Project 1
CSC 1052 - Algorithms and Data Structures II
Grading: 100 points
Due Date: March 2th, 2017

Description: In this project, you will be creating a Linked List based song player, see Figure 1. You will be provided with a fully functional Mp3 player that uses arraylists as its underlying data structure. Your task is to completely replace the arraylist and its methods with the Linked List equivalents. The Mp3Player.java file contains all code related to the GUI and playing of Mp3 files, as well as a main class that will test the data structure. **You do not modify this file**. In fact, you do not even need to understand all of the details of the Mp3Player.java file. However, you will need to completely rewrite the List.java file so that it uses Linked Lists. For your reference, I am using the javazoom mp3 decoder which is imported in the following way,

import javazoom.jl.player.*; // This is the MP3 decoder and player
import java.io.FileInputStream; // This allows us to read files from disk

For Eclipse, here are the instructions for adding the javazoom mp3 player to your classpath:
Drag and drop the jar file into your /src folder.
Right-click the project and select Properties
under the Java Build Path menu, select the Libraries button
On the right-hand side, press the Add jars button and add the javazoom jar from your project source folder

If you are having trouble getting music to play, you may need to modify one line in Mp3Player.java by adding the complete directory path in front of the String "song" that is passed into `new FileInputStream()`. For example, a Mac user would change

```
FileInputStream fis = new FileInputStream(song);
```

to contain the path to your source folder:

```
FileInputStream fis = new FileInputStream("/Users/YOURUSERNAME/Documents/workspace/
LASTNAME_project1/src/" + song);
```

Part 1 - List.java (80 points)
For the List, you will be creating a singly linked list with the following methods

1. `void insertItem (String name)` - This method will insert a String item to the end of the linked list.
2. `void insertItem (String name, int pos)` - This method will insert a String item to the given position in the linked list. You can assume that a position of 0 means the front of the list. (if position is out of bounds, do not insert, but output out of bounds)

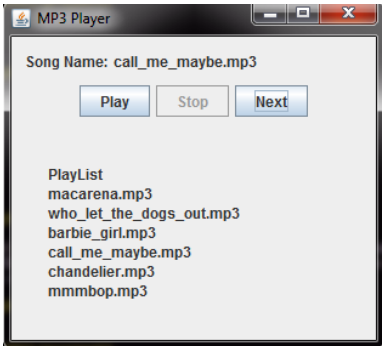

Figure 1: Visualization of the Mp3 player and correct song ordering.

3. `boolean removeItem( String aname)` - This method will find the String with the same value as aname and remove the first occurrence of that string from the list. The return of this method is either true (success) or false if the string does not exist in the list. Use the *.equals* method when comparing strings.
4. `void removeItem(int position)` - Remove the node at position (if position is out of bounds, do not remove, but output out of bounds)
5. `boolean contains(String name)` - Return true if the linked list contains the string.
6. `void clear()` - Remove all of the elements from the linked list
7. `String get(int index)` - Return the element at the specified position in the list.
8. `int size()` - Returns the number of elements in this list.
9. `String toString()` - Override the toString method to print out the names of all the songs in the list
10. `String toHTMLString()` -Return a String that starts with $< html >$ ends with $< /html >$ and uses the line breaks, $< br/ >$, in replacement of new line characters and prints out all the names of all the songs in the list. See the current implementation for details.

Next, create a LLNode class in a new java file with fields `name` and `link` of type String and LLNode, respectively. You will additionally be creating three constructors,

1. `public LLNode()` - constructor with no parameters, songName should be an empty string
2. `public LLNode(String name)` - constructor with 1 parameter, set the name of the song file
3. `public LLNode(String name, LLNode next)` - constructor with 2 parameters, name of the song and the next node in the list

Part 2 - PDF write up (20 points) Create a 1-2 page write up that summarizes the interesting parts of your program. Include any problems or insights that you encountered.

Deliverables: Submit the "lastname_project01.zip" file containing List.java, LLNode.java, and your report on Blackboard.