CS103-01 **Programming Assignment #3** **Due: 7/25/2017**

Overview:
This third programming assignment will continue building on what we've already learned. For this programming assignment, you'll have to make use of methods, loops, Strings, arrays, and random number generation. In this programming assignment, you will implement a version of the popular hangman word guessing game.

Brief Description:
For those unfamiliar with hangman, it is a common word guessing game where a person (or in this case, the computer) chooses a random word, and the player has to try to determine which word was chosen. The player guesses one letter at a time. For each guessed letter which appears in the randomly chosen word, the computer fills in the letter in the correct place on the word. For each incorrectly guessed letter, the letter is added to a list of wrong guesses. If this list grows to length 6, the player loses (i.e., the player must determine the word with fewer than 6 incorrect letter guesses).

Details:
When your program first runs, it should print out print out your name, the course number ("CS103-01"), and the programming assignment name ("Programming Assignment #3"), each on a new line. Then, the program should skip a line (i.e., print a blank line), and print a brief description of what the program does and the rules of the game.

The program should then choose a random word from the word bank (listed below). The word bank consists of an array of Strings.

Once a word is chosen, the program should start the game. In each round, the game should print out the randomly chosen word, using asterisks to represent letters which have not been guessed yet, and using the appropriate letter for already guessed letters. For example, if the randomly chosen word is "cat", and the user has guessed "a" but not "c" or "t", the game should print out "*a*". On the next line, after the word is printed, the game should print out the list of all "misses", or those letters which the user has guessed but are not present in the word. Finally, on the line after that, the game should present the user with a prompt to enter their next letter guess. After the user enters their next guess, the game should inform the user whether or not the guess appears in the word. Then, the next round should start.

The game ends when the user guesses all the letters in the randomly chosen word, or when the user has entered 6 incorrect guesses. Upon winning or losing, the game should inform the user that the game is over, and indicate whether the user won or lost.

Specific Implementation Requirements:

1. A random number should be generated to randomly select a word from the list of words.
2. You should implement at least two additional methods, besides main. It's up to you exactly what methods to implement. Anything reasonable is okay, as long as your code works correctly using these methods.
3. All methods should have appropriate comments explaining what data the method takes in, what data the method returns, and what the method does (this includes main!).
4. You can use whatever kinds of loops you want, but you need to ensure that the game continues running until either the user guesses the word, or the user runs out of guesses, at which point the game should inform the user whether they won or lost and exit.

CS103-01 **Programming Assignment #3** **Due: 7/25/2017**

Word bank:
Paste this code somewhere in your code. It declares, creates, and initializes an array of strings. This is the word bank from which your program should randomly choose a word.

```
final String[] words =   //the word bank
      {"mathematics", "cat", "yesterday", "java", "truck", "coffee",
      "fish", "token", "transit", "bottom", "apple", "cake",
      "remote", "pocket", "class", "arm", "cranberry", "tool",
      "caterpillar", "spoon", "watermelon", "laptop", "toe", "toad",
      "fundamental", "capitol", "capital", "anticipate", "cherry"};
```

Hints:

1. In order to store a copy of the word which has asterisks for each of the unguessed letters, it might be easier to use a character array. With character arrays, individual characters can easily be changed, whereas changing individual characters in strings is more complicated. It's also possible to easily create a String from a character array, or get a character array from a String.
2. Remember the indexOf method can be used to search for individual characters in a string.
3. You'll need to loop when searching for a character – what if the character appears in the word more than once?

Submission:
All programming assignments are to be submitted as hard copy **and** electronically. For the electronic submission, just submit your .java file on Canvas. Assignments should be turned in during class on the due date (or earlier). Grading late assignments is really time consuming (seriously – I didn't realize this until I started grading), so don't turn in any assignment late unless you've made arrangements with me.

As always, do your own work.

Example Run:

```
Jacob Hauenstein
CS103-01
Programming assignment 3

This program implements the common hangman game. A random word has been chosen from the
word bank. You must try to guess all the letters in the word. If you make 6 incorrect
guesses, you lose.

Word: *****
Misses:
Guess: a

You guessed a. a does appear in the word.

Word: a****
Misses:
Guess: o

You guessed o. o does NOT appear in the word.

Word: a****
Misses: o
Guess: e

You guessed e. e does appear in the word.

Word: a***e
Misses: o
Guess: p

You guessed p. p does appear in the word.

Word: app*e
Misses: o
Guess: l

You guessed l. l does appear in the word.

Word: apple
Misses: o
You win!
```

Hints:

Remember that to read in a single character, the book suggests to read in an entire line of text and then just grab the first character from it.