

1.Print element of Linked list

Code:

```
//{ 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  
*/
```

```
class Solution {  
    public:  
  
    void printList(Node *head) {  
        Node *current = head;  
        while(current != NULL) {  
            cout << current->data << " ";  
            current = current->next;  
        }  
    }  
};
```

```
//{ Driver Code Starts.
```

```
int main() {  
    int t;  
    cin >> t;  
    cin.ignore(); // Ignore the newline character after t  
  
    while (t-->0) {  
        string input;  
        getline(cin, input); // Read the entire line for the array elements
```

```

stringstream ss(input);

Node *head = nullptr, *tail = nullptr;

int x;

// Building the linked list from input
while (ss >> x) {
    if (head == nullptr) {
        head = new Node(x);
        tail = head;
    } else {
        tail->next = new Node(x);
        tail = tail->next;
    }
}

Solution ob;
ob.printList(head);

cout << endl;
cout << "~" << endl;
}

return 0;
}

// } Driver Code Ends

```

Output:

The screenshot shows a C++ IDE interface. On the left, the 'Output Window' is open, displaying 'Compilation Results'. It indicates 'Problem Solved Successfully' with a green checkmark. Below this, statistics are shown: 'Test Cases Passed: 1112 / 1112', 'Attempts: Correct / Total: 1 / 1', 'Accuracy: 100%', 'Points Scored: 1 / 1', and 'Time Taken: 0.07'. A 'Solve Next' button is at the bottom left. On the right, the code editor shows C++ code for a linked list. It includes a 'Node' struct, a 'Node(int x)' constructor, and a 'printList' function that traverses the list and prints its elements. The code is in C++ (g++ 5.4) and has a 'Start Timer' button at the top right.

2. Remove-duplicates-from-sorted-list

Code:

```
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;
```

```
            }
```

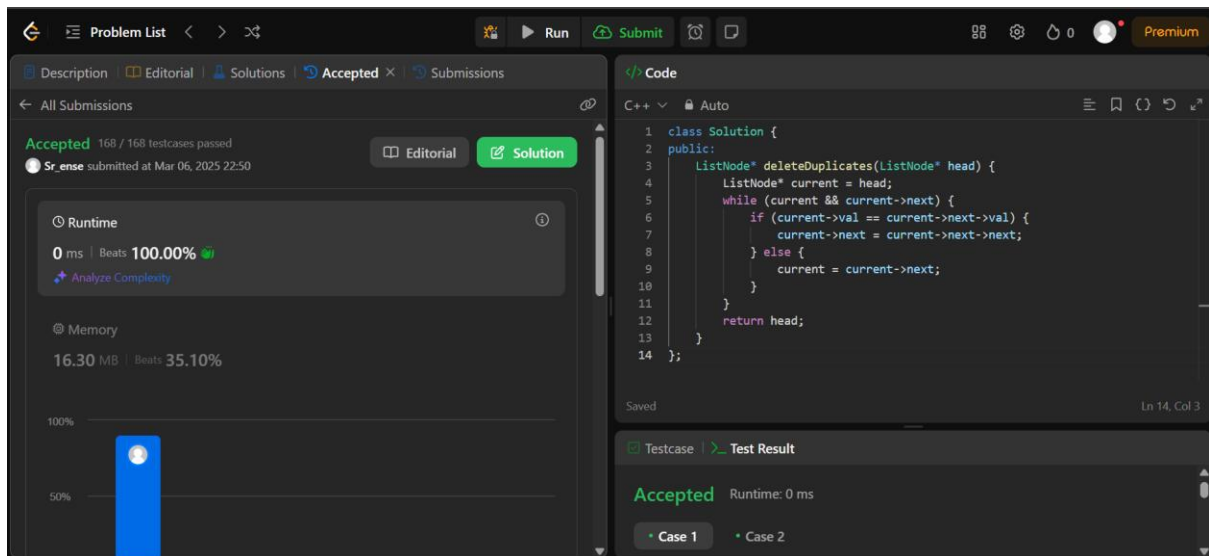
```
        }
```

```
        return head;
```

```
    }
```

```
};
```

Output:



3.Reverse a linked list

Code:

```
class Solution {
```

```
public:
```

```
    ListNode* reverseList(ListNode* head) {
```

```
        // Initialize pointers
```

```
        ListNode* prev = nullptr; // Previous node starts as NULL
```

```
        ListNode* next = nullptr; // Next node
```

```
        ListNode* curr = head; // Current node starts at the head
```

```
        // Traverse the list
```

```
        while (curr != nullptr) {
```

```
            // Save the next node
```

```
            next = curr->next;
```

```
            // Reverse the link
```

```
            curr->next = prev;
```

```
            // Move pointers forward
```

```
            prev = curr; // Move prev to the current node
```

```

        curr = next; // Move curr to the next node
    }

    // prev is now the new head of the reversed list
    return prev;
}
};

```

Output:

The screenshot shows a C++ IDE interface. On the left, the 'Submissions' tab is active, showing an 'Accepted' status for a submission by 'Sr_ense' on March 06, 2025. The runtime is 0 ms (100.00% beats) and memory is 13.47 MB (39.75% beats). On the right, the 'Code' editor shows the following C++ code:

```

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) {
11            // Save the next node
12            next = curr->next;
13
14            // Reverse the link
15            curr->next = prev;

```

At the bottom, the 'Test Result' panel shows three test cases, all of which are 'Accepted' with a runtime of 0 ms.

4. Delete the middle node of the linked list

Code:

```

/**
 * 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==NULL || head->next==NULL) return NULL;

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

```

Output:

