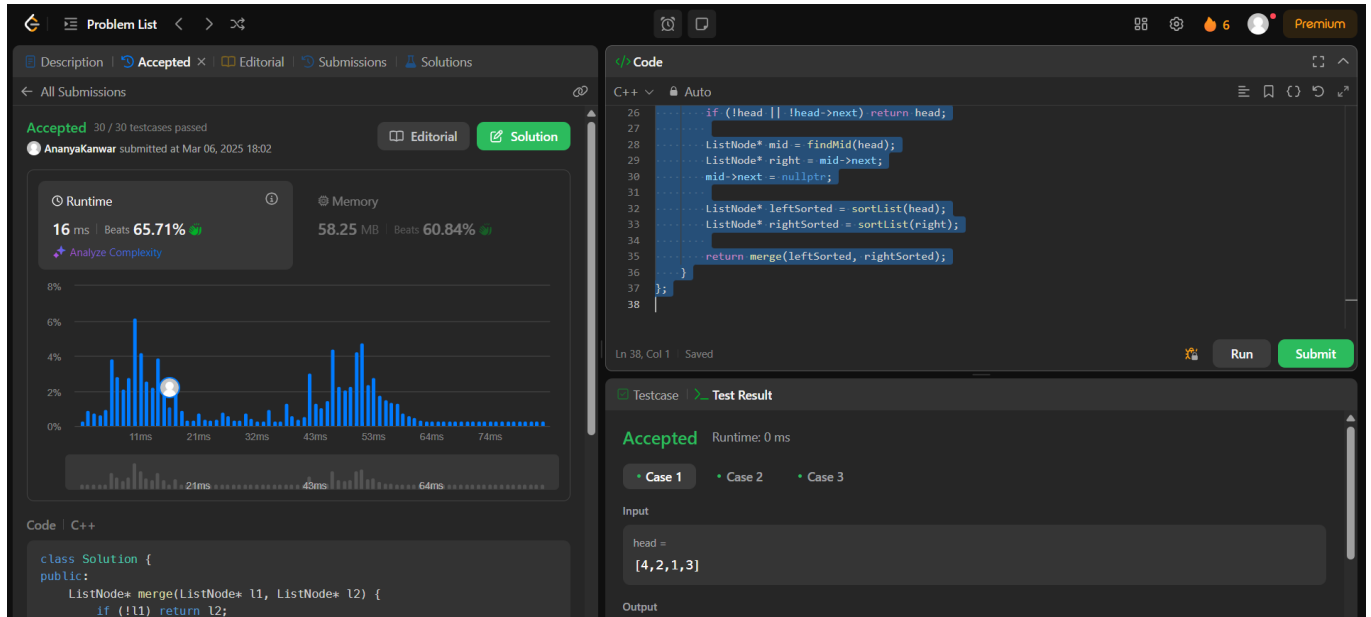



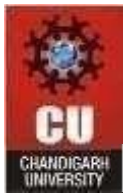
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        return merge(leftSorted, rightSorted);  
    }  
};
```

Submission: -





DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Ques 9: Merge k sorted lists

Code: -

```
#include <queue>

class Solution {
public:
    struct Compare {
        bool operator()(ListNode* a, ListNode* b) {
            return a->val > b->val; // Min-heap based on node values
        }
    };

    ListNode* mergeKLists(vector<ListNode*>& lists) {
        priority_queue<ListNode*, vector<ListNode*>, Compare> pq;

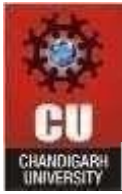
        // Push all non-null list heads into the priority queue
        for (auto list : lists) {
            if (list) pq.push(list);
        }

        ListNode dummy(0);
        ListNode* tail = &dummy;

        while (!pq.empty()) {
            ListNode* node = pq.top();
            pq.pop();
            tail->next = node;
            tail = node;

            if (node->next) pq.push(node->next);
        }

        return dummy.next;
    }
};
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Submission: -

Problem List

Description | Accepted | Editorial | Submissions | Solutions

All Submissions

Accepted 134 / 134 testcases passed

AnanyaKanwar submitted at Mar 06, 2025 18:10

Editorial Solution

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

18.47 MB | Beats 66.07%

75%

50%

25%

0%

1ms 25ms 50ms 75ms 100ms 125ms 150ms 175ms

Code | C++

```
#include <queue>

class Solution {
public:
    struct Compare {
        bool operator()(ListNode* a, ListNode* b) {
```

Code

C++ Auto

```
22 while (!pq.empty()) {
23     ListNode* node = pq.top();
24     pq.pop();
25     tail->next = node;
26     tail = node;
27     if (node->next) pq.push(node->next);
28 }
29 return dummy->next;
30 }
31 }
32 }
33 }
34 }
```

Ln 34, Col 1 | Saved

Run Submit

Testcase | Test Result

Accepted Runtime: 0 ms

Case 1 Case 2 Case 3

Input

lists =

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

Output

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



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.