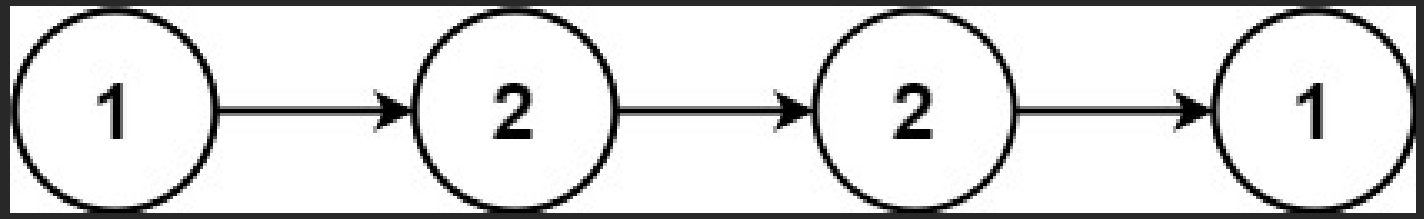
Check is the linked list a palindrome

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
```

pta palindrome:

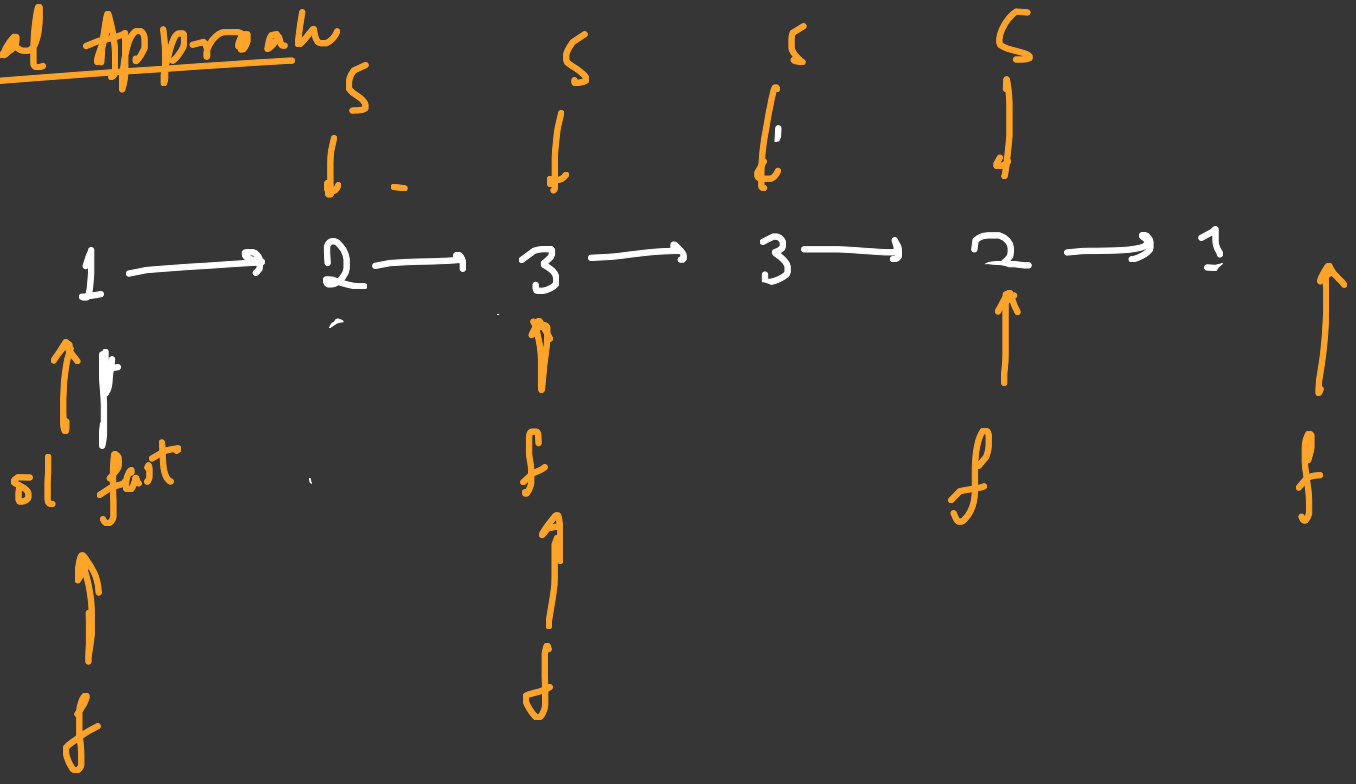
1 2 3 3 2 1

palindrome

Brute: (Using arr)

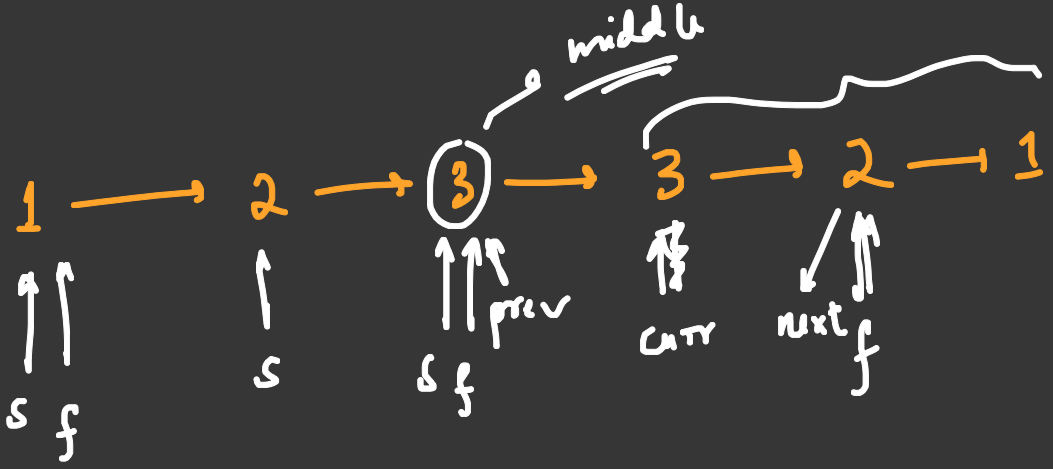
push linked list in arr and find the array is palindrome or not.

Optimal approach



Idea:

→ To find the middle element in list and reverse the second part of the list and check if the starting element and second part after middle is same or not



f → next ! = NULL

f → next → next = NULL

f → next = NULL

```
reverse(node)
    node* prev=NULL, next;
    while(head!=NULL
```

```
Node* reverseLinkedList(Node *head){
    Node* prev = NULL, *nex;
    while(head != NULL){
        nex = head->next;
        head->next = prev;
        prev = head;
        head = nex;
    }
    return prev;
}

Node *middleNode(Node *head){
    Node *slow = head, *fast = head;
    while(fast->next != NULL && fast->next->next != NULL){
        slow = slow->next;
        fast = fast->next->next;
    }
    return slow;
}

//Optimal Approach
//TC: O(N)
//SC: O(1)
bool isPalindromeOptimal(Node* &head){
    if(head == NULL || head->next == NULL)  return true;
    Node *middle = middleNode(head);
    Node* revNode = reverseLinkedList(middle->next);

    while(revNode != NULL){
        if(revNode->data != head->data) return false;
        revNode = revNode->next;
        head = head->next;
    }
    return true;
}
```

$O(\frac{N}{2})$

$O(N)$

$O(N)$

T.C → $O(N + \frac{N}{2} + N)$

$O(\frac{2.5}{2}N) \leq O(N)$

$2.5N$

$O(2N)$

S.C → $O(1)$