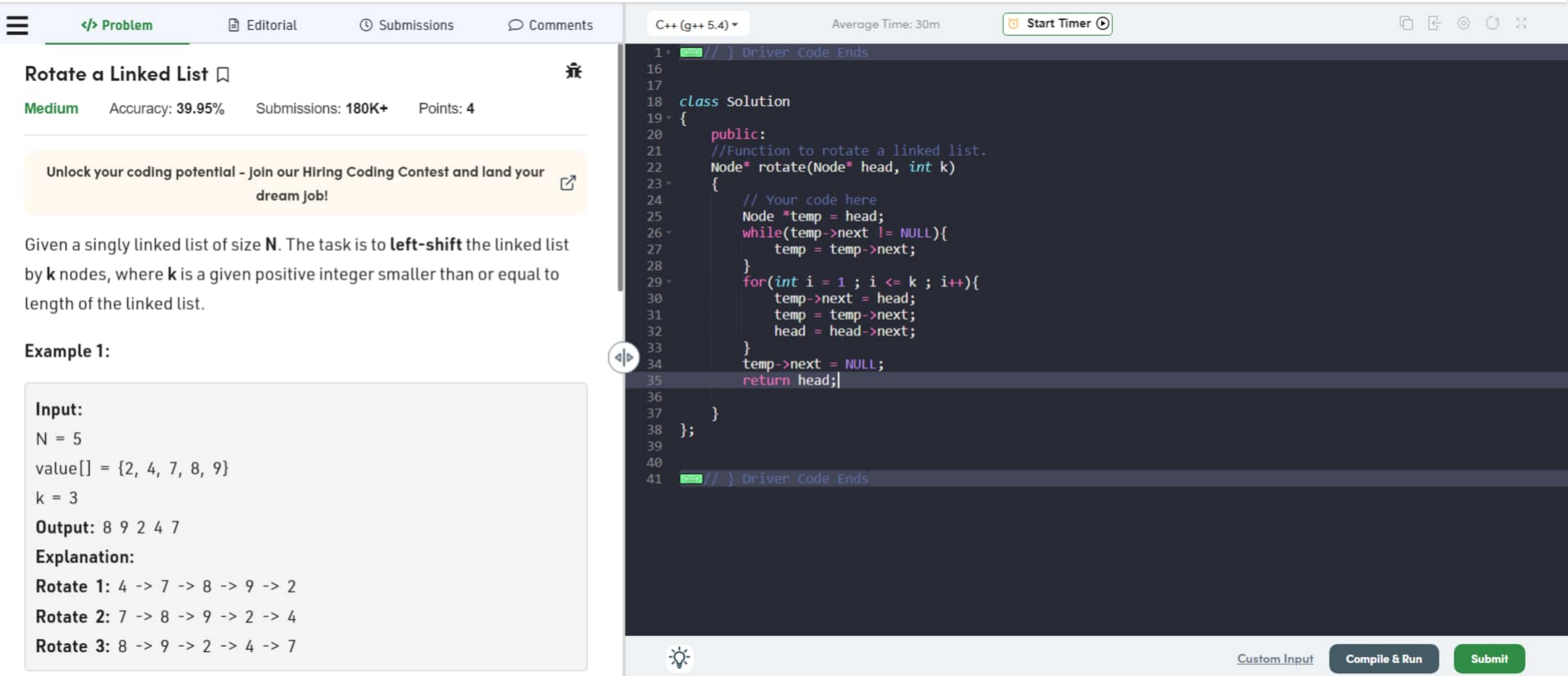


Rotate a Linked List

Medium Accuracy: **39.95%** Submissions: **180K+** Points: **4**

Given a singly linked list of size **N**. The task is to **left-shift** the linked list by **k** nodes, where **k** is a given positive integer smaller than or equal to length of the linked list.

Example 1:

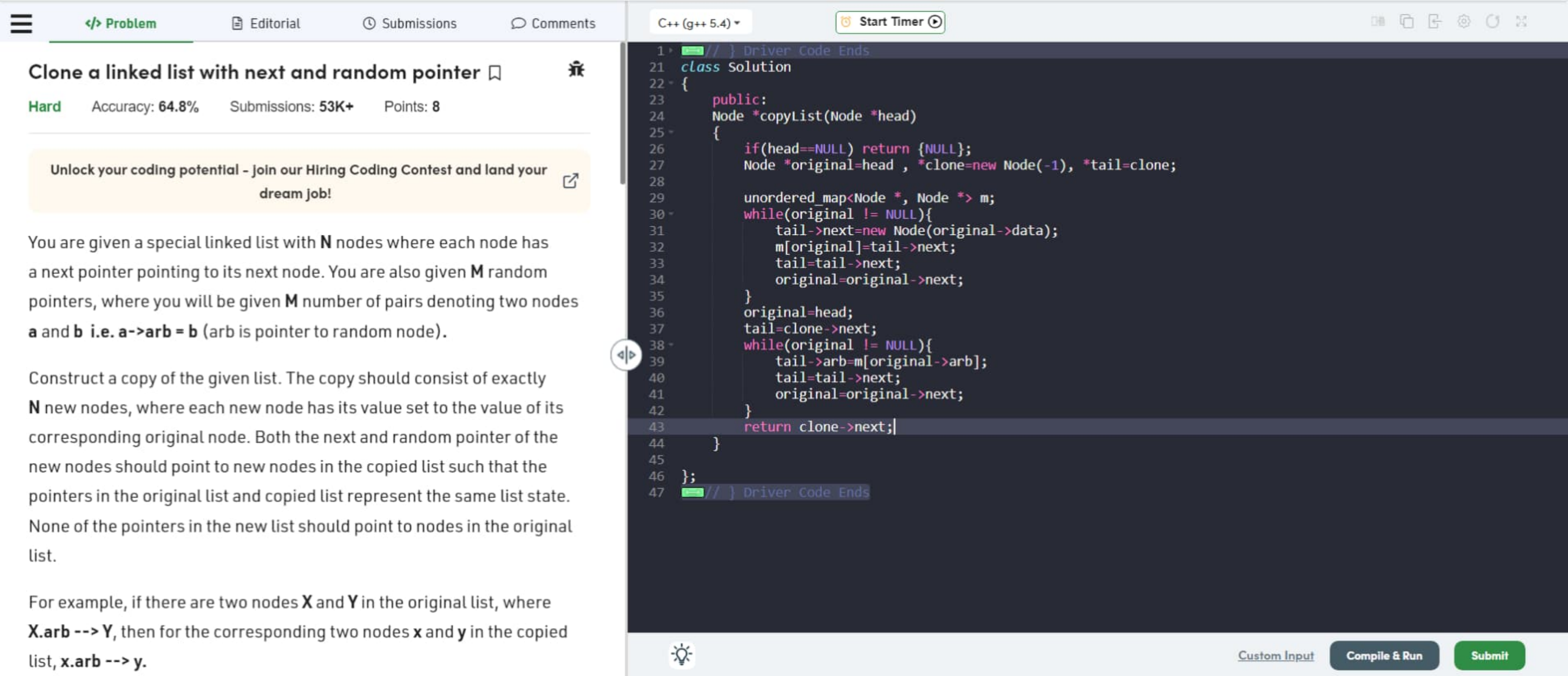
Input:
N = 5
value[] = {2, 4, 7, 8, 9}
k = 3
Output: 8 9 2 4 7
Explanation:
Rotate 1: 4 -> 7 -> 8 -> 9 -> 2
Rotate 2: 7 -> 8 -> 9 -> 2 -> 4
Rotate 3: 8 -> 9 -> 2 -> 4 -> 7

```
// } Driver Code Ends


class Solution
{
    public:
    //Function to rotate a linked list.
    Node* rotate(Node* head, int k)
    {
        // Your code here
        Node *temp = head;
        while(temp->next != NULL){
            temp = temp->next;
        }
        for(int i = 1 ; i <= k ; i++){
            temp->next = head;
            temp = temp->next;
            head = head->next;
        }
        temp->next = NULL;
        return head;

    }
};


// } Driver Code Ends
```



Clone a linked list with next and random pointer

Hard Accuracy: **64.8%** Submissions: **53K+** Points: **8**

You are given a special linked list with **N** nodes where each node has a next pointer pointing to its next node. You are also given **M** random pointers, where you will be given **M** number of pairs denoting two nodes **a** and **b** **i.e. a->arb = b** (arb is pointer to random node).

Construct a copy of the given list. The copy should consist of exactly **N** new nodes, where each new node has its value set to the value of its corresponding original node. Both the next and random pointer of the new nodes should point to new nodes in the copied list such that the pointers in the original list and copied list represent the same list state. None of the pointers in the new list should point to nodes in the original list.

For example, if there are two nodes **X** and **Y** in the original list, where **X.arb --> Y**, then for the corresponding two nodes **x** and **y** in the copied list, **x.arb --> y.**

```
// } Driver Code Ends
class Solution
{
    public:
    Node *copyList(Node *head)
    {
        if(head==NULL) return {NULL};
        Node *original=head , *clone=new Node(-1), *tail=clone;

        unordered_map<Node *, Node *> m;
        while(original != NULL){
            tail->next=new Node(original->data);
            m[original]=tail->next;
            tail=tail->next;
            original=original->next;
        }
        original=head;
        tail=clone->next;
        while(original != NULL){
            tail->arb=m[original->arb];
            tail=tail->next;
            original=original->next;
        }
        return clone->next;
    }

};
// } Driver Code Ends
```