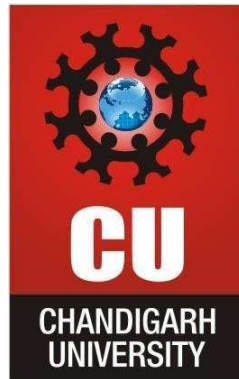




**CHANDIGARH
UNIVERSITY**
Discover. Learn. Empower.



Subject: Advance programming lab

Subject code: 22CSH-351

Assignment No.: 3

Student Name: Muskan

Submitted to: Er. Pratima sonalie

UID: 22BCS13343

Date of Submission: 04-03-25

Branch: BE-CSE

Section/Group: 22BCS_IOT-607(B)

90% Refund

Courses ▾

Tutorials ▾

Jobs ▾

Practice ▾

Contests ▾

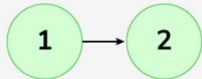
Print Linked List

Difficulty: Basic Accuracy: 60.71% Submissions: 142K+ Points: 1

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

Examples:

Input: LinkedList : 1 -> 2



Output: 1 2

Explanation: The linked list contains two elements 1 and 2. The elements are printed in a single line.

Input: Linked List : 49 -> 10 -> 30

C++ (g++ 5.4)

Start Timer

```
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         while (head != nullptr) {
40             cout << head->data << " ";
41             head = head->next;
42         }
43         //22BCS13343 MUSKAN
44     }
45 };
46 // } Driver Code Ends
```

Custom Input

Compile & Run

Submit

Top Stories

Virat Kohli emul...

Search

ENG IN

21:05

04-03-2025

[illegible]

Open-End xInbox (6,8: xmarksheet xWelcome t xChandigar xRemove D xNew Doc xScreensho x+
leetcode.com/problems/remove-duplicates-from-sorted-list/

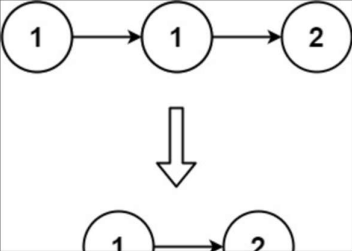
Problem List <> Run Submit
Description | Editorial | Solutions | Submissions

83. Remove Duplicates from Sorted List

EasyTopicsCompanies

Given the `head` of a sorted linked list, delete all duplicates such that each element appears only once. Return the linked list **sorted** as well.

Example 1:



9.1K10899 Online

Code | Testcase | Test Result

C++ Auto

```
11 class Solution {
12 public:
13     ListNode* deleteDuplicates(ListNode* head) {
14         if(head == nullptr || head->next == nullptr) return head;
15         ListNode *s1 = head;
16         ListNode *s2 = s1->next;
17         while( s1 != nullptr && s1->next != nullptr)
18         {
19             if(s1->val == s2->val)
20             {
21                 s1 -> next = s2 -> next;
22                 s2 = s1->next;
23             }
24             else
25             {
26                 s1 = s1->next;
27             }
28         }
29         return head; //22BCS13343 MUSKAN
30     }
31 };
```

SavedLn 29, Col 40

Temps to rise
Thursday

Search

ENG
IN

21:06
04-03-2025

Open-End xInbox (6,8 xmarksheet xWelcome t xChandigar xRemove D xNew Doc xScreensho x+
leetcode.com/problems/remove-duplicates-from-sorted-list/

Problem List <> < > < >

RunSubmit

0000Premium

DescriptionAccepted xEditorialSolutionsSubmissions

All Submissions

Accepted 168 / 168 testcases passed
Muskan submitted at Mar 04, 2025 21:07

EditorialSolution

Runtime 0 ms | Beats 100.00%
Analyze Complexity

Memory 16.22 MB | Beats 35.01%

100%
50%
0%
1ms2ms3ms4ms

1ms2ms3ms4ms

CodeTestcaseTest Result

Accepted Runtime: 0 ms

Case 1Case 2

Input
head =
[1,1,2]

Output
[1,2]

Expected
[1,2]

Contribute a testcase

6 Temps to rise Thursday

Search

ENG IN 21:07 04-03-2025

Open-End xInbox (6,8 xmarksheet xWelcome t xChandigar xReverse Li xNew Doc xScreensho x+
leetcode.com/problems/reverse-linked-list/description/

Problem List <> Run Submit

DescriptionEditorialSolutionsSubmissions

206. Reverse Linked List

EasyTopicsCompanies

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

Example 1:

```
graph LR; 1((1)) --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5)); 5 --> null; style null fill:none,stroke:none; 5r((5)) --> 4r((4)); 4r --> 3r((3)); 3r --> 2r((2)); 2r --> 1r((1)); 1r --> nullr; style nullr fill:none,stroke:none;
```

22.6K272367 Online

CodeTestcaseTest Result

C++Auto

```
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* reverseList(ListNode* head) {
14         ListNode* node = nullptr;
15
16         while (head != nullptr) {
17             ListNode* temp = head->next;
18             head->next = node;
19             node = head;
20             head = temp;
21         }
22         //228CS13343 MUSKAN
23         return node;
24     }
25 };
```

SavedLn 22, Col 28

6 Temps to rise Thursday

Search

ENG IN21:0804-03-2025

Open-Endi xInbox (6,8: xmarksheet xWelcome t xChandigar xReverse Li xNew Doc xScreensho x+

leetcode.com/problems/reverse-linked-list/submissions/1562711288/

Problem List<>RunSubmit

DescriptionAccepted xEditorialSolutionsSubmissions

All Submissions

Accepted28 / 28 testcases passed

Muskan submitted at Mar 04, 2025 21:08

EditorialSolution

Runtime0 ms | Beats 100.00%

Memory13.41 MB | Beats 39.71%

Analyze Complexity

150%100%50%0%

1ms2ms3ms4ms

CodeTestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2Case 3

Input

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

Output

[5,4,3,2,1]

Expected

[5,4,3,2,1]


Contribute a testcase

5 Temps to rise Thursday

Search

ENG IN

21:0804-03-2025

Open-End xInbox (6,8: xmarksheet xWelcome t xChandigar xDelete the xNew Doc xScreensho x+
leetcode.com/problems/delete-the-middle-node-of-a-linked-list/
Problem List <> Run Submit
Description | Editorial | Solutions | Submissions
2095. Delete the Middle Node of a Linked List Solved
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 $\lfloor n / 2 \rfloor^{\text{th}}$ 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]
4.5K 77 61 Online
Air: Moderate Now
Search
21:09 04-03-2025

Code | Testcase | Test Result
Java Auto
10 */
11 class Solution {
12 public ListNode deleteMiddle(ListNode head) {
13 int count = 0;
14 ListNode current = head;
15 while (current != null){
16 count++;
17 current = current.next;
18 }
19 current = head;
20 if(count == 1){
21 return current.next;
22 }
23 int n = (count / 2) - 1;
24 for(int i = 0; i < n; i++){
25 current = current.next;
26 }
27 current.next = current.next.next;
28 return head; //22BC513343 MUSKAN
29
30 }
31 }

Open-End xInbox (6,8 xmarksheet xWelcome t xChandigar xDelete the xNew Doc xScreensho x+
leetcode.com/problems/delete-the-middle-node-of-a-linked-list/submissions/1562712190/

Problem List<>RunSubmitRunSubmitPremium

DescriptionAccepted xEditorialSolutionsSubmissions

All Submissions

Accepted70 / 70 testcases passed
Muskan submitted at Mar 04, 2025 21:09

EditorialSolution

Runtime
3 ms | Beats 99.72%
Analyze Complexity

Memory
63.01 MB | Beats 81.86%

Runtime (ms)	Beats (%)
2ms	~10%
3ms	99.72%
4ms	~10%
5ms	~10%
6ms	~10%

CodeTestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2Case 3

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

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

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

Contribute a testcase

Air: Moderate Now

Search

ENG IN21:09 04-03-2025

Open-End... xInbox (6,8... xmarksheet xWelcome t xChandigar xLinked List xNew Doc xScreensho x+

leetcode.com/problems/linked-list-cycle/

Problem List <>🔍RunSubmit🔧🔧🔧🔧🔧Premium

DescriptionEditorialSolutionsSubmissions

141. Linked List Cycle

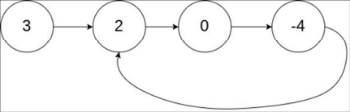
EasyTopicsCompanies

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:



16.2K360240 Online

CodeTestcaseTest Result

C++Auto

5ListNode* next;
6 *
7 ListNode(int x) : val(x), next(NULL) {}
8 *};
9
10 class Solution {
11 public:
12 bool hasCycle(ListNode* head) {
13 ListNode* slow = head;
14 ListNode* fast = head;
15 while (fast != NULL && fast->next != NULL) {
16 slow = slow->next;
17 fast = fast->next->next;
18 if (slow == fast) {
19 return true;
20 }
21 }
22 return false; //22BCS13343 MUSKAN
23 };
24 }

SavedLn 21, Col 42

Trending videosWhat Makes Pe...

Search

ENG IN21:1204-03-2025

Open-End xInbox (6,8 xmarksheet xWelcome t xChandigar xLinked List xNew Doc xScreensho x+
leetcode.com/problems/linked-list-cycle/submissions/1562715257/

Problem List<>RunSubmit

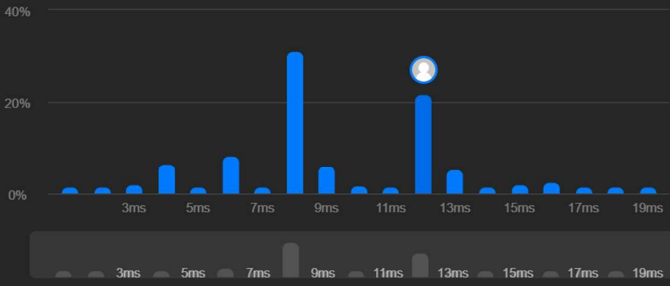
DescriptionAccepted xEditorialSolutionsSubmissions

All Submissions

Accepted 29 / 29 testcases passed
Muskan submitted at Mar 04, 2025 21:13

Runtime
12 msBeats 40.46%
Analyze Complexity

Memory
11.74 MBBeats 79.79%



CodeTestcaseTest Result

AcceptedRuntime: 0 ms

Case 1Case 2Case 3

Input

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

pos =
1

Output

true

Expected

true

Contribute a testcase

Trending videos
What Makes Pe...

Search

ENG IN21:13
04-03-2025

Open-End xInbox (6,8: xmarksheet xWelcome t xChandigar xRotate List xNew Doc xScreensho x+
leetcode.com/problems/rotate-list/

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:

1 → 2 → 3 → 4 → 5

rotate 1 5 → 1 → 2 → 3 → 4

rotate 2 4 → 5 → 1 → 2 → 3

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

10.2K 104 105 Online

Code | Testcase | Test Result

Java Auto

```
11 class Solution {
12     public ListNode rotateRight(ListNode head, int k) {
13         if (head == null || head.next == null || k == 0) {
14             return head; //22BCS13343 MUSKAN
15         }
16         ListNode temp = head;
17         int length = 1;
18         while (temp.next != null) {
19             temp = temp.next;
20             length++;
21         }
22         temp.next = head;
23         k = k % length;
24         int breakPoint = length - k;
25         ListNode newTail = head;
26         for (int i = 1; i < breakPoint; i++) {
27             newTail = newTail.next;
28         }
29         head = newTail.next;
30         newTail.next = null;
31         return head;
32     }
33 }
```

Saved Ln 14, Col 44

59°F Clear

Search

ENG IN 21:14 04-03-2025

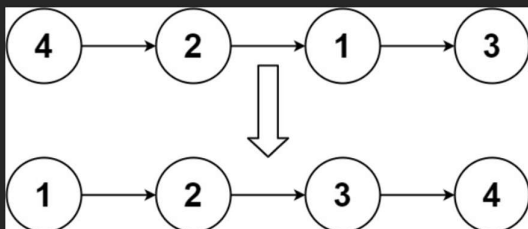
Open-End xInbox (6,8 xmarksheet xWelcome t xChandigar xRotate List xNew Doc xScreensho x+
leetcode.com/problems/rotate-list/submissions/1562716195/
Problem List <> Run Submit
Description Accepted xEditorial Solutions Submissions
All Submissions
Accepted 232 / 232 testcases passed
Muskan submitted at Mar 04, 2025 21:14
Editorial Solution
Runtime 0 ms | Beats 100.00%
Memory 42.49 MB | Beats 78.37%
Analyze Complexity
100%
0%50%100%
1ms2ms3ms4ms
1ms2ms3ms4ms
Code Testcase Test Result
Accepted Runtime: 0 ms
Case 1 Case 2
Input
head =
[1,2,3,4,5]
k =
2
Output
[4,5,1,2,3]
Expected
[4,5,1,2,3]
Contribute a testcase
59°F Clear
Search
ENG IN
21:14
04-03-2025

148. Sort List

Medium Topics Companies

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

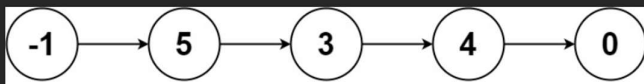
Example 1:



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

Output: [1,2,3,4]

Example 2:



[Code](#) | [Testcase](#) | [Test Result](#)

```

11 class Solution {
12     public ListNode sortList(ListNode head) {
13         if (head == null || head.next == null) return head;
14         ListNode slow = head, fast = head.next;
15         while (fast != null && fast.next != null) {
16             slow = slow.next;
17             fast = fast.next.next;
18         }
19         ListNode mid = slow.next;
20         slow.next = null;
21         ListNode left = sortList(head); // 22BCS13343 MUSKAN
22         ListNode right = sortList(mid);
23         return merge(left, right);
24     }
25     private ListNode merge(ListNode l1, ListNode l2) {
26         ListNode dummy = new ListNode(0);
27         ListNode tail = dummy;
28         while (l1 != null && l2 != null) {
29             if (l1.val < l2.val) {
30                 tail.next = l1;
31                 l1 = l1.next;
32             } else {
33                 tail.next = l2;
34                 l2 = l2.next;
35             }
36             tail = tail.next;
37         }
38         tail.next = (l1 != null) ? l1 : l2;
39         return dummy.next;
40     }
41 }

```

Problem List

Run

Submit

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted

30 / 30 testcases passed

Muskan submitted at Mar 04, 2025 21:18

Editorial

Solution

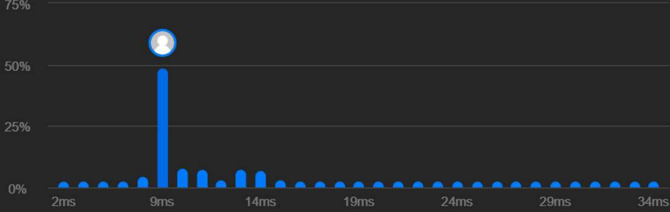
Runtime

9 ms | Beats 94.67%

Analyze Complexity

Memory

56.98 MB | Beats 45.39%



Runtime (ms)	Percentage (%)
2	~1
9	~65
14	~10
19	~5
24	~5
29	~5
34	~5

Code | Java

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

Code

Testcase

Test Result

Accepted

Runtime: 0 ms

Case 1

Case 2

Case 3

Input

head =

[4,2,1,3]

Output

[1,2,3,4]

Expected

[1,2,3,4]

Contribute a testcase

Solved

You are given an array of `k` linked-lists `lists`

Merge all the linked-lists into one sorted linked-list and return it.

Example 1:

Input: lists = [[1,4,5],[1,3,4],[2,6]]

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

Explanation: The linked-lists are:

```
[
    1->4->5,
    1->3->4,
    2->6
]
```

```
merging them into one sorted list:
```

1->1->2->3->4->4->5->6

Example 2:

Input: lists = []

Output: `[]`

Example 3:

👍 20.1K 🗨️ 253 ⭐ 📄 ⓘ

● 247 Online

[Code](#) | [Testcase](#) | [Test Result](#)

C++ Auto

```

1 #include <vector>
2 using namespace std;
3 class Solution {
4 public:
5     ListNode* mergeTwoLists(ListNode* l1, ListNode* l2) {
6         if (!l1) return l2;
7         if (!l2) return l1;
8         if (l1->val < l2->val) {
9             l1->next = mergeTwoLists(l1->next, l2);
10            return l1;
11        } else {
12            l2->next = mergeTwoLists(l1, l2->next);
13            return l2; //228CS13343 MUSKAN
14        }
15    }
16    ListNode* mergeKLists(vector<ListNode*>& lists) {
17        if (lists.empty()) return nullptr;
18        return divideAndConquer(lists, 0, lists.size() - 1);
19    }
20    ListNode* divideAndConquer(vector<ListNode*>& lists, int left, int right) {
21        if (left == right) return lists[left];
22        int mid = left + (right - left) / 2;
23        ListNode* l1 = divideAndConquer(lists, left, mid);
24        ListNode* l2 = divideAndConquer(lists, mid + 1, right);
25        return mergeTwoLists(l1, l2);
26    }
27 };

```


Problem List

Run

Submit

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted

134 / 134 testcases passed

Muskan submitted at Mar 04, 2025 21:20

Editorial

Solution

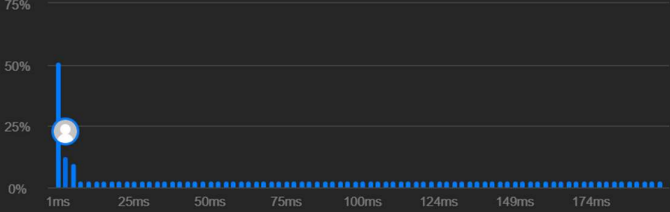
Runtime

3 ms | Beats 64.87%

Analyze Complexity

Memory

18.52 MB | Beats 50.76%



Code

C++

```
#include <vector>
using namespace std;
class Solution {
public:
    ListNode* mergeTwoLists(ListNode* l1, ListNode* l2) {
```

Code

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]

Expected

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

Contribute a testcase