Question 1

25 minutes to complete, 5 minutes to upload answer.

1. This question involves the implementation and extension of a `RandomStringChooser` class.

   (a) A `RandomStringChooser` object is constructed from an array of non-null String values. The `RandomStringChooser` class has a `getRandomSequence` method, which has the following behavior. A call to `getRandomSequence` returns a `String[]` array containing a random sequence of the available strings in the object.

   The following code segment shows an example of the behavior of `RandomStringChooser`.

```
String[] wordArray = {"wheels", "on", "the", "bus"};
String[] sChooser = new RandomStringChooser(wordArray);
for (String str : sChooser.getRandomSequence())
{
   System.out.print(str + " ");
}
```

   One possible output is shown below.

```
bus the wheels on
```

   WRITE YOUR SOLUTION ON THE NEXT PAGE.

Write the entire `RandomStringChooser` class. Your implementation must include an appropriate constructor and any necessary methods. Any instance variables must be private. The code segment in the example above should have the indicated behavior (that is, it must compile and produce a result like the possible output shown). Neither the constructor nor any of the methods should alter the parameter passed to the constructor, but your implementation may copy the contents of the array.

(b) The following partially completed `RandomLetterChooser` class is a subclass of the `RandomStringChooser` class. You will write the method `getSingleLetters` for the `RandomLetterChooser` class.

```
public class RandomLetterChooser extends RandomStringChooser
{
      /** Constructs a random letter chooser using the given string str.
       * Precondition:
       * str contains only letters.
       */
      public RandomLetterChooser(String str)
      {
           super( getSingleLetters(str) );
      }

      /** Returns an array of single-letter strings.
        * Each of these strings consists of a single letter from str.
        * Element k of the returned array contains the single letter at
        * position k of str.
        * For example, getSingleLetters("cat") returns the
        * array { "c", "a", "t" }.
        */
      public static String[] getSingleLetters(String str)
      { /* to be implemented in part (b) */ }
}
```

The following code segment shows an example of using `RandomLetterChooser`.

```
RandomLetterChooser letterChooser = new RandomLetterChooser("cat");
String[] strArray = letterChooser.getRandomSequence();
for (int k = 0; k < strArray.length; k++)
{
   System.out.print(strArray[k]);
}
```

The code segment will print the three letters in "`cat`" in one of the possible orders. One possible output is shown below.

```
act
```

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Complete the `getSingleLetters` method below.

```
/** Returns an array of single-letter strings.
 *  Each of these strings consists of a single letter from str. Element k
 *  of the returned array contains the single letter at position k of str.
 *  For example, getSingleLetters("cat") returns the
 *  array { "c", "a", "t" }.
 */

public static String[] getSingleLetters(String str)
```