```java
  1 import java.io.IOException;
  2
  3 /*****************************************************************
  4  * Dalton Nofs                                                  *
  5  * Login ID: nofs5491                                           *
  6  * CS-102, Summer 2017                                          *
  7  * Programming Assignment 5                                     *
  8  * Term class: High level binary tree of terms for database    *
  9  *****************************************************************/
 10 public class Term implements TermInterface
 11 {
 12     Tree<Course> courseTree;        // Tree of all the courses
 13     String term;                    // The term of the courses
 14
 15     /*****************************************************************
 16      * Method: Term()                                               *
 17      * Purpose: default constructor for linkedList obj              *
 18      *                                                              *
 19      * Parameters:                 N/A                              *
 20      * Returns: void:              N/A                              *
 21      *****************************************************************/
 22     Term()
 23     {
 24         courseTree = new Tree<Course>(); // init an empty tree
 25         term = null;
 26     }
 27     /*****************************************************************
 28      * Method: Term()                                               *
 29      * Purpose: constructor passed a string                         *
 30      *                                                              *
 31      * Parameters:                 N/A                              *
 32      * Returns: void:              N/A                              *
 33      *****************************************************************/
 34     Term(String term)
 35     {
 36         courseTree = new Tree<Course>(); // init a empty tree
 37         this.term = term;
 38     }
 39
 40     /*****************************************************************
 41      * Method: isEmpty()                                            *
 42      * Purpose: check to see if linkedList is empty                 *
 43      *                                                              *
 44      * Parameters:                 N/A                              *
 45      * Returns: boolean:        if list is empty                    *
 46      *****************************************************************/
 47     public boolean isEmpty()
 48     {
 49         if(courseTree.isEmpty())
 50             return true;
 51         return false;
 52     }
 53
 54     /*****************************************************************
 55      * Method: getSearched()                                        *
 56      * Purpose: get object from linked list at index                *
 57      *                                                              *
 58      * Parameters:                 N/A                              *
 59      * Returns: Object:        Object stored in index               *
 60      *****************************************************************/
 61     public Course getSearched()
 62     {
 63         return(courseTree.getSearched().getDatum());
 64     }
 65
 66     /*****************************************************************
 67      * Method: search()                                             *
```

```java
 68       * Purpose: searches tree for target                              *
 69       *                                                                *
 70       * Parameters: T:          target                                 *
 71       * Returns: boolean:       if found or not                    *
 72       ****************************************************************/
 73      public boolean search(Course target)
 74      {
 75          return courseTree.search(target);
 76      }
 77
 78      /****************************************************************
 79       * Method: add()                                              *
 80       * Purpose: add a object at specified index                 *
 81       *                                                            *
 82       * Parameters:                                               *
 83       *              Course:         Object to be placed          *
 84       *                                                            *
 85       * Returns: void:              N/A                           *
 86       ****************************************************************/
 87      public void add(Course item) throws IOException
 88      {
 89          courseTree.add(item);
 90      }
 91
 92      /****************************************************************
 93       * Method: remove()                                           *
 94       * Purpose: remove item from tree                          *
 95       *                                                            *
 96       * Parameters: Course:       course to remove                *
 97       * Returns: void:            nothing is returned             *
 98       ****************************************************************/
 99      public void remove(Course item)
100      {
101          courseTree.remove(item);
102      }
103
104      /****************************************************************
105       * Method: removeAll                                          *
106       *                                                            *
107       * Purpose: removes all nodes from array                     *
108       *                                                            *
109       * Parameters:              N/A                              *
110       * Returns: void:           N/A                              *
111      ****************************************************************/
112      public void removeAll()
113      {
114          courseTree.removeAll();
115      }
116
117      /****************************************************************
118       * Method: removeAll                                          *
119       *                                                            *
120       * Purpose: removes all nodes from array                     *
121       *                                                            *
122       * Parameters:              N/A                              *
123       * Returns: String          the term                        *
124      ****************************************************************/
125      public String getTerm()
126      {
127          return(this.term);
128      }
129
130      /****************************************************************
131       * Method: getRoot()                                          *
132       *                                                            *
133       * Purpose: gets tree root node                             *
134       *                                                            *
135       * Parameters:              N/A                              *
```

```java
136        * Returns: TreeNode<T>:        the root node                        *
137        *************************************************************/
138       public TreeNode<Course> getRoot()
139       {
140           return(this.courseTree.getRoot());
141       }
142 }
```