

leetcode.com/problems/linked-list-cycle-ii/submissions/1889824211/

Problem List Submit +

Description Editorial Solutions Submissions

142. Linked List Cycle II Solved

Medium Topics Companies

Given the `head` of a linked list, return *the node where the cycle begins*. If there is no cycle, return `null`.

There is a cycle in a linked list if there is some node in the list that can be reached again by continuously following the `next` pointer. Internally, `pos` is used to denote the index of the node that tail's next pointer is connected to (**0-indexed**). It is `-1` if there is no cycle. **Note that pos is not passed as a parameter.**

Do not modify the linked list.

Example 1:

```

graph LR
    N3((3)) --> N2((2))
    N2 --> N0((0))
    N0 --> N4((−4))
    N4 --> N2
  
```

Input: head = [3,2,0,-4], pos = 1
Output: tail connects to node index 1
Explanation: There is a cycle in the linked list, where tail connects to the second node.

15K 250 | ⭐ | 📁 | ⚙️ | ① | ● 168 Online | Testcase | Test Result

Accepted 18 / 18 testcases passed
Anupam_Pathak submitted at Jan 19, 2026 17:48

Editorial Solution

Runtime 7 ms Beats 66.41% Analyze Complexity

Memory 11.24 MB Beats 83.40% Analyze Complexity

Time Range (ms)	Percentage (%)
2ms	~1%
4ms	~1%
6ms	~1%
8ms	~1%
10ms	~30%
12ms	~1%
14ms	~1%

Code C++

```

1 class Solution {
2 public:
3     ListNode *detectCycle(ListNode *head) {
4         ...
5     }
6 }
  
```

Code | Accepted ×



← All Submissions



```
1 class Solution {
2 public:
3     ListNode *detectCycle(ListNode *head) {
4         ListNode* fast=head;
5         ListNode* slow=head;
6         while(fast!=NULL && fast->next!=NULL){
7             slow=slow->next;
8             fast=fast->next->next;
9             if(slow==fast){
10
11                 slow=head;
12                 while(slow!=fast){
13                     slow=slow->next;
14                     fast=fast->next;
15                 }
16                 return slow;
17             }
18         }
19         return NULL;
20     }
21 }
22 };
```

View less

The screenshot shows a programming environment with a toolbar at the top featuring icons for lock, play, submit, copy, and settings. The main area displays a code editor with the title 'Code | Accepted'. Below the title, there are tabs for 'Fold' and 'Alt', with 'All Submissions' selected. The code itself is written in C++ and defines a singly-linked list and a solution class to reverse it.

```
1  /**
2  * Definition for singly-linked list.
3  * struct ListNode {
4  *     int val;
5  *     ListNode *next;
6  *     ListNode() : val(0), next(nullptr) {}
7  *     ListNode(int x) : val(x), next(nullptr) {}
8  *     ListNode(int x, ListNode *next) : val(x), next(next) {}
9  * };
10 */
11 class Solution {
12 public:
13     ListNode* reverseList(ListNode* head) {
14         ListNode* temp = head;
15         vector<int> stor;
16         while (temp != NULL) {
17             stor.push_back(temp->val);
18             temp = temp->next;
19         }
20         temp = head;
21         for (int i = stor.size() - 1; i >= 0; i--) {
22             temp->val = stor[i];
23             temp = temp->next;
24         }
25         return head;
26     }
27 };
```

View less

Problem List < > ✎ Submit ⌂ ↻ 0 00:00:00 🔍

Description | Editorial | Solutions | Submissions

206. Reverse Linked List

Solved 2

Easy Topics Companies

Given the `head` of a singly linked list, reverse the list, and return *the reversed list*.

Example 1:

```
graph LR; 1((1)) --> 2((2)); 2 --> 3((3)); 3 --> 4((4)); 4 --> 5((5)); 5 --> null; -->|down| 5((5)) --> 4((4)); 4 --> 3((3)); 3 --> 2((2)); 2 --> 1((1)); 1 --> null;
```

Input: head = [1,2,3,4,5]
Output: [5,4,3,2,1]

Example 2:

```
graph LR; 1((1)) --> 2((2)); 2 --> null; -->|down| 2((2)) --> 1((1)); 1 --> null;
```

24.1K 384 ★ ⓘ

Code Accepted

All Submissions

```
1 /**
2 * Definition for singly-linked list.
3 * struct ListNode {
4 *     int val;
5 *     ListNode *next;
6 *     ListNode() : val(0), next(nullptr) {}
7 *     ListNode(int x) : val(x), next(nullptr) {}
8 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
9 * };
10 */
11 class Solution {
12 public:
13     ListNode* reverseList(ListNode* head) {
14         ListNode* temp=head;
15         vector<int> stor;
16         while(temp!=NULL){
17             stor.push_back(temp->val);
18             temp=temp->next;
19         }
20         temp=head;
21         for(int i=stor.size()-1; i>=0; i--){
22             temp->val = stor[i];
23             temp=temp->next;
24         }
25     }
26 }
27 }
```

View less

• 504 Online Testcase Test Result