

142. Linked List Cycle II

```
class Solution {  
public:  
    ListNode *detectCycle(ListNode *head) {  
        if (!head || !head->next) return nullptr;  
  
        ListNode *slow = head, *fast = head;  
  
        while (fast && fast->next) {  
            slow = slow->next;  
            fast = fast->next->next;  
            if (slow == fast) {  
                slow = head;  
                while (slow != fast) {  
                    slow = slow->next;  
                    fast = fast->next;  
                }  
                return slow;  
            }  
        }  
        return nullptr;  
    }  
};
```

leetcode.com/problems/linked-list-cycle-ii/submissions/1542127877/

Problem List

Run

Submit

Editorial

Solutions

Submissions

Description

Accepted

Editorial

Solutions

Submissions

All Submissions

Accepted

18 / 18 testcases passed

sashima_narula submitted at Feb 14, 2025 00:17

Editorial

Solution

Runtime

7 ms

Beats 68.16%

Memory

11.46 MB

Beats 24.87%

Analyze Complexity

Time Interval	Percentage
3ms	~15%
4ms	~10%
5ms	~10%
6ms	~10%
7ms	~25%
8ms	~10%
9ms	~10%
10ms	~10%
11ms	~10%
12ms	~10%
13ms	~10%

Code

C++

Auto

```
1 /**  
2  * Definition for singly-linked list.  
3  * struct ListNode {  
4  *     int val;  
5  *     ListNode *next;  
6  *     ListNode(int x) : val(x), next(NULL) {}  
7  * };  
8  */  
9 class Solution {  
10 public:  
11     ListNode *detectCycle(ListNode *head) {  
12         if (!head || !head->next) return nullptr;  
13         ListNode *slow = head, *fast = head;  
14         while (fast && fast->next) {  
15             slow = slow->next;  
16             fast = fast->next->next;  
17             if (slow == fast) {  
18                 return slow;  
19             }  
20         }  
21         return nullptr;  
22     }  
23 };
```

Testcase

Test Result

Accepted

Runtime: 3 ms

Case 1

Case 2

Case 3

Input

head = [3,2,0,-4]

pos = 1

More challenges