```java
import java.util.LinkedList;

/************************************************************
* Dalton Nofs                                               *
* Login ID: nofs5491                                        *
* CS-102, Summer 2017                                       *
* Programming Assignment 3                                  *
* ListInterface class: High level linked list of terms for  *
*                         database                           *
************************************************************/
public class Term implements TermInterface
{
    LinkedList<Course> courseList;  // Linked list of courses
    String term;                    // The term of the courses

    /************************************************************
    * Method: Term()                                            *
    * Purpose: default constructor for linkedList obj           *
    *                                                            *
    * Parameters:               N/A                             *
    * Returns: void:            N/A                             *
    ************************************************************/
    Term()
    {
        courseList = new LinkedList<Course>(); // init an empty
        term = null;
    }
    /************************************************************
    * Method: Term()                                            *
    * Purpose: constructor passed a string                       *
    *                                                            *
    * Parameters:               N/A                             *
    * Returns: void:            N/A                             *
    ************************************************************/
    Term(String term)
    {
        courseList = new LinkedList<Course>(); // init a empty
        this.term = term;
    }

    /************************************************************
    * Method: isEmpty()                                         *
    * Purpose: check to see if linkedList is empty              *
    *                                                            *
    * Parameters:               N/A                             *
    * Returns: boolean:         if list is empty                *
    ************************************************************/
    public boolean isEmpty()
    {
        if(courseList.isEmpty())
            return true;
        return false;
    }

    /************************************************************
    * Method: size()                                            *
    * Purpose: determine the size of linked list                *
    *                                                            *
    * Parameters:               N/A                             *
    * Returns: int:              the size of the array          *
    ************************************************************/
    public int size()
    {
        return(courseList.size());
    }

    /************************************************************
```

```
68      * Method: get()                                                  *
69      * Purpose: get object from linked list at index                 *
70      *                                                               *
71      * Parameters: int:           index                          *
72      * Returns: Object:           Object stored in index             *
73      *****************************************************************/
74      public Course get(int index)
75      {
76          return(courseList.get(index));
77      }
78
79      /*****************************************************************
80      * Method: add()                                                   *
81      * Purpose: add a object at specified index                      *
82      *                                                                *
83      * Parameters:                                                   *
84      *               int:           index                          *
85      *               Course:         Object to be placed             *
86      *                                                              *
87      * Returns: void:                N/A                            *
88      *****************************************************************/
89      public void add(int index, Course item)
90      {
91          courseList.add(index, item);
92      }
93
94      /*****************************************************************
95      * Method: remove()                                                 *
96      * Purpose: remove index postion and return object removed    *
97      *                                                               *
98      * Notes: calls func that can throw indexoutboundsexception     *
99      *                                                               *
100     * Parameters: int:           index                          *
101     * Returns: Object:           Object removed                     *
102     *****************************************************************/
103     public Course remove(int index)
104     {
105         return(courseList.remove(index));
106     }
107
108     /*****************************************************************
109     * Method: removeAll                                               *
110     *                                                              *
111     * Purpose: removes all nodes from array                         *
112     *                                                               *
113     * Parameters:              N/A                                *
114     * Returns: void:           N/A                                *
115     *****************************************************************/
116     public void removeAll()
117     {
118         courseList.removeAll(courseList);
119     }
120
121     /*****************************************************************
122     * Method: append                                                *
123     *                                                              *
124     * Purpose: appends the course to the end of the list         *
125     *                                                               *
126     * Parameters: Course:         courseIn                          *
127     * Returns: void:            N/A                                *
128     *****************************************************************/
129     public void append(Course courseIn)
130     {
131         courseList.addLast(courseIn);
132     }
133
134     /*****************************************************************
135     * Method: removeAll                                               *
```

```
136     *                                                               *
137     * Purpose: removes all nodes from array                          *
138     *                                                                *
139     * Parameters:              N/A                                  *
140     * Returns: String           the term                             *
141     ****************************************************************/
142     public String getTerm()
143     {
144         return(this.term);
145     }
146 }
```