**Problem Name:** Rotate list

**Topics:**  Linked list, Two Pointers

**Companies:**

**Level:** Medium

**Language:** C++

**Problem Statement**: Given the head of a linked list, rotate the list to the right by k places.

**Input Format:**

First line of the input contain integer n (size of list)

Second line contain n space separated integer list values.

Last line contain integer value pos representing index where to rotate.

Ex:

5

1 2 3 4 5

1

**Output Format:** Print linked list after rotating right by k.

**Constraints:**

* The number of nodes in the list is in the range [0, 500].
* -100 <= Node.val <= 100
* 0 <= k <= 2 \* 109

**Examples:**

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

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

**Brute force Solution:**

**Explanation:**  Make a for loop, each time go to the last second node and make last node as head

1. Calculate the length of linked list

2. take the mod of k with length as k is very large and if k > n its same as list rotating k/n + k%n times which is same as k%n times

**Code:**

#include <bits/stdc++.h>

using namespace std;

class ListNode

{

    public:

        int val;

        ListNode\* next;

        ListNode(int a){

            val = a;

            next = NULL;

        }

};

void insertNode(ListNode\* &head,int val) {

    ListNode\* newNode = new ListNode(val);

    if(head == NULL) {

        head = newNode;

        return;

    }

    ListNode\* temp = head;

    while(temp->next != NULL)

     temp = temp->next;

    temp->next = newNode;

    return;

}

void printList(ListNode \*node)

{

    while (node!=NULL)

    {

        cout<<node->val<<" ";

        node = node->next;

    }

}

ListNode\* rotateRight(ListNode\* head, int k) {

    if(head == NULL || head->next == NULL){

        return head;

    }

    ListNode \* temp = head;

    int length =0;

    while(temp!= NULL){

        temp = temp->next;

        length++;

    }

    if ( k%length == 0){

        return head;

    }

    int n ;

    n = k%length;

    for(int i = 1 ; i <= n ; i++ ){

        temp = head;

        while(temp->next->next!=NULL){

            temp = temp->next;

        }

        temp->next->next = head;

        head = temp->next;

        temp->next = NULL;

    }

    return head;

}

int main()

{

    ListNode\* a = NULL;

    ListNode\* res=NULL;

    int n, temp, pos;

    cin>>n;

    while(n--){

        cin>>temp;

        insertNode(a, temp);

    }

    cin>>pos;

    res = rotateRight(a, pos);

    printList(res);

    return 0;

}

**Time Complexity**: O(N2)

**Space Complexity:** O(1)

**Optimized Solution:**

**Explanation:** Take a temp pointer taking it to node which is (length - k)th node. Point the end of linked list to head and next of temp as NULL

**Example**

k = 2

Make another pointer p

take temp to length - k

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

t,p

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

t,p

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

t,p

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

t p

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

t p

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

**Code:**

#include <bits/stdc++.h>

using namespace std;

class ListNode

{

    public:

        int val;

        ListNode\* next;

        ListNode(int a){

            val = a;

            next = NULL;

        }

};

void insertNode(ListNode\* &head,int val) {

    ListNode\* newNode = new ListNode(val);

    if(head == NULL) {

        head = newNode;

        return;

    }

    ListNode\* temp = head;

    while(temp->next != NULL)

     temp = temp->next;

    temp->next = newNode;

    return;

}

void printList(ListNode \*node)

{

    while (node!=NULL)

    {

        cout<<node->val<<" ";

        node = node->next;

    }

}

ListNode\* rotateRight(ListNode\* head, int k) {

    if(head == NULL || head->next == NULL){

        return head;

    }

    ListNode \* temp = head;

    int length =0;

    while(temp!= NULL){

        temp = temp->next;

        length++;

    }

    if ( k%length == 0){

        return head;

    }

    int n ;

    n = k%length;

    ListNode\* temp1 = head;

    ListNode \* temp2 = head;

    for(int i = 1 ; i < (length-n) ; i++){

        temp1 = temp1->next;

        temp2 = temp2->next;

    }

    while(temp1->next != NULL){

        temp1 = temp1->next;

    }

    temp1->next = head;

    head = temp2->next;

    temp2->next = NULL;

    return head;

}

int main()

{

    ListNode\* a = NULL;

    ListNode\* res=NULL;

    int n, temp, pos;

    cin>>n;

    while(n--){

        cin>>temp;

        insertNode(a, temp);

    }

    cin>>pos;

    res = rotateRight(a, pos);

    printList(res);

    return 0;

}

**Time Complexity**: O(N)

**Space Complexity:** O(N)