

Leetcode Problem - 142

Code in java:

```
public class Solution {  
    public ListNode detectCycle(ListNode head) {  
        if (head == null || head.next == null) return null;  
  
        ListNode slow = head;  
        ListNode fast = head;  
  
        while (fast != null && fast.next != null) {  
            slow = slow.next;  
            fast = fast.next.next;  
  
            if (slow == fast) {  
                ListNode entry = head;  
                while (entry != slow) {  
                    entry = entry.next;  
                    slow = slow.next;  
                }  
                return entry;  
            }  
        }  
        return null;  
    }  
}
```

The screenshot shows a LeetCode problem submission page for Java. The code has been accepted with 28/28 testcases passed. The runtime is 0 ms (100.00% beats) and memory usage is 44.41 MB (19.20% beats). The code implements an O(n^2) solution for reversing a linked list by swapping each node's next pointer to its previous node.

```
Java  
1 *  
2 */  
3 class Solution {  
4     public ListNode reverseList(ListNode head) {  
5         ListNode prev = null;  
6         ListNode curr = head;  
7  
8         while (curr != null) {  
9             ListNode next = curr.next;  
10            curr.next = prev;  
11            prev = curr;  
12            curr = next;  
13        }  
14        return prev;  
15    }  
16 }
```

Testcase | Test Result
Accepted Runtime: 0 ms
Case 1 Case 2 Case 3

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

Code | Java

```
1 /**  
2  * Definition for singly-linked list.  
3 */
```