CS 1331 Homework 6

Due Thursday, February 21st , 2013 8:00pm

Introduction

This week the homework is meant to give you some hands on experience with arrays through creating computer generated MadLibs (http://en.wikipedia.org/wiki/Mad_Libs).

We will create one class that actually represents the MadLib. Then we will create a Driver to create 3 MadLibs. At this point, we could get user input to fill in the blanks but since we have been doing everything with user input so far we are going to go in a different direction here. We are going to randomly select the words from a list to fill in the blanks in the MadLib.

Also make sure to javadoc all your files!

If you get stuck, feel free to post on Piazza or come to office hours!

6.1 WordList.java

In order to randomly select words from a list, we first need something that can actually hold a list of words. This is a very simple class that contains an array of Strings, and a method to get a random word from the list. To populate this array, the constructor of the WordList will need to take in the name of a file that contains a list of words. You can assume that the format is the exact same for all of the files (the first line contains the number of words in the file, and then there is one word per line for the remainder of the file.)

Here is an outline of what you class should look like:

- Instance Variables(make sure these are the correct visibility!)
    - Random rand
    - String[] words
- Methods
    - Constructor
        - `public WordList(String fileName){…}`
    - random word method
        - `public String getRandomWord(){…}`
        - this method should get a random word from the list of words

6.2 MadLib.java

For this part of the assignment, we want to define the class that represents the logic of a single MadLib. The base text of the MadLib will also be contained in a file. The first line of the text file contains the number of blanks in the file. The blanks will be represented like this: [noun]. The word inside the brackets will be noun, verb, adjective, or adverb. Your program should not do anything to modify this file.

The MadLib class should keep a reference to that filename, and it should have two arrays for the missing words. One array should contain the information for the type of word that should go in the blank, and one array should contain the actual word that is going to be placed in that blank. The class will also have methods to get the type of the next blank, fill the next blank with a word, and get whether or not there are any blanks that still need to be filled.

It will also have a toString method that returns a string containing the text of the MadLib. If the blank has not been filled, the word type should be shown in its place. When the blank has been filled, the correct word should be in the space. The inserted word should be in all capital letters to show that this once was a blank space.

Here is an outline of the variables and methods your class should have:

- Instance Variables(make sure these are the correct visibility!)
    - String filename
    - String[] blanks
    - String[] blankTypes
- Methods
    - Constructor
        - `public MadLib(String filename){…}`
    - method to get the type of the next blank
        - `public String getNextType(){…}`
    - method to fill the next blank
        - `public void fillNextBlank(String word) {…}`
    - method to determine if there are any more blanks to be filled
        - `public boolean hasMoreBlanks(){…}`
    - to String method
        - `public String toString(){…}`

Note: It would be better style to use a 2D array for the blanks; each inner array would be for one particular word, and one index would be the type and another would be the actual word. If you would like to do it this way you are welcome to :]

6.3 MadLibDriver.java

This class will be where we actually create our WordLists and MadLibs. You should create an array that contains 3 MadLib objects. One for each of the provided text files. You will also create 4 WordLists, one for each of the provided word text files.

You should go through the MadLib array. For each element in the array, you should print out the madlib before any words have been added, then go through and fill all of the blanks with random words from the files (make sure you are putting nouns in noun places, etc.). Once the blanks have been filled, print out the resulting MadLib.

The output should be similar to this(the weird spacing has to do with Word wrapping lines around, but this way I formatted it, every line has 10 words on it):

```
-------------------------------------------------
B.F. Skinner was the founder of [noun] , a psychological school of [noun] that [verb]
all behavior can be [verb] by [adjective] conditioning or circumstances. This means
that [adjective] tendencies
have no influence on the [noun] or [noun] of people, and that a person could
be conditioned to be [adverb] [adjective] than he is now.

B.F. Skinner was the founder of PHONE , a psychological school of ANIMAL that POKE
all behavior can be BUY by IDIOTIC conditioning or circumstances. This means that SHY
tendencies
have no influence on the ANIMAL or DOG of people, and that a person could
be conditioned to be ANGRILY TENDER than he is now.
-------------------------------------------------
Today I went to the zoo. I saw a [adjective] [noun] jumping up and down
in its tree. He [verb] [adverb] through the large tunnel that led to its [adjective]
[noun] . I got some peanuts and passed them through the cage to a gigantic
gray [noun] towering above my head. Feeding that animal made me hungry. I went to
get a [adjective] scoop of ice cream. It filled my stomach. Afterwards I had to
[verb] [adverb] to catch our bus. When I got home I [verb] my mom for
a [adjective] day at the zoo.

Today I went to the zoo. I saw a BERSERK PEN jumping up and down
in its tree. He CLEAN CAREFULLY through the large tunnel that led to its HORRIBLE
DOG . I got some peanuts and passed them through the cage to a gigantic
gray DOG towering above my head. Feeding that animal made me hungry. I went to
get a HOMELY scoop of ice cream. It filled my stomach. Afterwards I had to
HUNT CAREFULLY to catch our bus. When I got home I ASK my mom for
a BEAUTIFUL day at the zoo.
-------------------------------------------------
Look at this [noun] , isn`t it neat? Wouldn`t you think my collection`s complete?
Wouldn`t
you think I`m the [noun] The [noun] who has everything? Look at this trove, treasures
untold How many wonders can one [noun] hold? [verb] around here, you`d think Sure,
she`s
got everything I`ve got [noun] and [noun] a-plenty I`ve got who`s-its and what`s-its
galore You
[verb] thing-a-mabobs? I`ve got twenty But who cares? No [adjective] deal. I [verb]
more

Look at this PICTURE , isn`t it neat? Wouldn`t you think my collection`s complete?
Wouldn`t
you think I`m the SUN The FOOD who has everything? Look at this trove, treasures
untold How many wonders can one BOY hold? BITE around here, you`d think Sure, she`s
got everything I`ve got COMPUTER and PHONE a-plenty I`ve got who`s-its and what`s-its
galore You
```

```
CRY thing-a-mabobs? I`ve got twenty But who cares? No FEW deal. I WISH more
```

Turn-in Procedure

Turn in the following files to T-Square. When you are ready, make sure that you have actually **submitted** your files, and not just saved them as a draft.

- WordList.java
- MadLib.java
- MadLibDriver.java

Note** Always submit .java files - never submit your .class files. And be 100% certain that the files you turn in compile and that the program runs - submissions that do not will receive an automatic 0. Also, make sure that your files are in on time; the real deadline is 8 pm. While you have until 2 am to get it turned in, we will not accept homework past 2 am for any reason. Don't wait until the last minute!

Verify the Success of Your HW Turn-in

Practice "safe submission"! Verify that your HW files were truly submitted correctly, the upload was successful, and that the files compile and run. It is solely your responsibility to turn in your homework and practice this safe submission safeguard.

1. After uploading the files to T-Square you should receive an email from T-Square listing the names of the files that were uploaded and received. If you do not get the confirmation email almost immediately, something is wrong with your HW submission and/or your email. Even receiving the email does not guarantee that you turned in exactly what you intended.
2. After submitting the files to T-Square, return to the Assignment menu option and this homework. It should show the submitted files.
3. Download copies of your submitted files from the T-Square Assignment page placing them in a new folder.
4. Recompile and test those exact files.
5. This helps guard against a few things.
    a. It helps insure that you turn in the correct files.
    b. It helps you realize if you omit a file or files. (If you do discover that you omitted a file, submit all of your files again, not just the missing one.)
    c. Helps find last minute causes of files not compiling and/or running.