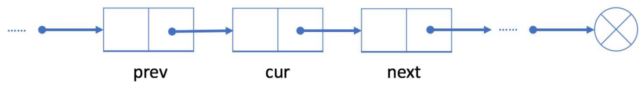
Delete Operation - Singly Linked List

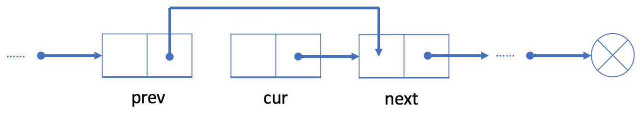
Report Issue (https://github.com/LeetCode-Feedback/LeetCode-Feedback/issues)

If we want to delete an existing node `cur` from the singly linked list, we can do it in two steps:

1. Find cur's previous node `prev` and its next node `next` ;



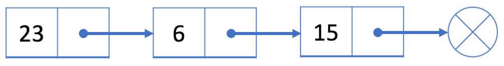
2. Link `prev` to cur's next node `next` .



In our first step, we need to find out `prev` and `next` . It is easy to find out `next` using the reference field of `cur` . However, we have to traverse the linked list from the head node to find out `prev` which will take `O(N)` time on average, where N is the length of the linked list. So the time complexity of deleting a node will be `O(N)` .

The space complexity is `O(1)` because we only need constant space to store our pointers.

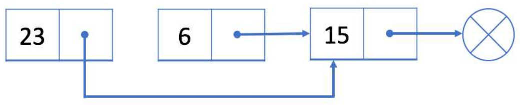
An Example



Let's try to delete node 6 from the singly linked list above.

1. Traverse the linked list from the head until we find the previous node `prev` which is node 23

2. Link `prev` (node 23) with `next` (node 15)



Node 6 is not in our singly linked list now.

Delete the First Node

If we want to delete the first node, the strategy will be a little different.

As we mentioned before, we use the head node `head` to represent a linked list. Our head is the black node 23 in the example below.