AP Java HW
Mr. Neat

Free Response

Write a complete Java program (including all necessary classes) to answer the following question (all parts). Submit a zipped folder with your solution to this GC assignment. Be sure to include a few sample outputs pasted into the starter.java file and commented out.

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. When the object is first constructed, all of the strings are considered available. The `RandomStringChooser` class has a `getNext` method, which has the following behavior. A call to `getNext` returns a randomly chosen string from the available strings in the object. Once a particular string has been returned from a call to `getNext`, it is no longer available to be returned from subsequent calls to `getNext`. If no strings are available to be returned, `getNext` returns `"NONE"`.

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

```
String[] wordArray = {"wheels", "on", "the", "bus"};
RandomStringChooser sChooser = new RandomStringChooser(wordArray);
for (int k = 0; k < 6; k++)
{
   System.out.print(sChooser.getNext() + " ");
}
```

One possible output is shown below. Because `sChooser` has only four strings, the string `"NONE"` is printed twice.

```
bus the wheels on NONE NONE
```

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 constructor 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)
   {  /* to be implemented in part (b) */  }

   /** 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)
   {  /* implementation not shown */  }
}
```

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

```
RandomLetterChooser letterChooser = new RandomLetterChooser("cat");
for (int k = 0; k < 4; k++)
{
   System.out.print(letterChooser.getNext());
}
```

The code segment will print the three letters in `"cat"` in one of the possible orders. Because there are only three letters in the original string, the code segment prints `"NONE"` the fourth time through the loop. One possible output is shown below.

```
actNONE
```

Assume that the `RandomStringChooser` class that you wrote in part (a) has been implemented correctly and that `getSingleLetters` works as specified. You must use `getSingleLetters` appropriately to receive full credit.

Complete the `RandomLetterChooser` constructor below.

```
/** Constructs a random letter chooser using the given string str.
 *  Precondition: str contains only letters.
 */
public RandomLetterChooser(String str)
```