**描述**

给定一个带有头结点 head 的非空单链表，返回链表的中间结点。

如果有两个中间结点，则返回第二个中间结点。

**样例 1:**

输入：1->2->3->4->5->null

输出：3->4->5->null

**样例 2:**

输入：1->2->3->4->5->6->null

输出：4->5->6->null

这道题完全就是链表第n个节点的特殊化。因此按之前的思路的话，又有两种方法了。

第一种方法和之前一样，先遍历链表得到链表长度。然后再设置一个变量指示链表的中间长度。然后通过while循环来不断逼近中间长度。最后得到的就是链表的中间节点了。

代码如下：

ListNode \* middleNode(ListNode \* head) {

// write your code here.

ListNode\*ptr=head;

int n=0;

while(ptr){

++n;

ptr=ptr->next;

}

int j=0;

ptr=head;

while(j<n/2){

j++;

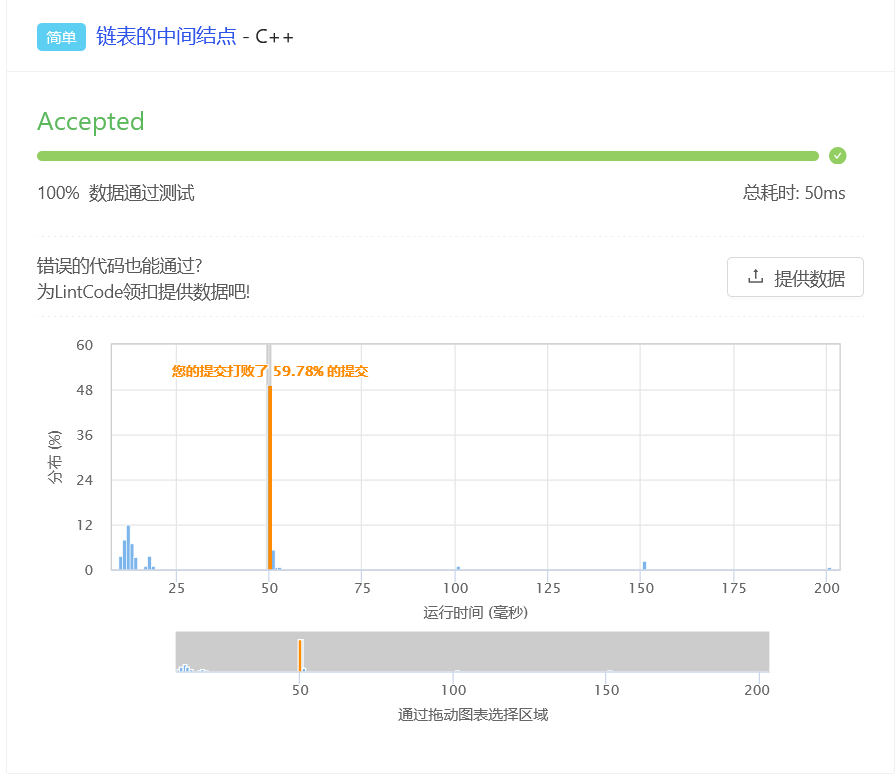
ptr=ptr->next;

}

return ptr;

}

以下是通过截图：



另外一种快慢指针就更简单了。直接让快指针走2步，慢指针在快指针走2步的时候只走1步。当然，要注意打破循坏的条件：当快指针的下根指针是null时或者快指针本身是null时就该打破循坏了。因为这个时候我们往往到了末尾。

代码如下：

ListNode \* middleNode(ListNode \* head) {

// write your code here.

ListNode\*fast=head;

ListNode\*slow=head;

while(fast!=nullptr&&fast->next!=nullptr){

fast=fast->next->next;

slow=slow->next;

}

return slow;

}

通过测试截图如下：

