描述：

找到单链表倒数第n个节点，保证链表中节点的最少数量为n。

样例：

Example 1:

Input: list = 3->2->1->5->null, n = 2

Output: 1

Example 2:

Input: list = 1->2->3->null, n = 3

Output: 1

这道题我用了两个方法。第一个方法就很平常了，先遍历一遍链表，然后得到链表的长度，再然后专门拿一个变量来计算出从链表的首节点到目标节点的路径长短，然后再设置一个循环迭代就可。

代码如下：

ListNode \* nthToLast(ListNode \* head, int n) {

// write your code here

ListNode\*p=head;

if(head==NULL)

return 0;

int j=0;

while(p->next!=NULL){

j++;

p=p->next;

}

int a=j-n+1;

p=head;

while(a>0){

a--;

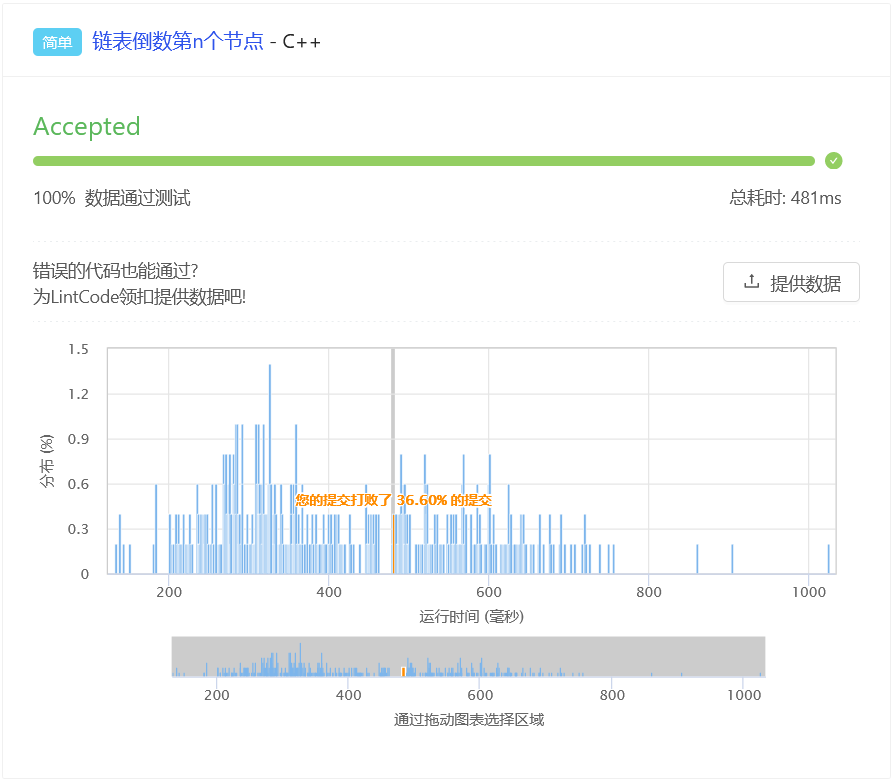
p=p->next;

}

return p;

}

通过截图如下：



第二个方法则是在力扣刷题上见过的，就是快慢指针了：

第一根指针指向开头，让它先走传进来的参数n步。第二根指针也指向开头。这个时候两根指针的距离是n步。然后慢指针和快指针同时出发，让快指针走到链表尽头，这个时候满指针所指的位置就是倒数第n个节点了。

数学上的证明很简单：

快指针总共走了k步，k步就是传进来的链表长度。一开始快指针走了n步。而慢指针走时是从头节点出发的。快指针第二阶段走了k-n步。那慢指针就走了k-n步。而k-n正好是我们需要的节点的序号。

代码如下：

ListNode \* nthToLast(ListNode \* head, int n) {

// write your code here

ListNode\*former=head;

ListNode\*latter=head;

if(head==NULL)

return 0;

for(int i=0;i<n;++i)

former=former->next;

while(former!=nullptr){

former=former->next;

latter=latter->next;

}

return latter;

}

通过截图如下：

