

//reverse linked list II

Problem List

92. Reverse Linked List II

Solved

Medium

Topics

Companies

Given the head of a singly linked list and two integers left and right where left ≤ right, reverse the nodes of the list from position left to position right, and return the reversed list.

Example 1:

Input: head = [1,2,3,4,5], left = 2, right = 4
Output: [1,4,3,2,5]

Example 2:

Input: head = [5], left = 1, right = 1
Output: [5]

Constraints:

11.2K 87

Code

Auto

```
1 /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     struct ListNode *next;
6  * };
7  */
8 struct ListNode* reverseBetween(struct ListNode* head, int left, int right) {
9     if (head == NULL || left == right) {
10         return head;
11     }
12     struct ListNode* p = malloc(sizeof(struct ListNode));
13     p->next = head;
14
15     struct ListNode *ptr, *ptr1;
16     ptr1 = p;
17     ptr = head;
18
19     for (int i = 1; i < left; i++) {
20         ptr1 = ptr;
21         ptr = ptr->next;
22     }
23
24     struct ListNode *first = ptr;
25     struct ListNode *last = ptr1;
26     struct ListNode *next = NULL;
27
28 }
```

Saved to local

Ln 1, Col 1

Testcase Test Result

Case 1 Case 2

Source

Problem List

92. Reverse Linked List II

Solved

Medium

Topics

Companies

Given the head of a singly linked list and two integers left and right where left ≤ right, reverse the nodes of the list from position left to position right, and return the reversed list.

Example 1:

Input: head = [1,2,3,4,5], left = 2, right = 4
Output: [1,4,3,2,5]

Example 2:

Input: head = [5], left = 1, right = 1
Output: [5]

Constraints:

11.2K 87

Code

Auto

```
23 }
24
25 struct ListNode *first = ptr;
26 struct ListNode *last = ptr1;
27 struct ListNode *next = NULL;
28
29 for (int i = left; i <= right; i++) {
30     struct ListNode *temp = ptr->next;
31     ptr->next = next;
32     next = ptr;
33     ptr = temp;
34 }
35
36 last->next = next;
37 first->next = ptr;
38
39
40
41
42 return p->next;
43 }
44
45
46 }
```

Saved to local

Ln 1, Col 1

Testcase Test Result

Case 1 Case 2

Source

Output

92. Reverse Linked List II Solved

Medium Topics Companies

Given the head of a singly linked list and two integers `left` and `right` where `left <= right`, reverse the nodes of the list from position `left` to position `right`, and return the reversed list.

Example 1:

Diagram showing a linked list with nodes 1, 2, 3, 4, 5. The nodes from position 2 to 4 are reversed, resulting in a new linked list with nodes 1, 4, 3, 2, 5.

Input: head = [1,2,3,4,5], left = 2, right = 4
Output: [1,4,3,2,5]

Example 2:

Input: head = [5], left = 1, right = 1
Output: [5]

Constraints:

- 1 <= left <= right <= length of list
- 1 <= length of list <= 5000
- 0 <= Node.val <= 1000

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

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

left = 2

right = 4

Output

[1,4,3,2,5]

Expected

[1,4,3,2,5]

Contribute a testcase

92. Reverse Linked List II Solved

Medium Topics Companies

Given the head of a singly linked list and two integers `left` and `right` where `left <= right`, reverse the nodes of the list from position `left` to position `right`, and return the reversed list.

Example 1:

Diagram showing a linked list with nodes 1, 2, 3, 4, 5. The nodes from position 2 to 4 are reversed, resulting in a new linked list with nodes 1, 4, 3, 2, 5.

Input: head = [1,2,3,4,5], left = 2, right = 4
Output: [1,4,3,2,5]

Example 2:

Input: head = [5], left = 1, right = 1
Output: [5]

Constraints:

- 1 <= left <= right <= length of list
- 1 <= length of list <= 5000
- 0 <= Node.val <= 1000

Testcase Test Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

head = [5]

left = 1

right = 1

Output

[5]

Expected

[5]

Contribute a testcase

Code

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
```

```

*      struct ListNode *next;
* };
*/
struct ListNode* reverseBetween(struct ListNode* head, int left, int right) {
    if (head == NULL || left == right) {
        return head;
    }
    struct ListNode* p = malloc(sizeof(struct ListNode));
    p->next = head;

    struct ListNode *ptr,*ptr1 ;
    ptr1=p;
    ptr= head;

    for (int i = 1; i < left; i++) {
        ptr1= ptr;
        ptr = ptr->next;
    }

    struct ListNode *first = ptr;
    struct ListNode *last = ptr1;
    struct ListNode *next = NULL;

    for (int i = left; i <=right; i++) {
        struct ListNode *temp = ptr->next;
        ptr->next = next;
        next = ptr;
        ptr = temp;
    }

    last->next = next;

    first->next = ptr;

    return p->next;
}

```