

## 2095. Delete the Middle Node of a Linked List

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

Problem List

Accepted

Editorial

Solutions

Submissions

Accepted 70 / 70 testcases passed

ashima\_narula submitted at Feb 13, 2025 21:57

Editorial

Solution

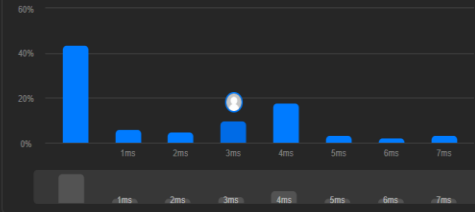
Runtime

3 ms | Beats 45.72%

Analyze Complexity

Memory

311.96 MB | Beats 83.80%



Code | C++

```
//
* 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 || !head->next) return nullptr;
        ListNode *slow = head, *fast = head, *prev = nullptr;
        while (fast && fast->next) {
            prev = slow;
            slow = slow->next;
            fast = fast->next->next;
        }
        prev->next = slow->next;
        delete slow;
        return head;
    }
};
```

Testcase

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]

Contribute a testcase