

```

1  /* Example 3 of Linked Lists: Object-oriented with lists inside the class
2  *   This class is a container for the data structures and algorithms but
3  *   allows information hiding inside this class.
4  *   See unit test main below for how to use this.
5  */
6
7  public class LinkedList {
8
9      private Node head;
10
11     private class Node {
12         public int item;
13         public Node next;
14
15         public Node() {                // for some reason, would not compile without
explicit declaration of this
16             item = 0;
17             next = null;
18         }
19
20         public Node(int k, Node p) {
21             item = k;
22             next = p;
23         }
24     }
25
26     public LinkedList() {
27         head = new Node();              // header node
28     }
29
30     public void printList() {
31         printListAux(head.next);
32     }
33
34     private void printListAux(Node p) {
35         if(p != null) {
36             System.out.println(p.item);
37             printListAux(p.next);
38         }
39     }
40
41     public void addInOrder(int k) {
42         head.next = addInOrderAux(k, head.next);
43     }
44
45     private Node addInOrderAux(int k, Node p) {
46         if(p == null || k < p.item)
47             return new Node(k, p);
48         else {
49             p.next = addInOrderAux(k, p.next);
50             return p;
51         }
52     }
53
54     }
55
56     public void mergeInto(LinkedList list) {    // Merge list into this one,
destroys argument list
57         head.next = merge(head.next, list.head.next);
58     }
59
60     private Node merge(Node p, Node q) {
61         if(p == null)
62             return q;
63         else if(q == null)
64             return p;
65         else if(p.item < q.item) {
66             p.next = merge(p.next, q);
67             return p;
68         }
69         else {
70             q.next = merge(q.next, p);
71             return q;
72         }
73     }
74
75     public static void main(String [] args) {
76
77         System.out.println("list 2:");
78         LinkedList list = new LinkedList();                // create a new list with
header node
79
80         list.addInOrder(7);
81         list.addInOrder(2);
82         list.addInOrder(13);
83         list.printList();
84         System.out.println("list 2:");
85         LinkedList list2 = new LinkedList();                // create a new list
with header node
86
87         list2.addInOrder(4);
88         list2.addInOrder(21);
89         list2.addInOrder(5);
90         list2.printList();
91         System.out.println("merged list 2:");
92     }
93 }

```

```
91         list.mergeInto(list2);
92         list.printList();
93     }
94 }
95
96 }
```