LINKED-LIST and their Applications

Section 1

I. Maintaining Sorted Linked-List.
II. Implementing Doubly Linked-List.
III. Write a function in the LinkedList class Reverse(): which should reverse the singly linked list.
IV. Reversing a Doubly linked-list: Now write the REVERSE function in doubly LinkedList.
V. BubbleSort of LinkedList

Section 2

CRACKING THE CODE INTERVIEW BOOK (for all these problems you should use STL-library), we will start again on these problems on MONDAY.

1. Write code to remove duplicates from an unsorted linked list.

   FOLLOW UP:
   How would you solve this problem if a temporary buffer is not allowed?

2. Implement an algorithm to find the nth to last element of a singly linked list.

3. Implement an algorithm to delete a node in the middle of a single linked list, given only access to that node.
   EXAMPLE
   Input: the node 'c' from the linked list a->b->c->d->e
   Result: nothing is returned, but the new linked list looks like a->b->d->e

4. You have two numbers represented by a linked list, where each node contains a single digit. The digits are stored in reverse order, such that the 1's digit is at the head of the list. Write a function that adds the two numbers and returns the sum as a linked list.
   EXAMPLE
   Input: (3 -> 1 -> 9 -> 9) + (5 -> 9 -> 2)
   Output: 8 -> 0 -> 2 -> 0 -> 1

5. Given a circular linked list, implement an algorithm which returns a node at the beginning of the loop.
   DEFINITION
   Circular linked list: A (corrupt) linked list in which a node's next pointer points to an earlier node, so as to make a loop in the linked list.
   EXAMPLE input: A -> B -> C -> D -> E -> C [the same C as earlier] output: C

NOTE: EVERY QUESTION's derivation of Time Complexity is MUST. 25% credit is writing and explaining about your algorithm's Time complexity (in comments section of the code).