**Reverse a singly linked list.**

* **Test Case 1**:  
  Input: List = [1, 2, 3, 4, 5]  
  Output: List = [5, 4, 3, 2, 1]
* **Test Case 2**:  
  Input: List = [10, 20, 30]  
  Output: List = [30, 20, 10]

explanation  
 Node class defines a basic node structure with data and next attributes. The SinglyLL class manages a singly linked list with methods for inserting nodes at the end (insertLast), displaying the list (display), and reversing the list (reverse). The insertLast method adds nodes to the end, updating the list structure appropriately. The display method iterates through the list, appending each node’s data to a string for display. The reverse method reverses the linked list by iterating through it and reversing the direction of the next pointers for each node, ultimately setting the first node to the last element traversed. The Question2 class tests these methods with predefined test cases, demonstrating the list's state before and after reversal  
  
flowchart  
 ┌───────────────────────────┐

│ Start │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ Create SinglyLL Instance │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ Insert Nodes (Last) │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ Display List (Before) │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ Reverse List │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ Display List (After) │

└───────────┬───────────────┘

│

▼

┌───────────────────────────┐

│ End │

└───────────────────────────┘

Time & space complexity  
The insertLast method has a time complexity of O(n) because it needs to traverse the entire list to insert at the end, and a space complexity of O(1) since it only requires additional space for a few variables. The display method also has a time complexity of O(n) as it traverses all nodes to print them, with a space complexity of O(1) for a constant space usage. The reverse method operates with a time complexity of O(n) since it processes each node once to reverse the list, and it uses O(1) space as it only needs a few pointers for the reversal process.  
  
output  
