CMSC 131 Quiz 6 Worksheet

The next quiz will be on **Monday**, Nov 1 during your lab session (either at 10 am or 11 am). The following list provides more information about the quiz:

- It will be a written quiz (not using any computer).
- It will be closed-book, closed-notes, and no calculator is allowed.
- Answers must be neat and legible. We recommend that you use pencil and eraser.
- The quiz will be based on the exercises you will find below.

The following exercises cover the material to be covered in Quiz #6. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with TAs and instructors during office hours.

When asked for a “piece of code” you do not need to provide an entire class definition or even an entire method. Just present the Java statements and any variable declarations as needed to make your solution clear.

**Problem 1**

Write a piece of code that determines whether two arrays of String references have the same elements, but perhaps in a different order. For example, given the following arrays:

```java
String[] a = {"Jose", "Rose", "Mary", "Jack"};
String[] b = {"Rose", "Jose", "Jack", "Mary"};
String[] c = {"Luis", "Laura"};
```

we can see that a and b are the only arrays that have the same elements.

**Problem 2**

Write a piece of code that removes the first element of an array of String references. A new array with a size that is one less the size of the original array should be created, and elements from the original array should be shifted down by one position. You may assume that the array contains at least one element to start.

**Problem 3**

Write a static method which takes as parameter an array of String references and returns a new array where null entries from the original array have been removed. For example, given the array:

```java
String[] a = {"Jose", "Rose", null, "Jack"};
```

the method will return an array with the following elements:

```java
String[] b = {"Jose", "Rose", "Jack"};
```
Problem 4

Write a static method that takes as parameter an array of String references and returns a complete duplicate of the array. That is, the String objects referred to by the new array are copies of the original String objects.

Problem 5

Write a static method that takes as parameters an array of String references and a String object. The method will insert the String into the array in such a way the array is kept sorted. You can assume that the String objects in the provided array are already sorted and they occupy the leftmost entries of the array. You can also assume that available (“empty”) entries of the array are set to null. If the array is already “full” then no insertion should occur.

Problem 6

Write a static method that takes as parameter an array of String references. The method will return the index of the array entry corresponding to the String with the maximum length in the array. If there are two or more String objects with the same maximum value then return the lowest index found.

Supplemental Problems

The following supplemental problems have been provided for your own practice. The quiz will not be based on these problems, but they will be good practice for upcoming exams.

Problem 1

Write a piece of code that appends the contents of one array of String references to the end of another. A new array (with a length corresponding to sum of the lengths of the originals arrays) should be created.

Problem 2

Write a static method that takes as parameters two arrays of String references and returns a new array representing the merging of the two arrays. The merging operation will create a new array by selecting alternate elements from the source arrays. For example, given the following two arrays:

```java
String[] a = {“Jose”, “Rose”, “Mary”, “Jack”};
String[] b = {“Pete”, “Larry”};
```

The new array will have the elements:

```java
String[] c = {“Jose”, “Pete”, “Rose”, “Larry”, “Mary”, “Jack”};
```
**Problem 3**

Write a static method that takes as parameter an array of String references and returns a String that represents the concatenation of all the String objects in the array. For example, given the following array:

```java
String[] c = {"Luis", "Laura", "Lupe"};
```

the method should return the String “LuisLauraLupe”.

**Problem 4**

Write a static method that takes as parameter an array of String references. The method will return an array of String references where duplicates from the parameter have been removed. The original array cannot be modified in any way.

**Problem 5**

Define a class called `Computer` which has the following specifications:

**Instance variables**

- `String name;` // computer name (e.g., twix.cs.umd.edu)
- `double diskCapacity;` // in megabytes
- `int numOfUsers;` // number of users

**Methods**

1. Define a default constructor that initializes the name, diskCapacity, and numOfUsers to “NONAME”, 0, and 0 respectively.
2. Define a constructor that initializes the data members of the object using a set of provided parameters. The prototype of the constructor is:

   ```java
   Computer(String nameIn, double diskCapacityIn, int numOfUsersIn);
   ```

3. Define a copy constructor.
4. Define an appropriate toString method.
5. Define appropriate accessor and mutator methods.
6. Define an equals method. Two computers are considered equal if they have the same name.
Problem 6

The following problem relies on the **Computer** class you defined above. Define a class called **Lab** with the following specifications:

### Instance variables

- String labName; // name of the computer lab
- Computer[] computers; // array of references to computer objects
- int currentNumComputers; // current number of computer objects in the array

### Methods

1. Define a default constructor which initializes the Lab instance variables to “NONAME”, null, and 0 respectively.

2. Define a constructor that has the following prototype:

   ```
   Lab(String labNameIn, int maxNumComputersIn);
   ```

   The constructor will create an array of computers with a length equal to maxNumComputersIn and it will initialize the name of the computer with the corresponding parameter. Notice that the currentNumComputers must be set to 0.

3. Define a copy constructor for the class.

4. Define an add method that will add a computer at the beginning of the array. The method will return true if the addition was successful and false otherwise (e.g., no more space available in the array). The prototype of the method is:

   ```
   boolean add(Computer computerIn)
   ```

   You must not add a computer who is already present in the collection. Each computer is uniquely identified by its name.

5. Define a remove method that will remove the computer with the specified name from the array. If necessary, elements will be shifted to the left to keep the entries arranged contiguously. The method will return true if the addition was successful and false otherwise (e.g., there is no computer with the specified name).

6. Define appropriate accessor and mutator methods for the Lab class.

7. Define a method called find that takes as parameter a String object. The method will return a reference to the computer associated with that name or null if the there is no such computer.

8. Define an appropriate toString method for the class
9. Write a main method that tests your class.

10. Write a memory diagram for the main method of your class.