Palindrome Linked List LeetCode

Write a C++ program to determine whether a given singly linked list is a palindrome or not.

Approach 1: Brute Force Approach: Convert the linked list to a string and check if it's a palindrome

- Convert the linked list's elements to a string.
- Compare characters from the beginning and the end of the string.
- If characters match for all corresponding positions, the linked list is a palindrome.

Time Complexity: O(n), where n is the length of the linked list.

Space Complexity: O(n), as an additional string is used to store the linked list's elements.

Approach 2: Stack Approach: Push elements onto a stack, then pop and compare with the linked list

- Push the linked list's elements onto a stack.
- Pop and compare elements from the stack while traversing the linked list.
- If all elements match, the linked list is a palindrome.

Time Complexity: O(n), where n is the length of the linked list.

Space Complexity: O(n), as a stack is used to store the linked list's elements.

Approach 3: Optimized Approach: Reverse the second half and compare with the first half

- Find the middle node of the linked list.
- Reverse the second half of the linked list.
- Compare the reversed second half with the first half.
- Restore the original structure of the linked list.

Time Complexity: O(n), where n is the length of the linked list.

Space Complexity: O(1), as no additional data structures are used.