Q.1 :

```
// Node class to represent a node in the linked list
class Node {
   int data;
   Node next;

   public Node(int data) {
      this.data = data;
      this.next = null;
   }
}

// LinkedList class to represent the linked list
class LinkedList {
   Node head;

   // Method to insert a new node at the end of the linked list
   public void insert(int data) {
      Node newNode = new Node(data);
      if (head == null) {
         head = newNode;
      } else {
         Node temp = head;
         while (temp.next != null) {
            temp = temp.next;
         }
         temp.next = newNode;
      }
   }
```

```
    // Method to find the length of the linked list
    public int findLength() {
        int length = 0;
        Node current = head;
        while (current != null) {
            length++;
            current = current.next;
        }
        return length;
    }
}

public class Main {
    public static void main(String[] args) {
        // Creating a sample linked list
        LinkedList linkedList = new LinkedList();
        linkedList.insert(1);
        linkedList.insert(2);
        linkedList.insert(3);
        linkedList.insert(4);
        linkedList.insert(5);

        // Finding and printing the length of the linked list
        int length = linkedList.findLength();
        System.out.println("Length of the linked list: " + length);
    }
}
```

Q.2 :

```
// Node class to represent a node in the linked list
class Node {
    int data;
    Node next;

    public Node(int data) {
        this.data = data;
        this.next = null;
    }
}

// LinkedList class to represent the linked list
class LinkedList {
    Node head;

    // Method to add 1 to the linked list representing a number
    public Node addOne(Node head) {
        Node dummy = new Node(0);
        dummy.next = head;
        Node lastNonNine = dummy, current = head;

        // Find the rightmost non-9 digit
        while (current != null) {
            if (current.data != 9) {
                lastNonNine = current;
            }
            current = current.next;
        }
```

```
        // Add 1 to the rightmost non-9 digit and set all following digits to 0
        lastNonNine.data++;
        current = lastNonNine.next;
        while (current != null) {
            current.data = 0;
            current = current.next;
        }

        // Check if the dummy node is still at the beginning, update head if needed
        return dummy.data == 1 ? dummy : dummy.next;
    }

    // Method to print the linked list
    public void printList(Node head) {
        Node current = head;
        while (current != null) {
            System.out.print(current.data + " ");
            current = current.next;
        }
        System.out.println();
    }
}

public class Main {
    public static void main(String[] args) {
        // Creating a sample linked list representing the number 123
        LinkedList linkedList = new LinkedList();
        linkedList.head = new Node(1);
```

```
        linkedList.head.next = new Node(2);
        linkedList.head.next.next = new Node(3);

        System.out.print("Original Linked List: ");
        linkedList.printList(linkedList.head);

        // Adding 1 to the linked list
        linkedList.head = linkedList.addOne(linkedList.head);

        System.out.print("Linked List after adding 1: ");
        linkedList.printList(linkedList.head);
    }
}
```