# 题目

给定一个有环链表，实现一个算法返回环路的开头节点。

有环链表的定义：在链表中某个节点的next元素指向在它前面出现过的节点，则表明该链表存在环路。

**示例 1：**

输入：head = [3,2,0,-4], pos = 1

输出：tail connects to node index 1

解释：链表中有一个环，其尾部连接到第二个节点。

**示例 2：**

输入：head = [1,2], pos = 0

输出：tail connects to node index 0

解释：链表中有一个环，其尾部连接到第一个节点。

**示例 3：**

输入：head = [1], pos = -1

输出：no cycle

解释：链表中没有环。

**进阶：**

你是否可以不用额外空间解决此题？

# 分析

## 方法一：双指针/快慢指针

/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

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

\* };

\*/

class Solution {

public:

ListNode \*detectCycle(ListNode \*head) {

if(!head||!head->next)

{

return NULL;

}

ListNode\* fast=head;

ListNode\* slow=head;

while(fast&&fast->next)

{

slow=slow->next;

fast=fast->next->next;

if(fast==slow)

{

break;

}

}

if(fast!=slow)

{

return NULL;

}

fast=head;

while(fast!=slow)

{

fast=fast->next;

slow=slow->next;

}

return fast;

}

};