CSC 212 Programming Assignment # 1
Implementing and Using Lists
Due date: 10/10/2019
Mark:+4. Plagiarism:-4.

Guidelines:	This is an **individual** assignment. The assignment must be submitted to **Web-CAT**

The goal of this assignment is to implement and use the ADT List. The assignment is divided into two parts. In the first part, you will implement the ADT List with additional operations using linked and array representations. In the second part, you will write methods that use these implementations to store and query information about movies.

1 Implementing List

Write two classes `ArrayList` and `LinkedList` that implement the interface `List` below. You may use code from the lecture notes for the methods similar to those discussed in class.

```
public interface List<T> {
  boolean empty();
  boolean full();
  void findFirst();
  void findNext();
  boolean last();
  T retrieve();
  void update(T e);
  void insert(T e);
  void remove();
  // Searches for the first element that satisfies a condition. If found,
       it is set as current and true is returned, otherwise current
     remains unchanged and false is returned. The condition is tested
     using cnd.
  boolean findFirstEle(Cond<T> cnd);
  // Searches and returns all elements that satisfy a condition. If none
     is found, the empty list is returned. This method does not change
     current. The condition is tested using cnd.
  List<T> findAllEle(Cond<T> cnd);
}
```

The interface `Cond` is given below:

```
public interface Cond<T> {
  // Return true if e satisfies the condition.
  boolean test(T e);
}
```

Example 1.1. *If you want to search for the first occurence of a given string in a list of strings, you can proceed as follows. First write the following implementation of the interface* `Cond`*:*

```
public class StringEqualCond implements Cond<String> {
  private String str;
  public StringEqualCond(String str) {
    this.str = str;
  }
  @Override
  public boolean test(String s) {
    return str.equals(s);
  }
}
```

You can the use it to serach for "hello" as follows:

```
public class SearchHello {
  public static void main(String[] args) {
    List<String> l = new LinkedList<String>();
    l.insert("cat");
    l.insert("hello");
    l.insert("dog");
    StringEqualCond cnd = new StringEqualCond("hello");
    System.out.println(l.findFirstEle(cnd)); // prints true
  }
}
```

2 Using List

Information about movies is stored in the class `Movie`:

```
public class Movie {
  public int id;
  public String title;
  public List<String> genres;

  public Movie(int id, String title, List<String> genres) {
    this.id = id;
    this.title = title;
    this.genres = genres;
  }
}
```

Note that IDs are unique but not titles and genres.

Example 2.1. *This is an example of using the class* `Movie`*:*

```
List<String> genres = new LinkedList<String>();
genres.insert("Adventure");
genres.insert("Animation");
genres.insert("Children");
genres.insert("Comedy");
genres.insert("Fantasy");
movies.insert(new Movie(1, "Toy␣Story␣(1995)", genres));
```

We want to write the following class that answers queries about movies:

```
public class MovieUtils {
  // Return the movie with the given id if found, null otherwise.
  public static Movie findMovieById(List<Movie> movies, int id) {
    return null;
  }
  // Return the list of movies having a given title. If none is found,
     empty list is returned.
  public static List<Movie> findMovieByTitle(List<Movie> movies, String
     title) {
    return null;
  }
  // Return the list of movies of a given genre. If none is found, empty
     list is returned.
  public static List<Movie> findMoviesByGenre(List<Movie> movies, String
     genre) {
    return null;
  }
  // Return the list of movies of given genres (a movie must have all
     genres to be in the list). If none is found, empty list is returned.
      Assume genres is not empty.
  public static List<Movie> findMoviesByGenres(List<Movie> movies, List<
     String> genres) {
    return null;
  }
}
```

Example 2.2. *An example of using the class* `MovieUtils` *can be found in the class* `Main` *attached to this assignment.*

3 Deliverable and rules

You must deliver:

1. Source code submission to Web-CAT. You have to upload the following classed in a zipped file in addition to any other classes you need:

   - `LinkedList.java`
   - `ArrayList.java`
   - `MovieUtils.java`

   Notice that you should **not upload** the interfaces `List`, `Cond` and the classes `Movie`, `Main`.

The submission **deadline** is: **10/10/2019**.

You have to read and follow the following rules:

1. The specification given in the assignment (**class and interface names, and method signatures**) must not be modified. Any change to the specification results in compilation errors and consequently the mark zero.

2. All data structures used in this assignment **must be implemented** by the student. The use of Java collections or any other data structures library is strictly forbidden.

3. This assignment is an individual assignment. Sharing code with other students will result in harsh penalties.

4. Posting the code of the assignment or a link to it on public servers, social platforms or any communication media including but not limited to Facebook, Twitter or WhatsApp will result in disciplinary measures against any involved parties.

5. The submitted software will be evaluated automatically using Web-Cat.

6. All submitted code will be automatically checked for similarity, and if plagiarism is confirmed penalties will apply.

7. You may be selected for discussing your code with an examiner at the discretion of the teaching team. If the examiner concludes plagiarism has taken place, penalties will apply.