Introduction
This exercise is meant to introduce you using key built-in classes (such as String and I/O classes) in Java, as well as researching classes and methods in the on-line Java API. You will be implementing a simple one-dimensional version of the game “Boggle”, in which your program generates a random string of characters, and the user then searches for words in that string. Your application class should be called Boggle.java.

Rules and Sample Run
The program should first generate a string of randomly chosen characters. To increase the odds of actually forming words, those characters will be chosen from the 20 letters a, b, d, e, f, g, h, i, l, m, n, o, p, r, s, t, u, w, x, y. For example:

```
oiwnbdpbfbapfydsrnfwbyymxuserdnlbxtlhpasyseamee
```

The user should then be repeatedly prompted for words that appear inside the string, until they choose to quit by entering the string QUIT. For example:

```
oiwnbdpbfbapfydsrnfwbyymxuserdnlbxtlhpasyseamee
Enter a word found in this puzzle (or QUIT to quit): by
Yes, that is in the puzzle
Enter another word found in this puzzle: use
Yes, that is in the puzzle
Enter another word found in this puzzle: user
Yes, that is in the puzzle
Enter another word found in this puzzle: pi
Yes, that is in the puzzle
Enter another word found in this puzzle: fa
Yes, that is in the puzzle
Enter another word found in this puzzle: seam
Yes, that is in the puzzle
Enter another word found in this puzzle: sea
Yes, that is in the puzzle
Enter another word found in this puzzle: me
Yes, that is in the puzzle
Enter another word found in this puzzle: am
Yes, that is in the puzzle
Enter another word found in this puzzle: as
Yes, that is in the puzzle
Enter another word found in this puzzle: QUIT
```
Listing words found

After the user quits, they should then be told how many words they found, and those words should be listed:

```
quit, that is in the puzzle
Enter another word found in this puzzle: QUIT
You found the following 10 words in the puzzle:
by
use
user
pi
fa
seam
sea
me
am
as
BUILD SUCCESSFUL (total time: 1 minute 54 seconds)
```

Words not in string

If a word entered by the user is not in the string, they should be told so:

```
euxlrgmoensinwrxpwdgmsiafphuigltlononudtemwhyryht
Enter a word found in this puzzle (or QUIT to quit): in
Yes, that is in the puzzle
Enter another word found in this puzzle: why
Yes, that is in the puzzle
Enter another word found in this puzzle: sing
Sorry, that word is not in the puzzle.
Enter another word found in this puzzle: |
```

Note that you do not have to determine whether something is actually a word or not! You simply have to determine whether the string entered by the user is a substring of the puzzle string.

Duplicate words entered by the user

If the user enters a word they have already found, they should be told so:
Program Structure

As with the first assignment, your program is to be a command line application with a main method. You should also have at least two other static methods called by main. I suggest that you have the following separate methods:

- A method that generates and returns the random 50-character puzzle string.
- A method that takes an array of strings (the words the user has guessed so far) and another string (the current guess) as parameters, and returns a Boolean indicating whether or not the string is in the array.

Hints and Suggestions:

Generating a random word

A simple way to select a random letter from a list of given letters (such as the 20 letters in this assignment) is to put them all in a single string. For example:

```java
String letters = "abdefghilmnopqrstuvwxyz";
```

You can then choose a random letter by generating a random number between 0 and 19, and getting the character at that position in the string. You would then build your random puzzle string by doing this 50 times.

Keeping track of previous words

In order to print the words guessed by the user at the end, and to detect words previously guessed, you will need to keep track of words (correctly) guessed by the user. The simplest way to do this is to keep track of an array of Strings.

This means that you will have to construct such an array at the beginning of your program. You may assume that the user will never guess more than 50 correct words in a game.
You will also need to keep track of how many words the user has correctly guessed so far, in order to know 1) what index to store the next word at, and 2) how many times to loop through the array in order to print/search for words.

---

### Researching Classes and Methods in the Java API

One of the goals of this assignment is for you to become familiar with finding information about classes and methods in the online Java API. You will need to use this to find the following:

- You will need some way to generate **random numbers between 0 and 19** to create your random string. This can be done in one of two ways:
  - Using a static method in the `Math` class.
  - Using the `Random` class in the `java.util` package.

- You will need to use several methods in the `String` class to do the following:
  - Extracting the random letter from your string of letters, and appending it to the end of the puzzle string you are constructing.
  - Determining whether or not the puzzle string contains the word entered by the user as a substring.

---

### What to Turn In

Email me the **source code** for your `Boggle.java` class at [jrsullins@ysu.edu](mailto:jrsullins@ysu.edu).

You can find the source code for your `Boggle.java` class on your computer under the `Documents/NetBeansProjects/Boggle/src` directory.

The subject line of your email should be **CSIS 3701 Exercise 2 from yourname.**

As with any program, your source code should be **well documented!**