Description | Editorial | Solutions (9K) | Submissions

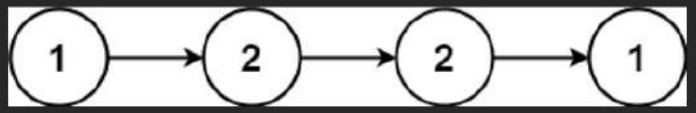
# 234. Palindrome Linked List

Easy 14.9K 812

Companies

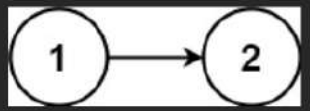
Given the `head` of a singly linked list, return `true` *if it is a palindrome or* `false` *otherwise.*

**Example 1:**




```
Input: head = [1,2,2,1]
Output: true
```

**Example 2:**




```
Input: head = [1,2]
Output: false
```

**Constraints:**

- The number of nodes in the list is in the range $[1, 10^5]$.

Java | Auto

```java
class Solution {
    public boolean isPalindrome(ListNode head) {
        ListNode slow = head, fast = head, prev, temp;
        while (fast != null && fast.next != null) {
            slow = slow.next;
            fast = fast.next.next;
        }
        prev = slow;
        slow = slow.next;
        prev.next = null;
        while (slow != null) {
            temp = slow.next;
            slow.next = prev;
            prev = slow;
            slow = temp;
        }
```

Ln 36, Col 2

Testcase | Result

**Accepted** Runtime: 0 ms

• Case 1 • Case 2

Input

head =
[1,2,2,1]

Output

true

Expected

Console | Run | Submit

Description | Editorial | Solutions (5.1K) | Submissions

## 239. Sliding Window Maximum

Hint

Hard | 16.4K | 550

Companies

You are given an array of integers `nums`, there is a sliding window of size `k` which is moving from the very left of the array to the very right. You can only see the `k` numbers in the window. Each time the sliding window moves right by one position.

Return *the max sliding window*.

**Example 1:**

```
Input: nums = [1,3,-1,-3,5,3,6,7], k = 3
Output: [3,3,5,5,6,7]
Explanation:
Window position                Max
---------------               -----
[1  3  -1] -3  5  3  6  7       3
 1 [3  -1  -3] 5  3  6  7       3
 1  3 [-1  -3  5] 3  6  7       5
 1  3  -1 [-3  5  3] 6  7       5
 1  3  -1  -3 [5  3  6] 7       6
 1  3  -1  -3  5 [3  6  7]      7
```

**Example 2:**

```
Input: nums = [1], k = 1
Output: [1]
```

Java | Auto

```java
import java.util.*;

class Solution {
    public int[] maxSlidingWindow(int[] nums, int k) {
        List<Integer> result = new ArrayList<>();
        Deque<Integer> deque = new LinkedList<>();

        for (int j = 0, i = 0; j < nums.length; j++) {
            while (!deque.isEmpty() && deque.peekLast() < nums[j]) {
                deque.pollLast();
            }
            deque.offerLast(nums[j]);

            if (j - i + 1 == k) {
                result.add(deque.peekFirst());
                if (deque.peekFirst() == nums[i]) {
```

Ln 25, Col 2

Testcase | Result

**Accepted** Runtime: 3 ms

• Case 1 • Case 2

Input

nums =
[1,3,-1,-3,5,3,6,7]

k =
3

Output

[3,3,5,5,6,7]

Console | Run | Submit