Odd and Even Linked List LeetCode

Given a singly linked list, implement a program to rearrange the nodes such that all odd-indexed nodes appear first, followed by all even-indexed nodes.

Approach 1: Rearrange the linked list with odd nodes first followed by even nodes

1. Traverse the linked list and distribute nodes into two separate lists: odd-indexed nodes and even-indexed nodes.
2. Connect the last node of the odd-indexed list to the first node of the even-indexed list.
3. Return the head of the odd-indexed list.
4. **Time Complexity:**
    - Traversing the linked list and distributing nodes: O(n)
    - **The overall time complexity is O(n).**
5. **Space Complexity:**
    - Additional space used for pointers and variables: O(1)
    - Two separate lists for odd and even nodes: O(n)
    - **The overall space complexity is O(n).**

Approach 2: Rearrange the linked list with odd nodes first followed by even nodes (Optimized approach)

1. Traverse the linked list while keeping track of odd and even nodes separately.
2. Use four pointers to maintain connections: **oddHead**, **oddTail**, **evenHead**, and **evenTail**.
3. Connect the last odd-indexed node to the first even-indexed node.
4. Return the **oddHead**.
5. **Time Complexity:**
    - Traversing the linked list and rearranging nodes: O(n)
    - **The overall time complexity is O(n).**
6. **Space Complexity:**
    - Additional space used for pointers and variables: O(1)
    - **The overall space complexity is O(1).**