

Rotate list leetcode

Problem List

61. Rotate List

Solved

Medium

Topics

Companies

Given the `head` of a linked list, rotate the list to the right by `k` places.

Example 1:

rotate 1

rotate 2

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

Example 2:

rotate 1

Code

C Auto

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     struct ListNode *next;
6  * };
7  */
8 struct ListNode* rotateRight(struct ListNode* head, int k) {
9
10     struct ListNode* p,*q;
11     if(head==NULL||head->next==NULL||k==0){
12         return head;
13     }
14     p=head;
15     int count=1;
16
17     while(p->next!=NULL){
18         p=p->next;
19         count++;
20     }
21     k=k%count;
22     if(k==0){
23         return head;
24     }
25     p->next=head;
26     p=head;
27
28     for(int i=0;i<count-k-1;i++){
29         p=p->next;
30     }
31     q=p->next;
32     p->next=NULL;
```

Saved to local

Ln 27, Col 5

Testcase

Test Result

Problem List

61. Rotate List

Solved

Medium

Topics

Companies

Given the `head` of a linked list, rotate the list to the right by `k` places.

Example 1:

rotate 1

rotate 2

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

Example 2:

rotate 1

Code

C Auto

```
8 struct ListNode* rotateRight(struct ListNode* head, int k) {
9
10     struct ListNode* p,*q;
11     if(head==NULL||head->next==NULL||k==0){
12         return head;
13     }
14     p=head;
15     int count=1;
16
17     while(p->next!=NULL){
18         p=p->next;
19         count++;
20     }
21     k=k%count;
22     if(k==0){
23         return head;
24     }
25     p->next=head;
26     p=head;
27
28     for(int i=0;i<count-k-1;i++){
29         p=p->next;
30     }
31     q=p->next;
32     p->next=NULL;
33     return q;
34 }
35
```

Saved to local

Ln 27, Col 5

Testcase

Test Result

Output

61. Rotate List Solved

Given the `head` of a linked list, rotate the list to the right by `k` places.

Example 1:

1 → 2 → 3 → 4 → 5

rotate 1 5 → 1 → 2 → 3 → 4

rotate 2 4 → 5 → 1 → 2 → 3

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

Example 2:

0 → 1 → 2

rotate 1 2 → 0 → 1

rotate 2 1 → 2 → 0

rotate 3 0 → 1 → 2

rotate 4 2 → 0 → 1

Input: `head = [0,1,2]`, `k = 4`
Output: `[2,0,1]`

Constraints:

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

```
struct ListNode* rotateRight(struct ListNode* head, int k) {
    struct ListNode* p,*q;
```

Testcase | Test Result

Accepted Runtime: 2 ms

Case 1 Case 2

Input

head =
[1,2,3,4,5]

k =
2

Output

[4,5,1,2,3]

Expected

[4,5,1,2,3]

Contribute a testcase

Example 2:

0 → 1 → 2

rotate 1 2 → 0 → 1

rotate 2 1 → 2 → 0

rotate 3 0 → 1 → 2

rotate 4 2 → 0 → 1

Input: `head = [0,1,2]`, `k = 4`
Output: `[2,0,1]`

Testcase | Test Result

Accepted Runtime: 2 ms

Case 1 Case 2

Input

head =
[0,1,2]

k =
4

Output

[2,0,1]

Expected

[2,0,1]

Contribute a testcase

Code

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     struct ListNode *next;
 * };
 */
struct ListNode* rotateRight(struct ListNode* head, int k) {
```

```

struct ListNode* p,*q;
if(head==NULL || head->next==NULL || k==0){
    return head;
}
p=head;
int count=1;

while(p->next!=NULL){
    p=p->next;
    count++;
}
k=k%count;
if(k==0){
    return head;
}
p->next=head;
p=head;

for(int i=0;i<count-k-1;i++){
    p=p->next;
}
q=p->next;
p->next=NULL;
return q;
}

```