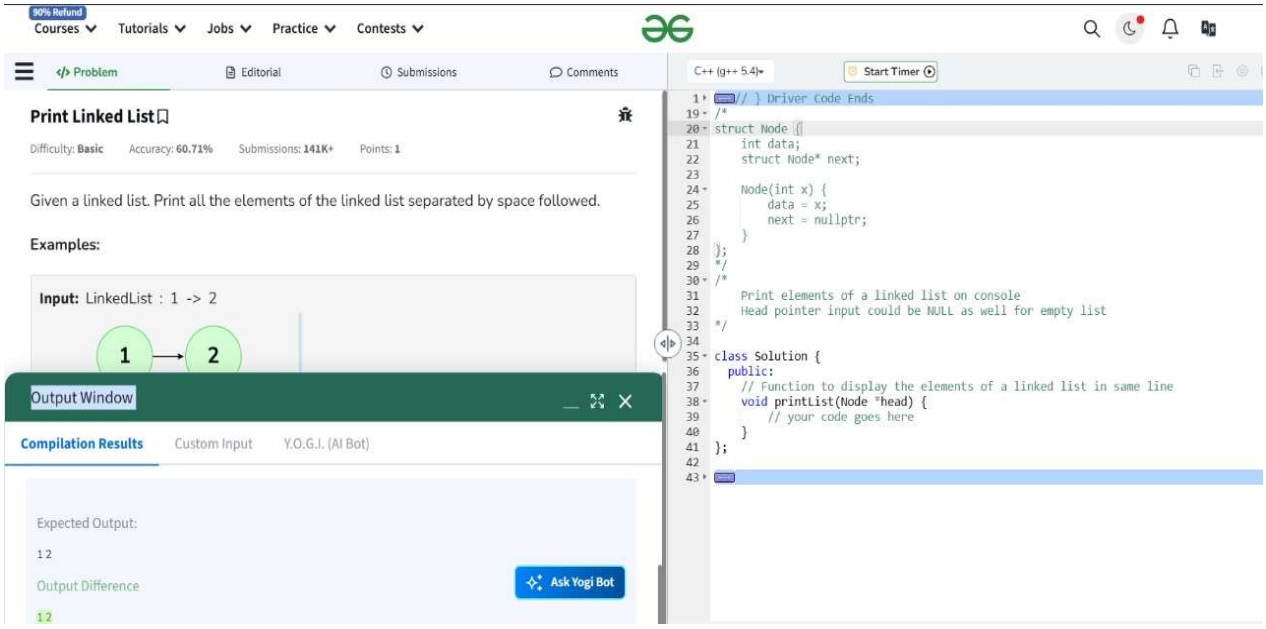
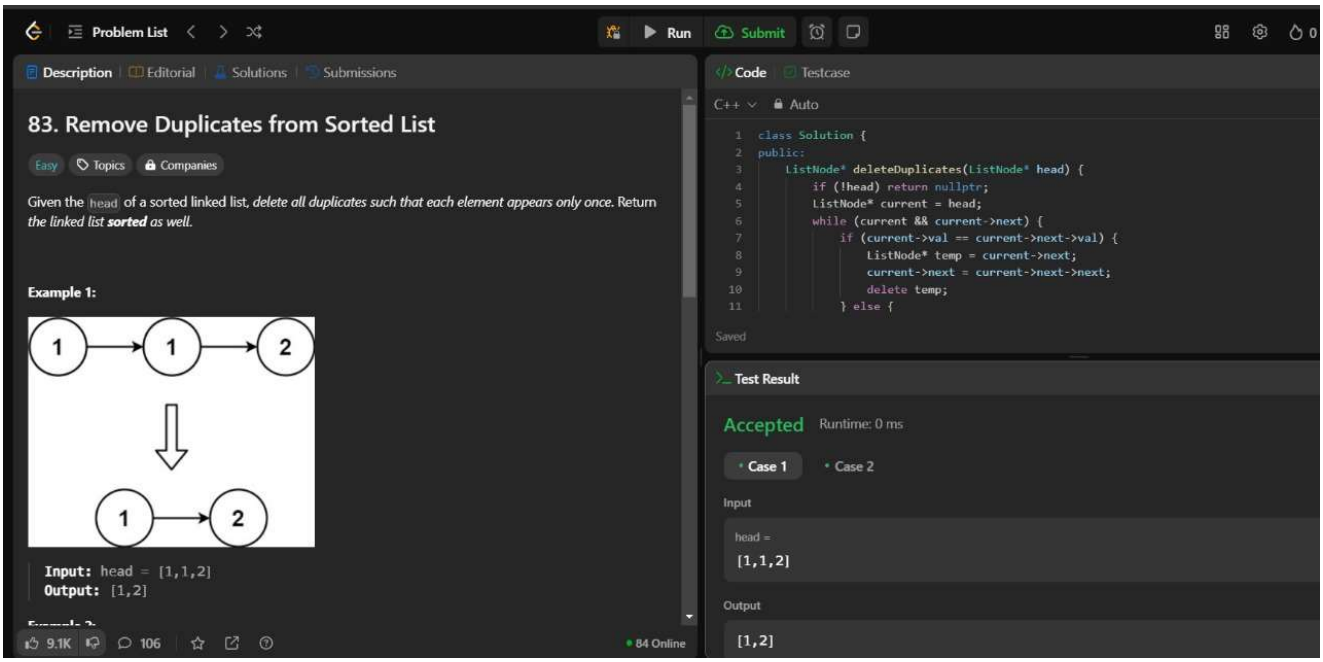
NAME: KRISH CHAUDHARY **SUBJECT: ADVANCED PROGRAMMING**

UID: 22BCS11598 **SECTION: IOT_612/B**





Problem List | Run | Submit

Description | Accepted | Editorial | Solutions | Submissions

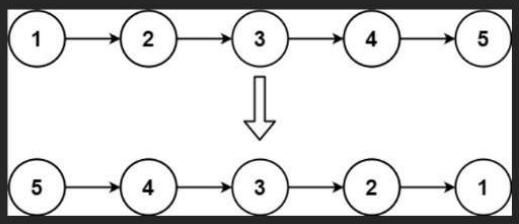
206. Reverse Linked List

Solved

Easy | Topics | Companies

Given the `head` of a singly linked list, reverse the list, and return *the reversed list*.

Example 1:



Input: head = [1,2,3,4,5]
Output: [5,4,3,2,1]

Example 2:

22.5K | 265 | 371 Online

Code | Testcase

C++ | Auto

```
        while (current) {
            ListNode* nextNode = current->next; // Store next node
            current->next = prev; // Reverse the link
            prev = current; // Move prev to current
            current = nextNode; // Move current to next
        }
        return prev; // New head of the reversed list
    }
};
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

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

Output

[5,4,3,2,1]

Problem List | Run | Submit

Description | Editorial | Solutions | Submissions

2095. Delete the Middle Node of a Linked List

Medium | Topics | Companies | Hint

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

The **middle node** of a linked list of size `n` is the $\lfloor n / 2 \rfloor^{th}$ node from the **start** using **0-based indexing**, where $\lfloor x \rfloor$ denotes the largest integer less than or equal to `x`.

- For `n` = `1`, `2`, `3`, `4`, and `5`, the middle nodes are `0`, `1`, `1`, `2`, and `2`, respectively.

Example 1:

Input: head = [1,3,4,7,1,2,6]
Output: [1,3,4,1,2,6]
Explanation:
The above figure represents the given linked list. The indices of the nodes are written below.
Since n = 7, node 3 with value 7 is the middle node, which is marked in red.
We return the new list after removing this node.

4.4K | 71 | 67 Online

Code | Testcase

C++ | Auto

```
        fast = fast->next->next;
    }

    prev->next = slow->next; // Delete the middle node
    delete slow;

    return head;
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

head =
[1,3,4,7,1,2,6]

Output

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

Expected

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

Problem List | Run | Submit

Description | Editorial | Solutions | Submissions

21. Merge Two Sorted Lists

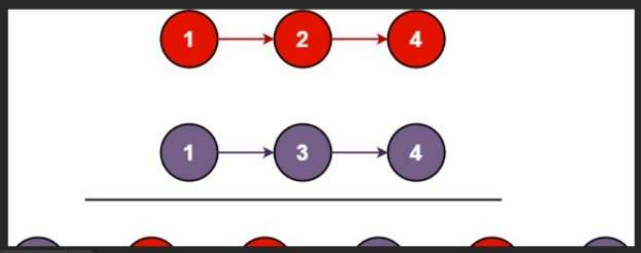
Easy | Topics | Companies

You are given the heads of two sorted linked lists `list1` and `list2`.

Merge the two lists into one **sorted** list. The list should be made by splicing together the nodes of the first two lists.

Return *the head of the merged linked list.*

Example 1:



22.9K | 411 | 480 Online

Code | Testcase

C++ | Auto

```
        ListNode* mergedHead = dummy->next;
        delete dummy;
        return mergedHead;
    }
};
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

list1 =
[1,2,4]

list2 =
[1,3,4]

Output

[1,1,2,3,4,4]

Problem List | Run | Submit

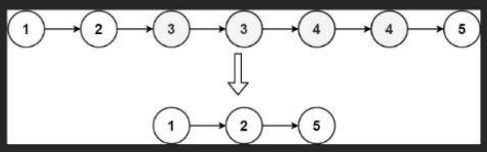
Description | Editorial | Solutions | Submissions

82. Remove Duplicates from Sorted List II

Medium | Topics | Companies

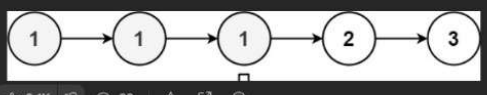
Given the `head` of a sorted linked list, *delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list.* Return *the linked list* ***sorted*** *as well.*

Example 1:




```
Input: head = [1,2,3,3,4,4,5]
Output: [1,2,5]
```

Example 2:



9.1K | 82 | 62 Online

Code | Testcase

C++ | Auto

```
        ListNode* newHead = dummy->next;
        delete dummy;
        return newHead;
    }
};
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2

Input

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

Output

[1,2,5]

Expected

[1,2,5]

Problem List | Run | Submit

Description | Editorial | Solutions | Submissions

141. Linked List Cycle

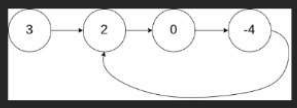
Easy | Topics | Companies

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`.

Example 1:




```
Input: head = [3,2,0,-4], pos = 1
Output: true
Explanation: There is a cycle in the linked list, where the tail connects to
the 1st node (0-indexed).
```

Example 2:

16.2K | 352 | 199 Online

Code | Testcase

C++ | Auto

```
        }

        return false;
    }
};
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

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

pos =
1

Output

true

Problem List | Run | Submit

Description | Editorial | Solutions | Submissions

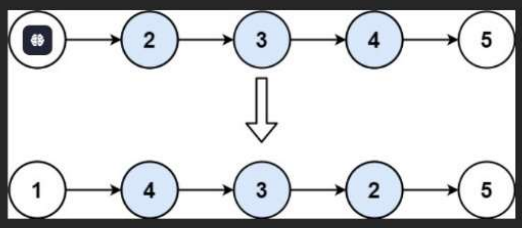
Run Ctrl '

92. Reverse Linked List II

Medium | Topics | Companies

Given the `head` of a singly linked list and two integers `left` and `right` where `left <= right`, reverse the nodes of the list from position `left` to position `right`, and return *the reversed list.*

Example 1:




```
Input: head = [1,2,3,4,5], left = 2, right = 4
Output: [1,4,3,2,5]
```

Example 2:

12K | 151 | 143 Online

Code | Testcase

C++ | Auto

```
            prev->next = nextNode;
        }

        return dummy->next;
    }
};
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2

Input

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

left =
2

right =
4

Problem List | Run | Submit

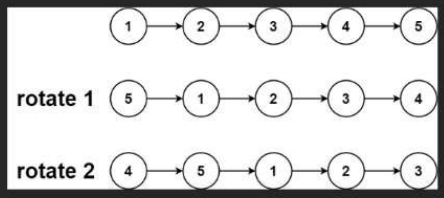
Description | Editorial | Solutions | Submissions

61. Rotate List

Medium | Topics | Companies

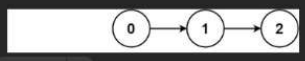
Given the `head` of a linked list, rotate the list to the right by `k` places.

Example 1:




```
Input: head = [1,2,3,4,5], k = 2
Output: [4,5,1,2,3]
```

Example 2:



10.2K | 101 | 108 Online

Code | Testcase

C++ | Auto

```
23          }
24
25          // Break the loop and set the new head
26          head = temp->next;
27          temp->next = nullptr;
28
29          return head;
30      }
31  };
32
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2

Input

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

k =
2

Problem List | Run | Submit

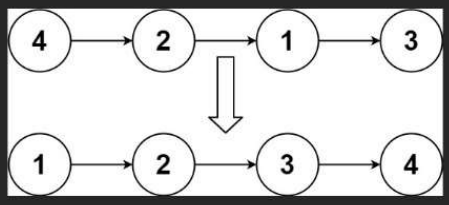
Description | Editorial | Solutions | Submissions

148. Sort List

Medium | Topics | Companies

Given the `head` of a linked list, return *the list after sorting it in* ***ascending order***.

Example 1:




```
Input: head = [4,2,1,3]
Output: [1,2,3,4]
```

Example 2:

12.1K | 108 | 134 Online

Code | Testcase

C++ | Auto

```
39              tail = tail->next;
40          }
41          tail->next = l1 ? l1 : l2;
42          return dummy.next;
43      }
44  };
45
```

Saved

Test Result

Accepted Runtime: 0 ms

Case 1 | Case 2 | Case 3

Input

head =
[4,2,1,3]

Output

[1,2,3,4]

Problem List | Run | Submit

Description | Editorial | Solutions | Submissions

142. Linked List Cycle II

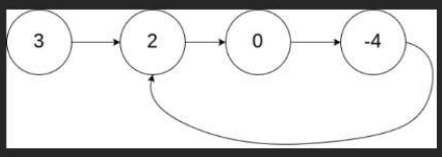
Medium | Topics | Companies

Given the `head` of a linked list, return *the node where the cycle begins. If there is no cycle, return* `null`.

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 (**0-indexed**). It is `-1` if there is no cycle. **Note that** `pos` **is not passed as a parameter**.

Do not modify the linked list.

Example 1:




```
Input: head = [3,2,0,-4], pos = 1
Output: tail connects to node index 1
Explanation: There is a cycle in the linked list, where tail connects to the
```

14.1K | 179 | 112 Online

Code | Testcase

C++ | Auto

```
                    slow = slow->next;
                    fast = fast->next;
                }
                return slow;
            }
        }
        return nullptr;
    }
};
```

Saved

Test Result

Accepted Runtime: 2 ms

Case 1 | Case 2 | Case 3

Input

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

pos =
1