Queens College
Computer Science Department

CS 212 – Spring 2024 – Project 2

Assigned: 20 March 2024
Due: 5 April 2024
Cutoff: 11 April 2024

Improving on the Word Game

Add the following improvements to the word game. (1) The first letter of the subject letters (the first line of the input file) must be contained in all the correct guessed words. (2) If a guessed word contains ALL of the subject letters, that is worth 3 points. (3) Display the correctly guessed words in alphabetical order.

Lists of Words

Create a class called *WordNode* which has fields for the *data* (a Word) and *next* (WordNode) instance variables. Include a one-argument constructor which takes a Word as a parameter. (For hints, see the PowerPoint on "Static vs. Dynamic Structures".)

```
      public WordNode (Word w) {  . . }
```

The instance variables should have protected access.

Create an abstract linked list class called *WordList.* This should be a linked list with head node as described in lecture. Modify it so that the data type in the nodes is *Word*. The no-argument constructor should create an empty list with *first* and *last* pointing to an empty head node, and *length* equal to zero. Include an append method in this class.

Create two more linked list classes that extend the abstract class *WordList*: One called *UnsortedWordList* and one called *SortedWordList*, each with appropriate no-argument constructors. Each of these classes should have a method called *add(Word)* that will add a new node to the list. In the case of the UnsortedWordList it will add it to the end of the list by calling the append method in the super class. In the case of the SortedWordList it will insert the node in the proper position to keep the list sorted.

Instantiate two linked lists, one sorted and one unsorted. Add the solutions from the input file to the unsorted linked list. This list will be searched to see if a guessed word matches. As words are correctly guessed, add them to the sorted list and display the contents of that list in the TextArea for the guessed words. Check the method *setText* in class TextArea to update the contents of the TextArea.

Submit a jar file.

Rather than upload all the files for this project separately, we will use Java's facility to create the equivalent of a zip file that is known as a **J**ava **AR**chive file, or "jar" file.

Instructions on how to create a jar file using Eclipse are on Blackboard. Be sure to include **source** files (.java files), not class files in your jar. Create a jar file called **Project2.jar** and submit that.