**LinkedList.java**

```java
package esof.utils;

/**
 * Represents a linked list of values (objects) of any type T.
 * Supports appending and traversing the list sequentially.
 * Changed to also support inserting and retrieving values by
 * position and finding a value in the list.
*/
public class LinkedList<T> {
    private Node<T> firstNode = null;
    private Node<T> lastNode = null;
    int length = 0;

    /**
     * Appends a value "x" at the end of the list.
     */
    public void append(T x) {
        Node<T> newNode = new Node<T>(x);
        if (firstNode == null)
            firstNode = newNode;
        else
            lastNode.setNextNode(newNode);
        lastNode = newNode;
    }

    /**
     * Retrieves the first node in the list (null if list is empty).
     * To get the value in that node and next node, see class Node.
     */
    public Node<T> getFirstNode(){
        return firstNode;
    }

    /**
     * Retrieves the value in position "n" of list (0 to length-1).
     */
    public T getNthValue(int n){
        Node<T> node = firstNode;
        for (int i = 0; i < n; i++)
            node = node.getNextNode();
        return node.getValue();
    }

    /**
     * Inserts value "x" in position "n" of the list (0 to length).
     */
    public void insert(T x, int n){
        Node<T> newNode = new Node<T>(x);
        if (n == 0) {
            newNode.setNextNode(firstNode);
            firstNode = newNode;
        }
        else {
            Node<T> prev = firstNode;
            for (int i = 0; i < n; i++)
                prev = prev.getNextNode();
            prev.setNextNode(newNode);
        }
    }
```

```
62
63      /**
64       * Retrieves the position of the first occurrence of value "x"
65       * in the list (between 0 and length), or -1 if not found.
66       */
67      public int find(T x){
68          int index = -1;
69          Node<T> node = firstNode;
70          while (node != null && node.getValue().equals(x)) {
71              node = node.getNextNode();
72              index++;
73          }
74          return index;
75      }
76   }
```

**Node.java**

```
77
78
79   package esof.utils;
80
81   /**
82    * Represents a node in a linked list of values (objects) of type T.
83   */
84
85   public class Node<T> {
86      private T value;
87      private Node<T> nextNode;
88
89      /**
90       * Creates a new node containing a value "x".
91       * Should only be called from LinkedList.
92       */
93      Node(T x) {
94          value = x;
95          nextNode = null;
96      }
97
98      /**
99       * Sets the next node.
100      * Should only be called from LinkedList.
101      */
102     void setNextNode(Node<T> nextNode) {
103         this.nextNode = nextNode;
104     }
105
106     /**
107      * Retrieves the value stored in this node.
108      */
109     public T getValue(){
110         return value;
111     }
112
113     /**
114      * Retrieves the next node (null if there is none).
115      */
116     public Node<T> getNextNode(){
117         return nextNode;
118     }
119  }
```