Problem | Submissions | Comments

Remove duplicates from a sorted doubly linked list

Basic | Accuracy: 49.11% | Submissions: 2K+ | Points: 1

Given a doubly linked list of **n** nodes sorted by values, the task is to remove duplicate nodes present in the linked list.

Example 1:

Input:
n = 6
1<->1<->1<->2<->3<->4
Output:
1<->2<->3<->4
Explanation:
Only the first occurance of node with value 1 is retained, rest nodes with value = 1 are deleted.

Example 2:

C++ (g++ 5.4) | Average Time: 10m | Start Timer

```
// } Driver Code Ends

class Solution
{
public:

    Node * removeDuplicates(struct Node *head)
    {
        // Your code here
        Node *curr = head , *back = head;
        while(curr != NULL){
            if(curr->data != back->data){
                back->next = curr;
                curr->prev = back;
                back = curr;
            }
            curr = curr->next;
        }
        back->next = NULL;

        return head;
    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

Problem | Editorial | Submissions | Comments

Find pairs with given sum in doubly linked list

Easy Accuracy: 66.01% Submissions: 17K+ Points: 2

Given a sorted doubly linked list of positive distinct elements, the task is to find pairs in a doubly-linked list whose sum is equal to given value **target**.

Example 1:

Input:
1 <-> 2 <-> 4 <-> 5 <-> 6 <-> 8 <-> 9
target = 7
Output: (1, 6), (2,5)
Explanation: We can see that there are two pairs (1, 6) and (2,5) with sum 7.

C++ (g++ 5.4) | Average Time: 10m | Start Timer

```cpp
// } Driver Code Ends
//User function Template for C++

class Solution
{
public:
    vector<pair<int, int>> findPairsWithGivenSum(Node *head, int target)
    {
        // code here
        vector<pair<int , int>>res ;
        Node *slow = head , *fast = head;
        while(fast->next != NULL){
            fast = fast->next;
        }
        while(slow->data < fast->data){
            if(slow->data + fast->data == target){
                res.push_back(make_pair(slow->data , fast->data));
                slow = slow->next;
                fast = fast->prev;
            }else if(slow->data + fast->data < target){
                slow = slow->next;
            }else if(slow->data + fast->data > target){
                fast = fast->prev;
            }
        }
        return res;
    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

Problem | Submissions | Comments

Delete all occurrences of a given key in a doubly linked list

Medium | Accuracy: 45.08% | Submissions: 3K+ | Points: 4

You are given the head of a doubly Linked List and a Key. Your task is to delete all occurrences of the given key and return the new DLL.

Note :- There exists atleast 1 distinct element other than key.

Example:

Input:
2<->2<->10<->8<->4<->2<->5<->2
2
Output:
10<->8<->4<->5
Explanation:
All Occurences of 2 have been deleted.

C++ (g++ 5.4) | Average Time: 30m | Start Timer

```
// } Driver Code Ends
// User function Template for C++

class Solution {
  public:
    void deleteAllOccurOfX(struct Node** head, int x) {
        // Write your code here
        Node *curr = *head;
        Node* curr_next = NULL;

        if(*head == NULL) return ;
        while(curr != NULL){
            if(curr->data == x){
                curr_next = curr->next;

                if(curr->prev == NULL){
                    *head = curr->next;
                }
                if(curr->next != NULL){
                    curr->next->prev = curr->prev;
                }
                if(curr->prev != NULL){
                    curr->prev->next = curr->next;
                }
                delete curr;

                curr = curr_next;
            }else{
                curr = curr->next;
            }
        }
    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit