



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

WORKSHEET 2

Student Name: Parth Arora

UID: 22BCS16661

Branch: BE-CSE

Section/Group: 22BCS_NTPP-602-A

Semester: 6th

Date of Performance: 22/01/2025

Subject Name: AP LAB - II

Subject Code: 22CSP-351

1. Aim: Given a linked list. Print all the elements of the linked list separated by space followed.

2. Source Code:

```
class Solution {  
    // Function to display the elements of a linked list in the same line  
    void printList(Node head) {  
        // Traverse the linked list  
        while (head != null) {  
            // Print the data of the current node followed by a space  
            System.out.print(head.data + " ");  
            // Move to the next node  
            head = head.next;  
        }  
    }  
}
```

3. Screenshots of outputs:

The screenshot displays a coding interface with two main panels. The left panel, titled 'Output Window', shows 'Compilation Results' with a green checkmark indicating 'Problem Solved Successfully'. It also displays 'Test Cases Passed' as 1112 / 1112, 'Attempts' as 2 / 4, and 'Accuracy' as 50%. The right panel shows the source code in Java 1.8, which includes a Node class and a Solution class with a printList method. The code is identical to the one provided in the previous block.

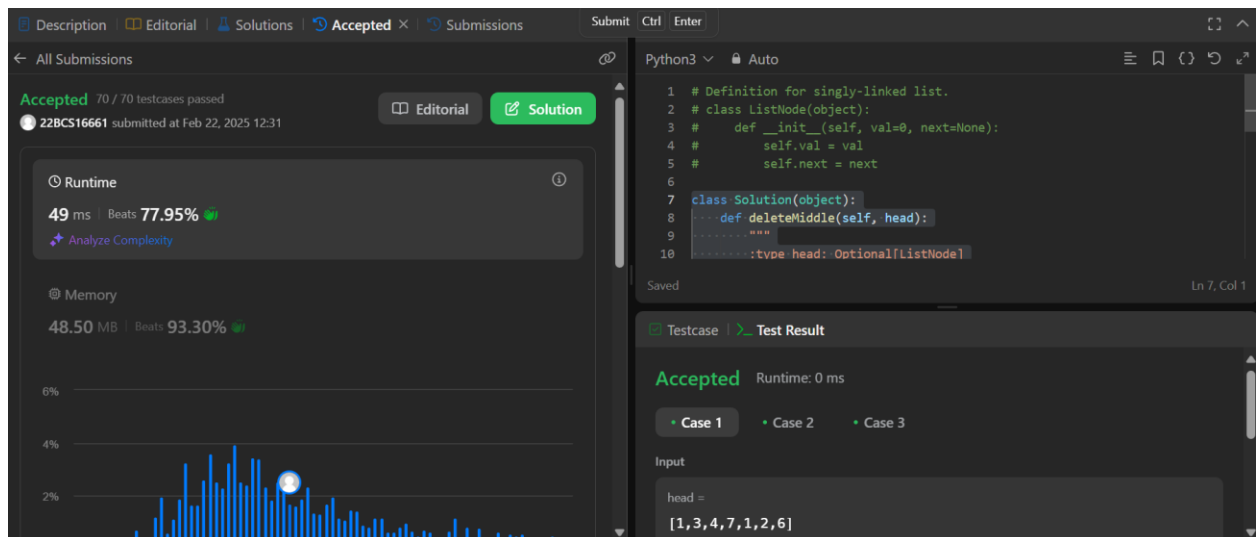
2.

Aim: You are given the head of a linked list. **Delete** the **middle node**, and return *the head of the modified linked list*.

Source Code:

```
class Solution(object):
    def deleteMiddle(self, head):
        """
        :type head: Optional[ListNode]
        :rtype: Optional[ListNode]
        """
        if not head or not head.next:
            return None
        slow = head
        fast = head
        prev = None
        while fast and fast.next:
            fast = fast.next.next
            prev = slow
            slow = slow.next
        prev.next = slow.next
        return head
```

Screenshots of outputs:



3.

Aim: Given head, the head of a linked list, determine if the linked list has a cycle in it. There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the next pointer. Internally, pos is used to denote the index of the node that tail's next pointer is connected to. **Note that pos is not passed as a parameter.** Return true *if there is a cycle in the linked list*. Otherwise, return false.

Source Code:

```
class Solution:
    def hasCycle(self, head: Optional[ListNode]) -> bool:

        fast = head
        slow = head

        while fast and fast.next:
            fast = fast.next.next
            slow = slow.next

        if fast == slow:
            return True

        return False
```

4. Screenshots of outputs:

