# 题目

反转从位置 m 到 n 的链表。请使用一趟扫描完成反转。

**说明:**

1 ≤ m ≤ n ≤ 链表长度。

**示例:**

输入: 1->2->3->4->5->NULL, m = 2, n = 4

输出: 1->4->3->2->5->NULL

# 分析

## 方法一：原地翻转

/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

\* ListNode(int x) : val(x), next(NULL) {}

\* };

\*/

class Solution {

public:

ListNode\* reverseBetween(ListNode\* head, int m, int n) {

ListNode\* dummy=new ListNode(-1);

ListNode\* pre=dummy;

dummy->next=head;

for(int i=0;i<m-1;i++)

pre=pre->next;

ListNode\* cur=pre->next;

for(int i=m;i<n;i++){

//原地翻转链表

ListNode\* t=cur->next;

cur->next=t->next;

t->next=pre->next;

pre->next=t;

}

return dummy->next;

}

};

## 方法二：双指针

**思路：**

以第m-1节点为"头节点"，将[m,n]范围的节点不断插入"头节点"后，最后，第m个节点指向第n+1个节点。

**代码：**

/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

\* ListNode(int x) : val(x), next(NULL) {}

\* };

\*/

class Solution {

public:

ListNode\* reverseBetween(ListNode\* head, int m, int n) {

ListNode \*dummy = new ListNode(-1);

dummy->next = head;

// 第m-1个节点

ListNode \*pre=dummy;

for(int i=1; i<m; i++){

pre = pre->next;

}

// 第m个节点

ListNode \*t, \*cur=pre->next, \*mNode=pre->next;

// 头插法

for(int i=m; i<=n; i++){

t = cur->next;

cur->next = pre->next;

pre->next = cur;

cur = t;

}

// 第m个节点指向第n+1个节点

mNode->next = cur;

return dummy->next;

}

};