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。

示例 1：

输入： `[1,2,3,4,5]`
输出： 此列表中的结点 `3` (序列化形式：`[3,4,5]`)
返回的结点值为 `3` 。 (测评系统对该结点序列化表述是 `[3,4,5]`)。
注意，我们返回了一个 `ListNode` 类型的对象 `ans`，这样：
`ans.val = 3, ans.next.val = 4, ans.next.next.val = 5,` 以及 `ans.next.next.next = NULL.`

示例 2：

输入： `[1,2,3,4,5,6]`
输出： 此列表中的结点 `4` (序列化形式：`[4,5,6]`)
由于该列表有两个中间结点，值分别为 `3` 和 `4`，我们返回第二个结点。

提示：

- 给定链表的结点数介于 `1` 和 `100` 之间。

思路，改节点的规律，若是把链表看成从0开始的数组，那么每次返回的节点，都是为度/2的位置。记住这个规律，然后设置返回，当位置=度/2时返回该节点，否则返回下一层遍历的节点结果。

代码

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode(int x) : val(x), next(NULL) {}
 * };
 */
class Solution {
public:
    ListNode* dfs(ListNode* head,int wz,int &length){
        length++;
```

```
        ListNode* result=NULL;
        if(head->next!=NULL){
            result=dfs(head->next,wz+1,length);
        }
        if(wz==(length>>1)){
            result=head;
        }
        return result;
    }
    ListNode* middleNode(ListNode* head) {
        int i=0;
        return dfs(head,0,i);
    }
};
```