```
public class merge_sort_OnLL {

    public static void main(String[] args) {
        int[] items = new int[7];
        Node head = null;
        for (int i = items.length; i >= 0; i--) {
            head = add_item(head, i);
        }

        print_Llist(head);
        System.out.println();
        merge_sort(head);
        print_Llist(head);
    }


    public static Node merge_sort(Node root) {
        if (root == null || root.link == null)
            return root;
        Node[] split = splitter(root);
        Node front = split[0], back = split[1];
        front = merge_sort(front);
        back = merge_sort(back);
        return merge(front, back);
    }


    private static merge_sort_OnLL.Node merge(merge_sort_OnLL.Node front,
merge_sort_OnLL.Node back) {
        if (front == null)
            return back;
        else if (back == null)
            return front;
        Node result;
        if (front.item <= back.item) {
            result = front;
            result.link = merge(result.link, back);
        } else {
            result = back;
            result.link = merge(front, result.link);
        }
        return result;
    }


    private static merge_sort_OnLL.Node[] splitter(merge_sort_OnLL.Node root) {
        if (root == null || root.link == null)
            return new Node[] { root, null };
        Node fast = root.link, slow = root;
        while (fast != null) {
            fast = fast.link;
            if (fast != null) {
                slow = slow.link;
                fast = fast.link;
            }
        }
        Node[] rtn_arr = new Node[] { root, slow.link };
        slow.link = null;
        return rtn_arr;
    }


    static class Node {
        int item;
        Node link;

        public Node(int item) {
            this.item = item;
            link = null;
        }
    }


    public static Node add_item(Node root, int val) {
        if (root == null)
            return new Node(val);
        Node tmp = root;
        while (tmp.link != null)
            tmp = tmp.link;
        tmp.link = new Node(val);
        return root;
    }


    public static void print_Llist(Node root) {
        if (root != null) {
            System.out.print(root.item + " ");
            print_Llist(root.link);
        }
    }

}
```