Task 5: Removing Duplicates from a Sorted Linked List
A sorted linked list has been constructed with repeated elements. Describe an algorithm to remove all duplicates from the linked list efficiently.

ANS:

```
package Assigmentday12.com;
public class Task5 {
  static class ListNode {
     int val;
     ListNode next;
     ListNode(int val) {
        this.val = val;
        this.next = null;
     }
  }
  static class DuplicateRemover {
     public static ListNode removeDuplicates(ListNode head) {
        ListNode current = head;
        while (current != null) {
           ListNode nextDistinct = current.next;

           // Find the next distinct node
           while (nextDistinct != null && nextDistinct.val ==
current.val) {
              nextDistinct = nextDistinct.next;
           }
           // Update current node's next pointer
           current.next = nextDistinct;
           current = nextDistinct;
        }
        return head;
     }
  }
  public static void main(String[] args) {

     ListNode head = new ListNode(1);
     head.next = new ListNode(1);
     head.next.next = new ListNode(2);
```

```
        head.next.next.next = new ListNode(3);
        head.next.next.next.next = new ListNode(3);
        head.next.next.next.next.next = new ListNode(3);
        System.out.println("Original List: ");
        printList(head);
        ListNode updatedList =
DuplicateRemover.removeDuplicates(head);
        System.out.println("List after removing duplicates: ");
        printList(updatedList);
    }
    public static void printList(ListNode head) {
        ListNode current = head;
        while (current != null) {
            System.out.print(current.val + " ");
            current = current.next;
        }
        System.out.println();
    }
}
package Assigmentday12.com;
public class Task5 {
    static class ListNode {
        int val;
        ListNode next;
        ListNode(int val) {
            this.val = val;
            this.next = null;
        }
    }
    static class DuplicateRemover {
        public static ListNode removeDuplicates(ListNode head) {
            ListNode current = head;
            while (current != null) {
                ListNode nextDistinct = current.next;

                // Find the next distinct node
```

```
            while (nextDistinct != null && nextDistinct.val ==
current.val) {
                nextDistinct = nextDistinct.next;
            }
            // Update current node's next pointer
            current.next = nextDistinct;
            current = nextDistinct;
        }
        return head;
    }
  }
  public static void main(String[] args) {

    ListNode head = new ListNode(1);
    head.next = new ListNode(1);
    head.next.next = new ListNode(2);
    head.next.next.next = new ListNode(3);
    head.next.next.next.next = new ListNode(3);
    head.next.next.next.next.next = new ListNode(3);
    System.out.println("Original List: ");
    printList(head);
    ListNode updatedList =
DuplicateRemover.removeDuplicates(head);
    System.out.println("List after removing duplicates: ");
    printList(updatedList);
  }
  public static void printList(ListNode head) {
    ListNode current = head;
    while (current != null) {
      System.out.print(current.val + " ");
      current = current.next;
    }
    System.out.println();
  }
}
```