Unit 7 FRQ

This question involves the creation of user names for an online system. A user name is created based on a user's first and last names. A new user name cannot be a duplicate of a user name already assigned. You will write the constructor and one method of the `UserName` class. A partial declaration of the `UserName` class is shown below.

```
public class UserName
{
    // The list of possible user names, based on a user's first and last
  names and initialized by the constructor.
    private ArrayList<String> possibleNames;

    /** Constructs a UserName object as described in part (a).
     * Precondition: firstName and lastName have length greater than 0
     * and contain only uppercase and lowercase letters.
     */
    public UserName(String firstName, String lastName)
    { /* to be implemented in part (a) */ }

    /** Returns true if arr contains name, and false otherwise. */
    public boolean isUsed(String name, String[] arr)
    { /* implementation not shown */ }

    /** Removes strings from possibleNames that are found in usedNames
  as described in part (b).
     */
    public void setAvailableUserNames(String[] usedNames)
    { /* to be implemented in part (b) */ }
}
```

(a) Write the constructor for the `UserName` class. The constructor initializes and fills `possibleNames` with possible user names based on the `firstName` and `lastName` parameters. The possible user names are obtained by concatenating `lastName` with different substrings of `firstName`. The substrings begin with the first character of `firstName` and the lengths of the substrings take on all values from 1 to the length of `firstName`.

The following example shows the contents of `possibleNames` after a `UserName` object has been instantiated.

Example

```
UserName person = new UserName("john", "smith");
```

After the code segment has been executed, the `possibleNames` instance variable of `person` will contain the following `String` objects in some order.

```
"smithj", "smithjo", "smithjoh", "smithjohn"
```

Write the `UserName` constructor below.

```
/** Constructs a UserName object as described in part (a).
 * Precondition: firstName and lastName have length greater than 0
 * and contain only uppercase and lowercase letters.
 */
public UserName(String firstName, String lastName)
```

Please respond on separate paper, following directions from your teacher.

(b) Write the `UserName` method `setAvailableUserNames`. The method removes from `possibleNames` all names that are found in `usedNames`. These represent user names that have already been assigned in the online system and are therefore unavailable.

A helper method, `isUsed`, has been provided. The `isUsed` method searches for `name` in `arr`. The method returns `true` if an exact match is found and returns `false` otherwise.

The example below shows the intended behavior of the `setAvailableUserNames` method.

| **Statement** | **Strings in `possibleNames` after statement execution** |
|---|---|
| `String[] used = {"harta", "hartm", "harty"};` | |
| `UserName person2 = new UserName("mary", "hart");` | `"hartm", "hartma", "hartmar", "hartmary"` |
| `person2.setAvailableUserNames(used);` | `"hartma", "hartmar", "hartmary"` |

Assume that the constructor works as specified, regardless of what you wrote in part (a). You must use `isUsed` appropriately to receive full credit.

Complete the `setAvailableUserNames` method below.

```
/** Removes strings from possibleNames that are found in usedNames as
described in part (b).
*/
public void setAvailableUserNames(String[] usedNames)
```

Please respond on separate paper, following directions from your teacher.

Unit 7 FRQ Answer sheet Name: _____

Part a:

**Write the `UserName` constructor below.**

```
/** Constructs a UserName object as described in part (a).
 * Precondition: firstName and lastName have length greater than 0
 * and contain only uppercase and lowercase letters.
 */
public UserName(String firstName, String lastName)
```

Part b:

**Complete the `setAvailableUserNames` method below.**

```
/** Removes strings from possibleNames that are found in usedNames as
described in part (b).
 */
public void setAvailableUserNames(String[] usedNames)
```

(more on back)

c) Write one line of code to create a UserName object "**hp**" using the name "Harry Potter".

d) After creating **hp**, what are the contents of the **hp**'s variable possibleNames?

e) As System Administrator (a title you proudly hold), you know that the user name "potterh" has already been taken. Write code to update the **hp** object with this information. (You need to use the setAvailableUserNames method to do this.)

f) What is wrong with the following code?

```
UserName Draco Malfoy = new UserName("draco","malfoy");
```

g) Write code to ask the user for a first and last name using the given Scanner and then create a UserName object **user** using their information.

```
Scanner scan = new Scanner(System.in);
```